Ciudad de La Habana, Noviembre de 2012

Universidad de las Ciencias Informáticas. Facultad 2

SUBSISTEMA PARA LA SINCRONIZACIÓN DE INFORMACIÓN EN TIEMPO REAL EN LOS SISTEMAS DE GESTIÓN DE EMERGENCIAS CIUDADANAS

Trabajo final presentado en opción al título de

Máster en Informática Aplicada

Autora: Ing. Yahima Vigo Valdés
Tutores: Dr. Ana María García Pérez
Dr. Rafael Arturo Trujillo Rasúa

Agradecimientos

Agradecer a mis tutores por la ayuda que me han dado a pesar del poco tiempo que disponen. Siempre supieron encontrar un huequito para atenderme.

A mis padres por todo el apoyo que me han brindado y la fuerza que me dan para que siga adelante. A mis abuelos, a mis tíos y a mi familia en general por confiar en mí.

A mi novio por su paciencia infinita hacia mi persona y darme el soporte que necesitaba.

A todos mis amigos que han servido de mucha ayuda como Adrian, Gilberto, Rammel, Wilson, Rasiel, Manfred, Zory, Yadira, Yero, Puchi, y otros que han estado preocupados por la culminación de la misma.

Dedicatoria

A mis padres, Maritza y Nemesio.

A mi abuela Mamina.

DECLARACIÓN JURADA DE AUTORÍA Y AGRADECIMIENTOS

Declaro por este medio que yo Yahima Vigo Valdés, con carné de identidad 84033018058, soy el autor principal del trabajo final de maestría Subsistema para la sincronización de información en tiempo real en los sistemas de gestión de emergencias ciudadanas, desarrollada como parte de la Maestría en Informática Aplicada y que autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Y para que así conste, firmo la presente declaración jurada de autoría en Ciudad de La Habana a los días del mes de noviembre del año 2012.

Resumen

Los centros de gestión de emergencias tienen como finalidad la recepción de las llamadas de auxilio y su gestión ante los servicios oportunos, realizando un seguimiento de la evolución de la emergencia hasta su cierre. Estos centros poseen sistemas automatizados de gestión de emergencias para agilizar el proceso de atención a las solicitudes de la población. El Sistema de Gestión de Emergencias y Seguridad Ciudadana 171 es un sistema que será desarrollado para la República Bolivariana de Venezuela.

En los sistemas de gestión de emergencias existentes actualmente en Venezuela la información que se comparte entre varios subsistemas no se actualiza en el momento que ocurre el cambio sobre un elemento, sino cada cierto tiempo definido en el sistema. Esto conlleva a que en ocasiones los subsistemas trabajen con información desactualizada, provocando retraso en la atención de las emergencias.

Para garantizar la coherencia y actualización de la información en varios subsistemas cuyos datos son críticos para la garantía de la vida humana se propone implementar un mecanismo de sincronización de información en tiempo real.

En este trabajo se incluyen conceptos relacionados con los mecanismos de sincronización y se recogen los resultados de las investigaciones realizadas durante el proyecto logrando el diseño e implementación de la propuesta realizada. Finalmente se muestra la validación de la solución y se plasman algunas recomendaciones para próximas versiones del mecanismo de sincronización.

Palabras claves: Mecanismos de sincronización, sincronización de datos, sistemas distribuidos, tecnologías de sincronización.

Abstract

The emergency management centers are intended for the reception of emergency calls and its management for appropriate services. Monitoring the evolution of emergency until it is closed. These centers have automated systems for emergency management to facilitate the process of attention to requests from the population.

The proposed emergency management system is distributed with a central database, with subsystems communication through a LAN, which should not allow access from Internet or communicate with external networks. It consists of several subsystems that share information, so you need to communicate constantly to work with updated data and carry out the response, in an efficient way, to requests from the population.

To ensure consistency and updating information in multiple subsystems for which data are critical to the security of human life the designing of a mechanisms for synchronization of software is proposed.

This paper describes the solution for the sub synchronization system, as part of the emergency management center, which includes concepts related to synchronization mechanisms and reflects the results of research conducted.

Keywords: Data synchronization, distributed systems, technologies for synchronization, synchronization mechanisms.

Índice

Introducción **1**

Capítulo 1. Fundamentación teórica **8**

1.1 Sistemas distribuidos 8
1.1.1 Características 9
1.2 Arquitecturas de sistemas distribuidos 10
1.2.1. Patrones arquitectónicos 10
1.3 Paradigmas de comunicación 11
1.3.1 Comunicación entre procesos 13
1.3.2 Invocación remota 14
1.3.3 Comunicación indirecta 14
1.4 Protocolos de comunicación 16
1.4.1 Protocolo TCP 17
1.4.2 Protocolo UDP 18
1.5 Sistemas de gestión de emergencias 18
1.6 Sincronización de datos 21
1.6.1 Tiempo real y sincronización de datos en el sistema de gestión de emergencias 23
1.7 Fundamentación del objetivo 25
1.8 Tecnologías de sincronización 25
1.8.1 Colas de mensajería de Microsoft 26
1.8.2 Lightstreamer 26
1.8.3 Conclusiones parciales 27
1.9 Servicios de Microsoft Windows 27
Conclusiones del capítulo 28

Capítulo 2. Propuesta del subsistema de sincronización de datos **29**

2.1 Sistema de gestión de emergencias de seguridad ciudadana 171 29
2.2 Escenarios de sincronización de datos en el SIGESC 30
2.3 Aspectos arquitectónicos en el mecanismo de sincronización 32
2.3.1 Modelo arquitectónico seleccionado 32
2.3.2 Patrones arquitectónicos seleccionados 34
2.3.3 Paradigmas de comunicación seleccionados 36
2.4 Arquitectura de sincronización de datos en el SIGESC 37
2.4.1 Organización arquitectónica 37
2.4.2 Interacción entre componentes 38
2.5 Arquitectura del subsistema de sincronización 39
2.6 Subsistema servidor de sincronización 40
2.6.1 Modelos de suscripción seleccionados del paradigma publica-suscribe 40
2.6.2 Mecanismo de comunicaciones 42
2.6.3 Protocolo de sincronización 44
2.6.4 Notificaciones 46
2.6.5 Vistas arquitectónicas 47
2.7 Subsistema de administración de sincronización 53
2.7.1 Mecanismo de administración 53
2.7.2 Distribución 54
2.7.3 Vistas Arquitectónicas 55
Conclusiones del capítulo 61

Capítulo 3 Validación de la solución **62**

3.1 Prueba de rendimiento del sistema 62
3.1.1 Selección de niveles para las variables independientes 63
3.1.2 Rendimiento del subsistema de sincronización en un escenario 64
3.1.3 Rendimiento del proceso sincronización 66
Conclusiones del capítulo 68

Conclusiones **69**
Recomendaciones **70**
Referencias bibliográficas **71**

Introducción

Todo Estado debe garantizar la organización de los órganos de seguridad ciudadana, para evitar o enfrentar situaciones que constituyan riesgos para la integridad física de las personas, sus propiedades, el disfrute de sus derechos y el cumplimiento de sus deberes. (CIDH y OEA, 2010)

Para que los órganos de seguridad ciudadana de un país puedan actuar de manera efectiva frente a una situación de emergencia, existen vías que facilitan a la población reportar los sucesos acontecidos. Varios países en el mundo han aprovechado el uso de las Tecnologías de la Información y las Comunicaciones (TIC) para automatizar los procesos de gestión de emergencias, propiciando una mayor disponibilidad de la información necesaria en los centros de gestión de emergencia (CGE) para proceder a solucionarlas.

Los centros de gestión de emergencias contribuyen a garantizar la seguridad ciudadana en un Estado, conllevando a salvaguardar las vidas humanas de las personas. Se entiende por seguridad ciudadana, la acción integrada que desarrolla el Estado, con la colaboración de la ciudadanía, destinada a asegurar su convivencia pacífica, la erradicación de la violencia y la utilización pacífica de las vías y espacios públicos, así como la contribución a la prevención de la comisión de delitos y faltas. (PARLAMENTARIA, 2005)

Los CGE son instituciones que se encargan de recibir llamadas de auxilio de la población u otros sectores, administrando, desde uno o varios puntos, los esfuerzos de los diferentes órganos de seguridad, para dar atención a las solicitudes de emergencias realizadas por la población (JURÍDICAS, 2010). Se caracterizan por laborar los 365 días del año y las 24 horas del día. Deben mantener un servicio de comunicaciones que permita garantizar la adecuada supervisión y la mejora de la capacidad de respuesta de los organismos.

Los CGE basan su funcionamiento en procesos que prestan servicios a la población y procesos de apoyo o soporte que permiten dar servicios de una manera más efectiva. Los procesos que prestan el servicio a la población son:

- Recepción de llamadas telefónicas o mensajería (e-mail, fax, sms, etc): el proceso inicia desde que el usuario intenta su comunicación para reportar el incidente de seguridad y se recibe en el centro. Luego el Operador de recepción realiza preguntas, siguiendo un procedimiento, hasta clasificar la emergencia y

obtener el lugar donde ocurre (GODDARD, 2009). Finalmente se procede al envío telemático de la información completa, íntegra y veraz a las agencias de Despacho para que den una respuesta coordinada y efectiva según sus competencias.

- Despacho: luego de recibir la información de la emergencia, se encarga de contactar con las unidades en servicio para movilizarlas a dar respuesta a cada situación de emergencia reportada por los ciudadanos. Se realiza un seguimiento de las solicitudes hasta su cierre, supervisando la actividad de las unidades en tiempo real, registrando las acciones que se llevan a cabo en la atención de las solicitudes y a través del sistema de comunicaciones que utilice el centro (GOBIERNO, 2005). Los puestos de despacho pueden estar dentro del propio centro de emergencia o pueden estar en otro local.

  Existen centros donde no se coordina directamente con los recursos, sino que se contacta con los órganos de seguridad para que estos coordinen con los recursos que tengan disponibles.

  Como apoyo al despacho de las solicitudes se puede utilizar un mapa digital del área que corresponde atender. Éste contiene la ubicación de las unidades en servicio en tiempo real, indicando su estado (ocupado, disponible, etc) (2MARES y SATDATA, 2011), solicitudes y sitios de interés. Además, en el mapa se pueden realizar cálculos para sugerir las unidades más cercanas a una emergencia, las rutas más cercanas para llegar a un lugar específico, lo que posibilita una mejor apreciación de la situación en el territorio y por tanto permite tomar decisiones más precisas. (GUNES y KOVEL, 2000)
- Supervisión: controla y supervisa la calidad del trabajo de las otras áreas. Se encarga de controlar la gestión de operadores (2MARES, 2011), despachadores y funcionamiento en general del Centro. Cuenta para ello con varias aplicaciones que posibilitan, de forma permanente, monitorear las diferentes acciones que realiza el personal del Centro de acuerdo a sus competencias. La supervisión implica, esencialmente, la asesoría, orientación y el seguimiento para conocer, comprender e intervenir en la obtención del mejoramiento de la calidad del trabajo realizado por otros o por uno mismo. Dicha actividad se debe realizar sistemáticamente y debe ser rigurosa, coherente, flexible y ética.

Los centros tienen otros procesos que contribuyen a su correcto funcionamiento. Los procesos de administración, que van a garantizar el control de los trabajadores y sus funciones, así como los parámetros y normas a tener en cuenta para su correcto funcionamiento. También, los procesos complementarios, que incorporan elementos al servicio para mejorar la calidad de la atención, entre los que se pueden mencionar el análisis estadístico y la administración de los recursos.

Estos centros utilizan sistemas automatizados, llamados sistemas de gestión de emergencias, compuestos por subsistemas de cómputo, telefonía, radio e información operativa. Los sistemas de gestión de emergencias permiten mejorar los tiempos de respuesta a las llamadas de auxilio; facilitando, además de la atención adecuada de las llamadas que se produzcan, una actuación coordinada y eficaz de los servicios públicos que ayude a controlar el evento y preserve de esta forma la vida de los ciudadanos involucrados.

El objetivo de estos sistemas siempre está encaminado a aumentar la confiabilidad en la solicitud por parte de la población, y lograr una mayor disponibilidad y accesibilidad de la información por los organismos encargados de dar solución a las emergencias.

Un sistema de gestión de emergencias tiene entre sus funciones: (GOBIERNO, 2005)

- Desarrollar y mantener un subsistema de recepción y transferencia de llamadas de emergencias a los Centros de Despacho y organismos públicos y privados que corresponda durante las veinticuatro horas de todos los días del año sin interrupción alguna.
- Recibir, procesar automáticamente y atender de manera centralizada las llamadas de emergencias dirigidas al número establecido a tales fines.
- Contener y orientar inmediatamente a los usuarios del sistema.
- Coordinar la elaboración de los protocolos de comunicación para la actuación concreta de los distintos órganos involucrados.
- Realizar un seguimiento integral del incidente y controlar la calidad de la prestación final según la emergencia.
- Organizar y mantener actualizado un registro de las emergencias recibidas, de las que fueran derivadas.
- Optimizar la utilización de los recursos disponibles para la atención de emergencias y promover las acciones necesarias tendientes a lograrlo ante las autoridades competentes.

- Promover la suscripción de convenios necesarios con las administraciones y entidades públicas o privadas que dispongan de los recursos indispensables para la implementación del sistema, así como para establecer procedimientos de atención y coordinación.

El desarrollo de un sistema de gestión de emergencias es una tarea compleja, por el volumen de información que se maneja diariamente al tener que almacenar la información de todas las llamadas realizadas al centro y las acciones realizadas por los órganos de seguridad para darle solución. Además, los sistemas deben interactuar con tecnologías telemáticas novedosas de radio, telefonía, video vigilancia, Sistema de Posicionamiento Global (GPS), entre otras, para lograr mayor efectividad en la solución de las solicitudes. Estos sistemas necesitan en su mayoría la interacción con sistemas externos, requieren de una alta disponibilidad al prestar servicios de forma ininterrumpida y deben tener capacidad de crecimiento al poder modificar la cantidad de usuarios (operadores, despachadores o supervisores) para responder ante una situación de emergencia, entre otras.

Los sistemas de gestión de emergencias generalmente están compuestos por varios subsistemas que comparten información. Los subsistemas automatizan los procesos que tenga el centro para prestar el servicio de atención a emergencias. Están ubicados en computadoras autónomas que pueden estar distantes geográficamente, que necesitan comunicarse constantemente para trabajar con los datos actualizados y llevar a cabo la atención a las solicitudes realizadas por la población de forma eficiente. Además, esta información se maneja en tiempo real y en ocasiones puede ser modificada por algún subsistema mientras que otros la están utilizando.

En un sistema de gestión de emergencias es necesario trabajar con la información actualizada, puesto que constantemente se toman decisiones para resolver un incidente a partir de la información visualizada en los subsistemas. Un despachador debe conocer el estado y posición real de las unidades móviles que gestiona, para poder asignar el recurso con estado disponible más cercano al lugar donde ocurre una emergencia. Si el estado no está actualizado puede provocar pérdida de tiempo, pues se cuenta con una unidad visualizada como “disponible”, cuando en realidad no puede ser utilizada en la atención a la emergencia porque un supervisor la asignó a otra tarea desde otro subsistema. En el caso de la posición, sucede algo similar, el despachador traza una estrategia para la resolución de una emergencia a partir de la información que tiene visible, por lo que si no se tiene la ubicación real de las unidades puede

ocurrir que la unidad que se muestra como la más cercana a un incidente no lo sea realmente. Esto provoca que se asigne al incidente una unidad que solucionará el problema, pero tomará más tiempo en llegar al lugar de los hechos que una más cercana y el tiempo en un sistema de gestión de emergencias puede significar la vida de las personas.

También, en un centro de atención de emergencias, se reciben llamadas de la población para informarse sobre el estado de un incidente reportado. En estos casos el operador telefónico debe tener la información actualizada del incidente, para indicar al ciudadano si se asignaron unidades por parte de los despachadores y el estado de las mismas. Si los operadores no tienen la información veraz del incidente que están resolviendo los despachadores, el centro no podrá brindar este servicio de manera eficiente y puede provocar que la población deje de reportar emergencias que pueden ser resueltas en un período de tiempo adecuado.

El hecho de que la información compartida entre varios subsistemas no se actualice en tiempo real, trae consigo un mal funcionamiento en los sistemas de gestión de emergencias que puede provocar la toma de decisiones erradas, que conllevan a que aumenten los tiempos de respuesta a las situaciones de emergencias, se cree un estado de inseguridad en la población y se tengan repercusiones para la gobernación de la ciudad.

En correspondencia con lo anterior el **problema de investigación** consiste en: ¿Cómo actualizar y mantener coherente la información compartida entre los procesos de gestión de emergencias en el menor tiempo posible para contribuir a garantizar la vida humana en condiciones de emergencia?

El **objeto de investigación** es: las soluciones automatizadas empleadas para mantener la información actualizada y en estado coherente en varios procesos. Se plantea como **objetivo general** de la investigación: desarrollar un subsistema informático de sincronización de información en sistemas de gestión de emergencias ciudadanas, para garantizar la coherencia y actualización de la información en tiempo real en los subsistemas que la comparten.

El **campo de acción** lo constituyen: los sistemas informáticos de sincronización de información en los procesos de gestión de emergencias ciudadanas en sistemas distribuidos.

Las **preguntas científicas** que guiarán la investigación, de acuerdo con el problema científico y las necesidades detectadas, son:

- ¿Cuál es el estado del arte en los sistemas informáticos para la atención a emergencias ciudadanas?
- ¿Qué modelo arquitectónico es apropiado utilizar en un sistema de gestión de emergencias para mantener la información actualizada en los subsistemas que la comparten?
- ¿Qué paradigma de comunicación emplear para lograr la sincronización de la información en tiempo real en un sistema de gestión de emergencias?
- ¿Cómo debe ser el subsistema informático de sincronización de información para la atención a emergencias ciudadanas?
- ¿Qué factores afectan la actualización de la información en tiempo real?

Derivado de la relación entre el problema científico, las preguntas científicas, el objeto y el objetivo de la investigación se desarrollaron los siguientes **objetivos específicos**:

- Realizar un análisis de los conceptos relacionados con la investigación referente a los sistemas de gestión de emergencias, la sincronización de datos, los sistemas distribuidos y sus modelos arquitectónicos.
- Realizar un análisis de las técnicas y tecnologías empleadas actualmente para mantener la información actualizada en varios subsistemas de un sistema y en los procesos de gestión de emergencias.
- Definir el modelo arquitectónico del componente para la sincronización de información en los procesos de atención a emergencias.
- Desarrollar el subsistema para la sincronización de información en los procesos de atención a emergencias.
- Validar la solución propuesta a través del estudio de caso para comprobar que la información se actualiza en tiempo real.

En correspondencia con el objetivo propuesto se proponen utilizar los siguientes **métodos científicos:**

Métodos generales: Se utilizó el método histórico-lógico y el dialéctico para el estudio crítico de trabajos anteriores, y para utilizar estos como punto de referencia y comparación de los resultados alcanzados.

Métodos lógicos: El método analítico-sintético al descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución de la propuesta; el método de idealización-modelación para explicar por qué el componente propuesto es el que más se ajusta a las características de los procesos de gestión de emergencias.

Métodos empíricos: El método coloquial para la presentación y discusión de los resultados en sesiones científicas; el método de la entrevista y la observación para determinar los problemas presentes en los sistemas de gestión de emergencias estudiados; el método experimental para comprobar la utilidad de los resultados obtenidos a partir del subsistema implementado.

Como resultado de la investigación se obtendrá un subsistema de sincronización de datos para aplicar en el Sistema de Gestión de Emergencias de Seguridad Ciudadana 171 (SIGESC), con la definición de las técnicas y herramientas a utilizar. Se espera que con la aplicación de este subsistema se actualice en tiempo real. También, que las personas que utilizan el sistema se sientan confiadas en la información mostrada en el mismo y puedan tomar mejores decisiones que contribuyan a garantizar la seguridad ciudadana. Además, con el uso del subsistema de sincronización se pretende que la población se sienta más segura y protegida al mejorar los tiempos de respuesta del centro donde se implante el sistema.

El documento de la presente investigación está **estructurado** en tres capítulos. En el capítulo uno se estudian los conceptos relacionados con la investigación y se realiza un estudio del estado del arte en la aplicación de tecnologías informáticas que se han usado para mantener actualizada y coherente la información en varios subsistemas que la visualizan y modifican al mismo tiempo.

