Universidad Autónoma de Querétaro

Facultad de Informática

Maestría en Ingeniería de Software Distribuido

"Acceso Inteligente a Bases de Datos Remotas por medio de Agentes Móviles"

TESIS

Que como parte de los requisitos para obtener el grado de

Maestro en Ingeniería de Software Distribuido

Presenta:

Ing. Rebeca Eugenia Aguilar Durón

Dirigido por:

Dr. Saúl Tovar Arriaga

Centro Universitario

Querétaro, Qro.

Junio del 2013

México

Universidad Autónoma de Querétaro
Facultad de Informática
Maestría en Ingeniería de Software Distribuido

ACCESO INTELIGENTE A BASES DE DATOS REMOTAS POR MEDIO DE AGENTES MÓVILES

TESIS

Que como parte de los requisitos para obtener el grado de

Maestra en Ingeniería de Software Distribuido

Presenta:
Ing. Rebeca Eugenia Aguilar Durón

Dirigido por:
Dr. Saúl Tovar Arriaga

SINODALES

Dr. Saúl Tovar Arriaga
Presidente — Firma

Dr. Jesús Carlos Pedraza Ortega
Secretario — Firma

Dr. Juan Manuel Ramos Arreguín
Vocal — Firma

Dr. Marco Antonio Aceves Fernández
Suplente — Firma

Dr. José Emilio Vargas Soto
Suplente — Firma

M. C. Ruth Angélica Rico Hernández
Directora de la Facultad de Informática

Dr. Irineo Torres Pacheco
Director de Investigación y Posgrado

Centro Universitario
Querétaro, Qro.
29 de Junio del 2012
México

RESUMEN

El presente trabajo trata de dar solución al problema de tiempo de acceso a bases de datos remotas, mediante la ejemplificación del uso de tecnologías para acceso a bases de datos remotas con Agentes Móviles. Para su implementación se ha utilizado un sistema de agentes móviles Aglets, basado en la tecnología Java. Se trata de comprobar, principalmente, dos aspectos de la tecnología, la mejora en cuanto al tiempo de acceso respecto a la tecnología cliente/servidor y la disminución de ancho de banda en la red. Se iniciará con una visión general de esta tecnología y sus ventajas, y más adelante se describirán las características generales de la aplicación y su funcionamiento. Finalizando en el análisis de los resultados experimentales y sus conclusiones.

(**Palabras clave**: Aglets, agentes móviles, cliente-servidor, red.)

ABSTRACT

In this paper attempts to provide a solution to the problem of access time to remote databases through the exemplification of the use of technologies to access remote databases with mobile agents. For its implementation we used a Aglets mobile agents system based on Java technology. This is test, which mostly, two aspects of technology, improvement in access time on the client-server technology and the decrease of bandwidth in the network data transmission. This starts with an overview of this technology (mobile agents with aglets) and its benefits, and later will describe the characteristics of the implementation and operation. Finishing with the analysis of experimental results and conclusions.

(**Key Words**: Aglet, mobile agents, client-server, network.)

A mi hijo Rafael

“Por su valiosa compañía y su gran paciencia”

AGRADECIMIENTOS

Primeramente a *Dios*, por ser mi compañero incondicional en cada uno de los días de mi vida, gracias por darme las herramientas suficientes para ser la persona que soy.

Gracias a mi *Madre* y a mi *Padre*+, por su amor y por su apoyo, porque me dejaron ser libre para elegir mi camino.

Un muy cariñoso agradecimiento también, a mi hijo *Rafael*, ya que fue mi compañero en la mayoría de las clases a las que tuve que asistir, y con su disposición y paciencia me dio la fortaleza que necesitaba para continuar.

A mi maestro y director de tesis *Saúl*, porque con su sencillez aprendí que todas somos personas valiosas en la vida, gracias Saúl por la confianza puesta en mi.

También un profundo agradecimiento a *Carlos Pedraza*, por su amistad y valiosos consejos.

Un agradecimiento muy especial a mis mejores amigas, *Gema* por ser un apoyo fundamental para poder terminar esta etapa de mi vida académica y *Laura,* ya que con su amistad, entusiasmo y simpatía, alegra siempre mi vida.

A todas aquellas personas que de forma indirecta me apoyaron para terminar este proyecto les doy mi más profundo agradecimiento ya que fueron parte importante para este buen término.

Í N D I C E

ÍNDICE DE CUADROS

ÍNDICE DE FIGURAS

1. Introducción

La obtención de la información de una base de datos remota en un sistema, generalmente es hecha mediante una conexión tradicional de Cliente/Servidor. En ocasiones, es suficiente con obtener esta información sin importar cuanto espacio en la red se ocupe. Pues bien, este trabajo de investigación pretende demostrar lo útil que puede ser la Tecnología de Agentes Móviles en la recuperación de la información sin sacrificar el ancho de banda de la red, ya que actualmente esta Tecnología ofrece soluciones a diversos problemas que se presentan al momento de elaborar sistemas en red.

1.1 Motivación

Los Agentes Móviles surgen de la necesidad de usar de manera efectiva la información disponible en las redes de computadoras. Forman parte de la evolución de algunos campos relacionados con la Inteligencia Artificial, el Procesamiento Distribuido así como el de la Computación Distribuida.

Con el advenimiento de la era del Internet y la globalización económica, cada vez son más las empresas que experimentan la necesidad de compartir recursos geográficamente muy distantes unos de otros. En este caso las bases de datos ocupan un lugar muy importante entre los recursos que más necesitan compartir las empresas ya que es información con la que tienen que tomar decisiones día con día. De esta necesidad se deriva el problema de la saturación del ancho de banda de la red, siendo ya un problema clave a solucionar en la actualidad, es aquí en donde intervienen los Agentes Móviles, al permitir que interactúen de forma local con bases de datos remotas.

La tecnología de Agentes Móviles soluciona (o pretende solucionar) diversos problemas referentes al alto consumo del ancho de banda en una aplicación distribuida que se produce en la red al utilizar la arquitectura Cliente/Servidor. Este ancho de banda en una aplicación distribuida es muy escaso y por tanto valioso.

1.2 Definición del Proyecto

En la actualidad cada vez son más las empresas e instituciones públicas que experimentan la necesidad de compartir recursos geográficamente muy distantes unos de otros. De estos recursos, la información almacenada en bases de datos empresariales ocupa un lugar esencial (Lange B.D., 1999). La red Internet ofrece la infraestructura adecuada para conectar estos recursos a través de una amalgama de máquinas, sistemas operativos y redes de ordenadores de diferentes tipos.

En este contexto, la saturación del ancho de banda de la red se convierte en el problema clave a solucionar. Conforme avanza la tecnología, las bases de datos son cada vez más complejas y contienen más volumen de información, especialmente aquellas bases de datos que están distribuidas en diferentes nodos. Desafortunadamente esto provoca que los tiempos de espera para la obtención de la información sean muy largos e incómodos para los usuarios. Es aquí en donde la tecnología de agentes móviles puede cumplir una función fundamental para ahorrar al usuario tiempos espera prolongados.

Específicamente, este trabajo de investigación pretende alcanzar los siguientes objetivos:

1.3 Objetivos

General:

Implementar un algoritmo de búsqueda de información en bases de datos remotas basado en tecnología de Agentes Móviles para la solicitud de información que permita al usuario consultar datos con un tiempo de espera más adecuado.

Específicos:

- Se desarrollará un estudio de los diferentes algoritmos existentes basándonos en tecnología de Agentes Móviles para la obtención de información en situaciones críticas de tráfico.

- Con base al estudio anterior se elegirá el algoritmo para cumplir con los requerimientos necesarios o en su caso se mejorará.

- Se complementará esta búsqueda con un sistema que otorgue seguridad (sistema robusto ante desconexiones del cliente) y haga un uso óptimo de los recursos de la red.

- Se propondrá una interface más adecuada que permita al usuario decidir si desea esperar en línea para obtener la información requerida o si desea desconectarse y obtener la información hasta el momento en que vuelva a acceder a Internet.