El segundo capítulo describe la solución empleada en el desarrollo del subsistema para la sincronización de información, explicando su funcionamiento, modo de trabajo, utilidad y elementos que lo componen.

En el tercero se realiza una validación de la solución dada para el SIGESC, revisando el rendimiento de la aplicación en diferentes escenarios, teniendo en cuenta los recursos utilizados.

Capítulo 1. Fundamentación teórica

En este capítulo se plasman definiciones que son necesarias para la realización de la investigación, relacionadas con los sistemas distribuidos, los sistemas de gestión de emergencias y la sincronización de datos. Además, se presentan diversas tecnologías de sincronización que se han utilizado en la Informática, para tratar el problema de numerosos subsistemas compartiendo la misma información y que debe ser actualizada en tiempo real.

1.1 Sistemas distribuidos

En la actualidad, los sistemas distribuidos están cada vez más presentes en la sociedad, motivado por la necesidad de intercambio de la información así como de compartir los recursos tanto de software como de hardware a través de la red.

Algunas definiciones de sistemas distribuidos son:

- "Sistema en el cual múltiples procesadores autónomos, posiblemente de diferentes tipos, están interconectados por una subred de comunicación para interactuar de una manera cooperativa en el logro de un objetivo global". (LELANN, 1981)
- "Un sistema distribuido es aquel en el que los componentes localizados en computadores conectados en red, comunican y coordinan sus acciones únicamente mediante el paso de mensajes." (COULOURIS *et al.*, 2001)
- "Conjunto de computadores independientes que se muestran al usuario como un sistema único coherente." (TANENBAUM y VAN STEEN, 2006)

En sentido general, un sistema distribuido está compuesto por aplicaciones que se ejecutan en diferentes computadoras geográficamente distribuidas, que pueden encontrarse distantes unas de otras. Las computadoras se encuentran interconectadas por una red, en un constante intercambio de mensajes que permite la transferencia de información para lograr que el sistema funcione de forma correcta y cooperativa para alcanzar un objetivo común. Es un sistema único, por lo que tiene un único estado global compartido por todas las computadoras que lo componen. El estado global se refiere a datos como la hora actual y datos compartidos por procesos de distintos ordenadores.

1.1.1 Características

Los sistemas distribuidos tienen características que deben cumplir para que se comporten según esperan los usuarios. Coulouris estableció ocho características o desafíos principales que se describen a continuación: (COULOURIS *et al.*, 2012)

Abiertos: son sistemas que pueden ser extendidos de diversas maneras. La apertura de los sistemas distribuidos se determina por el grado hacia el que nuevos servicios de compartición de recursos se pueden añadir sin perjudicar ni duplicar a los ya existentes.

Concurrencia: la presencia de múltiples usuarios es una fuente de peticiones concurrentes de sus recursos. Cada recurso compartido debe ser diseñado para ser salvado de un entorno concurrente, donde sus operaciones deben estar sincronizadas de tal manera que sus datos se mantengan consistentes.

Transparentes: la meta es hacer ciertos aspectos de distribución invisible para el usuario y los programadores del sistema. El sistema debe ser percibido como un todo, en vez de una colección de componentes independientes. La transparencia ejerce una gran influencia en el diseño del software de sistema.

Tolerante a fallos: cualquier proceso, computadora o red puede fallar independientemente de los otros. Sin embargo, cada componente tiene que ser consciente de las posibles formas en que los componentes de los que depende pueden fallar y estar diseñados para hacer frente a cada uno de esos fracasos adecuadamente.

Heterogéneos: deberán ser construidos a partir de una variedad de diferentes redes, sistemas operativos, hardware y lenguajes de programación. Los protocolos de comunicación de Internet pueden enmascarar la diferencia en redes, *middleware* y pueden hacer frente a las otras diferencias.

Seguros: el cifrado se puede utilizar para proporcionar una protección adecuada de los recursos compartidos y de mantener en secreto la información confidencial cuando se transmite en mensajes a través de una red.

Escalables: un sistema distribuido es escalable si el costo de agregar un usuario es una cantidad constante en términos de los recursos que deben ser añadidos. Los algoritmos utilizados para acceder a los datos compartidos deben evitar los “cuellos de botella” de rendimiento y los datos deben ser estructurados jerárquicamente para obtener los mejores tiempos de acceso.

Calidad de servicio: no es suficiente con proporcionar acceso a los servicios en sistemas distribuidos. También es importante proporcionar garantías sobre las cualidades asociadas con el acceso de dicho servicio. Ejemplos de tales cualidades pueden incluir parámetros relacionados con el rendimiento, seguridad y fiabilidad.

1.2 Arquitecturas de sistemas distribuidos

Los nodos de los sistemas distribuidos están interconectados a través de la red y se comunican mediante el paso de mensajes. Decidiendo sobre los componentes de software, su interacción y colocación, se pueden identificar varios modelos de arquitecturas del sistema. Las más conocidas son los enfoques centralizados y descentralizados, así como varias formas híbridas. (TANENBAUM y VAN STEEN, 2006)

Dentro de la arquitectura centralizada se encuentra el modelo cliente-servidor, donde los componentes del sistema presentan varios roles. En este modelo los procesos del sistema distribuido son divididos en dos grupos, un servidor que implementa un determinado servicio, que puede ser un servicio de base de datos (BD), y un cliente que solicita los servicios del servidor mediante el envío de una solicitud y espera una respuesta del servidor. Además, se pueden encontrar los modelos multicapas (*n-tiers) (TANENBAUM y VAN STEEN, 2006)*, que es una generalización del anterior, donde los componentes pueden emitir peticiones y responder a las peticiones sobre los recursos concretos que gestionan (clientes y servidores a la vez).

En la arquitectura descentralizada el modelo arquitectónico más usado es el *peer-to-peer* (punto a punto) (TANENBAUM y VAN STEEN, 2006). Los procesos que constituyen un sistema *peer-to-peer* son todos iguales, cada proceso actuará como un cliente y un servidor al mismo tiempo.

La arquitectura híbrida realiza combinaciones del enfoque descentralizado con el enfoque centralizado.

1.2.1. Patrones arquitectónicos

Los patrones arquitectónicos proporcionan una estructura compuesta que ha sido probada y funciona bien en circunstancias dadas. Los patrones se pueden combinar y pueden conllevar a dar una solución para un problema de dominio determinado.

Varios patrones se han identificado para sistemas distribuidos, pero solo se van a analizar los esenciales, que incluyen la arquitectura en capas (*layering*) y arquitectura escalonada (*tiered*) (COULOURIS *et al.*, 2012). La primera se refiere a la distribución

vertical del software, que cuando es muy complejo es necesario dividir en un número de capas, con una capa haciendo uso de los servicios de la capa inferior como se muestra en la Figura 1. Las capas *Hardware* y Sistema operativo prestan servicios a las capas superiores. *Middleware* es una capa de software cuyo objetivo es enmascarar la heterogeneidad y proporcionar un modelo de programación muy práctico para los programadores de aplicaciones. Está representado por los procesos u objetos en un conjunto de equipos que interactúan entre sí para implementar la comunicación y el intercambio de recursos de apoyo para las aplicaciones distribuidas. La capa Aplicaciones corresponde a las aplicaciones que tienen la lógica del negocio a resolver.

Figura 1. Capas de servicios hardware y software en sistemas distribuidos. (Tanenbaum, y otros, 2006)

El segundo patrón arquitectónico se complementa con el basado en capas, usa la fragmentación como técnica para organizar la funcionalidad de una determinada capa y colocar esta funcionalidad en servidores apropiados y, en un segundo orden de prioridad, en los nodos físicos. Esta técnica se aplica a todas las capas de una arquitectura de sistemas distribuidos, un ejemplo se muestra en la Figura 2 en la capa aplicación entre los nodos 2 y 3.

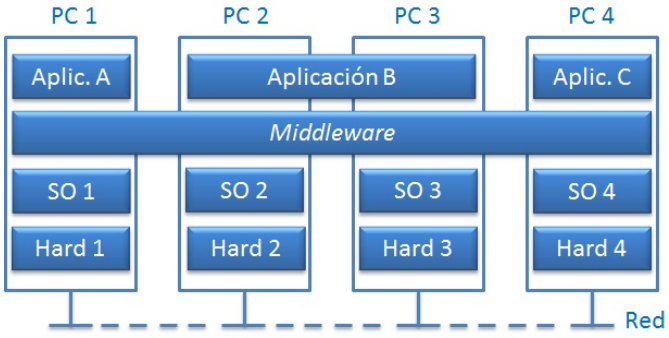

Figura 2. Sistema distribuido organizado como middleware. (Coulouris, y otros, 2012)

1.3 Paradigmas de comunicación

La comunicación desempeña un rol crucial entre las diversas entidades de los sistemas distribuidos. Para comunicar los nodos de los sistemas distribuidos se conocen tres

paradigmas de comunicación: comunicación entre procesos, invocación remota y comunicación indirecta. (COULOURIS *et al.*, 2001)

El primer paradigma comprende técnicas de bajo nivel para comunicar procesos, como son: paso de mensajes (JIA y WANLEI, 2005), programación con *sockets* (STEPISNIK, 2007) y *multicast. (COULOURIS et al., 2012)*

La invocación remota (VÖLTER *et al.*, 2005) abarca una serie de técnicas basadas en un intercambio de doble vía entre las entidades comunicantes dando lugar a la invocación de una operación, procedimiento o método a distancia, como son: protocolos de solicitud-respuesta (*Request-reply protocols*), llamadas a procedimientos remotos (*Remote procedure calls*, RPC) (BIRMAN, 2012) e invocación a métodos remotos (*Remote method invocation*, RMI) (GROSSO, 2002).

La comunicación indirecta se caracteriza por existir un intermediario entre el emisor y el receptor. Los dos paradigmas anteriormente descritos tienen en común que la comunicación representa una relación de dos vías entre un emisor y un receptor con remitentes dirigiendo explícitamente los mensajes o invocaciones a los receptores asociados. Los receptores conocen de la identidad de los remitentes, y en la mayoría de los casos ambas partes deben existir al mismo tiempo. Un número de técnicas han surgido para la comunicación indirecta, como: comunicación en grupo (*Group communication*) (RIZZO y FDIDA, 1999), sistemas Publica-suscribe (*Publish-subscribe systems*) (TARKOMA, 2012), colas de mensajes (*Message queues*) (BABER *et al.*, 2003), entre otros, lo que permite un alto grado de desacoplamiento entre emisores y receptores. En particular:

- Los remitentes no necesitan conocer los destinatarios de sus envíos (desacoplamiento de espacio).
- Los remitentes y receptores no tienen que existir al mismo tiempo (tiempo de desacoplamiento).

En la tabla 1 se muestra un resumen de los paradigmas de comunicación.

Tabla 1. Paradigmas de comunicación.

Paradigmas de comunicación (cómo se comunican)		
Comunicación entre procesos	**Invocación remota**	**Comunicación indirecta**
Paso de mensajes	Protocolos de solicitud-respuesta	Grupo de comunicación
Sockets	RPC	Publica-suscribe
Multicast	RMI	Colas de mensajes

1.3.1 Comunicación entre procesos

El paso de mensajes entre un par de procesos puede ser sustentado por dos operaciones de comunicación entre mensajes, enviar y recibir. Para comunicarse un proceso envía un mensaje (una secuencia de *bytes*) a un destino y en el destino otro proceso recibe el mensaje (JIA y WANLEI, 2005). Los mensajes pueden tener diferentes tamaños. Esta actividad puede implicar la sincronización de los dos procesos, el de envío y el de recepción. La comunicación puede ser sincrónica o asincrónica. El destino de los mensajes es identificado por una dirección IP (por sus siglas en inglés, Internet Protocol) y un puerto.

Sockets es un concepto abstracto por el cual dos procesos pueden intercambiar datos. La comunicación consiste en transmitir un mensaje entre un *socket* en un proceso y un *socket* de otro proceso. Para que un proceso pueda recibir mensajes, su *socket* debe estar vinculado a la dirección IP y un puerto de la computadora en la que se ejecuta.

Los mensajes enviados a una dirección IP y número de puerto sólo pueden ser recibidos por un proceso cuyo *socket* tenga esa dirección IP y número de puerto (STEPISNIK, 2007). Los procesos pueden utilizar el mismo *socket* para enviar y recibir mensajes. Cada ordenador tiene un número de puertos grande que pueden ser usados por los procesos locales para recibir mensajes. Cualquier proceso puede hacer uso de varios puertos para recibir mensajes, pero un proceso no puede compartir puertos con otros procesos en el mismo equipo. Sin embargo, los procesos pueden enviar mensajes al mismo puerto. Cada protocolo está asociado a un protocolo en específico, ya sea TCP (por sus siglas en inglés, *Transmission Control Protocol*) o UDP (por sus siglas en inglés, *User Datagram Protocol*).

Multicast es un método para transmitir datagramas IP a un grupo de receptores interesados. El número de receptores es transparente para el proceso que envía (COULOURIS *et al.*, 2012).

1.3.2 Invocación remota

Los protocolos de solicitud-respuesta constituyen patrones establecidos basados en el servicio de paso de mensajes que sirven de apoyo para la comunicación en el modelo cliente–servidor. Los protocolos suelen incluir un intercambio de parejas de mensajes, desde el cliente al servidor y de servidor a cliente. El primer mensaje contiene código de la operación que se ejecuta en el servidor, así como una cadena de *bytes* con los argumentos asociados, y el segundo mensaje contiene los resultados de la operación, y es nuevamente codificado como una cadena de *bytes*. Este paradigma es bastante primitivo y utilizado realmente en sistemas embebidos donde el rendimiento es primordial (COULOURIS *et al.*, 2012).

Las RPC permiten realizar llamadas a los procedimientos que se encuentran en otras computadoras ocultando los detalles propios de la comunicación que son comunes en los sistemas distribuidos, como la codificación y decodificación de parámetros y resultados, el paso de mensajes y la preservación de la semántica necesarias para la llamada de procedimiento (BIRMAN, 2012). Este enfoque soporta el modelo cliente-servidor con servidores que ofrecen un conjunto de operaciones a través de una interfaz de servicio y clientes que llaman estas operaciones directamente como si estuvieran disponibles localmente.

Las RMI permiten la colaboración de objetos que están localizados remotamente (GROSSO, 2002). Un objeto cliente puede invocar un método en un objeto remoto. El objeto remoto prepara la información requerida y envía la respuesta al cliente. Se asemejan a las RPC en que ocultan los detalles de la comunicación para el usuario. Suelen tener mayor integración con los lenguajes de programación orientados a objetos.

1.3.3 Comunicación indirecta

Entre los paradigmas de comunicación indirecta se encuentra la Comunicación en grupo, que realiza la entrega de los mensajes a un grupo de destinatarios, permitiendo así la comunicación uno a muchos (RIZZO y FDIDA, 1999). Se basa en la abstracción de un grupo que está representado en el sistema por un identificador y todos los interesados en recibir los mensajes que se envían a un grupo deben unirse al mismo. Los mensajes se envían al grupo a través del identificador de grupo, y por lo tanto no es necesario conocer los destinatarios del mensaje.

Los sistemas publica-suscribe están compuestos por publicadores, servidores publica-suscribe (*broker,* traducido al español como intermediario) y suscriptores (TARKOMA, 2012). Los publicadores no envían los mensajes directamente a los destinatarios, sino que los envían al intermediario que se encarga de gestionar que los mensajes se dirijan a los interesados en consumirlos, que son los llamados suscriptores. Puede existir un solo intermediario o varios. Los suscriptores deben registrar previamente en el intermediario su interés por los cambios realizados a un elemento. Una vez que el publicador notifica el cambio en un elemento, el intermediario identifica los interesados y envía automáticamente una notificación a cada uno. Con este paradigma los publicadores no tienen que preocuparse por la ubicación de los receptores, dejando esta responsabilidad al intermediario. Tiene como características distintivas que es heterogéneo, pues se necesita como medio de comunicación solamente una notificación que tiene que ser interpretada por los involucrados. Además es asincrónico, los publicadores no tienen que esperar una respuesta a su notificación.

En el sistema publica-suscribe son necesarios los modelos de suscripciones o filtrado, que permiten realizar el proceso de selección de los mensajes para la recepción y tratamiento de los mismos (COULOURIS *et al.*, 2012). Son usados frecuentemente los modelos de suscripción basado en tópico, basado en contenido y basado en tipo. En el modelo de suscripción basado en tópicos cada notificación es expresada en términos de un número de campos y un campo denota el tópico. Los suscriptores de un sistema basado en tópico recibirán todos los mensajes publicados en los tópicos a los que están inscritos y todos los suscriptores de un tópico recibirán el mismo mensaje. El publicador es el responsable de definir las clases de mensajes a las que se deben suscribir. El modelo basado en contenido es una generalización del enfoque basado en tópico, donde las expresiones de suscripción están formadas por una serie de campos de una notificación, o sea, es una cadena definida en términos de restricciones sobre los valores de los atributos de los eventos. Otro modelo de suscripción del sistema publica-suscribe es el basado en tipo, donde los eventos son vistos como objetos. Los suscriptores exponen sus preferencias basándose en los tipos de los objetos, y lo combinan con el paradigma basado en contenido teniendo en cuenta los atributos públicos de este tipo de evento.

Las colas de mensajes son otro paradigma de comunicación indirecta que provee comunicación punto a punto, a diferencia de los sistemas publica-suscribe y la comunicación en grupo. Los procesos producidos pueden enviar mensajes a una cola

específica y los procesos consumidores pueden recibir los mensajes de la cola o ser notificados de la llegada de un nuevo mensaje a la cola. Una vez que la cola ha sido creada, cualquier proceso puede escribir mensajes en la cola de mensajes así como leer mensajes de la cola de mensajes. Por tanto, es posible que varios procesos envíen mensajes a una misma cola y que múltiples procesos reciban mensajes de la cola de mensajes. Para obtener comunicación *full-duplex* se utilizan dos colas, una para cada dirección y es conveniente borrar las colas de mensajes cuando no van a ser utilizadas, ya que en caso contrario se malgastará de forma innecesaria la memoria del ordenador.

1.4 Protocolos de comunicación

Un protocolo de comunicación es una serie de convenciones y reglas que establecen cómo deben comunicarse las computadoras a través de la red (TANENBAUM, 2003). Para que las computadoras puedan funcionar en redes necesitan de estos protocolos, que transmiten la información fragmentada.

Un protocolo de comunicación describe los siguientes aspectos:

- el tiempo relativo al intercambio de mensajes entre dos sistemas de comunicación.
- el formato que el mensaje debe contener para que el intercambio entre dos computadoras que emplean protocolos diferentes puedan establecer comunicación.
- las acciones que deben realizarse en el caso de producirse error en la comunicación.
- las suposiciones acerca del medio ambiente en el cual se ejecutará el protocolo.

En definitiva, los protocolos de comunicación son programas (software) que se instalan tanto en la computadora de origen como en la destino. Estos programas añaden una serie de datos de control, a la información que se pretende trasmitir. Los datos de control son agregados por el trasmisor y suprimidos por el receptor antes que la información llegue al usuario.

Los protocolos pueden ser orientados a conexión y orientados a no-conexión. Los protocolos orientados a conexión operan en tres fases (TANENBAUM, 2003). La primera fase es configuración de la conexión, durante la cual las entidades correspondientes establecen la conexión y negocian los parámetros que definen la

conexión. La segunda fase es transferencia de datos, durante la cual las entidades correspondientes intercambian mensajes bajo el amparo de la conexión. Finalmente, la fase de liberación de la conexión, en la cual ambas entidades se ponen de acuerdo para terminar la conexión.

Los protocolos orientados a no-conexión difieren de los orientados a conexión, ya que están siempre en la fase de transferencia de datos, y no realizan las fases restantes de configuración y liberación de una conexión.

El medio de comunicación de un sistema de gestión de emergencias es guiado, o sea que está constituido por un cable que se encarga de la conducción de las señales desde un extremo al otro. Conociendo esta característica se estudian los protocolos de comunicación más empleados en este medio de comunicación.

1.4.1 Protocolo TCP

TCP (Protocolo de control de transmisión) es un protocolo confiable, orientado a la conexión, que permite que un flujo de *bytes* que se origina en una máquina se entregue sin errores en cualquier otra máquina en la red. Divide el flujo de *bytes* entrantes en mensajes discretos y pasa cada uno de ellos a la capa de red (TANENBAUM, 2003).

El modelo TCP/IP lo integran el protocolo TCP y el IP, o Protocolo de Internet. IP realiza el “ruteo” (o “encaminamiento“) de los paquetes desde la dirección IP de origen hasta la de destino. TCP es el protocolo encargado de la transferencia de paquetes, libre de errores con servicio de puertos, pudiendo recuperar paquetes perdidos o desordenados producidos por el protocolo IP. Coloca los datagramas nuevamente en orden cuando vienen del protocolo IP y permite el monitoreo del flujo de los datos evitando la saturación de la red. Si TCP determina que un paquete no ha sido recibido, intentará volver a enviarlo hasta que sea recibido correctamente.

El TCP/IP es un modelo básico sobre el que funcionan otros protocolos destinados a tareas específicas. Todos los procesos de nivel de aplicación que utilicen protocolos TCP/IP se deben identificar mediante un número de puerto. Este número se utiliza por las dos computadoras para identificar el programa de aplicación que va a recibir el tráfico entrante. El uso de número de puerto proporciona capacidades de multiplexación, ya que varios programas de usuarios se pueden comunicar de forma concurrente con un programa de aplicación como TCP. Los números de puerto sirven para identificar a cada aplicación.

La distribución del protocolo TCP/IP consta de cuatro niveles (TANENBAUM, 2003): aplicación, transporte, red y física o enlace de datos.

1.4.2 Protocolo UDP

El protocolo UDP (Protocolo de datagrama de usuario) es un protocolo no orientado a conexión de la capa de transporte del modelo TCP/IP. Permite a las aplicaciones enviar datagramas IP sin establecer conexión previa con el otro extremo. No tiene confirmación, ni control de flujo, los mensajes se envían y pueden duplicarse o llegar desordenados al destino (TANENBAUM, 2003). Además no es fiable, los mensajes se pueden perder o llegar dañados, y no se sabe porque no hay confirmación de entrega o de recepción. Cualquier tipo de garantías para la transmisión de la información, deben ser implementadas en capas superiores.

UDP se emplea mayormente en tareas de control y en la transmisión de audio y video a través de una red. No introduce ningún retardo para establecer una conexión. Utiliza el protocolo IP para trasportar sus mensajes.

Cuando es más importante la velocidad que la fiabilidad, se utiliza UDP. En cambio, TCP asegura la recepción en destino de la información a transmitir.

1.5 Sistemas de gestión de emergencias

Los sistemas de gestión de emergencias (SGE) son sistemas distribuidos (S.A, 2012). Sus subsistemas se instalan en computadoras independientes, que necesitan comunicarse, intercambiar y compartir datos. Permiten la deslocalización geográfica de nodos, posibilitando que se puedan utilizar subsistemas fuera del área del centro si se tienen los permisos necesarios.

Los SGE se encargan de optimizar la operatividad de los centros de gestión de emergencias y proporcionar las herramientas necesarias para integrar y gestionar tecnologías modernas en las comunicaciones (INSTITUTION, 2012), telefonía, GPS, mapeo digital, entre otras. Permiten mejorar los tiempos de respuesta a las solicitudes de auxilio, y contribuyen a garantizar una actuación coordinada y eficaz de los servicios públicos, mejorando la administración de los recursos disponibles para cada una de las instituciones involucradas. Además, facilitan la operatividad diaria y la toma decisiones, proponiendo opciones de respuesta que mejor se adaptan a cada situación, asistiendo al operador y ofreciendo información actualizada. (INDRA, 2011)

Los objetivos esencialmente de un SGE son: recopilar la información sobre la situación por la cual se inicia una solicitud y los procedimientos que se precisan, proporcionar

canales de comunicación seguros y confiables, manejar recursos, informar sobre las solicitudes, gestionar respuestas de un modo coordinado y propiciar información geográfica para contribuir a tomar mejores decisiones.

Por lo general componentes de los sistemas de gestión de emergencias se pueden dividir en:

- subsistemas para el registro de la información referente a la emergencia, que permite registrar los datos de la atención, tratamiento y seguimiento de la emergencia desde el inicio hasta el fin para garantizar la trazabilidad de la misma. Generalmente forman parte de este grupo los subsistemas: atención al cliente, despacho, supervisión.
- subsistema de información geográfica (GIS por sus siglas en inglés), que permite visualizar en tiempo real desde el centro de control la localización de los recursos, emergencias, sitios de interés y realizar el seguimiento de los servicios efectuados, sobre cartografía digital. Existen sistemas donde el mapa está muy vinculado a los procesos de negocio de despacho y permiten además de lo anterior, asignar recursos a emergencias (2MARES y SATDATA, 2011) estando fusionadas las funcionalidades de ambos. Otros sistemas vistos como el Centro Integral de Atención y Coordinación de Seguridad Ciudadana en Bolívar (1-7-1, 2010), el Sistema Integrado de Atención a Emergencias 171 de Aragua (ARAGUA, 2010), el Servicio Autónomo de Emergencias Lara 171 (LARA, 2006), entre otros, utilizan el mapa como apoyo para la toma de decisiones más precisas, que presentan interfaces para integrarse con el resto de los subsistemas del sistema y/o también pueden funcionar de forma independiente.
- subsistemas de comunicación, que permiten la interacción con tecnologías de radio, telefonía, etc. Realizan funciones como distribución automática de llamadas (ACD por sus siglas en inglés), monitorización y escucha de conversaciones ajenas, transferencias, conferencias de recursos, cambio de canal, entre otros, que posibilitan una atención a las emergencias más eficiente. Se puede utilizar como apoyo al subsistema, una cola unificada de interacciones que se encargue de la gestión de interacciones con los actores de la emergencia mediante colas en un solo contenedor común (fax, e-mail, sms, mensajería de voz, y ACD). (2MARES y SATDATA, 2011) (2MARES, 2011)

- subsistema de localización automática de vehículos (AVL por sus siglas en inglés) que permite representar sobre el sistema de información geográfica la posición de los vehículos dotados de receptor mediante sistema GPS. (VALENCIANA, 2012)
- subsistema de grabación de comunicaciones de radio, telefonía y captura de pantallas que permite el cifrado de conversaciones, además de la grabación selectiva, total, bajo demanda, etc. (2MARES y SATDATA, 2011) (2MARES, 2011)
- subsistemas de soporte, que contribuyen al adecuado funcionamiento del sistema. Ejemplos: estadísticas para medir los resultados del centro y su contribución a la seguridad ciudadana; administración y configuración del sistema según las necesidades de los clientes. También se pueden mencionar los subsistemas que permiten gestionar información útil a todos los trabajadores del centro, que generalmente se hace a través de una intranet.

Las características comunes más importantes de los sistemas de gestión de emergencias son: (FEDETEC y TELEMÁTICOS, 2008)

- Alta disponibilidad – tolerancia a fallos: La atención a las emergencias debe estar disponible en todo momento, 24 horas al día y 365 días al año, incluso a pesar de que puedan fallar algunos de sus componentes. Para ello es necesario disponer de redundancia en los componentes críticos, y tener la inteligencia del sistema distribuida por la red.
- Comunicaciones transparentes: las comunicaciones son el elemento básico en los momentos de crisis para la resolución rápida de las emergencias, por ello el medio de transporte de la voz y los datos, debe ser transparente para el agente o el operador que está resolviendo una emergencia. En cada momento se usará automáticamente el medio que esté disponible, sin intervención de las personas implicadas.

  Los SGE integran varios medios de comunicación como: telefonía analógica, telefonía RDSI (Red Digital de Servicios Integrados) básica, GSM (Sistema Global para las Comunicaciones Móviles), Radio *Trunking* analógico, Radio *Trunking* digital, TETRA (*Terrestrial Trunked Radi*), Interfonía, Megafonía, entre otros.
- Seguridad: dadas las características de modularidad y de sistema distribuido, y debido a la importancia crítica del sistema, es imprescindible dotarlo de un alto

nivel de seguridad. Tanto los datos como las comunicaciones de audio deben tener la máxima seguridad, para dar la mayor confianza de confidencialidad a los usuarios del sistema.

- Interoperabilidad e integración: la sinergia lograda con la integración de todos los subsistemas de gestión aumenta la efectividad del sistema y facilita la toma de decisiones del operador y el despachador. Si en la respuesta a la emergencia han de involucrarse distintos organismos o emplearse recursos de diferentes procedencias, es imprescindible disponer de una comunicación e intercambio de información entre distintos sistemas.
- Escalabilidad y modularidad: un SGE debe ser modular para permitir la adaptación a las distintas necesidades de los diferentes servicios de emergencias. De esta manera se permite a los distintos organismos públicos o privados adaptar el sistema a las necesidades tecnológicas y económicas particulares de cada uno.

  Un SGE de estas características debe permitir crecer cuando las necesidades se incrementan, sin necesidad de sustituir elementos o desechar subsistemas anteriores.
- Consistencia de los datos: debe mantener un estado coherente de los datos en todo el sistema. Es frecuente en sistemas de gestión de emergencias compartir recursos, porque varios subsistemas visualizan la misma información y en algunas ocasiones más de un subsistema pueden modificarla. La comunicación entre los subsistemas para lograr la actualización de los datos se debe realizar en el menor tiempo posible luego de ocurrido un cambio.

1.6 Sincronización de datos

La sincronización de datos hace posible que en los sistemas de información los datos que se administran de forma independiente por múltiples aplicaciones obtengan la información actualizada. El objetivo principal de la sincronización de datos es garantizar la integridad de los datos de todas las aplicaciones que participan en un proceso de negocio. (APTE, 2002)

Según la librería MSDN de Microsoft la sincronización de los datos se refiere al proceso de propagación de los cambios en los datos y el esquema entre el publicador y los suscriptores después de haber aplicado la instantánea inicial en el suscriptor. (MICROSOFT, 2012c)

La segunda definición de sincronización de datos será la utilizada en la presente investigación.

La sincronización de datos se puede realizar de diferentes maneras:

- A petición del usuario decidiendo éste el momento de realizarla.
- Al cabo de un tiempo definido en el sistema donde él mismo accedería a recursos compartidos en el lugar de almacenamiento de la información para actualizarse.
- Una vez que ocurran los cambios sobre un elemento notificar el cambio a todas las aplicaciones implicadas.

En los sistemas de gestión de emergencias si se decide emplear el primer modo de sincronización se perdería tiempo en presionar una opción para actualizar los datos cada vez que se fuera a atender una emergencia. Para el caso del seguimiento y control de las unidades móviles y del trabajo de los supervisores habría que estar constantemente presionando dicha opción para tomar decisiones acertadas. Este no es el modo adecuado para utilizar en un sistema de gestión de emergencias donde la información compartida cambia con frecuencia y es necesario mantener actualizados a los subsistemas que la comparten.

El segundo modo de sincronización evita perder el tiempo de presionar una opción y permite la actualización de la información pasado un tiempo que debe definirse en el sistema. Aparentemente esta solución resuelve el problema de mantener actualizada la información, pero tiene el inconveniente de los accesos innecesarios a la BD para los casos en que pasado el tiempo de actualización no se hayan realizado cambios sobre ningún elemento. Otro inconveniente sería para el caso en que se realicen cambios sobre un elemento y todavía no le corresponda al subsistema que se está sincronizando actualizar la información. Este último problema se puede minimizar poniendo un tiempo pequeño de actualización, pero entonces se incrementan los accesos a la BD aumentando la probabilidad que sean accesos innecesarios y que pueda colapsar la misma.

El último modo de sincronización evita los problemas de los anteriores (pérdida de tiempo por acceso a un botón y accesos innecesarios a la BD), pues propone que cuando ocurra un cambio se notifique a todos los subsistemas implicados y luego es responsabilidad de cada subsistema realizar las operaciones propias de su lógica de negocio asociada al aviso recibido. Este modo de sincronización en la presente investigación será conocido como sincronización de datos en tiempo real. Entiéndase

por tiempo real un tiempo menor de 5 segundos para actualizar los subsistemas que comparten información luego de ocurrido un cambio en el sistema. Este tiempo fue definido tomando como referencia las entrevistas realizadas en tres centros de gestión de emergencias en Venezuela, donde la actualización encuestando a un recurso compartido, como lo propone el segundo modo de sincronización, oscilaba entre 5 y 10 segundos.

1.6.1 Tiempo real y sincronización de datos en el sistema de gestión de emergencias

Los sistemas de tiempo real son sistemas computacionales en los que la computadora está fuertemente ligada a un entorno que cambia con el tiempo. Por esta causa, en los sistemas de tiempo real el funcionamiento correcto no sólo depende de los resultados del cálculo, sino también del instante de tiempo en el que estos cálculos son finalizados (STANKOVIC y RAMAMRITHAM, 1988). Este tipo de sistema es muy utilizado en sistemas controladores de robots, en la supervisión de procesos industriales, en la transmisión y el procesado de imagen para tener una visión artificial en tiempo real, en sistemas de control y navegación de vehículos como automóviles, aviones, o naves espaciales, entre otros.

La definición de un Sistema de Tiempo Real (STR) dada por Rajib Mall consiste en:

“Un sistema se denomina un sistema de tiempo real, cuando se necesita de expresión cuantitativa del tiempo para describir el comportamiento del sistema.” (MALL, 2007)

En la mayoría de los sistemas de tiempo real las actividades que el sistema debe ejecutar se conocen por adelantado, ya que existe un determinado código activo en la memoria de la computadora que está preparado para ejecutar. La ejecución de una actividad es activada por un evento o por el tiempo y la actividad propiamente dicha se conoce como la respuesta a ese evento. Deben ser tolerantes a fallas y la comunicación en los sistemas distribuidos de tiempo real debe ser de alto desempeño.

Los sistemas de tiempo real pueden ser clasificados en tiempo real duro y tiempo real suave (KOPETZ, 2011). El tiempo real duro significa que los eventos que llegan al sistema pueden tener impuestos plazos estrictos, y se considera que falla el sistema si no cumple con alguno de sus plazos. En el sistema de tiempo real suave se puede incumplir alguno de sus plazos de vez en cuando, produciendo así una disminución de las prestaciones o de la calidad del sistema, pero no un fallo total.

Los sistemas de gestión de emergencias son sistemas distribuidos de tiempo real que necesitarían cumplir con ambas clasificaciones. Cuando se trata del envío de las

emergencias de un subsistema de atención al cliente al subsistema de despacho para que puedan ser procesadas, lidiarían con el tiempo real duro, pues la no llegada de la misma en un tiempo límite podría ocasionar la pérdida de vidas humanas. Pero cuando se trata de mantener la información actualizada en tiempo real para varios subsistemas que la comparten, en casos como visualizar la situación operativa de la ciudad, conocer el estado, posición y velocidad de los recursos móviles en la vía, se corresponde con el tiempo real suave, pues son procesos que afectan la usabilidad del sistema donde su no cumplimiento con el tiempo establecido no constituye una falla total del sistema, la información persistente y actualizada debe estar almacenada en la base de datos. Puede ocurrir que un supervisor tenga una información desactualizada de los procesos que están ocurriendo sin mayores consecuencias aunque dicha situación no sea deseable. También puede suceder que debido a la información incorrecta poseída, se tomen decisiones incorrectas que puedan traer afectaciones para los ciudadanos, siendo uno de los motivos por los que es importante mantener la información actualizada en tiempo real en un sistema de gestión de emergencias.

En los sistemas de gestión de emergencias el proceso de sincronización consiste en notificar la ocurrencia de cambios en los datos, a todos los subsistemas que los compartan y estén suscritos en el publicador al elemento modificado en un tiempo menor que cinco segundos. No tiene sentido avisarle a todos los nodos de la red sobre un cambio, sino que solamente se notificaría a los nodos que soliciten actualización de un elemento que estén sincronizando. También ocurre en este tipo de sistema que el suscriptor solo necesita informarse de los cambios ocurridos en un atributo de un elemento, por tanto no siempre que ocurra un cambio en el elemento se debe enviar una notificación, sino que también hay que tener en cuenta el atributo que se está sincronizando en el nodo. Un ejemplo lo constituyen las unidades en servicio, que pueden ser sincronizadas por velocidad, posición, estado, entre otros. Existen subsistemas como el de despacho que solo necesitan conocer el estado de las unidades en servicio, mientras que el subsistema de mapa que utilizan los despachadores debe ser actualizado de los cambios en los tres atributos para mostrar el lugar, velocidad y estado del recurso en el menor tiempo posible y ayudar al despachador a tomar mejores decisiones ante una emergencia.

1.7 Fundamentación del objetivo

Mantener sincronizados los datos que se muestran en los subsistemas es uno de los problemas que se presentan al diseñar un sistema de gestión de emergencias, sobre todo conociendo que la información debe ser actualizada en tiempo real. Se debe tener en cuenta que el número de nodos del sistema puede variar en cualquier momento, y los que se incorporan pueden visualizar y modificar los mismos recursos que otros nodos existentes. Además, en cada sistema de gestión de emergencias los recursos que se comparten no son los mismos, ni se manejan los mismos datos de cada recurso. Por otro lado existen sistemas donde las emergencias tienen un estado global que depende de los estados de las atenciones de cada una de las agencias de despacho, permitiendo conocer de manera general la situación del incidente, que puede ser usada para informar al cliente.

Las posibles soluciones a este problema no se encuentran en la bibliografía consultada de los sistemas de gestión de emergencias (LARA, 2006) (FEDETEC y TELEMÁTICOS, 2008) (2MARES y SATDATA, 2011) (VALENCIANA, 2012) (S.A, 2012) (1-7-1, 2010). Debido a que son sistemas que tributan a la seguridad ciudadana de un Estado, donde un error puede traer como consecuencia la pérdida de vidas humanas, y tratándose de un problema tan específico en su diseño, es difícil que sea publicado.

En las entrevistas realizadas en tres centros de Venezuela, se pudo conocer que los sistemas no realizan la actualización de la información en tiempo real cuando la misma es modificada, sino que los subsistemas acceden a la BD cada cierto tiempo predefinido en el sistema, que no puede ser modificado si no se conocen las herramientas utilizadas.

Por tal motivo se impone realizar un estudio de los mecanismos que existen para lograr mantener coherente la información compartida entre varios subsistemas de un sistema distribuido donde es necesario la actualización de la información en un intervalo de tiempo menor de cinco segundos una vez ocurrido un cambio.

1.8 Tecnologías de sincronización

Para realizar la investigación de tecnologías que implementen mecanismos de sincronización en tiempo real, es necesario tener en cuenta que la solución debe ser compatible con la plataforma .NET, debido a que el sistema de gestión de emergencias al que debe integrarse está desarrollado sobre esta plataforma.

1.8.1 Colas de mensajería de Microsoft

La cola de mensajería de Microsoft, por sus siglas en inglés *Microsoft Message Queuing* (MSMQ), es una infraestructura de mensajería y una herramienta de desarrollo para crear aplicaciones distribuidas de mensajería para el sistema operativo Microsoft Windows (MICROSOFT, 2012a). Ha sido comúnmente usada en sistemas desarrollados con Visual Studio.NET.

MSMQ permite a las aplicaciones que se ejecutan en tiempos distintos comunicarse a través de redes o sistemas que pueden estar temporalmente fuera de línea. Las aplicaciones se comunican enviando o leyendo mensajes de colas. Además, proporciona un medio de entrega garantizada de mensajes, encaminamiento eficaz, seguridad y mensajería basada en criterios de prioridad. Se puede utilizar para implantar soluciones para escenarios de mensajería asíncrona o síncrona.

MSMQ provee autenticación, cifrado de mensajes, y el uso de transacciones externas (DICKMAN y HOUSTON, 1998), por ejemplo administradas por SQL Server, lo que facilita la implementación de las aplicaciones que utilizan la cola de mensajes como forma de realizar sus tareas. En el caso de la firma digital, es posible definir una cola de mensajes que únicamente acepte mensajes firmados y MSMQ se encarga de verificar la integridad del mensaje, es decir, que el mismo no haya sido modificado. Corresponde a la aplicación que recibe el mensaje determinar si confía en la persona que lo envió o no.