1.4 Alcance

El alcance de este trabajo de investigación es la aplicación de Agentes Móviles a búsquedas de información en bases de datos del sistema de control escolar de una Institución de Educación Superior.

1.5 Organización de la Tesis

Las siguientes líneas muestran la presentación progresiva del material de investigación.

1. Introducción

En este capítulo se hará una breve explicación del alcance, objetivos y el por qué de este trabajo de investigación.

2. Estado del Arte

Este capítulo contiene conceptos de los agentes móviles, describe las ventajas de esta tecnología comparada con otras ya existentes, así como también hace referencia a las herramientas que se pueden utilizar en la programación de Agentes Móviles.

3. Implementación de Aglets.

En este capítulo se describe de manera detallada la implementación de los Aglets, desde la instalación del software necesario, descripción del Caso de Estudio, descripción del sistema a realizar y la demostración de la ejecución de dicho sistema utilizando dos tecnologías para acceso a información remota, agentes móviles y cliente servidor.

4. Análisis de los resultados.

En este apartado se realiza una evaluación exhaustiva de los resultados obtenidos para corroborar la validez de la hipótesis hecha en un inicio de ésta investigación.

5. Conclusiones y trabajo futuro.

Se discuten los resultados obtenidos y se mencionan las conclusiones que arrojo la investigación. Se hace un análisis de las posibles medidas a tomar para mejorar el proyecto así como las líneas de investigación adecuadas para su ejecución.

2. Estado del Arte

Es importante la descripción de los conceptos de los que estará compuesto este proyecto de investigación. Esto ayudará a entender, por qué es importante hacer este trabajo. Por otro lado, analizar toda la investigación que existe alrededor de este tema, Agentes Móviles, ayudará al término de este trabajo de investigación.

2.1 Conceptos Básicos

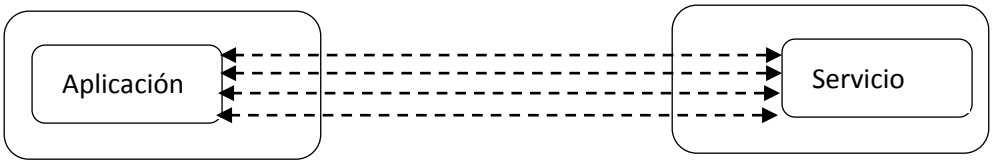
Los agentes se pueden definir como entidades de software, los cuales tienen la característica de ser inteligentes y autónomos. Dado que son inteligentes, poseen capacidades muy peculiares, tales como capacidad de comunicación con diferentes servidores, la movilidad entre diferentes nodos y por consecuencia el aprendizaje que adquieren.

Entre los tipos de agentes se tienen los llamados:

Agentes estacionarios o estáticos: Estos agentes son ejecutados solamente en el sistema donde fue iniciada su ejecución, no tienen capacidad de movilidad. Si estos necesitan información que no esté en el sistema o necesita interactuar con un agente en un diferente sistema, usa un mecanismo de comunicación típica (Grimshaw D., 2001) tal como el *Remote Procedure Call* (RPC) (Perez L., 1998).

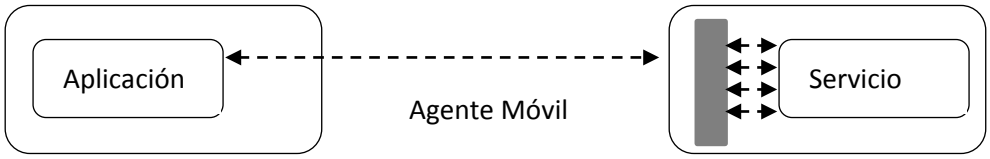
Agente Móvil: Tiene la habilidad de transportarse por sí mismo desde un sistema en la red a otra. Esta habilidad que tiene para viajar, le permite moverse a un sistema que contiene el objeto con el cual el agente quiere interactuar (Pérez L., 1998).

a)



R P C (en el contexto Cliente/Servidor)

b)



Host A Host B

Figura 1 Agentes Móviles y la reducción en la carga de la red (Lange B. D., Oshima M., 1999). Se hace una comparación entre la constante comunicación de dos nodos con RCP (sobre carga en la red) y la comunicación de ida y vuelta (menos carga en la red) entre dos nodos con Agentes Móviles.

En la figura 1 se describen dos formas de solicitar información a un servidor. En el inciso *a*, se muestra la forma más común de interactuar el cliente y el servidor, en este caso puede ser una comunicación por procedimiento remoto utilizando una arquitectura cliente/servidor. Aquí se muestra la cantidad de veces que es necesario comunicarse el cliente con el servidor para la solicitud de información. La principal característica de la llamada a procedimiento remoto (*RPC*) es que, en cada interacción entre el cliente y el servidor existen dos actos de comunicación, uno para enviar al servidor la petición y los argumentos correspondientes y otra para enviar como respuesta los resultados de la petición. Este tipo de interacción hace que se incremente de forma dramática el tráfico de la red, sobre todo, si el número de clientes y/o el número de peticiones de servicio se incrementan (Boquera E., 2003).

En el inciso *b*, se muestra como el cliente hace una solicitud y el servidor se encarga de solucionar dicha solicitud sin necesidad de llevar a cabo comunicación frecuente con el cliente, esto es utilizando agentes. En este tipo de comunicación, cuando el cliente envía una petición al servidor, no solo envía los argumentos

necesarios para la ejecución del servicio, sino que también manda el procedimiento requerido para la ejecución. Cada mensaje que la red transporta contiene no solo *n* procedimientos que la computadora receptora ejecuta sino también los datos (argumentos) necesarios para la ejecución (Pérez L., 1998).

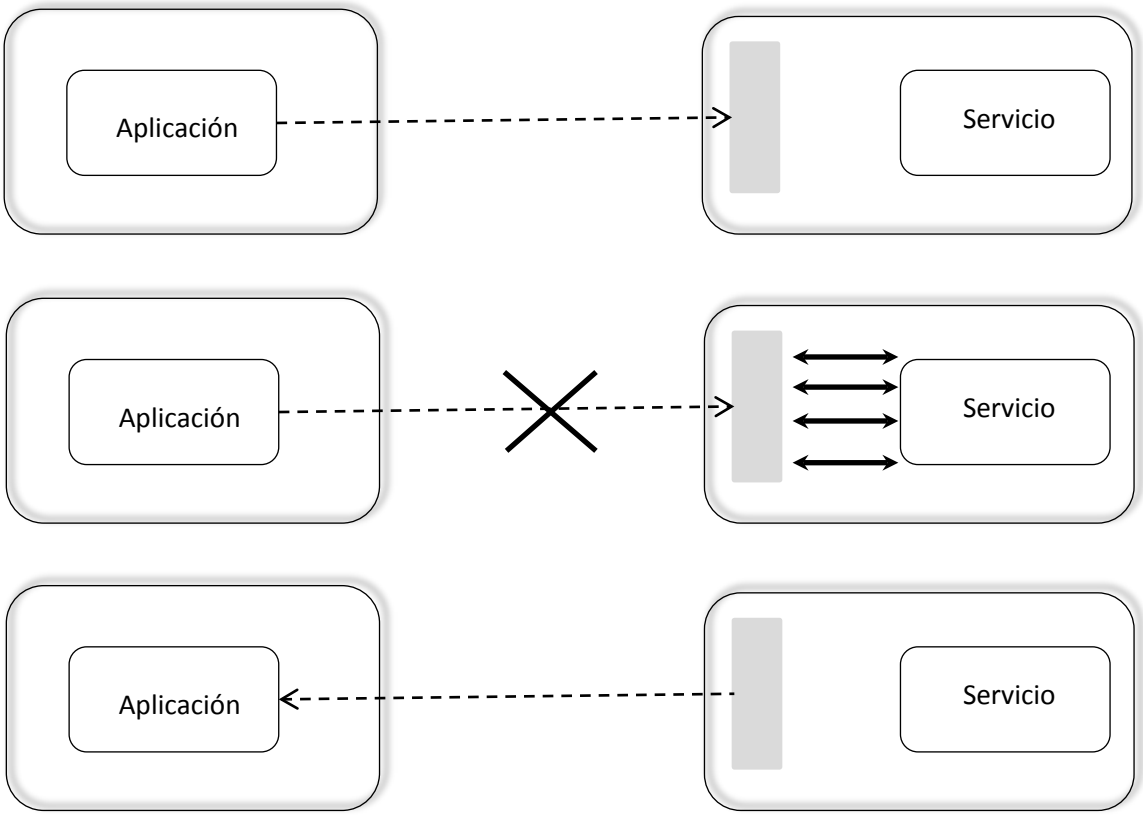


Figura 2 Operación de Agente Móvil y desconexión. Los Agentes Móviles tienen la capacidad de ejecutarse en el servidor sin necesidad de que el cliente continúe conectado. Una vez que el cliente reinicia la conexión, el Agente devuelve la información encontrada.