1.8.2 Lightstreamer

Lightstreamer es una tecnología para la entrega en tiempo real de datos a través de la red. Ha sido elegido por muchas empresas, así como las nuevas compañías, para los sistemas críticos de negocio y de misión crítica, donde los requisitos básicos son: alta escalabilidad, el consumo de ancho de banda bajo, listo para atravesar el firewall, adaptación de límite y fiabilidad total. (WESWIT, 2006a)

Permite el envío de datos en tiempo real para y desde cualquier navegador y aplicaciones de escritorio o móvil. Es compatible con HTML, JavaScript, iOS, Android, WinPhone, BlackBerry, Flash, Flex, Silverlight, Java y NET. (WESWIT, 2006a)

Está basado en la tecnología *Push* que establece que a la llegada de un mensaje se notifiquen a todos los interesados. Realiza adaptación de mensajes cuando la red está saturada teniendo control sobre la composición de paquetes TCP que se envían por la red. La conexión es punto a punto, se realiza a través del protocolo HTTP (por sus

siglas en inglés, *Hypertext Transfer Protocol*) y se abre una única conexión permanente con el servidor para reducir los tiempos de respuesta (WESWIT, 2006b). El servidor es escalable, permite hacer *clustering* basado en el balanceo de carga.

1.8.3 Conclusiones parciales

MSMQ no puede ser usada en el sistema propuesto a pesar de las funcionalidades que ofrece, debido a que la sincronización en este marco tiene una lógica de negocio muy específica, determinada por diferentes reglas o restricciones sobre la información que se maneja. Tal es el caso de las solicitudes de emergencias que poseen un estado en un momento determinado, que puede ser Pendiente, En proceso y Culminado. Este estado de las solicitudes depende del estado de todos los despachos de solicitud creados para atenderla, y se realizan cálculos propios del negocio para mantenerlo actualizado.

Lightstreamer es una poderosa herramienta que se integra con .NET, pero no permite adicionarle la lógica de sincronización necesaria del sistema de gestión de emergencias que se propone. Además, utiliza el protocolo HTTP para comunicarse, que se utiliza más en aplicaciones que no tienen requerimientos de tiempo donde otros protocolos tienen mejor rendimiento (STACKOVERFLOW, 2011).

La propuesta es desarrollar un subsistema de sincronización sobre la plataforma .NET porque no se encuentra en la investigación realizada un modelo que se ajuste a las necesidades del negocio del SIGESC.

1.9 Servicios de Microsoft Windows

Los servicios de *Windows* son pequeños programas o aplicaciones que se ejecutan de forma automática al iniciar *Windows* y quedan residentes en memoria corriendo en un segundo plano, consiguiendo que el máximo posible de situaciones sean resueltas con la mínima intervención del usuario. Ofrece soporte para otras aplicaciones y controlan una tarea muy específica en el sistema.

Los servicios de *Microsoft Windows*, permiten crear aplicaciones ejecutables de larga duración, que se ejecutan en sus propias sesiones de Windows. Estos servicios pueden iniciarse automáticamente cuando el equipo arranca, se pueden pausar y reiniciar, y no muestran ninguna interfaz de usuario. Estas características hacen que los servicios resulten perfectos para ejecutarse en un servidor o donde se necesite una funcionalidad de ejecución larga que no interfiera con los demás usuarios que trabajen en el mismo equipo. (MICROSOFT, 2007)

Existen tres opciones posibles de inicio:

- Manual: Se puede iniciar y detener manualmente cuando se desee u otro servicio puede hacerlo automáticamente. En un principio estaría detenido.
- Automático: Se inician junto con el sistema operativo.
- Deshabilitado: No se puede iniciar manualmente ni otro servicio puede hacerlo.

Conclusiones del capítulo

Los subsistemas de los sistemas de gestión de emergencias necesitan mantener sus datos actualizados y coherentes. El hecho de que los subsistemas de un sistema de gestión de emergencias tengan información desactualizada puede conllevar a errores de decisiones que impliquen la pérdida de ciudadanos. Por tanto cuando exista alguna modificación sobre un dato que es compartido entre varios subsistemas, se debe notificar en tiempo real a todos los subsistemas que estén utilizando la información.

Se realizó un estudio del estado del arte de herramientas informáticas que se han usado para implementar mecanismos de sincronización de información en sistemas distribuidos de tiempo real, llegando a la conclusión que no pueden ser usados en el sistema a desarrollar. Por tal motivo es necesaria la implementación de un subsistema de sincronización para el SIGESC.

Capítulo 2. Propuesta del subsistema de sincronización de datos

En el capítulo se describe el Sistema de gestión de emergencias de seguridad ciudadana 171 con sus escenarios de sincronización característicos. Según la situación problemática planteada y en base al estudio realizado en el capítulo anterior, se define el mecanismo de sincronización propuesto. Luego se describe cada componente de la solución, así como la relación entre ellos.

Además, se especifican las funcionalidades que debe cumplir el subsistema que se propone, lo que permite hacer una concepción general del mismo e identificar mediante diagramas de caso de uso, las relaciones de los actores que interactúan con el subsistema y las acciones que se realizan. Además se expone el diseño de la solución, el diagrama de implementación y la validación de la misma.

2.1 Sistema de gestión de emergencias de seguridad ciudadana 171

El sistema de gestión de emergencias de seguridad ciudadana 171 (SIGESC) (MEDINA, 2003) es una solución integral que contribuye a gestionar de forma eficiente los servicios de atención a emergencias policiales, salud pública, bomberos, protección civil, tránsito, entre otras. Está compuesto por varios subsistemas instalados en computadoras diferentes, que necesitan intercambiar información constantemente y se comunican a través de una red de área local (LAN). Presenta una BD centralizada, donde se almacenan los datos referentes a las emergencias que se registran en el sistema, así como el historial de las acciones realizadas por cada usuario y recurso que interviene en la atención de la misma. Además permite la integración con diferentes tecnologías de comunicación como, seguimiento vehicular con GPS, GIS, etc. (Figura 3, Figura 4)

Todos los subsistemas del sistema están implementados sobre la plataforma .NET y lenguaje de programación C# y deben cumplir los mismos requerimientos de software y hardware.

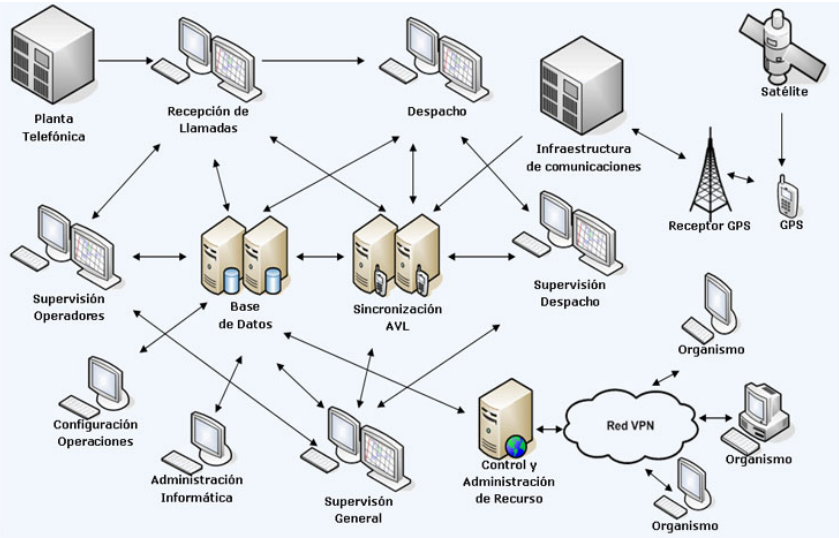

Figura 3. Diagrama de despliegue del SIGESC

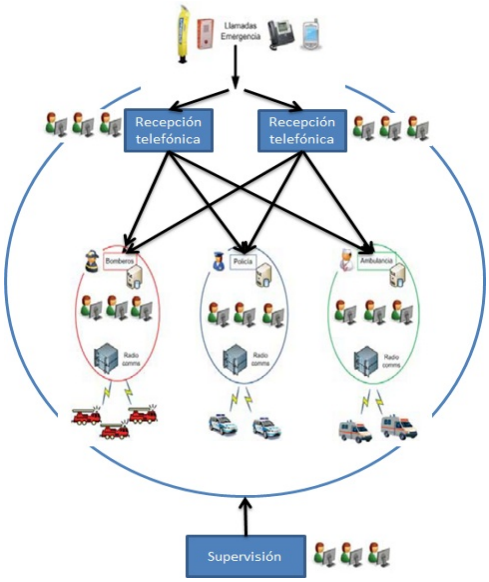

Figura 4. Funcionamiento del SIGESC

2.2 Escenarios de sincronización de datos en el SIGESC

En el SIGESC los subsistemas comparten información por disímiles motivos. Uno de ellos consiste en que la solicitud de emergencia atendida por las diferentes áreas es la misma y todos los subsistemas trabajan independientemente en función de solucionarla. Además existen subsistemas de supervisión y control al trabajo de otros subsistemas, y por tanto requieren conocer el estado de los elementos que administra y gestiona el área supervisada en tiempo real.

En el subsistema de Atención al Cliente, entre otras funcionalidades, se registran los datos de las solicitudes de emergencias para enviarla al subsistema de Despacho de las agencias que deben darle atención. En cada agencia se crea un despacho a partir de los datos de la solicitud, que en dependencia del avance de la atención brindada tiene un estado, que no tiene que ser el mismo para todos los despachos. A partir de los estados de cada despacho se determina el estado de la solicitud. La solicitud puede ser visualizada por: el subsistema de Atención al Cliente desplegado en las distintas computadoras para darle información al cliente de la emergencia en caso que lo solicite; el subsistema de Supervisión para conocer la situación operativa de la ciudad pudiendo determinar por el tipo de incidente si debe llevar una atención diferenciada y tomar otras decisiones; el subsistema de Mapa, que puede encontrarse en dos variantes, el que está integrado a los subsistemas mencionados anteriormente o el que está independiente donde se pueden visualizar todos los elementos que existen en el sistema en tiempo real. Por tanto los datos de una solicitud han de compartirse entre varios subsistemas del SIGESC y constantemente se deben sincronizar. El estado de una solicitud lo modifican los despachos que tiene asociados y es consultado por el resto de los subsistemas.

Otro dato compartido entre varios subsistemas es el relacionado con los recursos y unidades móviles habilitadas para atender solicitudes de emergencias. Las unidades que poseen tecnología GPS son monitoreadas en tiempo real en el subsistema de Mapa que utiliza el centro, ya sea integrado a los subsistemas Despacho, Supervisión o independiente. En un sistema de gestión de emergencias se necesita conocer la posición de las unidades móviles, de modo que se pueda asignar a una emergencia la unidad que esté más cercana y así ganar prontitud. También, se debe monitorear la velocidad de las unidades, por ejemplo, para conocer si la unidad llegará a tiempo teniendo una velocidad determinada. En el SIGESC existe un servidor de AVL, que recibe en tiempo real las señales de los GPS de las unidades en la vía indicando su localización y velocidad, que deben ser enviadas a todos los subsistemas que la visualicen. Otro atributo que tienen las unidades es su estado, que indica si la unidad está disponible, en camino para la atención a una emergencia, en el sitio de la emergencia o en mantenimiento, que permiten diferenciar si puede ser asignada a una emergencia en un momento determinado. El estado de las unidades puede ser visualizado y modificado en los subsistemas de despacho y supervisión, y es visualizado, además en el subsistema mapa en conjunto con los otros atributos.

2.3 Aspectos arquitectónicos en el mecanismo de sincronización

Los sistemas de gestión de emergencias son sistemas distribuidos de tiempo real, donde es preciso actualizar la información en los subsistemas que la comparten en el menor tiempo posible una vez ocurridos los cambios en los datos. Para elaborar el mecanismo de sincronización entre los subsistemas del sistema se deben tener en cuenta las características, modelos y patrones arquitectónicos, así como el paradigma de comunicación entre los subsistemas.

2.3.1 Modelo arquitectónico seleccionado

Para realizar la selección del modelo arquitectónico, una solución puede ser el enfoque descentralizado con el modelo *peer-to-peer*. En este modelo no es necesario tener un administrador del sistema a tiempo completo, lo que aumenta la fiabilidad del sistema al no tener una dependencia central, donde el fallo de un nodo no afecta al funcionamiento de los demás.

El subsistema que modifique la información tiene la responsabilidad de avisar al resto de los subsistemas que la estén utilizando que la misma ha sido cambiada. Esto implica que un subsistema que vaya a utilizar una información que pueda ser compartida, tiene que tener conocimiento de todos los subsistemas que la están utilizando en ese momento, por lo que constantemente tiene que estar actualizando sus listas de destinatarios por información. Mantener actualizada la lista de destinatarios, sería una lógica adicional que tendría cada subsistema que comparta la información, que sería compleja de implementar y además necesaria en cada nuevo subsistema que vaya a utilizar la información. Para el descubrimiento de otros nodos que están compartiendo información, con el objetivo de minimizar la complejidad de desarrollo, se puede utilizar un servidor de nombres central que mantenga una lista de todas las aplicaciones que compartan la información al que puedan acceder todos los nodos. Luego cada uno tendría que notificar la actualización de la información a cada integrante de la lista. Además, existen otros medios, como los algoritmos de descubrimiento *multicast* IP (APONTE, 2006) (TANENBAUM, 2003), que se pudieran emplear.

También es preciso analizar que en el modelo *peer-to-peer* aumenta la carga en el sistema de comunicaciones que se emplee. El número de canales de comunicaciones, abiertos o por abrir, sin importar de qué tipo son, sería considerablemente alta, aproximadamente igual a $\sum_{1}^{i=n-1} i$, donde *n* es la cantidad de subsistemas que están

compartiendo la información, debido a que el número de mensajes de comunicación es el número de aristas de un grafo completo. Este cálculo es considerando un modelo *peer-to-peer* sin la presencia de un servidor de nombres, donde sería aún mayor la cantidad de canales abiertos.

A medida que la red crece se vuelve más difícil de coordinar y operar. Además, el número de computadoras que se pueden conectar a otra computadora es limitado, por lo que cada uno tendría que encargarse de mantener un canal de escucha siempre abierto para que en caso de incorporación de otra computadora, esta pueda conectarse y mantenerse actualizada.

Otra alternativa sería el enfoque centralizado con el modelo cliente-servidor, donde se encapsularía la lógica de sincronización en un servidor. Este servidor tendría la responsabilidad de notificar cualquier cambio, gestionar las listas de destinatarios para cada información específica, así como los canales de comunicación para cada caso. Este modelo simplifica el desarrollo que se tendría que hacer en cada subsistema (cliente), pues sólo tendrían que notificar al servidor que están utilizando una determinada información y los cambios de estado de la misma y éste se encargaría de notificar a todos los interesados.

Además, se disminuiría la carga en el sistema de comunicaciones, porque el número de canales sería aproximadamente igual a *n*, donde *n* es la cantidad de subsistemas que están compartiendo la información.

Uno de los inconvenientes de esta alternativa es precisamente que toda la lógica de sincronización está centralizada en un “servidor de sincronización”, donde la disponibilidad y el rendimiento serían dos factores vitales. Para lograr un mayor rendimiento en el servidor y tratando de minimizar este inconveniente, se debe instalar el servidor en una computadora con características de hardware avanzadas teniendo en cuenta la cantidad de usuarios concurrentes que pueden acceder. En cuanto a la disponibilidad, se puede implementar un mecanismo de redundancia donde, se pueden preparar otros puestos de trabajo para que asuman temporalmente dicha responsabilidad en caso que el mismo no responda.

Cada modelo tiene sus ventajas y sus desventajas. En este trabajo se propone que la segunda alternativa sea la que se utilice en el SIGESC como modelo arquitectónico del mecanismo de sincronización (Figura 5). La selección se realiza teniendo en cuenta que:

- a pesar de que el número de subsistemas que conforman el sistema de gestión de emergencias puede variar en un momento dado, se beneficia la transparencia y extensibilidad del sistema como un todo, puesto que los subsistemas no tienen que conocer quiénes están utilizando o compartiendo una información determinada y se podrían adicionar nuevos subsistemas al sistema de sincronización sin hacer ningún cambio o ninguna lógica adicional.
- la cohesión de cada subsistema se consolida encargándose solamente de su lógica de negocio.
- el desarrollo se hace menos complejo, al no tener que mantener las listas de los subsistemas que comparten información ni tener que mantener un canal de comunicación siempre abierto escuchando peticiones.

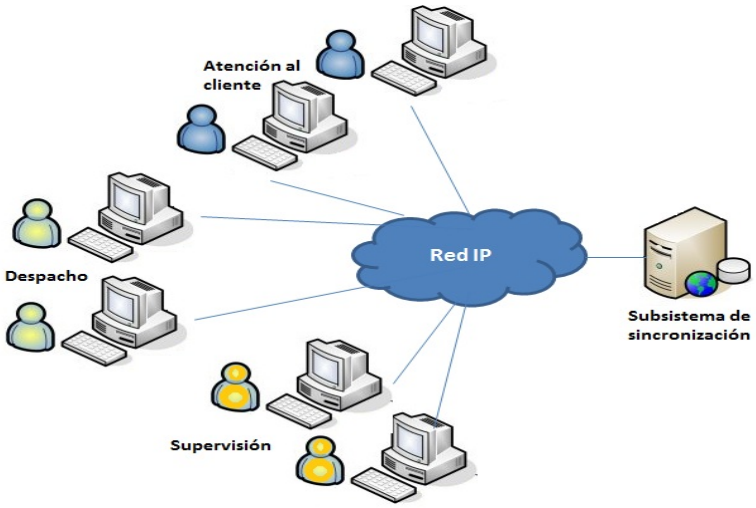

Figura 5. Modelo cliente-servidor

2.3.2 Patrones arquitectónicos seleccionados

En el sistema de gestión de emergencias en cada nodo se aplica el patrón de arquitectura en capas (*layering*). En los nodos clientes, en la capa Aplicación se encuentran los subsistemas del SIGESC que necesiten sincronizarse (Atención al cliente, Despacho, Supervisión, entre otras), en la capa *Middleware* se encuentra el sistema de comunicación del *framework* DMAX (AGRAMONTE *et al.*, 2008), en la capa Sistema Operativo se requiere de *Windows XP* y como requerimientos de *hardware* cada nodo debe tener como mínimo un procesador Pentium IV 3.00 GHz, 512 MB de memoria RAM y 122MB de espacio libre en disco.

En el nodo servidor, en la capa Aplicación se encuentra el subsistema de sincronización; en la capa de *Middleware* se encuentra, de la misma manera que en el cliente, el sistema de comunicación del *framework* DMAX; en la capa Sistema

Operativo se requiere Windows Server 2003 y como requerimientos de *hardware* se debe tener como mínimo 2 procesadores a 3.0 Ghz, 2 GB de memoria RAM y 2 discos de 72 GB en RAID1. (Figura 6)

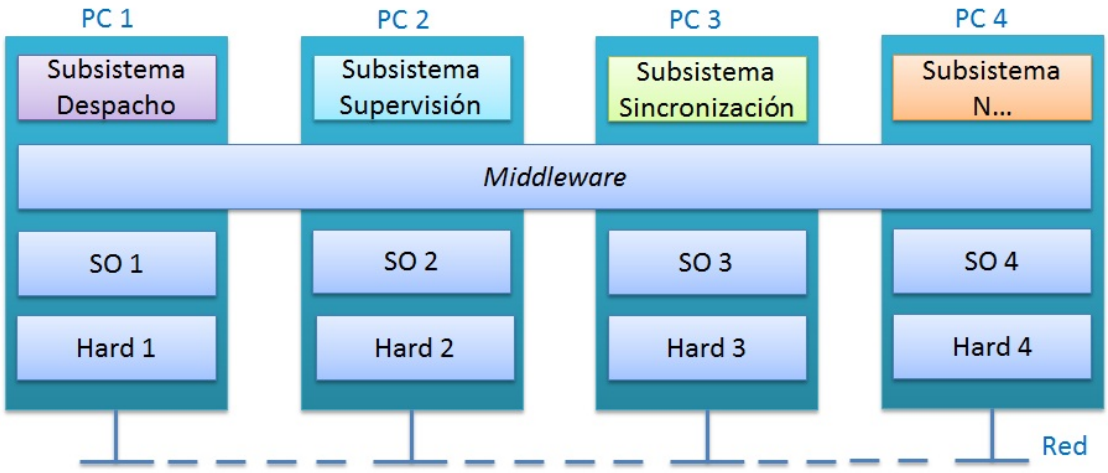

Figura 6. Patrón de arquitectura en capas del SIGESC

En la capa Aplicación se aplica el patrón arquitectónico escalonado (tiered), porque la descomposición funcional del SIGESC consiste en la lógica de presentación, la lógica de aplicación y la lógica de datos. Las dos primeras se encuentran ubicadas en la misma computadora; pero la lógica de datos, se encuentra ubicada en otra computadora donde está instalado el servidor de base de datos Oracle 10g (HITCHCOCK, 2005). Teniendo en cuenta que el modelo arquitectónico seleccionado fue el cliente-servidor, se tendría una arquitectura de dos niveles (*two-tiered*) (Figura 7).

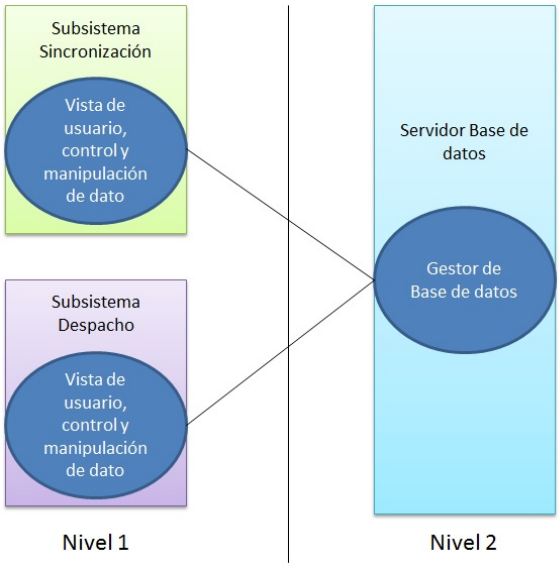

Figura 7. Patrón de arquitectura escalonado del SIGESC

En general, en el sistema de gestión de emergencias para el mecanismo de sincronización la estructura propuesta está compuesta por el patrón de arquitectura en capas (*layering*) combinado con el escalonado (*tiered*).

2.3.3 Paradigmas de comunicación seleccionados

En el SIGESC se emplea el paradigma *Socket* para la comunicación entre procesos, que permite la comunicación punto a punto entre los subsistemas de los nodos de la red, ya sea en máquinas cliente o servidor. Este mecanismo está implementado en el *framework* DMAX que utiliza el sistema y se especifica en el epígrafe 2.6.2.

Como propuesta de la tesis, se seleccionó el paradigma de comunicación indirecta publica-suscribe (*Publish-Subscribe*) para usar en el mecanismo de sincronización entre los subsistemas clientes (Atención al cliente, Despacho, entre otros) y el subsistema servidor (sincronización). Esta selección se realiza teniendo en cuenta que, los subsistemas en los nodos clientes no están interesados en todos los tipos de mensajes que se envían por la red, sino en diferentes combinaciones de ellos. En este caso, haciendo una analogía con los elementos que componen el paradigma de comunicación publica-suscribe, los subsistemas de los nodos clientes serían los publicadores de datos y se pueden suscribir a cambios en los mismos; mientras que el subsistema servidor de sincronización sería el llamado intermediario (*Broker*), que se encarga de recibir mensajes que indiquen el cambio en algún elemento y notificar a todos los subsistemas suscritos, como se muestra en la Figura 8.

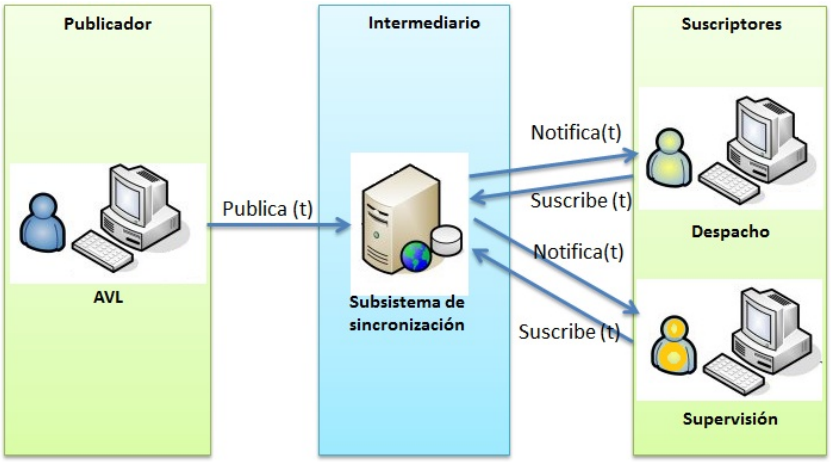

Figura 8. Publica-suscribe en el SIGESC

En este paradigma se beneficia la escalabilidad del sistema, porque el que envía la notificación no necesita saber la identidad de los que deben recibirla, solo debe conocer la identidad del intermediario, permitiendo aumentar el número de subsistemas que compartan información en la red, sin tener que notificar a los existentes.

Se incluyó para implementar el paradigma publica-suscribe los siguientes aspectos:

- ampliar la infraestructura de comunicación mediante la creación de temas o formas dinámicas de inspeccionar el contenido de los mensajes.
- habilitar subsistema de escucha para suscribirse a mensajes específicos.
- crear un mecanismo que envía mensajes a todos los usuarios interesados.
- la selección de una de las variantes del paradigma: basada en tópico, basada en contenido y basada en tipo. (COULOURIS *et al.*, 2012)

2.4 Arquitectura de sincronización de datos en el SIGESC

2.4.1 Organización arquitectónica

El mecanismo de sincronización está compuesto por un subsistema de sincronización, el gestor de base de datos del SIGESC y los subsistemas del SIGESC que necesiten sincronizarse. Todos en conjunto colaboran entre sí para garantizar que las diferentes reglas sobre la información se cumplan y minimizar así la probabilidad de entrar en un estado incoherente.

La estructura fundamental del mecanismo de sincronización de datos en el SIGESC (Figura 9) está basada en un enfoque centralizado, siguiendo el modelo arquitectónico cliente-servidor, donde los subsistemas del SIGESC serían los clientes y el subsistema de sincronización sería el servidor y se utiliza el paradigma de comunicación publica-

suscribe entre ambos elementos. El gestor de base de datos constituye un elemento fundamental en el mecanismo propuesto.

La descripción de cada componente se realiza a continuación:

- Subsistema de sincronización: Encapsula y gestiona todos los servicios de sincronización en el SIGESC, así como define los diferentes protocolos para consumirlos.
- Gestor de base de datos del SIGESC: Es el elemento rector en el sistema de sincronización, ya que posee la información confiable y actualizada. Dentro del mecanismo de sincronización es una fuente de consulta para verificar la fiabilidad de la información que se maneja.
- Subsistemas del SIGESC que necesiten sincronizarse: Son aquellos subsistemas que por alguna razón utilizan los recursos compartidos ya sea para consultarlos o modificarlos y por tanto necesitan conocer el estado real en que se encuentra el mismo.

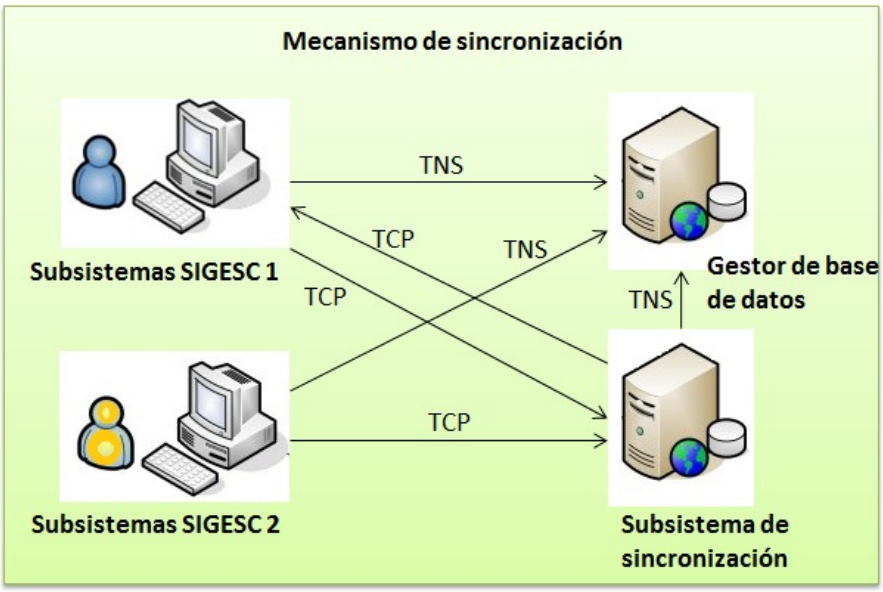

Figura 9. Estructura del mecanismo de sincronización

2.4.2 Interacción entre componentes

Los subsistemas del SIGESC que necesiten sincronizarse se comunican con el gestor de BD, específicamente Oracle 10g, a través del protocolo de comunicación TNS (Substrato de Red Transparente (por sus siglas en inglés, *Transparent Network Substrate*)) (ORACLE, 2005) cuando reciben una notificación de actualización de algún elemento al que está suscrito en el subsistema de sincronización para verificar la autenticidad de la información. Además, se comunican con el subsistema de

sincronización a través de un protocolo de comunicación para consumir sus servicios. Algunos de estos subsistemas se suscriben en las listas de los elementos que pueden ser compartidos y así pueden ser notificados por el subsistema de sincronización lo más pronto posible de los cambios realizados a dicho elemento.

El subsistema de sincronización establece una comunicación con el gestor de BD a través del protocolo TNS para cargar las configuraciones de comunicación y poder conocer el puerto que debe utilizar para recibir las peticiones de sincronización de los clientes.

2.5 Arquitectura del subsistema de sincronización

Considerando que el subsistema de sincronización debe prestar sus servicios de forma ininterrumpida, con baja probabilidad de fallos y sin necesidad de un operador, se propone que se divida en dos subsistemas (Figura 10):

- Subsistema servidor de sincronización: se encarga de prestar todos los servicios de sincronización. El tipo de aplicación es un servicio de Windows y por tanto es totalmente autónomo, diseñado para funcionar por largos períodos de tiempos, optimizando su rendimiento y minimizando algunos factores que ponen en riesgo su integridad. Además al encapsular solo la lógica necesaria para prestar los servicios de sincronización se facilita la gestión de errores, memoria y concurrencia, obviando aspectos como la representación de la información que se maneja.
- Subsistema de administración de sincronización: el servidor de sincronización al ser un servicio de Windows no posee interfaz gráfica por lo que se dificulta el control y la supervisión sobre el funcionamiento del mismo por parte de los administradores del sistema. De ahí que se necesite un subsistema de administración que se conecte al servidor para conocer el estado de éste y enviar determinadas órdenes que se necesiten, como de rectificación de errores, configuraciones de seguridad y comunicaciones.

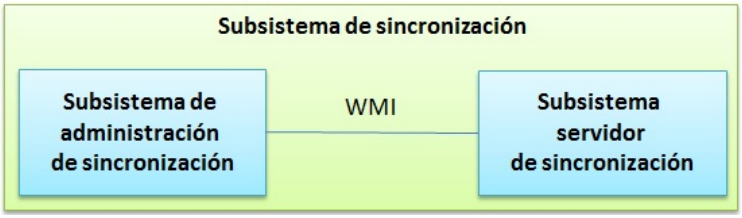

Figura 10. Estructura del subsistema de sincronización

2.6 Subsistema servidor de sincronización

Teniendo en cuenta que se seleccionó publica-suscribe como paradigma de comunicación en el mecanismo de sincronización, el subsistema servidor de sincronización debe desempeñar el rol de intermediario (*Broker*).

2.6.1 Modelos de suscripción seleccionados del paradigma publica-suscribe

Las expresiones del sistema publica-suscribe se determinan por el modelo de suscripción, con un número de esquemas definidos. Como se mencionó en el epígrafe 2.2 en el SIGESC existen subsistemas que requieren conocer todos los cambios sobre cualquiera de los atributos de un elemento (solicitud o unidad) a sincronizar. El otro caso son los subsistemas que solo necesitan ser notificados cuando cambien uno o varios atributos en un elemento.

Se selecciona el esquema Basado en Tipo (*Type-based*), creando objetos de tipo notificación. El tipo del objeto notificación sería el identificador que permite distinguir el elemento al que deben suscribirse los interesados que serán notificados cuando ocurra un cambio en el elemento. Esto trae como ventaja que se integra al lenguaje de programación orientado a objeto y además se puede comprobar la corrección del tipo de las suscripciones desde que se recibe el mensaje permitiendo eliminar algunos errores de suscripción que pudieran ocurrir.

Pero solamente utilizando el esquema basado en tipo no se resuelve el caso en que los subsistemas que necesiten ser notificados de los cambios ocurridos en los atributos de un elemento, por lo que es necesario combinarlo con el esquema Basado en Contenido (*Content-based*) para distinguir no solo el elemento, sino atributos dentro del elemento.

Para la implementación del esquema basado en tipo solamente, el tipo de notificación que se envía debe permitir identificar el elemento al que se desea inscribir o del que desea publicar información. Para la implementación de los dos esquemas juntos, la identificación del elemento y los atributos que se desean sincronizar están dentro del mensaje, donde el tipo de notificación ayudaría a distinguir fácilmente lo que se está solicitando e invocar a los métodos que deben darle tratamiento en el negocio.

Además, en el SIGESC ocurre que todos los publicadores no son suscriptores por lo que esta información también debe estar en el mensaje a enviar (ver epígrafe 2.6.4). La interacción con el intermediario se realiza a través de una serie de mensajes de

punto a punto, utilizando el mecanismo de comunicación del *framework* DMAX (ver epígrafe 2.6.2).

En el servidor de sincronización se necesita de una lista de elementos sincronizados, que pueden ser unidades, despachos y solicitudes, identificados inequívocamente en todo el SIGESC y además estos elementos pueden tener los atributos: estado, categoría, posición y velocidad. Por cada elemento de la lista, se tiene otra lista con las descripciones de los puntos de los subsistemas que soliciten sincronizar por ese elemento. El elemento a sincronizar con su lista de suscriptores constituye un paquete. Cada descripción del punto tiene el tipo de sincronización que pueden ser: estado, velocidad y localización en dependencia del elemento sincronizado, y el identificador del punto conformado por el IP y el puerto.

Cuando se recibe un mensaje, después que es clasificado, se comienza a procesar la lógica de negocio a la que está asociado. Si llega otro mensaje que tiene la misma clasificación y por tanto debe realizar el mismo proceso se pone en espera, evitando problemas de concurrencia.

Para lograr un mayor entendimiento de la utilización de los dos esquemas seleccionados, basado en contenido y basado en tipo, se presenta un escenario de sincronización donde un subsistema desea suscribirse para ser notificado de los cambios que ocurran sobre un elemento.

1. Un subsistema del SIGESC envía una notificación al servidor de sincronización de manera asincrónica.
2. El servidor de sincronización clasifica la notificación teniendo en cuenta el nombre de la notificación e invoca el proceso del negocio que debe ejecutarse asociado al tipo de notificación. Conociendo el tipo de notificación se conoce también si el subsistema que envía la notificación desea suscribirse a un elemento o solo publicar información.
3. El servidor de sincronización verifica que no se está ejecutando el proceso asociado al tipo de notificación. Si se está ejecutando, se pone a esperar hasta que el proceso que se está ejecutando termine, para evitar problemas de concurrencia.
4. El servidor de sincronización busca el elemento a sincronizar en la lista de los elementos sincronizados utilizando la información contenida en la notificación que permite identificar el elemento. Si el elemento no está en la lista se agrega.

5. El servidor de sincronización adiciona el subsistema a la lista de suscriptores del elemento que se desea sincronizar, culminando así el proceso de suscripción.

Cuando ocurre un cambio sobre un elemento, los subsistemas que comparten información deben notificar al servidor de sincronización para que este notifique a todos los suscritos en la lista del elemento. El proceso que se sigue en esos casos es el siguiente:

1. Se realizan los pasos del 1 al 3 del proceso descrito anteriormente.
2. El servidor de sincronización busca el elemento a sincronizar en la lista de los elementos sincronizados utilizando la información contenida en la notificación que permite identificar el elemento.
3. El servidor de sincronización verifica que el tipo de sincronización del elemento cambió con respecto al que él tenía almacenado y cambia el tipo de sincronización. El tipo de sincronización es un atributo del elemento, que puede ser: estado, velocidad o posición. Si el tipo de sincronización del elemento no cambió se culmina el proceso.
4. El servidor de sincronización busca en la lista de suscritos a ese elemento los subsistemas que desean que les avise por el cambio en el tipo de sincronización de ese elemento. Luego le envía una notificación a cada elemento encontrado indicando que debe actualizar la información.
5. Los subsistemas del SIGESC notificados tienen un filtro donde clasifican las notificaciones que le llegan, permitiendo identificar las notificaciones que van a tratar en el negocio de su aplicación.

2.6.2 Mecanismo de comunicación

El mecanismo de comunicación es el *Middleware* que permite la interacción entre los subsistemas del SIGESC que necesiten sincronizarse y el servidor de sincronización. Utilizará el subsistema de comunicaciones definido por el *framework* DMAX para el SIGESC, que está basado en el paradigma de comunicación *Socket* y el protocolo TCP/IP.

Subsistema de comunicaciones del framework DMAX

Los *Sockets* son altamente genéricos e implementan diferentes protocolos que abarcan los más disímiles escenarios, pero la implementación de los protocolos no es especializada y se dificulta su programación. TCP se encuentra entre los protocolos

que implementa Socket.Net (MICROSOFT, 2012b) y es el utilizado en el subsistema de comunicaciones del *framework* DMAX. Además, en un entorno donde la gestión de las conexiones de comunicación es vital como es el caso del Servidor de Sincronización se utilizan algunos de los servicios especializados altamente importantes, como:

- Gestión de estados de conexión.
- Gestión de errores.
- Mecanismos de serialización de datos.
- Seguridad.
- Filtrado de datos.
- Notificaciones de envío y recepción.
- Gestión de configuración.

Tales servicios, no los brinda *Socket.Net* por lo que fue necesaria la utilización de librerías de comunicaciones más especializadas en TCP/IP, que constituyen el subsistema de comunicaciones del *framework* DMAX. Dicho subsistema de comunicaciones está formado por tres capas (Figura 11):

- Socket.Net: son las librerías que provee la plataforma .Net, es el núcleo del subsistema de comunicaciones de DMAX.
- TCP/IP especializada cliente-servidor: provee un servidor y un cliente basados en TCP/IP que brindan diferentes servicios especializados como: modos de conexiones, modos de escucha, notificaciones de cambios de estados, gestión de errores, serialización de datos, modos de envío y recepción de datos, entre otros.
- TCP/IP especializada de alto nivel: está montada sobre la capa TCP/IP especializada cliente-servidor y provee un interfaz de alto nivel que facilita la gestión de las comunicaciones, abstrayendo al desarrollador de elementos de la arquitectura cliente/servidor e introduciendo nuevos conceptos como canales, filtros, solicitudes de conexión, entre otros.

Figura 11. Estructura del subsistema de comunicaciones de DMAX

Este subsistema de comunicaciones del DMAX se puede utilizar para la comunicación entre cualesquiera dos nodos en la red. En el caso de la comunicación entre el subsistema servidor de sincronización y los subsistemas del SIGESC que necesiten sincronizarse fue necesaria una lógica adicional, por lo que se definió el Protocolo de sincronización de datos del SIGESC (PSDS), quedando el mecanismo de comunicación de sincronización como se muestra en la Figura 12.

Figura 12. Estructura del subsistema de sincronización

2.6.3 Protocolo de sincronización

El PSDS es un protocolo que provee el mecanismo de comunicación entre el servidor de sincronización y cualquier otro subsistema que necesite sincronizar datos. Está montado sobre el protocolo TCP y entre sus funciones fundamentales se encuentran disminuir la carga de datos en el servidor y establecer las reglas necesarias para consumir los servicios de sincronización.

Esencialmente el servidor de sincronización está a la escucha de cualquier solicitud de conexión. Una vez que se establece la conexión, el protocolo TCP se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores. A partir de este momento es que se comienza a utilizar el PSDS.

Descripción del PSDS

Regla de conexión

Cuando una conexión es interrumpida por cualquier motivo se cancelan todos los servicios que se estaban prestando al subsistema asociado a dicha conexión. Por tanto es responsabilidad del subsistema que consume los servicios volver a sincronizar todos sus datos al restablecerse la conexión, así como los diferentes filtros que se necesiten. El servidor no conserva los estados de sincronización una vez que se termina una conexión para no sobrecargarse, garantizando así que no existan elementos sincronizados que no se les dé seguimiento.