Cada cliente es responsable de la correcta especificación del servicio que necesita y debe ser descrito en el código enviado al servidor remoto. Por su naturaleza, un agente móvil detecta cuando una conexión de red es lenta (por ejemplo modem), y si es así, éste envía el programa al servidor para su ejecución. De esta manera no será necesario mantener una conexión permanente (figura 2) con el mismo, excepto para la transmisión del resultado final. Estas características hacen de los agentes móviles una tecnología muy atractiva para el desarrollo de nuevos sistemas.

Algunas diferencias entre agentes móviles y RPC (Lange B.D., 2000) (Braun P., 2007):

a) Los *Agentes Móviles* pueden ser inteligentes, tienen movilidad entre servidores, pueden ser autónomos, traslada su proceso al servidor.
b) Los RPC, son estáticos, no son inteligentes, hace la petición y el servidor genera un proceso para darle respuesta, no son autónomos, no se mueven entre servidores.

Para que un *Agente Móvil* sea útil, debe interactuar con su nodo y otros agentes. Debe acceder a la información que el cliente (computadora) ofrece y/o negociar con otros agentes sobre el intercambio de servicios. Los Agentes también deben ser capaces de moverse dentro de las redes heterogéneas de computadoras. Esto es posible solo si existe un marco de trabajo común para las operaciones de agentes a través de la red completa: una infraestructura de agentes estandarizada (UMBC, 2000).

Esta infraestructura debe ofrecer soporte básico para la movilidad y comunicación de agentes. También debe proteger a la computadora de accesos no autorizados y salvaguardar la integridad de los agentes, tanto como sea posible (Pérez L., 1998).

Algunas de las aplicaciones de esta tecnología son:

a) Recuperación de la Información en la red: Puede ser soportada de una forma mucho más eficiente si un agente representa un *query* (consulta) que puede moverse al lugar en donde los datos están almacenados, en lugar de tener que mover todos los datos a través de la red para revisarlos y posteriormente descartar la mayoría. Esta técnica tiene ahorro en el ancho de banda.
b) Manejo de la red: Esto es, en la detección de errores en donde están involucradas cientos de computadoras. En este caso se hace muy difícil el monitoreo, ya que esta tarea involucraría grandes cantidades de datos. En estas situaciones es factible utilizar *Agentes Móviles* que observen detalladamente el sistema enviando reportes sobre el estado de las computadoras.
c) Comercio electrónico (Castillo C., 2004): Los negocios en internet son una realidad, los agentes móviles pueden ayudar, a localizar cuales sitios ofrecen los servicios o productos más económicos.
d) En sistemas de cómputo móviles: En la actualidad las computadoras son más pequeñas y a su vez más poderosas, pero el acceso a la infraestructura de información fija se hace lento, debido a que existen restricciones en la transmisión. Para minimizar el consumo y el costo de la transmisión, los usuarios desearían no tener que permanecer en línea mientras que alguna consulta complicada está siendo ejecutada en su nombre por los recursos de cómputo fijo. Los agentes móviles ofrecen, una forma en que se puede resolver este problema de espera en línea. Los usuarios simplemente liberan un agente móvil que realizará de forma autónoma las búsquedas solicitadas. El cliente, si así lo desea, puede desconectarse o esperar, el agente regresará con la información solicitada y, si es el caso, esperará hasta que el cliente vuelva a conectarse para regresar con la información.

Existen siete razones por las que se debería usar Agentes Móviles, tales como:(Lange B.D., 1999)(Acebal C., 2000)

a) Reducen la carga de la red. Esto es, los sistemas distribuidos frecuentemente hacen comunicaciones sobre protocolos que involucran múltiples interacciones

para completar la tarea requerida. El resultado de estas tareas es una cantidad de tráfico en la red. Los Agentes Móviles permiten que la conversación y el proceso se ejecute en el servidor destino, por lo tanto las interacciones no tendrán que viajar en la red sino que serán de forma local.

b) They overcome (reducen latencia en la red). Los sistemas críticos en tiempo real, tales como los robots en los procesos de manufactura, necesitan responder en tiempo real a las diferentes peticiones del ambiente en donde se desenvuelven, en estos casos el problema de latencia de red no es aceptable. Los Agentes Móviles ofrecen una solución, porque estos pueden ser despachados desde un controlador central que actúa de manera local y directa ejecutando la dirección del control.

c) Encapsulan protocolos. Cuando los datos son intercambiados dentro de un sistema distribuido, cada servidor tiene el código que implementan los protocolos que se necesitan propiamente para el código de los datos que salen e interpreta los datos que entran, respectivamente. Los A*gentes Móviles* pueden moverse al servidor remoto para establecer "canales" basados propiamente en los protocolos de comunicación.

d) Se ejecutan de forma asíncrona y automáticamente. Frecuentemente los dispositivos móviles deberán ser confiables sobre costosas o frágiles conexiones de red. Las tareas que requieren una conexión continuamente abierta entre dispositivos móviles y una red propiamente fija, generalmente no son económicas o factibles tecnológicamente hablando. Para resolver este problema, las tareas pueden ser embebidas en un *Agente Móvil*, el cual puede ser despachado dentro de la red. Después de que es despachado, el *Agente Móvil* se crea el proceso de manera independiente y puede operar de forma asíncrona o síncrona.

e) Se adaptan de forma dinámica. Los *Agentes Móviles* tienen la habilidad identificar el ambiente de ejecución y reaccionar de forma autónoma a los cambios. Múltiples *Agentes Móviles* poseen la única habilidad para distribuirse ellos mismos entre los diferentes servidores de la red, así como mantener la configuración óptima para resolver un problema en particular.

f) Son naturalmente heterogéneos. En las redes de computadoras en fundamental la heterogeneidad, desde el punto de vista de hardware así como de software. Los *Agentes Móviles* son ejecutados, independientemente de del ambiente y las condiciones de las plataformas de software y hardware.

g) Son robustos y tolerantes a fallos. Los *Agentes Móviles* poseen la habilidad de reaccionar de forma dinámica a situaciones poco favorables.

Los *Agentes Móviles* pueden revolucionar el diseño y desarrollo de sistemas distribuidos (Lange B.D., 1999). Para poner esto dentro de una perspectiva, se hará la comparación de 3 paradigmas de programación para componentes (Mohamed A.T., 2007) distribuidos: cliente/servidor, código sobre demanda y *Agente Móvil*.

Paradigma Cliente/Servidor: (Lange B.D., 1999)

En este paradigma, un servidor ofrece un conjunto de servicios que dan acceso a algunos recursos (tal como bases de datos). El código que implementan esos servicios son hospedados localmente por el servidor. Finalmente es el servidor quien ejecuta los servicios y tiene las capacidades de procesar. Si el cliente está interesado en acceder a un recurso de almacenado en el servidor, el cliente simplemente usará uno o más de los servicios que provee el servidor. Es importante mencionar que el cliente necesitará alguna inteligencia para decidir cuál de los servicios usará. El servidor tiene todo esto: el conocimiento de recursos y procesador. De manera remota los sistemas distribuidos tendrán que estar basados en este paradigma (figura 3).