Descripción de los servicios de sincronización

- Inicialización de sincronización de dato: indica al servidor que un elemento determinado necesita estar sincronizado, estableciéndose el tipo de sincronización, la categoría de sincronización y la conexión asociada al subsistema que sincroniza dicho elemento.

  El tipo de sincronización puede ser por:

  - Estado.
  - Velocidad.
  - Georreferencia.
  - Cualquier combinación de las tres anteriores.

  La categoría de sincronización determina una agrupación lógica de sincronización de los diferentes elementos sincronizados. Sobre dichas categorías se pueden establecer filtros para el seguimiento de estos grupos, (ver la descripción “Filtros por categorías de sincronización”).

  Cuando el elemento se inicializa en la sincronización se verifica si el mismo se encuentra en el servidor, de no estar se introduce y si está se suscribe el subsistema que realizó la solicitud a la lista de destinatarios que han sincronizado este elemento y se le envía el estado actual del elemento a dicho subsistema.
- Sincronizar dato: indica al servidor que un elemento fue modificado y por tanto debe notificarse a todos los subscriptores.

  Primeramente se verifica si el elemento está en el servidor, si no está se realiza una inicialización del mismo con tipo de sincronización por estado y categoría vacía, (ver la descripción “Inicialización de sincronización de dato”).

Si el elemento se encuentra en el servidor se verifica si se procede a la actualización teniendo en cuenta el tipo de sincronización a la que está subscrito el subsistema que realiza la petición y la diferencia de valores, ya sea de estado, velocidad, georreferencia, según corresponda. En caso de realizar la actualización se notifica a todos los subscriptores exceptuando al subsistema que realizó la solicitud.

- Desincronizar dato: indica al servidor que un elemento va a dejar de ser sincronizado por un subsistema determinado. Se elimina la subscripción del mismo y si no existen otros subscriptores a dicho elemento se elimina el elemento del servidor.
- Filtros por categorías de sincronización: estable o elimina el seguimiento a una categoría de sincronización por parte de un conjunto de subsistemas.
  El seguimiento de una categoría consiste en notificar a todos los subscriptores interesados de cuándo se adiciona, se elimina o se actualiza un elemento de dicha categoría.

2.6.4 Notificaciones

Un aspecto importante de los sistemas distribuidos es que la comunicación se lleva a cabo mediante la descripción de las características de los elementos de datos que se van a intercambiar. Como consecuencia, el nombre que identifique el elemento juega un papel crucial. Las notificaciones desde su llegada utilizando el nombre pueden ser usadas para distinguir la operación que desea realizar el subsistema que envía el mensaje, que puede ser publicación o suscripción. Los nombres de las notificaciones deben ser previamente acordados por todos los subsistemas que intervengan en el proceso de sincronización. Esto se relaciona con la lógica de negocio del sistema.

En el SIGESC se estableció un conjunto de elementos que permite identificar una notificación. La misma está compuesta por:

- Código: es un número único definido en todo el sistema para cada notificación lo que permite al receptor clasificarla.
- Tipo de notificación: es una descripción de cada notificación que permite identificarla.
- Punto de conexión respuesta: indica el canal de comunicación (IP y puerto) del subsistema al que se debe responder.

- Punto de conexión destino: indica el canal de comunicación (IP y puerto) del subsistema al que debe llegar la notificación.
- Dato: es variable, dependiendo del tipo de notificación. Representa la información que se quiere trasmitir de un elemento.

Una notificación es un objeto, que para enviarlo por la red se utiliza el mecanismo de serialización del subsistema de comunicaciones de DMAX. Para el envío de notificaciones es necesario establecer un canal de comunicación (IP y puerto) entre el remitente y el receptor. Los canales en DMAX son clasificados en permanentes y temporales. En el caso de la interacción entre el servidor de sincronización y los subsistemas del SIGESC que necesiten sincronizarse se plantea que el canal debe ser permanente, teniendo en cuenta el gran flujo de información entre ellos. Mantener el canal de comunicación permanente entre los clientes y el servidor de sincronización tiene como ventajas que:

- el tiempo de respuesta del sistema mejora, las suscripciones que desencadenan el inmediato envío de los cambios se realiza mejor cuando la conexión ya existe.
- la reducción del número de conexiones, de manera que se necesita invertir menos recursos del sistema tanto en el lado del cliente como en el servidor.

2.6.5 Vistas arquitectónicas

2.6.5.1 Modelo de casos de uso

El subsistema de sincronización para apoyar y contribuir a garantizar la coherencia de la información en todos los subsistemas que la comparten tiene que realizar un conjunto de funcionalidades. Estas funcionalidades se enumeran a continuación:

- Cargar configuración inicial: carga la identidad del servicio, que son las credenciales con las cuales se conectará a la BD, el puerto por donde el subsistema de sincronización recibe las solicitudes de sincronización de los subsistemas del SIGESC que necesiten sincronizarse e inicia el proveedor de administración.
- Gestionar elementos sincronizados: realiza las funcionalidades del protocolo PSDS que son:
    - Inicializar sincronización de datos.
    - Sincronizar dato.
    - Desincronizar dato.

  - Filtrar por categoría de sincronización.

- Gestionar sincronización de solicitud: gestiona los elementos sincronizados de la categoría “solicitud”, debido a que el cambio de estado de una solicitud no depende de un subsistema en sí, sino del conjunto de subsistemas en que se está atendiendo.
- Proveer información de administración: recibe peticiones de la aplicación de administración y procesa todos los comandos necesarios para proveer la información de administración al subsistema.

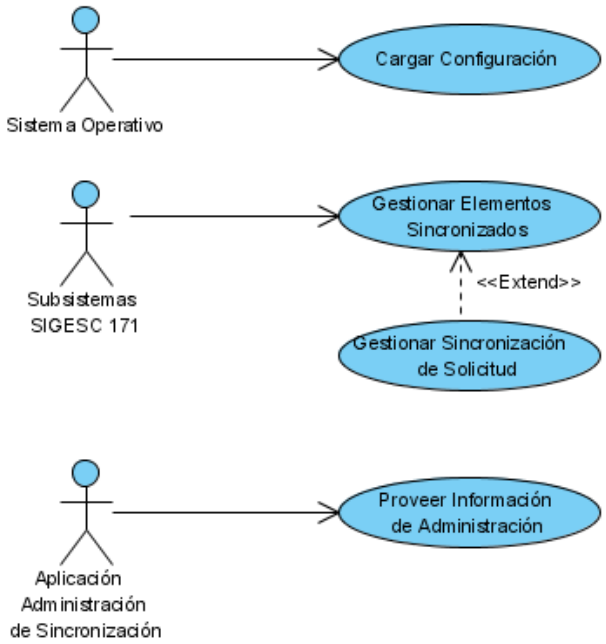

Figura 13. Diagrama de CU del subsistema servidor de sincronización

2.6.5.2 Modelo de diseño

El diagrama de clases de diseño que se muestra a continuación es una vista de alto nivel de la realización de las funcionalidades de los CU del subsistema servidor de sincronización que especifica el esquema de implementación del subsistema en el marco arquitectónico definido para el SIGESC.

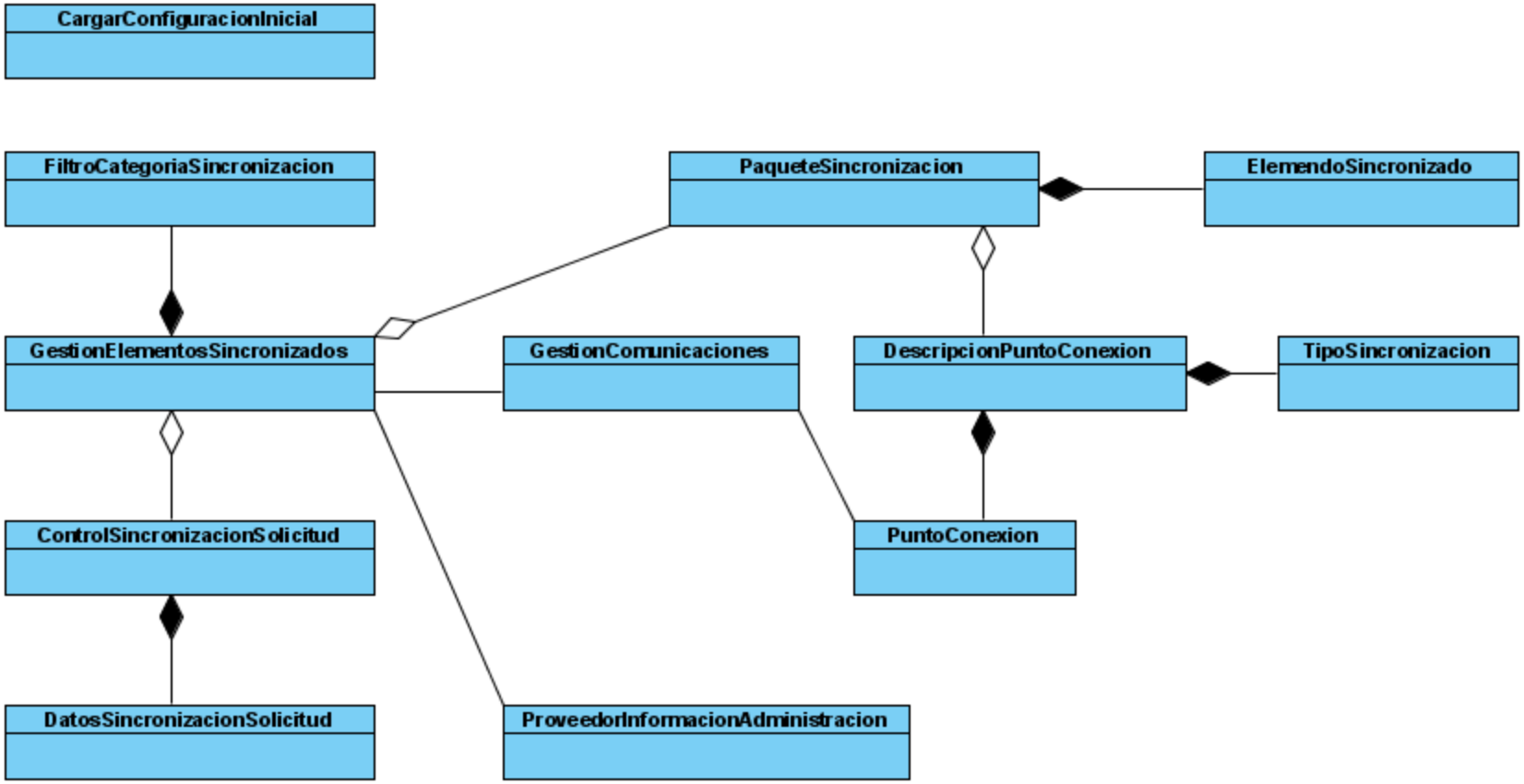

Figura 14. Diagrama de diseño del Subsistema servidor de sincronización

CargarConfiguracionInicial: carga de la BD la configuración establecida para el sistema. Los datos que obtiene son: identidad del servicio y puerto de comunicación por donde el subsistema de sincronización recibe las solicitudes de sincronización de los subsistemas del SIGESC.

FiltroCategoriaSincronizacion: encapsula la información de un filtro de categoría de sincronización, específicamente el identificador de la categoría y el grupo de subsistemas que le dan seguimiento a la misma.

GestionElementosSincronizados: gestiona todos los elementos sincronizados en el servidor prestando los servicios siguientes:

- Inicialización de Sincronización.
- Sincronizar.
- Desincronizar.
- Gestión filtros de categoría de sincronización.

Además gestiona la interacción con el servidor AVL para aquellos elementos que necesiten tal seguimiento y se pueden enviar mensajes por difusión a todos los elementos sincronizados.

GestionComunicaciones: gestiona las solicitudes de conexión o desconexión y las notificaciones (enviar y recibir) de los subsistemas que requieren la sincronización de elementos.

ControlSincronizacionSolicitud: se gestiona de forma especializada los elementos sincronizados de la categoría “solicitud” debido a que el cambio de estado de una solicitud no depende de un subsistema en sí, sino del conjunto de subsistemas en que se está atendiendo.

DatosSincronizacionSolicitud: encapsula el identificador de la solicitud y el conjunto de subsistemas que están atendiendo a dicha solicitud.

ElementoSincronizado: representa un elemento sincronizado, específicamente el identificador del elemento, la categoría a la que pertenece, su estado, velocidad y georreferencia.

PaqueteSincronizacion: representa un paquete de sincronización que está compuesto por un elemento sincronizado, el conjunto de subsistemas que le dan seguimiento y el tipo de sincronización por cada uno de estos subsistemas.

PuntoConexion: contiene los datos para establecer comunicación TCP con un subsistema. Los datos son IP y puerto.

TipoSincronizacion: indica los tipos por los que se puede sincronizar un elemento: estado, punto de georreferencia, velocidad o cualquier combinación de los anteriores.
DescripcionPuntoConexion: encapsula el tipo de sincronización de un elemento y el punto de conexión que requiere de la actualización sobre los cambios que se realicen en la característica del elemento especificado en el tipo de sincronización.
ProveedorInformacionAdministracion: provee la información requerida por la aplicación de administración.

2.6.5.3 Modelo de implementación

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. El uso más importante de estos diagramas es mostrar la estructura de alto nivel del modelo de implementación. Se utilizan para modelar la vista estática de un sistema. Muestran la organización y las dependencias lógicas entre un conjunto de componentes software, sean éstos componentes de código fuente, librerías, binarios o ejecutables.
A continuación se modela un diagrama de componentes, con los componentes principales que constituyen el subsistema servidor de sincronización. Para crearlo se tuvieron en cuenta las pautas del proyecto donde cada fichero de diseño está representado por un componente.

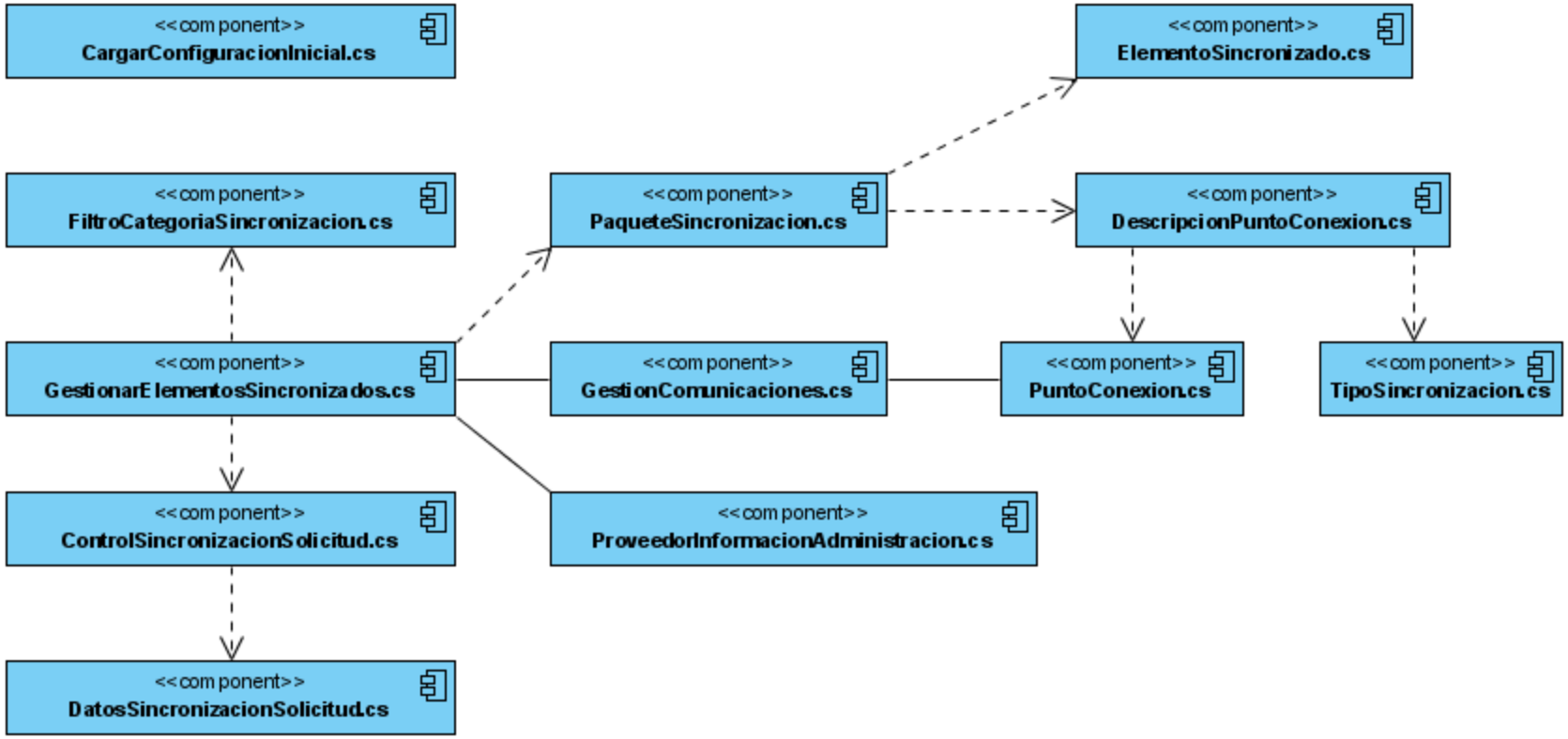

Figura 15. Diagrama de implementación del subsistema servidor de sincronización

2.7 Subsistema de administración de sincronización

El servidor de sincronización es un servicio *Windows* por lo que no posee interfaz para que el usuario pueda interactuar con él. Debido a que se necesita configurarlo, ver el estado en que se encuentra y darle seguimiento, surge la necesidad de crear un subsistema de administración que se encargue de tales funciones.

2.7.1 Mecanismo de administración

Definir el mecanismo para administrar el servidor es un punto de análisis esencial, teniendo en cuenta aspectos como administración remota, seguridad, comunicaciones, API disponibles en la plataforma .Net, complejidad de desarrollo, entre otras, indispensables para garantizar el éxito.

La siguiente tabla refleja una comparación entre tres mecanismos estudiados en cuanto a los aspectos mencionados.

Tabla 2. Comparación entre mecanismos de administración.

Mecanismos de Administración	Seguridad	API	Complejidad de Desarrollo (Estimada)
.Net Remoting	No	Sí	Media
Socket	No	Sí	Alta
WMI	Seguridad Basada en Windows	Sí	Baja

2.7.1.1 Instrumental de administración de *Windows* (WMI)

WMI es un componente del sistema operativo *Windows* que permite obtener acceso mediante programación a información de administración en un entorno empresarial. WMI en .NET *Framework* se basa en la tecnología WMI original y permite el mismo desarrollo de aplicaciones y proveedores, además de las ventajas que ofrece la programación en .NET *Framework*.

Se pueden escribir scripts o aplicaciones WMI para automatizar tareas administrativas en computadoras remotas o locales. Provee una interfaz uniforme para cualquier aplicación local o remota o scripts que obtienen datos administrativos de un sistema operativo o de una red.

WMI provee gran cantidad de servicios de administración de sistema incorporados dentro del sistema operativo *Microsoft Windows*. Un espectro ancho de aplicaciones, servicios y dispositivos están disponibles para usar WMI con el fin de proveer

características administrativas extensivas para operaciones de tecnología de la información (TI). El uso de sistemas administrativos basados a WMI conduce a ambientes de computación más robustos y mayor fiabilidad de sistema.

La escritura de una aplicación cliente o de un proveedor con WMI en .NET *Framework* proporciona diversas ventajas con respecto a WMI original, y en especial para los desarrolladores que utilizan C#. Algunas de las ventajas de WMI en .NET *Framework* son las siguientes:

- Aprovechamiento de características de *Common Language Runtime*.
- Simplicidad de uso.
- Acceso a todos los datos WMI. Las aplicaciones cliente tienen el mismo tipo de acceso a datos WMI y pueden realizar las mismas operaciones con éstos que en WMI original.

El desarrollo de proveedores y aplicaciones de administración empresarial es más rápido con WMI en .NET *Framework*, pero existen algunas limitaciones. Estas limitaciones afectan principalmente a los proveedores de objetos instrumentados con código administrado y se presentan a continuación:

- Los proveedores de código administrado no pueden definir métodos.
- Aunque WMI admite objetos incrustados y referencias a otros objetos con WMI en .NET *Framework*, sólo se pueden utilizar objetos incrustados cuando se definen nuevas clases.
- No se puede crear un proveedor del consumidor de eventos en código administrado.
- WMI en .NET *Framework* no admite actualizadores.