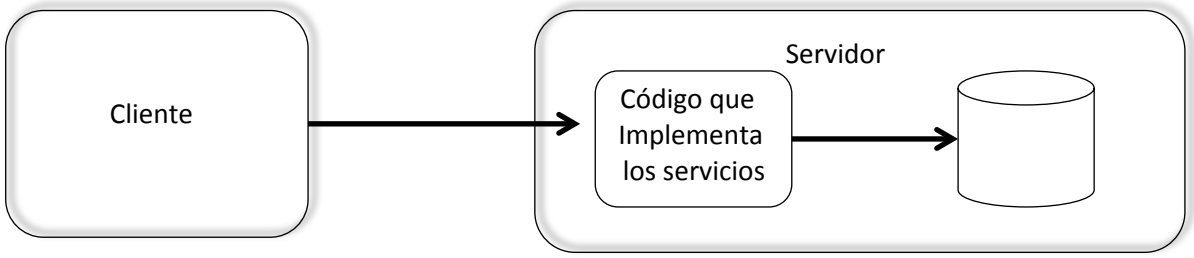


Figura 3 Paradigma Cliente / Servidor

Paradigma Código sobre Demanda. (Lange B.D., 1999)

Se supone que el cliente inicialmente esta deshabilitado para ejecutar esta tarea ya que le falta código. Un host en la red provee el código necesario. Uno de los códigos es recibido por el cliente, el cálculo es realizado fuera del cliente. El cliente tiene la capacidad de procesar así como los recursos locales. En contraste con el paradigma clásico cliente-servidor, el cliente no necesita preinstalar el código, todo el código necesario deberá ser descargado. Se dice que el cliente tiene los recursos y procesador y el host tiene el código de implementación. Los applets de java y servlets son un ejemplo clásico de este paradigma. Los applets son descargados en la web del navegador y son ejecutados localmente, mientras que los servlets son descargados de forma remota del servidor web y ejecutados ahí (figura 4).

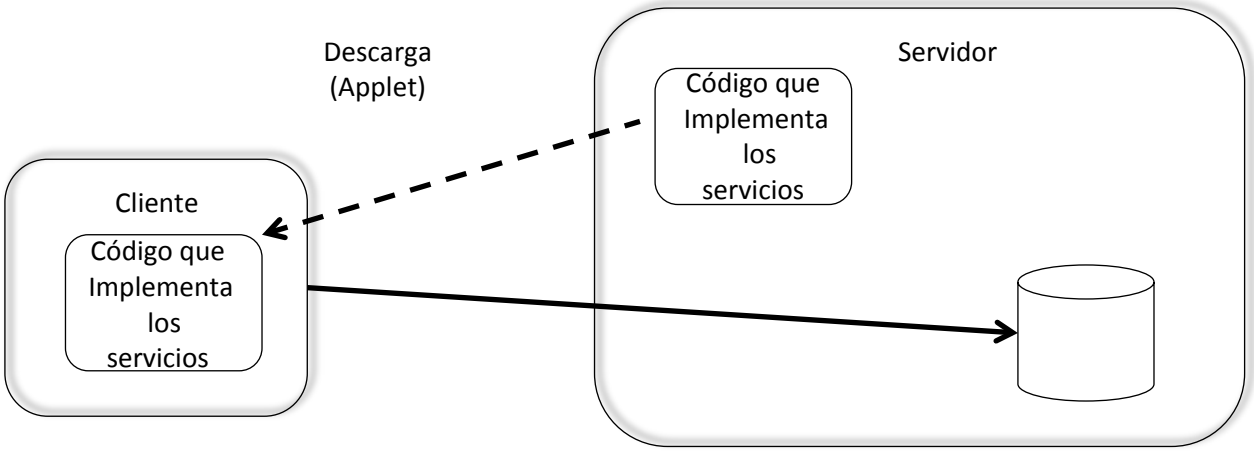


Figura 4 Paradigma Código sobre Demanda

Paradigma de Agente Móvil.(Lange B.D., 1999)

Una característica clave del *Agente Móvil* es que cualquier host en la red tiene un alto grado de flexibilidad y posee cualquier mezcla de código de implementación, recursos y procesador. Esta capacidad de procesamiento puede ser combinada con recursos locales. El código de implementación (en forma de *Agente Móvil)*, este no está ligado a un simple host pero está disponible para cualquier parte de la red (figura 5).

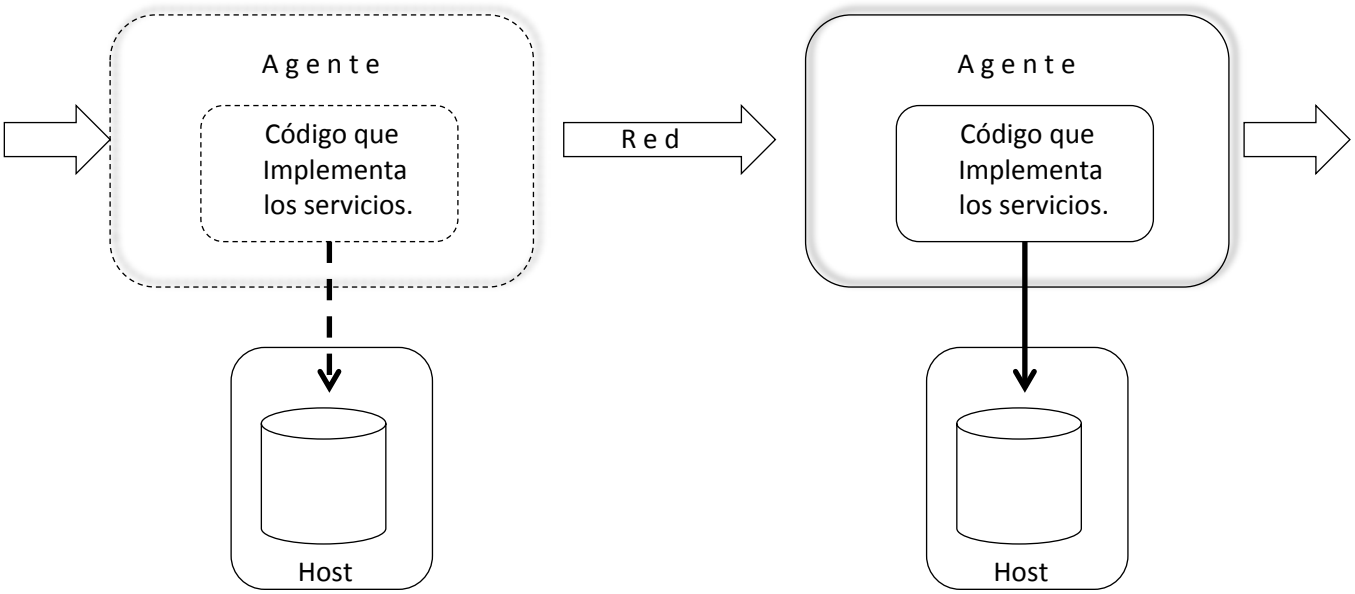


Figura 5 Paradigma de Agente Móvil

Si se comparan estos tres paradigmas, se notará la tendencia cronológica hacia una mayor flexibilidad. El cliente y el servidor se fusionan y se convierten en un host (Lange B.D., 1999).

Un *Agente Móvil*, es un programa que es emitido por una computadora (cliente) u ordenador fuente. Este programa tiene la habilidad de recorrer una serie de servidores en la red con la finalidad de ejecutar una tarea específica. Unas de las características de estos programas es que es completamente autónomo (Solve real problems with aglet, type of mobile agent 1997) e independiente a la computadora que lo emitió, sin dejar de considerar que ésta puede continuar comunicándose con el agente. De esta forma, cuando el agente termina su tarea, regresa al cliente que lo originó con sus resultados.

Entre las características de los *Agentes Móviles* encontramos las siguientes (Barba-Matrtí A., 2002):

a) Es un programa que actúa en nombre de otra entidad.

b) Es autónomo y orientado a objetivos.
c) Tiene la capacidad de trasladarse de un nodo a otro dentro de una red de comunicaciones.
d) Es considerado como un proceso informático ya que consta de un código, estado y datos que va recogiendo a lo largo de su proceso.