Para suplir algunas de las limitantes heredadas de WMI.Net se utilizan controladores de servicios de Windows especializados que provee .Net para el envío de comandos así como arrancar, detener o pausar el servicio.

2.7.2 Distribución

El subsistema de administración puede residir en la misma computadora del servidor de sincronización o en otra. La tecnología de administración WMI que utiliza lo abstrae de la lógica necesaria para establecer la comunicación remota y seguridad.

2.7.3 Vistas arquitectónicas

2.7.3.1 Modelo de casos de uso

El subsistema de administración implementa un conjunto de funcionalidades para poder configurar, dar seguimiento y conocer el estado en que se encuentra el servidor de sincronización. Entre sus funcionalidades se encuentra:

- Cargar configuración: obtener los datos necesarios para conectarse con el servidor de sincronización.
- Consumir información: obtener del servidor de sincronización los datos relacionados con los elementos que se están sincronizando.
- Representar elemento sincronizado: mostrar los datos de los elementos que se están sincronizando, que son:
    - Identificador.
    - Categoría.
    - Estado.
    - Punto de georreferencia.
    - Velocidad.
- Establecer identidad del servicio: indica cuáles son los datos de autenticación que utilizará el servicio de sincronización para ejecutarse. Se introducen las credenciales de sesión y de BD necesarias para configurar la identidad del servicio. Los datos de la identidad del servicio son:
    - Credenciales de la Sesión.
    - Usuario.
    - Contraseña.
    - Dominio.
    - Credenciales de la BD.
    - Usuario.
    - Contraseña.
    - Servidor.
- Ver detalles de elemento sincronizado: muestra los datos en detalles de un elemento sincronizado. Obtiene los datos complementarios que no se muestran en la interfaz principal como detalles de un elemento sincronizado. Los datos son:
    - Lista de descripción de los puntos de conexión.
    - Punto de Conexión.

  - Número IP.
  - Puerto de comunicación.
  - Tipo de Sincronización.
- Administrar servicio: obtiene y muestra los datos del servicio de sincronización. Los datos son:
  - Nombre.
  - Nombre que muestra.
  - Estado.
  - Corriendo.
  - Pendiente Corriendo.
  - Parado.
  - Pendiente Parado.

A partir del estado en que se encuentre el servicio de sincronización es posible detener o iniciar el servicio.

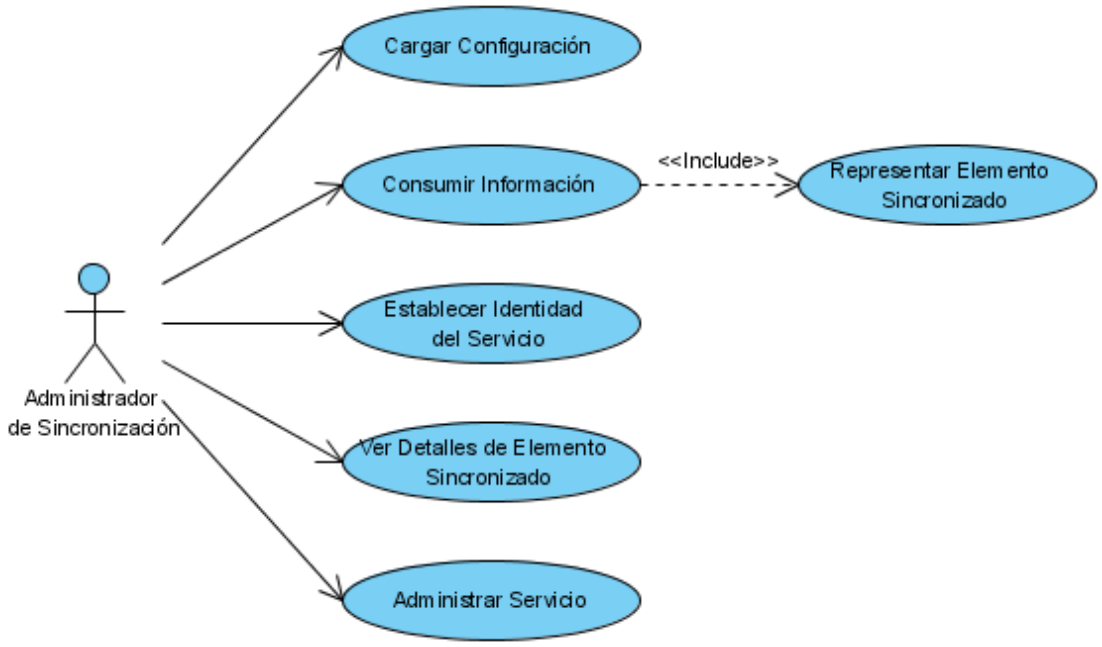

Figura 16. Diagrama de CU del subsistema de administración de sincronización

2.7.3.2 Modelo de diseño

El diagrama de clases de diseño que se muestra a continuación es una vista de alto nivel de la realización de las funcionalidades de los CU del subsistema de administración.

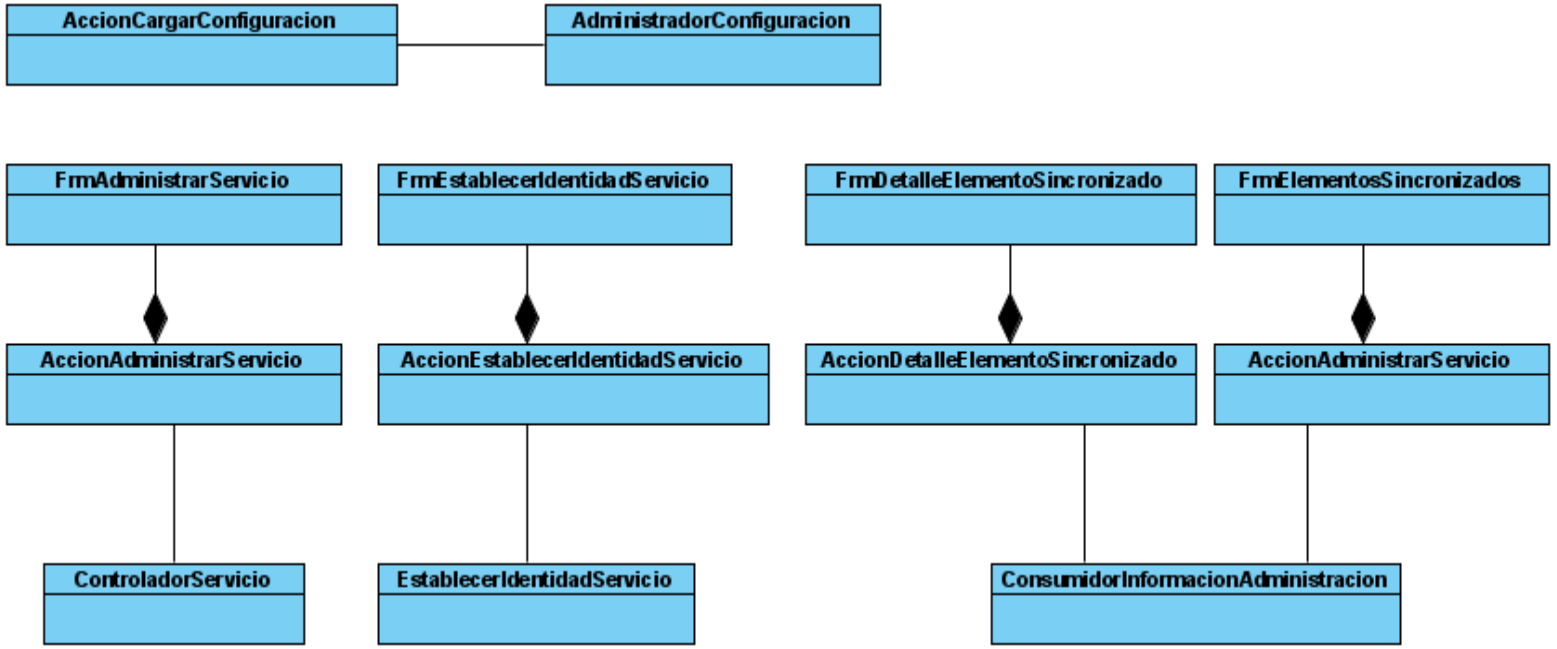

Figura 17. Diagrama de diseño del subsistema de administración de sincronización

AccionCargarConfiguracion: inicializa las operaciones para obtener la configuración establecida para el subsistema de administración de sincronización.

AdministradorConfiguracion: obtiene de la BD la configuración definida para el subsistema de administración de sincronización.

FrmAdministrarServicio: interfaz visual que contiene los controles necesarios para establecer el estado del servicio y mostrar información sobre él.

AccionAdministrarServicio: define los manipuladores de eventos que se ejecutarán sobre la clase FrmAdministrarServicio.

ControladorServicio: permite la manipulación de los estados del servicio y provee información sobre el mismo.

FrmEstablecerIdentidadServicio: interfaz visual a través de cual se indica los datos que definen la identidad del servicio. Los datos son:

- Credenciales de la Sesión.
- Usuario.
- Contraseña.
- Dominio.
- Credenciales de la BD.
- Usuario.
- Contraseña.
- Servidor.

AccionEstablecerIdentidadServicio: define los manipuladores de eventos que se ejecutarán sobre la clase FrmEstablecerIdentidadServicio.

EstablecerIdentidadServicio: gestiona la identidad del servicio almacenando los datos necesarios en la máquina local. Los datos almacenados son encriptados.

FrmDetalleElementoSincronizado: interfaz visual que muestra los datos en detalles de un elemento que se está sincronizando. Los datos son:

- Lista de descripción de los puntos de conexión.
- Punto de Conexión.
- Número IP.
- Puerto de comunicación.
- Tipo de Sincronización.

AccionDetalleElementoSincronizado: define los manipuladores de eventos que se ejecutarán sobre la clase FrmDetalleElementoSincronizado.

FrmElementosSincronizados: interfaz visual que muestra el listado de los elementos que se están sincronizando. Los datos que se visualizan en la lista de cada elemento son:

- Identificador.
- Categoría.
- Estado.
- Punto de georreferencia.
- Velocidad.

AccionAreaTrabajoSincronizacion: define los manipuladores de eventos que se ejecutarán sobre la clase FrmElementosSincronizados.

ConsumidorInformacionAdministracion: interactúa con el proveedor de información del servicio de sincronización para obtener información sobre el funcionamiento del mismo y establecerle parámetros de configuración.

2.7.3.3 Modelo de implementación

A continuación se modela un diagrama de componentes, con los componentes principales que constituyen el subsistema de administración de sincronización. Para crearlo se tuvieron en cuenta las pautas del proyecto donde cada fichero de diseño está representado por un componente.

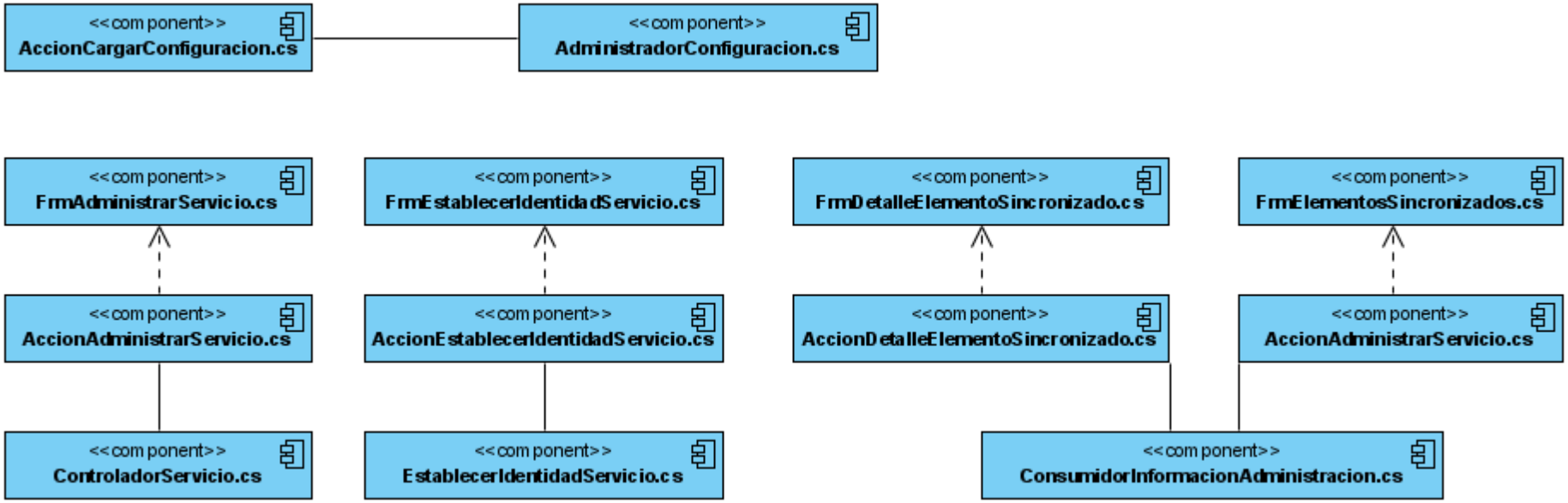

Figura 18. Diagrama de implementación del subsistema de administración de sincronización

Conclusiones del capítulo

En el presente capítulo se demostró que existen varios escenarios en el SIGESC 171 donde es necesario que existan mecanismos de sincronización para lograr mantener la información coherente y actualizada en los subsistemas, aspecto que es vital para tomar decisiones más precisas en situaciones de emergencias.

Se identificaron los aspectos arquitectónicos que conforman el mecanismo de sincronización del SIGESC que resuelven la problemática planteada, definiendo:

- que es necesario un enfoque centralizado utilizando el modelo cliente-servidor.
- como paradigma de comunicación publica-suscribe, con los modelos de suscripción basado en tipo y basado en contenido.
- la estructura de los objetos notificaciones.
- un protocolo de sincronización para establecer la comunicación entre el servidor y los subsistemas que necesiten consumir sus servicios.

Capítulo 3 Validación de la solución

"*La calidad y objetividad de una investigación científica se mide mediante los criterios de validez y fiabilidad de sus resultados.*" (CARAZO, 2006)

La validación de esta investigación se centra en demostrar que la propuesta de solución con la incorporación del subsistema de sincronización de información en los procesos de gestión de emergencias como modelo arquitectónico centralizado permite actualizar la información compartida entre varios subsistemas del SIGESC 171 en "tiempo real". Además, demostrar la influencia que tienen la cantidad de subsistemas que estén compartiendo la información, así como la cantidad de mensajes a trasmitir por la red y la latencia de la misma en el tiempo de respuesta del subsistema de sincronización.

3.1 Prueba de rendimiento del sistema

Para realizar una valoración del rendimiento del subsistema de sincronización propuesto, es necesario definir entornos de prueba. El entorno de prueba definido está en correspondencia con la situación real de los centros de gestión de emergencias para los que fue creado el sistema, que se corresponde con una red LAN.

Se tiene una computadora con procesador Intel(R) Core(TM) i3-2330M @ 2.20GHz de velocidad y 1 Gb de memoria RAM donde está instalado el subsistema servidor de sincronización. El simulador de despacho se encuentra distante geográficamente en una computadora con procesador Intel(R) Core(TM)2 Duo @ 2.66GHz de velocidad y 788 MB de memoria RAM. En ambas computadoras se tiene instalado el sistema operativo Windows XP.

Además se tiene una computadora Pentium(R) Dual-Core con CPU E5300 @ 2.60GHz de 2 Gb de memoria RAM como servidor de base de datos que se encuentra distante geográficamente de la computadora cliente. Está montado sobre el sistema operativo CentOS release 5.6.

La aplicación simulador de despacho permite crear varios canales de comunicación con el servidor de sincronización, simulando por cada canal un subsistema del SIGESC que necesite sincronizar un elemento. También permite mandar notificaciones que impliquen: suscribirse en la lista de elementos sincronizados y actualizar el estado de un elemento de la lista de elementos sincronizados donde tiene que mandarle a todos los suscritos a ese elemento menos a él, la actualización del estado.

Para validar la propuesta de solución es necesario aplicar un enfoque cuantitativo de estudio de casos múltiples (CARAZO, 2006). Los casos en esta investigación, constituyen un conjunto de situaciones de prueba para conocer el tiempo que se demora en actualizar la información, teniendo en cuenta diferentes valores de los tres factores mencionados al inicio del capítulo (la cantidad de subsistemas que comparten información, la cantidad de mensajes trasmitidos en un instante de tiempo y la latencia de la red). Por lo tanto, se tienen como variables independientes los tres factores y como variable dependiente el tiempo de procesamiento del subsistema de sincronización. Entiéndase por latencia de la red la suma de retardos temporales dentro de la red desde que un mensaje es enviado, origen, hasta la llegada a su destino (DUATO *et al.*, 2002).

En los experimentos que existen más de una variable independiente se emplean los diseños factoriales (SAMPIERI *et al.*, 2001) e incluyen dos o más niveles de presencia en cada una de las variables independientes. En el caso de la presente investigación se definieron tres niveles de presencia de las variables cantidad de notificaciones enviadas y cantidad de subsistemas suscritos a un elemento sincronizado, para tener mayor exhaustividad en los resultados y se tienen dos niveles de la variable latencia de la red, por lo que se corresponde con un diseño 3^2 x 2^1. Para mostrar los resultados se decide emplear dos gráficas, una para cada nivel de latencia de red, que es equivalente a tener una gráfica con la representación del 3^2 en cada caso.

3.1.1 Selección de niveles para las variables independientes

Variable independiente: latencia

Para la definición del nivel de latencia es necesario realizar un registro histórico de al menos un mes del comportamiento de la red y así poder establecer valores de latencia bajos y altos en la red donde se va a implantar el subsistema. Ese estudio no fue posible realizarlo, quedando como una recomendación de esta investigación. Lo que se hizo para obtener los valores de la latencia de la red fue asumir que en el momento de las pruebas era baja, a pesar de que la computadora donde se encontraba el servidor estaba en una subred distinta a la computadora donde se encontraba el simulador de despacho y que pudieron existir cientos de usuarios conectados a las mismas. El valor alto de latencia se obtuvo suponiendo que sería 10 veces más que el valor bajo asumido.

En el sistema a probar la latencia es medida como el tiempo que demora un mensaje o un conjunto de mensajes desde que se envía/n desde un subsistema y hasta el momento en que llega/n al subsistema de sincronización. Se realizaron varias pruebas con diferentes números de mensajes enviados para comprobar en situaciones de mayor cantidad de tráfico cuánto se retardan los mensajes.

Variable independiente: cantidad de subsistemas suscritos

En el caso de la variable cantidad de subsistemas conectados se definieron tres niveles. El nivel bajo es asumido cuando solamente está suscrito a la lista de elementos sincronizados 1 subsistema. El nivel medio se asume cuando existen 10 subsistemas suscritos en la lista y el nivel alto cuando existen 20 subsistemas.

Para determinar dichos valores se tuvieron en cuenta las características de los centros de gestión de emergencias donde iba a ser implantado el sistema. Mencionar que en un centro puede haber más puestos de trabajo, pero no todos sincronizan información, como se mencionó en los capítulos anteriores.

Variables independientes: cantidad de notificaciones

Esta variable se refiere a la cantidad de notificaciones que pueden llegar concurrentemente al subsistema de sincronización. Para este caso se definieron tres niveles también. Teniendo en cuenta los valores otorgados a la variable independiente anterior se asumió como nivel bajo 1 notificación, como nivel medio 10 notificaciones y como nivel alto 20 notificaciones.

3.1.2 Rendimiento del subsistema de sincronización en un escenario

El escenario de prueba escogido se corresponde con el de sincronizar el estado de un elemento. En el caso de llegar una notificación de actualización de estado debe notificarse que ha ocurrido un cambio a la cantidad de subsistemas suscritos a los cambios de estado del elemento a sincronizar menos el subsistema que envió la actualización.

A continuación se muestra el gráfico de diseño 3^2 correspondiente al tiempo de respuesta del subsistema de sincronización ante determinadas combinaciones de las variables independientes. En el eje de las x se encuentra la cantidad de notificaciones enviadas concurrentemente y en el eje y la cantidad de subsistemas suscritos a los cambios que ocurran en un elemento. La intercepción de ambas variables indica el tiempo en segundos que se demora el subsistema sincronización en procesarlas y enviar las notificaciones. (Tabla 3)

Tabla 3. Rendimiento del subsistema sincronización.

cantidad de subsistemas		cantidad de notificaciones		
		1	10	20
	1	0.0156	0.0156	0.0312
	10	0.0312	0.1406	0.3594
	20	0.0469	0.3125	0.6719
Intel Corei3 SO W XP				

Como se puede apreciar el tiempo de procesamiento es siempre menor a 1 segundo, incluso para el escenario donde se tienen 20 subsistemas conectados y se envían 20 notificaciones al subsistema de sincronización. En este caso el subsistema debe enviar 20*(20-1) cantidad de notificaciones, además de buscar por cada elemento de la lista los subsistemas que estén sincronizando el elemento por el estado. El proceso de sincronización del elemento para el escenario más crítico, se realiza en un tiempo aceptable acorde a la autora de 0.6719 segundos.

Se puede comprobar además que manteniendo constante la cantidad de subsistemas y aumentando de la cantidad de notificaciones el tiempo de procesamiento del servidor de sincronización aumenta, aspecto que era esperado debido a que a medida que aumenta la cantidad de notificaciones recibidas aumenta el número total de notificaciones a enviar, a parte del proceso de búsqueda que debe hacer por todos los subsistemas suscritos.

Se puede comprobar también que manteniendo constante la cantidad de notificaciones y aumentando de la cantidad de subsistemas el tiempo de procesamiento del servidor de sincronización aumenta, aspecto que era esperado debido a que por cada notificación recibida de actualizar elemento se deben enviar tantas notificaciones como cantidad de subsistemas - 1 tenga en la lista de elementos suscritos, a parte del proceso de búsqueda que debe hacer por todos los nodos suscritos.

El número de notificaciones de actualización de estado que debe enviar el subsistema de sincronización se corresponde con lo siguiente:

(Número de subsistemas suscritos al cambio - 1)* Número de notificaciones

En este escenario es importante destacar que si se aumentan las prestaciones de *hardware* de la computadora del subsistema de sincronización, los tiempos de respuesta serían menores.

3.1.3 Rendimiento del proceso sincronización

En el escenario de prueba anterior no se tuvo en cuenta la variable independiente latencia de la red, que puede influir negativamente en los tiempos de respuesta del sistema. En un sistema de tiempo real como el que se aborda en la investigación, no es solamente importante tener en cuenta el tiempo de procesamiento del servidor, es necesario conocer el tiempo desde que se envió una notificación de actualización de un elemento hasta que se avisa a todos los suscritos a los cambios de dicho elemento. Es decir, el tiempo desde que ocurre el cambio en un elemento hasta que todos estén sincronizados con la información actualizada del elemento.

Para realizar este tipo de prueba, se mantuvieron los valores de las variables independientes cantidad de notificaciones y cantidad de subsistemas del escenario anterior. Se tomó la hora que correspondía con el inicio del envío de la notificación hacia sincronización de actualización de estado del elemento, hasta que la notificación de actualización le llega al último subsistema en sincronizarse. En la Figura 19 el tiempo fin se corresponde con el tiempo de llegada del mensaje del paso 3.

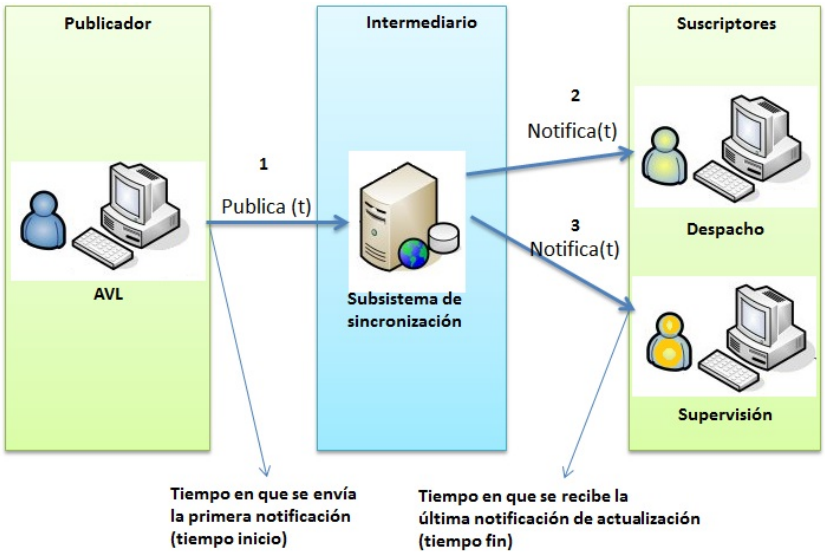

Figura 19. Proceso de sincronización

Como se puede apreciar en este escenario interviene la red, que introduce retardos en los tiempos de la actualización de la información, por lo tanto es una variable que hay que observar su comportamiento.

Como se mencionó anteriormente para hacer corresponder el diseño 3^2 x 2^1, con la modelación 3^2 se presentan dos tablas (Tabla 4, Tabla 5), una para el caso de la latencia de red baja y otra para la alta, en la que se supone que la latencia va a ser 10 veces mayor que la latencia baja.

Tabla 4. Diseño 3^2 con latencia baja.

cantidad de subsistemas		cantidad de notificaciones		
		1	10	20
	1	0.0312	0.0625	0.1562
	10	0.0600	0.1903	0.4306
	20	0.0801	0.3705	0.7110
Latencia de la red baja				

Tabla 5. Diseño 3^2 con latencia alta.

cantidad de subsistemas		cantidad de notificaciones		
		1	10	20
	1	0.1562	0.4844	1.2500
	10	0.6000	1.9030	4.3059
	20	0.8010	3.7050	7.1100
Latencia alta suponiendo que es 10 veces mayor que la baja				

En la Tabla 4 se puede apreciar el mismo resultado en cuanto a proporcionalidad que en la Tabla 3. Al aumentar el número de subsistemas y mantener constante el número de notificaciones, aumenta el tiempo en que se actualiza la información en los subsistemas y viceversa. Se puede apreciar que para el escenario más crítico de sincronización donde 20 subsistemas están suscritos a la lista del elemento modificado y se envían 20 notificaciones de cambios sobre el mismo, el tiempo máximo de actualización a todos los subsistemas es 0.7110 segundos, considerado por la autora como un tiempo aceptable en sistemas de gestión de emergencias con esa concurrencia de usuarios. Comparando los resultados con la Tabla 3, se puede comprobar que la influencia de la latencia de la red afecta inversamente el tiempo de respuesta del sistema.

En la Tabla 5 donde la latencia es aumentada 10 veces con respecto a la de bajo nivel, el tiempo máximo de actualización de los datos es 7.1100 segundos. Este tiempo, a criterio de la autora, puede ser un tiempo adecuado, aceptado en sistemas de tiempo real para el tipo de emergencias tratadas. Cabe destacar que los sistemas de gestión de emergencias en cuestión, se encuentran en una red LAN donde la latencia de la red pudiera llegar a ser menor que en la Universidad de las Ciencias Informáticas donde se realizaron las pruebas y se obtuvieron resultados favorables siempre en el orden de los milisegundos.

Conclusiones del capítulo

Se pudo comprobar que el tiempo de procesamiento del servidor de sincronización cuando la latencia de la red es cero, es de 0.67 segundos para el caso más crítico donde se tienen 20 subsistemas suscritos a la lista de elementos sincronizados y se envían 20 notificaciones, siendo un tiempo aceptable para este tipo de sistemas.

Además se demostró que la latencia de la red es una variable que se debe tratar en los sistemas de emergencias de tiempo real, pues influye considerablemente en los tiempos de respuesta del sistema. Considerando que en un centro de gestión de emergencias exista una red como en la UCI se tiene como tiempo de actualización a todos los subsistemas en el caso más crítico 0.71 segundos, considerado un tiempo admisible. Para el caso simulado donde se aumenta 10 veces la latencia de la red el tiempo que se demora en actualizar todos los subsistemas es aproximadamente igual a 7.11 segundos, que para este tipo de sistemas puede ser aceptado.

Se pudo apreciar que para el primer modelo (descentralizado) la cantidad de canales de comunicaciones es casi 6 veces mayor que para el modelo centralizado por lo que el consumo de recursos es menos en este último.

Conclusiones

Luego de terminada la investigación y realizadas las pruebas al subsistema de sincronización de información propuesto, se concluye:

- Investigaciones realizadas demuestran que aunque es necesaria la actualización de los datos en tiempo real en todos los subsistemas que comparten información en un sistema de gestión de emergencias, algunos sistemas no tienen implementados mecanismos para garantizarlo y el resto de los sistemas consultados no brindan información sobre cómo gestionan su realización.
- Ante la tarea de crear el Sistema de Gestión de Emergencias de Seguridad Ciudadana 171 se hizo necesario definir e implementar un subsistema de sincronización para mantener la información actualizada en todo el sistema.
- Se desarrolló el subsistema de sincronización que desde el punto de vista práctico su aplicación constituye un aporte en el diseño de los sistemas de gestión de emergencias de manera general.
- En el diseño del subsistema de sincronización constituye un aporte el uso del enfoque centralizado utilizando el modelo cliente-servidor, teniendo como clientes a todos los subsistemas que compartan datos y como servidor el subsistema de sincronización. Además, otro aporte lo constituye el uso del paradigma de comunicación publica-suscribe con los modelos de suscripción basado en tipo y basado en contenido, y el protocolo creado para la sincronización de información en el SIGESC para utilizar sobre el protocolo TCP.
- Se realizó la validación del subsistema implementado en un entorno creado para simular el comportamiento del sistema en un ambiente real, donde se pudo comprobar que el tiempo de procesamiento de las peticiones al servidor de sincronización tiene valores menores o iguales a 0.71 segundos para las condiciones de la red UCI, que es aceptable en sistemas de gestión de emergencias médicas, policiales y de protección civil de tiempo real.

El subsistema realizado está implantado desde el 2008 en cinco centros de gestión de emergencias de la República Bolivariana de Venezuela, funcionando adecuadamente, lo que sustenta el aporte práctico de la tesis.

Recomendaciones

Con el objetivo de mejorar la solución propuesta se recomienda para versiones posteriores:

- Incorporar al mecanismo de sincronización clúster para garantizar la alta disponibilidad minimizando el riesgo de los “cuellos de botella” que pueden ocurrir utilizando el modelo centralizado propuesto.
- Incorporar la identificación de saturación en la red para aplicar manipulación del ancho de banda en el mecanismo de comunicación que se emplea y hacer los mensajes adaptativos.
- Incorporar el almacenamiento de los registros de los mensajes enviados y recibidos.
- Hacer un estudio de la latencia de la red a partir de un registro histórico de los valores de la latencia al menos por un período de un mes, para situaciones similares en el orden de clientes y las características físicas de enlace entre dichos clientes con el subsistema de sincronización.

Referencias bibliográficas

1-7-1, S. A. D. E. B. *Emergencias Bolívar 171.* 2010, [Consultado el: Mayo/2012]. Disponible en: http://www.e-171.gob.ve/.

2MARES. *Centratel Contact Center.* 2011, [Consultado el: Mayo/2012]. Disponible en: http://www.2mares.com/Descargas/2.-Catalogo_CENTRATEL_CU_ACD_CTI_B2M.pdf.

2MARES y SATDATA. *SIGEME Sistema de gestión de emergencias.* 2011, [Consultado el: febrero/2012]. Disponible en: http://www.2mares.com/Descargas/Catalogo_SIGEME.pdf.

AGRAMONTE, A. M.; TELLEZ, E. S.*, et al.* Especificación de la arquitectura del SIGESC 171. 2008, nº

APONTE, R. *A Modular Multicasting Algorithm with a Contour Discovery Approach.* ProQuest Information and Learning Company, 2006. ISBN 3232036.

APTE, A. *Java Connector Architecture: Building Custom Connectors and Adapters.* Sams Publishing, 2002. ISBN 0672323109.

ARAGUA, A. C. B. V. D. *Sistema Integrado de Atención a Emergencias 171. Aragua.* 2010, [Consultado el: mayo/2012]. Disponible en: http://www.171aragua.org.ve/index.php.

BABER, S. B.; BRITTON, K. H.*, et al. Methods, systems and computer program products for differencing data communication using message queue.* 2003. ISBN 6546428.

BIRMAN, K. P. *Guide to Reliable Distributed Systems: Building High-Assurance Applications and Cloud-Hosted Services.* New York: Springer, 2012. ISBN 9781447124160.

CARAZO, P. C. M. El método de estudio de caso. Estrategia metodológica de la investigación científica 2006, vol. 20, nº Disponible en: http://ciruelo.uninorte.edu.co/pdf/pensamiento_gestion/20/5_El_metodo_de_estudio_de_caso.pdf. ISSN 1657-6276.

CIDH, C. I. D. D. H. y OEA, O. D. L. E. A. *Informe sobre seguridad ciudadana y derechos humanos.* Copyright 2011 OEA, 2010, Disponible en: http://www.cidh.org/countryrep/seguridad/seguridadindice.sp.htm.

COULOURIS, G.; DOLLIMORE, J.*, et al. "Distributed Systems: Concepts and Design" 3rd. Edition.* Addison-Wesley, 2001. ISBN 0-13-214301-1.

---. *"Distributed Systems: Concepts and Design" 5th. Edition.* Addison-Wesley, 2012. ISBN 0-13-214301-1.

DICKMAN, A. y HOUSTON, P. *Designing Applications with MSMQ: Message Queuing for Developers.* Boston: Addison-Wesley Longman Publishing Co, 1998. ISBN 0201325810.

DUATO, J.; YALAMANCHILI, S.*, et al. Interconnection Networks.* Elsevier Science, 2002. ISBN 1558608524.

FEDETEC y TELEMÁTICOS, D. D. I. D. S. *Gestión de Emergencias Distribuido. Proyecto GEMYC D.* 2008, [Consultado el: Mayo/2012]. Disponible en: http://web.dit.upm.es/~gemycd/.

GOBIERNO, D. D. *Decreto 747. SISTEMA DE ATENCION TELEFONICA DE EMERGENCIA DE LA PROVINCIA DE BUENOS AIRES.* Editado por: Gobierno, D. D. publicado el: Abril 20 de 2005, última actualización: Abril 20. [Consultado el: Febrero/2012]. Disponible en: http://www.mjys.gba.gov.ar/.

GODDARD, L. A. *¿Cómo opera un sistema de atención de emergencias?* publicado el: Diciembre 7 de 2009, última actualización: Diciembre 7. Disponible en: http://www.contactforum.com.mx/articulos/2866.html.

GROSSO, W. *Java RMI.* Sebastopol: O´Reilly Media, Inc, 2002. ISBN 9781565924529.

GUNES, A. E. y KOVEL, J. P. Using GIS in Emergency Management Operations. *Journal of Urban Planning and Development*, 2000, vol. 126, nº 3, Disponible en: http://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9488%282000%29126%3A3%28136%29. ISSN 1943-5444.

HITCHCOCK, B. *Oracle Database 10g New Features.* Oracle Press, 2005. ISBN 0072229470.

INDRA. *iSafety Sistema integral de gestión de emergencias.* 2011, [Consultado el: mayo/2012]. Disponible en: http://www.indracompany.com/sites/default/files/isafety_1.pdf.

INSTITUTION, B. S. *Nueva norma ISO para la gestión de emergencias.* publicado el: enero 9 de 2012, última actualización: enero 9. [Consultado el: mayo/2012]. Disponible en: http://www.bsigroup.es/es/certificacion-y-auditoria/Sistemas-de-gestion/Novedades/Noticias-2012/LD-News-Source-/Nueva-norma-ISO-para-la-gestion-de-emergencias/.

JIA, W. y WANLEI, Z. *Distributed Network Systems from Concepts to Implementations.* New York: Springer Science+Business Media, Inc, 2005. ISBN 038723839.

JURÍDICAS, N. Ley 13/2010, de 23 de noviembre, de la Generalitat, de Protección Civil y Gestión de Emergencias. . *Noticias jurídicas*, 2010, nº [Consultado el: Febrero/2012]. Disponible en: http://noticias.juridicas.com/base_datos/CCAA/va-l13-2010.t6.html.

LARA, D. D. T. *Servicio Autónomo de Emergencias Lara 171* 2006, [Consultado el: mayo/2011]. Disponible en: http://www.sel171.gob.ve/.

LELANN, G. *"Motivations, Objectives, and Characterization of Distributed Systems" in Distributed Systems, Architecture and Implementation.* Springer-Verlag, 1981. ISBN 3-540-10571-9.

MEDINA, Y. T. Proyecto técnico del proyecto 171. 2003, nº

MICROSOFT. Aplicaciones de servicios de Windows. 2007, nº Disponible en: http://msdn.microsoft.com/es-es/library/d56de412%28v=vs.90%29.aspx.

---. *Message Queuing (MSMQ)* [Consultado el: mayo de 2012]. Disponible en: http://msdn.microsoft.com/en-us/library/windows/desktop/ms711472%28v=vs.85%29.aspx.

---. Socket Class. 2012b, nº Disponible en: http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.aspx.

---. *Synchronize Data.* Microsoft, 2012c, [Consultado el: diciembre/2011]. Disponible en: http://msdn.microsoft.com/en-en/library/ms151793.aspx.

ORACLE. Oracle® Database Net Services Reference 10g Release 2 (10.2). En 2005,

PARLAMENTARIA, C. D. I. *Seguridad Ciudadana.* publicado el: Mayo de 2005, última actualización: Mayo. [Consultado el: Marzo/2012]. Disponible en: http://www12.georgetown.edu/sfs/clas/pdba/Security/citizensecurity/peru/evaluaciones/seguridadciudadanacip.pdf.

RIZZO, L. y FDIDA, S. *Networked Group Communication: first international COST 264 Workshop, NGC´99.* New York: Springer, 1999. ISBN 3540667822.

S.A, A. S. *Gestión de emergencias. Sistema multiagencia y distribuido para gestión de incidencias.* 2012, [Consultado el: febrero/2012]. 2 p. Disponible en: http://www.desca.com/latam/mseg/Folletos/ESPA%C3%91OL/ESFCC02_Folleto_Emergencias.pdf.

SAMPIERI, R. H.; COLLADO, C. F.*, et al. METODOLOGÍA DE LA INVESTIGACIÓN.* McGRAW - HILL INTERAMERICANA DE MÉXICO, S.A. de C.V., 2001, Disponible en: http://www.upsin.edu.mx/mec/digital/metod_invest.pdf. ISBN 968-422-931-3.

STACKOVERFLOW. *TCP Vs. Http Benchmark* Disponible en: http://stackoverflow.com/questions/1196623/tcp-vs-http-benchmark.

STEPISNIK, B. *Distributed Object-Oriented Arquitectures: Sockets, Java Rmi and Corba.* Hamburg: Diplomica GmbH, 2007. ISBN 9783836650335.

TANENBAUM, A. S. *Redes de computadoras. Cuarta Edición.* México: Pearson Education, 2003. ISBN 9702601622.

TANENBAUM, A. S. y VAN STEEN, M. *"Distributed Systems: Principles and Paradigms" 2da Edición.* Prentice Hall, 2006. ISBN 0-13-239227-5.

TARKOMA, S. *Publish/subscribe systems. Design and principles.* John Wiley & Sons Ltd, 2012. ISBN 9781119951544.

VALENCIANA, E. C. *Teléfono de emergencias 112.* 2012, [Consultado el: mayo/2012]. Disponible en: http://www.112cv.com/ilive/srv.112CV.QueEsEl112.

VÖLTER, M.; KIRCHER, M.*, et al. Remoting Paterns: Foundations of Enterprise, Internet and Realtime. Distributed Object Middleware.* England: John Wiley & Sons, Ltd, 2005. ISBN 0470856629.

WESWIT. *Lightstream. Real-time data push* [Consultado el: diciembre de 2011]. Disponible en: http://www.lightstreamer.com/?gclid=CP-z3O_G67ICFQqZ4AodRgkA8w.

---. *Lightstreamer. White paper.* publicado el: 5 junio de 2006b, última actualización: 5 junio. [Consultado el: 2011]. Disponible en: http://www.lightstreamer.com/docs/Lightstreamer_WhitePaper.pdf.