Algunas de las ventajas del utilizar este tipo de *Agentes Móviles* son (Barba- Matrtí A., 2002), (Roa O.L., 2005), (Acebal C., 2000):

a) Los agentes pueden continuar recorriendo la red y operando en nombre de los usuarios aún si no se encuentran conectados (operación asincrónica).
b) Reducen el tráfico en la red con respecto a la conexión de cliente estática.
c) Su flexibilidad facilita y acelera el diseño de aplicaciones a medida de los usuarios comparado con el cliente/servidor que resulta estático para poder adaptarse rápidamente a los cambios.
d) Le reducen la carga a los clientes ya que la búsqueda se hace directamente en el servidor.
a) Se puede gestionar el análisis, control del tráfico y rendimiento de una forma proactiva.
b) Administra el ancho de banda

En cuanto al funcionamiento básico de un sistema de *Agentes Móviles* este es el siguiente (Pérez L., 1998):

- Primero, un agente es lanzado desde una aplicación de un cliente que se encuentra en ejecución en un dispositivo del usuario.
- El cliente puede emitir el destino (Servidor) hacia donde se dirigirá el agente aunque también es posible dar una lista de servidores hacia donde se dirigirán los agentes a realizar la tarea especificada.
- El agente recorrerá los servidores indicados, semejante a una ruta, hasta lograr sus objetivos.

- Cuando un servidor recibe a un agente, este es transferido hacia el entorno de ejecución de agentes, una vez que esta en este entorno, este es autentificado. Después de la validación el entorno de ejecución examina la descripción del agente y se evalúan las solicitudes de servicio que contienen, con la finalidad de determinar si estos se encuentran disponibles en el servidor.
- Una vez que el agente finaliza su tarea, recopila su estado y la información que obtuvo en el servidor y solicita que lo transporten a otro nodo de la red.
- Finalmente el agente regresa la información obtenida al cliente que lo originó.

2.2 Herramientas de programación para *Agentes Móviles*.

Los *Agentes Móviles* suelen programarse normalmente en lenguajes interpretados, -Java, Telescrip- ya que éstos dan un mejor soporte a entornos heterogéneos, permitiendo que los programas y sus datos sean independientes de la plataforma utilizada.

Para la implementación de *Agentes Móviles* se tomará como base el modelo de lenguaje de programación de Java. Estos *Agentes Móviles* se implementan como objetos Java con capacidad de trasladarse de una computadora a otra para su ejecución, transportando, para lo anterior, su estado, su código y los datos que recopila. En cuanto a las plataformas utilizadas para su ejecución, esencialmente se trata de máquinas virtuales (JVM *Java Virtual Machine*) (Venners Bill, 1999) con funcionalidades adicionales, entre las que podemos mencionar, escuchar los puertos de conexión a la red para detectar la llegada de los agentes. Algunos ejemplos de sistemas con las característica necesarias para ejecutar *Agentes Móviles* están las siguientes tecnologías: Aglet, Concordia y Voyager (UMBC, 2000).

Aglet: (AGent+appLETS) (De Ockham G., 2005)

- Diseñado por Dany B. Lange (1996) en el laboratorio de investigación de IBM en Tokyo (IBM Tokyo Research Laboratory).
- Aglets Software Development Kit (ASDK o Aglets Workbench):http://sourceforge.net/projects/aglets.

- Libre distribución.
- Interfaz de programación basado en Java.
- Servidor de aglets (Tahiti).
- Posiblemente uno de los sistemas abiertos mas conocidos y utilizados actualmente.
- No proporciona soporte.
- Un aglet es un objeto móvil escrito en Java.

Concordia: (De Ockham G., 2005) (Franco-Borré D.A., 2008)

- Desarrollado por Mitsubishi Electri Research Laboratorioes (Cambridge, Massachusset) en 1998: (De Ockham G., 2005).
- Basado en lenguaje Java.
- Cumple con las normas específicas en MASIF por el grupo OMG.
- Incorpora mecanismos de seguridad.

Voyager: (De Ockham G., 2005)

- Desarrollado por la empresa ObjectSapce, Incorporated (Dallas, Texas) en 1998:(Roa O.L., 2005).
- Basado en el lenguaje Java.
- Intenta aglutinar las ventajas de Aglet, Mole y Odissey.
- Soporta movilidad de objetos y agentes móviles para el desarrollo de aplicaciones distribuidas.
- Interfaz de transporte: Java RMI,DCOM,CORBA IIOP.

En este proyecto de investigación se programarán las tareas de los *Agentes Móviles* utilizando *Aglets*. La plataforma de Aglets Software Development Kit (ASDK) presenta una herramienta de desarrollo precisamente para *Agentes Móviles*, esta fue creada por los laboratorios de IBM en Tokio (UMBC, 2000)(León S.S., 2005). El motivo de haber elegido esta plataforma es el hecho de que se cuenta con una base de conocimiento de la misma, aunado al hecho de que es una de las más conocidas y tradicionales para este tipo de soluciones.

2.3 Aglets y Aplicaciones

Un Aglet es un objeto Java el cual puede moverse de forma autónoma y espontánea de un servidor a otro trasladando su código a cada servidor. Puede ser programado para ejecutarse de forma remota mostrando diferentes comportamientos según el servidor. Este objeto permite (Agentes Móviles y CORBA, 1999):

- Cifrar el código y los datos de un Aglet utilizando el método de seriación de Java (JOS).
- Trasladar a agentes utilizando el protocolo para el transporte de Aglets (ATP).
- Ofrecer una interfaz de programación para Aglets (A-API).
- Interconexión e intercambio de información entre Aglets y otros objetos mediante paso de mensajes.
- El ciclo de vida de un Aglet puede tratarse por métodos basados en captura de eventos. Los eventos definidos son: (Lange B.D., 1999)

  Create(): Un Aglet es creado dentro de un contexto. Cuando esto sucede, su hilo de ejecución empieza y al agente se le da un número de identificador propio.

  Clone(): Este proceso copia el Aglet original con un identificador distinto. Debido a limitantes de Java, no es posible para el agente clonado reflejar el estado del hilo de ejecución, por lo que se ejecuta desde el principio.

  Dispose():Este método puede ser usado cuando el trabajo de un agente finaliza. Este proceso termina la ejecución y remueve el Aglet del contexto. Si no quedan referencias será sujeto del recolector de basura.

  Retract(): Un Aglet es removido del contexto a donde fue despachado y retorna a su lugar de origen. Al ser transferido su ejecución empieza de nuevo.

  Dispatch(): Un Aglet es removido de su contexto actual y enviado a otro. En el nuevo lugar su ejecución empieza de nuevo.

  Activate(): Cuando el ciclo de suspensión de un Agente expira, es activado y su ejecución comienza de nuevo.

Deactivate(): La ejecución de un Agente es suspendida y su estado es transferido a un lugar de almacenamiento secundario. El agente no es removido sino que se queda dormido por el tiempo especificado.

- Control de seguridad mediante definición de autoridades y de sus privilegios y preferencias.

Para tener acceso y controlar un Aglet, es necesario usar su proxy, (basado en el patrón de diseño que lleva el mismo nombre). Al crear un Aglet no se obtiene como tal un objeto de este tipo sino una referencia a su proxy.

La figura 6 y figura 7, muestra un diagrama de arquitectura y un diagrama físico del sistema Aglets respectivamente (Agentes Móviles y CORBA, 1999).

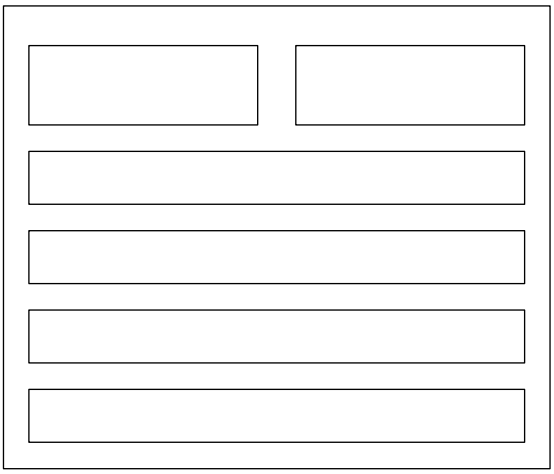


Figura 6 Arquitectura de la Plataforma Aglets

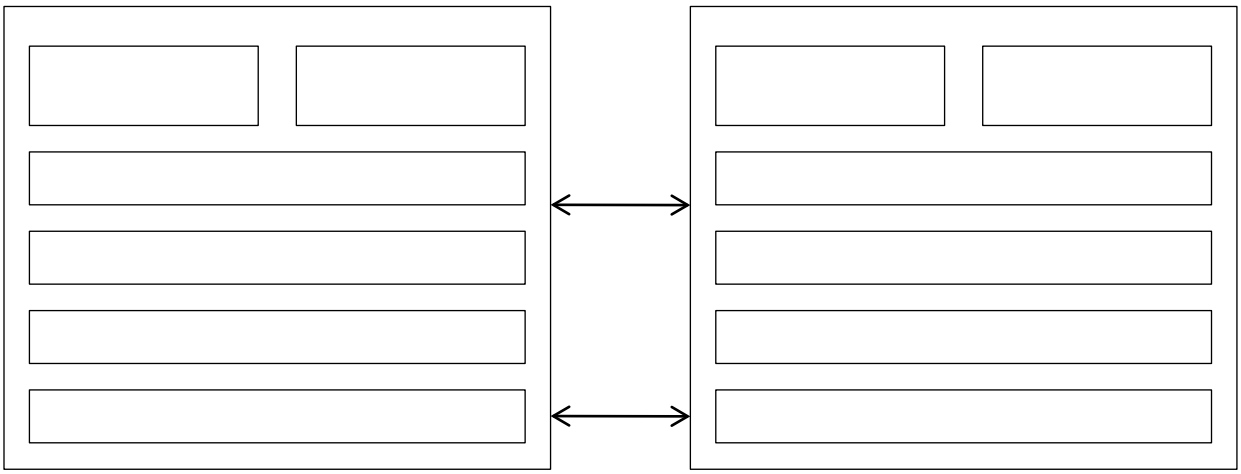


Figura 7 Diagrama Físico de la plataforma Aglets

Algunas de las características de los Aglets son (Agentes Móviles y CORBA, 1999):

- Comunicación entre agentes mediante mensajes sincrónicos y asincrónicos.
- Un esquema global único para agentes.
- Un itinerario de viaje, para la especificación de patrones complejos de viajes con múltiples destinos y manejos de fallas automáticos.
- Un mecanismo white-board que permite que múltiples agentes colaboren y compartan información asíncronamente.

- Un esquema de transmisión de mensajes que soporta una unión asíncrona desahogada tan bien como una comunicación síncrona entre agentes.
- Una carga de clases dinámica que permite que el código Java de los agentes y la información de su estado viajen a través de la red.
- Un contexto de ejecución que proporciona un ambiente independiente del sistema actual sobre el cual se están ejecutando.

En la figura 8 se describen los principales elementos de la arquitectura Aglet son (Agentes Móviles y CORBA, 1999), los cuales son:

- Aglet: define los métodos básicos para el control de un *Agente Móvil*.
- Representante: aísla el Aglet del entorno, permitiendo un mayor grado de seguridad y transparencia respecto a la localización del agente.
- Contexto: ofrece al Aglet una interfaz con su entorno de operación.
- Mensaje: objetos utilizados para la comunicación entre Aglets.

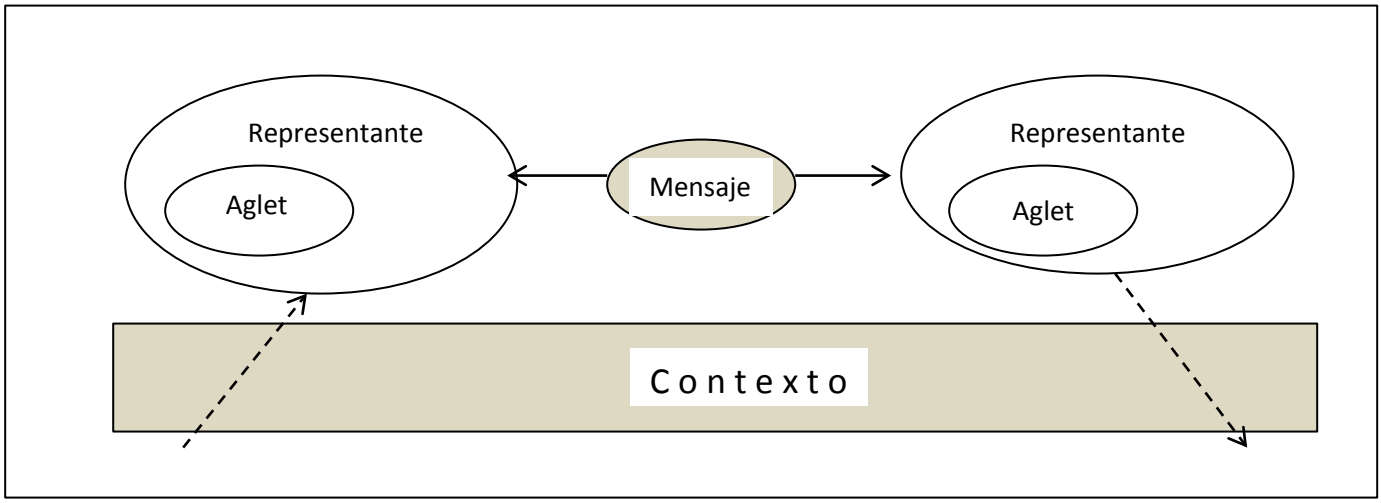


Figura 8 Arquitectura de Aglet (Lange B.D., 1999). Se muestran los principales elementos que necesita un Aglet para su ejecución.

Como los Aglets se basan en programación orientada a objetos, estos poseen clases e interfaces para la creación de *Agentes Móviles* en Java. Estas clases se encuentran en el paquete llamado *com.ibm.Aglet* en Java.

Para el caso de estudio se adaptará la herramienta implementada con Aglets en el que se podrán hacer consultas de datos académicos (calificaciones, maestros, historial académico y materias). Estos datos se encuentran albergados en una base de datos de una institución de educación superior.

Esta consulta tendrá tres elementos principales, el usuario, el administrador y el agente mensajero (*Agente Móvil*). El usuario es el que trabaja con la aplicación y solicitará la ejecución de un servicio. Éste proporcionará a los agentes los datos necesarios dependiendo del servicio a ejecutar. El administrador es el encargado de la configuración del sistema, administrará a los usuarios y los servicios, ingresándolos, modificándolos y eliminándolos. Por último el agente mensajero es el encargado de realizar las peticiones del cliente basándose en los datos de entrada que le fueron proporcionados.

La figura 9 muestra el diagrama de una consulta a base de datos remota mediante un *Agente Móvil* (Agentes Móviles y CORBA,1999),(Fonnegra M.N., 2002) .

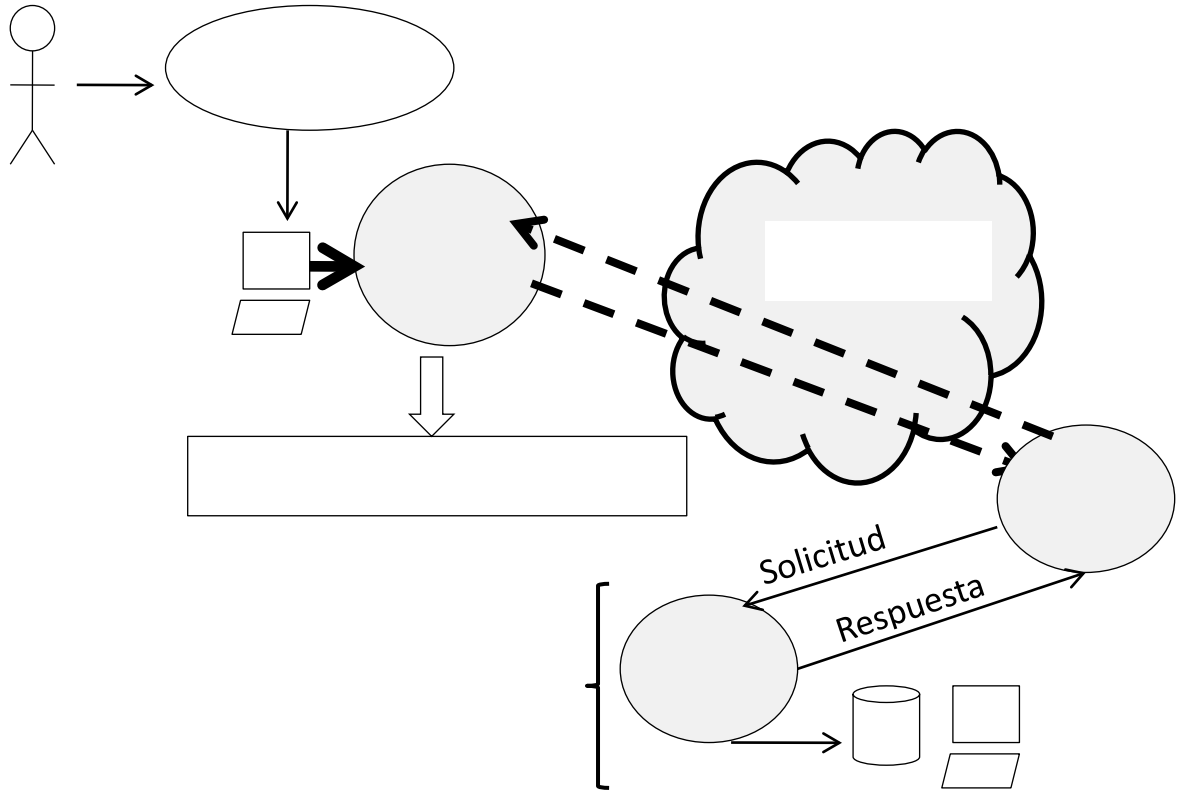


Figura 9 Diagramas de una consulta remota mediante Agentes. Esta figura representa la ejecución de un agente a un servidor remoto.

2.4 Tecnología Cliente / Servidor.

En este punto se hará una descripción de la tecnología Cliente/Servidor, ya que en este trabajo pretende hacer una aplicación para comparar dos tecnologías, y una de ellas es precisamente Cliente / Servidor (CS).

En el mundo de las TCP/IP, las comunicaciones entre computadoras se rigen básicamente por lo que se llama tecnología Cliente / Servidor, este es un modelo que intenta proveer usabilidad, flexibilidad, interoperabilidad y escalabilidad en las

comunicaciones (Caballe S., 2006). El término Cliente / Servidor fue usado por primera vez en 1980 para referirse a PC's en red.

El modelo CS empezó a ser aceptado a finales de los 80's (Caballe S., 2006), su funcionamiento es sencillo: se tiene una máquina cliente, que requiere un servicio de una máquina servidor, y éste realiza la función para la que está programado (nótese que no tiene que tratarse de máquinas diferentes; es decir, una computadora por sí sola puede ser ambos, cliente y servidor, dependiendo del software de configuración).

Desde el punto de vista funcional, se puede definir la computación Cliente/Servidor como una arquitectura distribuida (Boger M., 2001) que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma (Caballe S., 2006).

En este modelo, el cliente envía un mensaje solicitando un servicio determinado a un servidor (hace una petición o varias), y éste envía uno o varios mensajes con la respuesta (provee el servicio). Este modelo se puede ver en la figura 10:

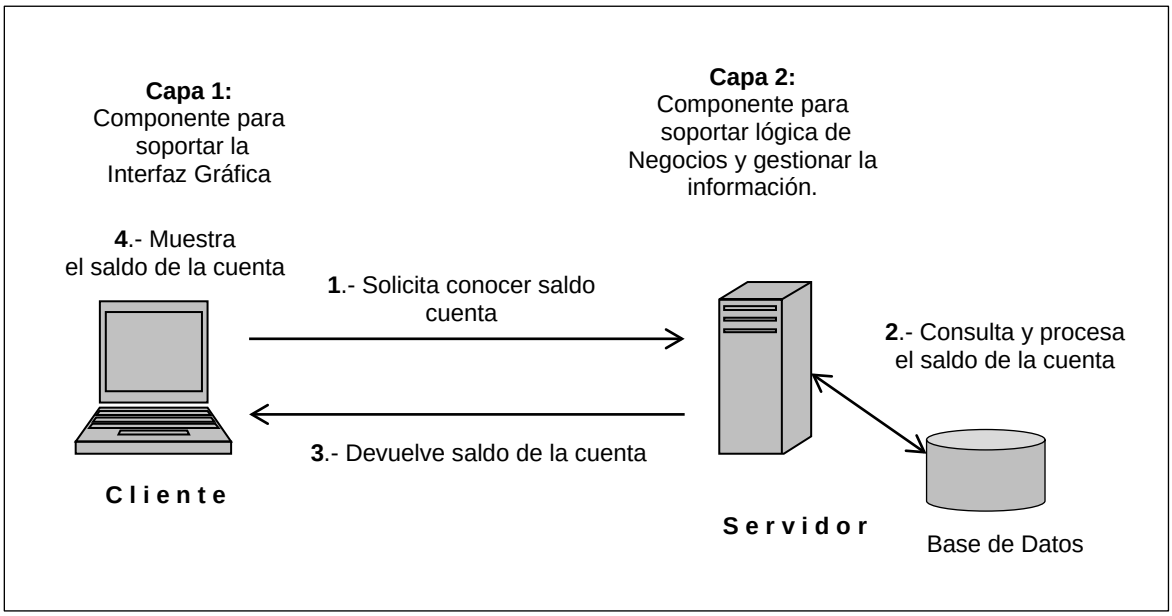


Figura 10 Tecnología Cliente/Servidor

Cliente

El cliente es el proceso que permite al usuario formular los requerimientos y pasarlos al servidor, se le conoce con el término de *front-end* (Sommerville I., 2004).

El cliente normalmente maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de la red.

Las funciones que lleva a cabo el cliente son las siguientes:

- Administrar la interfaz de usuario.
- Interactuar con el usuario.
- Procesar la lógica de la aplicación y hacer validaciones locales.
- Generar requerimientos de bases de datos.
- Recibir resultados del servidor.
- Dar formato a resultados.

Servidor

Es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. El proceso servidor se le conoce con el término de *back-end* (Sommerville I., 2004). El servidor maneja, generalmente, todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos.

Las funciones que lleva a cabo el proceso de servidor se resume en los siguientes puntos:

- Aceptar los requerimientos de bases de datos que hacen los clientes.

- Procesar requerimientos solicitados por los cliente (acceso a bases de datos, imprimir, correo, etc).
- Formatear datos para transmitirlos al cliente.
- Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

Las características básicas de una arquitectura Cliente / Servidor son (Sommerville I., 2004):

- El proceso del cliente proporciona la interfaz entre el usuario y el resto del sistema. El proceso del servidor actúa como un motor de software que maneja recursos compartidos tales como, bases de datos, impresoras, módems, etc.
- Las tareas del cliente y del servidor tienen diferentes requerimientos en cuanto a recursos de cómputo, como velocidad del procesador, memoria, velocidad y capacidades del disco, así como input-output dispositivos.
- Existe una clara distinción de funciones basada en el concepto de servicio que se establece entre clientes y servidores.
- La relación establecida puede ser de muchos a uno, en la que un servidor puede dar servicio a muchos clientes, regulando su acceso a recursos compartidos.
- No existe otra relación entre clientes y servidores que no sea la que se establece a través del intercambio de mensajes entre ambos. El mensaje es el mecanismo para la petición y entrega de solicitudes de servicio.
- El ambiente es heterogéneo. La plataforma de software y hardware del cliente y servidor no siempre son los mismos. Precisamente una de las principales ventajas de esta arquitectura es la posibilidad de conectar clientes y servidores independientemente de sus plataformas.
- El concepto de escalabilidad tanto horizontal como vertical es aplicable a cualquier sistema Cliente/Servidor. La escalabilidad horizontal permite agregar mas estaciones de trabajo activas sin afectar significativamente el

rendimiento. La escalabilidad vertical permite mejorar las características del servidor o agregar múltiples servidores.

Entre las ventajas y desventajas de la tecnología Cliente/Servidor se encuentra lo siguiente (Sommerville I., 2004).

Ventajas	Desventajas
1. Uso de plataformas de bajo costo.	1. El mantenimiento a los sistemas es mas difícil pues implica la interacción de diferentes partes de hardware y software.
2. Uso de componentes de hardware y software de diferentes fabricantes que favorece la flexibilidad en la implantación y actualización de soluciones.	2. Son necesarias estrategias para el manejo de errores y para mantener la consistencia de los datos.
3. Facilita la integración entre sistemas diferentes compartiendo la información.	3. Se deben hacer verificaciones tanto en el cliente así como en el servidor para proporcionar seguridad en los accesos.
4. Es rápido el mantenimiento y desarrollo de aplicaciones.	4. El desempeño es un problema en este tipo de conexiones ya que se pueden presentar por congestión de tráfico en la red.
5. La estructura inherente modular facilita la integración de nuevas tecnologías y el crecimiento de la estructura computacional, favoreciendo a la escalabilidad de las soluciones.	

Cuadro 1 Ventajas y Desventajas de la Tecnología Cliente/Servidor

2.5 Métricas para la evaluación del software.

Dentro del contexto de la Ingeniería de Software, una medida proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o

tamaño de algunos de los atributos del proceso o producto. El *Standard Glossary of Software Engineering Terms*, define métrica como <<una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado>> (Pressman R.S., 2002).

En este trabajo se hará una comparación entre dos tecnologías, para ambas se desarrollará una aplicación. La métrica de software utilizada para la comparación (o medición) de las dos aplicaciones, es la medición de software llamada Punto de Función.

La métrica orientada a la Función, utilizan una medida de la funcionalidad entregada por la aplicación como un valor de normalización. Los puntos de función se derivan con una relación empírica según las medidas contables (directas) del dominio de información del software y las evaluaciones de la complejidad del software.

Los puntos de función se calculan en base a 5 características de dominio de información, y se pondera la respuesta en base a tres factores (simple, medio, complejo), como se muestra el la figura 11 (Pressman R.S., 2002):

Parámetro de medida	**Cuenta**	**Factor de Ponderación**				**Resultado**
			Simple	**Medio**	**Complejo**	
Número de entradas de Usuario		X	**3**	4	6	
Número de salidas de usuario		X	4	5	7	
Número de peticiones al usuario		X	3	4	6	
Número de archivos		X	7	10	15	
Número de interfaces externas		X	5	7	10	
Cuenta - total	----------→					

Figura 11 Cálculo de puntos de función

Los valores de dominio de información se definen de la siguiente forma:

Número de entradas de usuario: Se cuenta cada entrada de usuario que proporciona diferentes datos orientados a la aplicación.

Número de salida de usuarios: Se cuenta cada salida que la aplicación le proporciona al usuario de información.

Número de peticiones de usuario: Es una entrada interactiva que produce la generación de alguna respuesta del software inmediata en forma de salida interactiva.

Número de Archivos: Son las tablas que interactúan con la aplicación.

Número de interfaces externas: Son todos aquellos dispositivos que se utilizan para transmitir información a otro sistema.

Una vez que se han capturado los datos en la tabla, se procede ha hacer el conteo, el cual estará conformado por la sumatoria de la última columna (resultado).

Para calcular el punto de función (PF), se utiliza la relación siguiente:

PF = Cuenta-Total X (0.65 + 0.01 X Fi)

En donde:

Cuenta-Total es la suma de todas las entradas PF obtenidas en la figura 10.

Fi es la sumatoria de los valores de ajuste de complejidad. Estos valores se muestran en la siguiente figura (Pressman R.S., 2002):

0	1	2	3	4	5
Sin influencia	Incidental	Moderado	Medio	Significativo	Esencial
1.- ¿Requiere el sistema copias de seguridad y recuperación fiables?					
2.- ¿Se requieren comunicaciones de datos?					
3.- ¿Existen funciones de procesamiento distribuido?					
4.-¿Es crítico el rendimiento?					
5.- ¿Será ejecutado el sistema en un entorno operativo existente y fuertemente utilizado?					
6.- Requiere el sistema entrada de datos interactiva?					
7.- ¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas o variadas operaciones?					
8.- ¿Se actualizan los archivos maestros de forma interactiva?					
9.- ¿Son complejas las entradas, las salidas los archivos o las peticiones?					
10.- ¿Es complejo el procesamiento interno?					
11.- ¿Se ha diseñado el código para ser reutilizable?					
12.- ¿Están incluidas en el software la conversión y la instalación?					
13.- ¿Se ha soportado el sistema para soportar múltiples instalaciones en diferentes organizaciones?					
14.- Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizadas por el usuario?					
Resultado ----------→					

Figura 12 Valores de ajuste de complejidad

Cada una de las preguntas de la figura anterior es respondida en un rango de escala desde 0 (no importante o aplicable) hasta 5 (absolutamente esencial).

Una vez calculados los PF se utilizan como una forma de normalizar las medidas de productividad, calidad y otros atributos del software.

3. Implementación de Aglets

Este capítulo describirá de manera práctica la investigación hecha en este trabajo. Se describirán los elementos relacionados con la aplicación que se desarrolló para comprobar la hipótesis referente a *Agentes Móviles*, tales como Sistema Operativo utilizado, herramienta de programación, descripción de la instalación del servidor *Tahiti*, entre otros. Y se mostrarán las interfaces de dicha aplicación que demostrarán los tiempos de acceso a la información en las dos tecnologías que se están comparando.

3.1 Arquitectura del Sistema

Este proyecto demuestra que la Tecnología Aglets puede utilizarse en la consulta masiva de información remota sin incrementar el uso de ancho de banda de la red y en un tiempo de respuesta menor.

Para probar esta tecnología se desarrolló un sistema con una arquitectura como la de la figura 6, en donde se muestra la configuración final del sistema. Es importante mencionar cuáles son los principales componentes que necesita dicho sistema:

1) *Sistema Operativo*: Se utilizará un Sistema Operativo Linux, distribución Ubuntu versión 10, debido a que es un sistema operativo abierto y no tiene ningún costo adquirirlo. Cabe mencionar que se puede utilizar cualquier sistema operativo (Windows, Unix, Linux, OS/2), debido a que los Aglets se programan en Java y la cual es una herramienta de programación de sistemas abiertos y de libre adquisición.
2) *Gestor de Base de Datos MySQL 5*: Se utilizará este gestor de bases de datos dado que es un software libre. Otro motivo por el cual se eligió este gestor, es por el hecho de que la mayoría de las herramientas de programación la soportan.

3) *Servidor Tahiti*: Este es un software indispensable para el manejo de los Aglets, la versión a utilizar es la 2.0.2. Este servidor almacenará todos aquellos Aglets que viajarán para la búsqueda de la información así como también aquellos que son estáticos. Este servidor es necesario que esté instalado tanto en la parte del cliente así como en la del servidor.
4) *Aglet Principal*: Este es un programa en Java que utiliza las API´s (los paquetes de Java) de Aglet. Este es el agente principal, el cual será estático ya que solo enviará la orden para que el agente móvil vaya a buscar la información en el servidor.
5) *Aglet de Búsqueda*: También es un programa en Java que utiliza las API´s de Aglets. Este es un agente móvil el cual tendrá la función de desplazarse a las diferentes bases de datos que se encuentran en diferentes servidores, para hacer la búsqueda de la información requerida.
6) *Driver JDBC (Java Data Base Connection)* (Papastavrou S.,, 2000)*:* Este es una clase en Java necesaria para que el Aglet pueda tener conexión con Base de Datos.

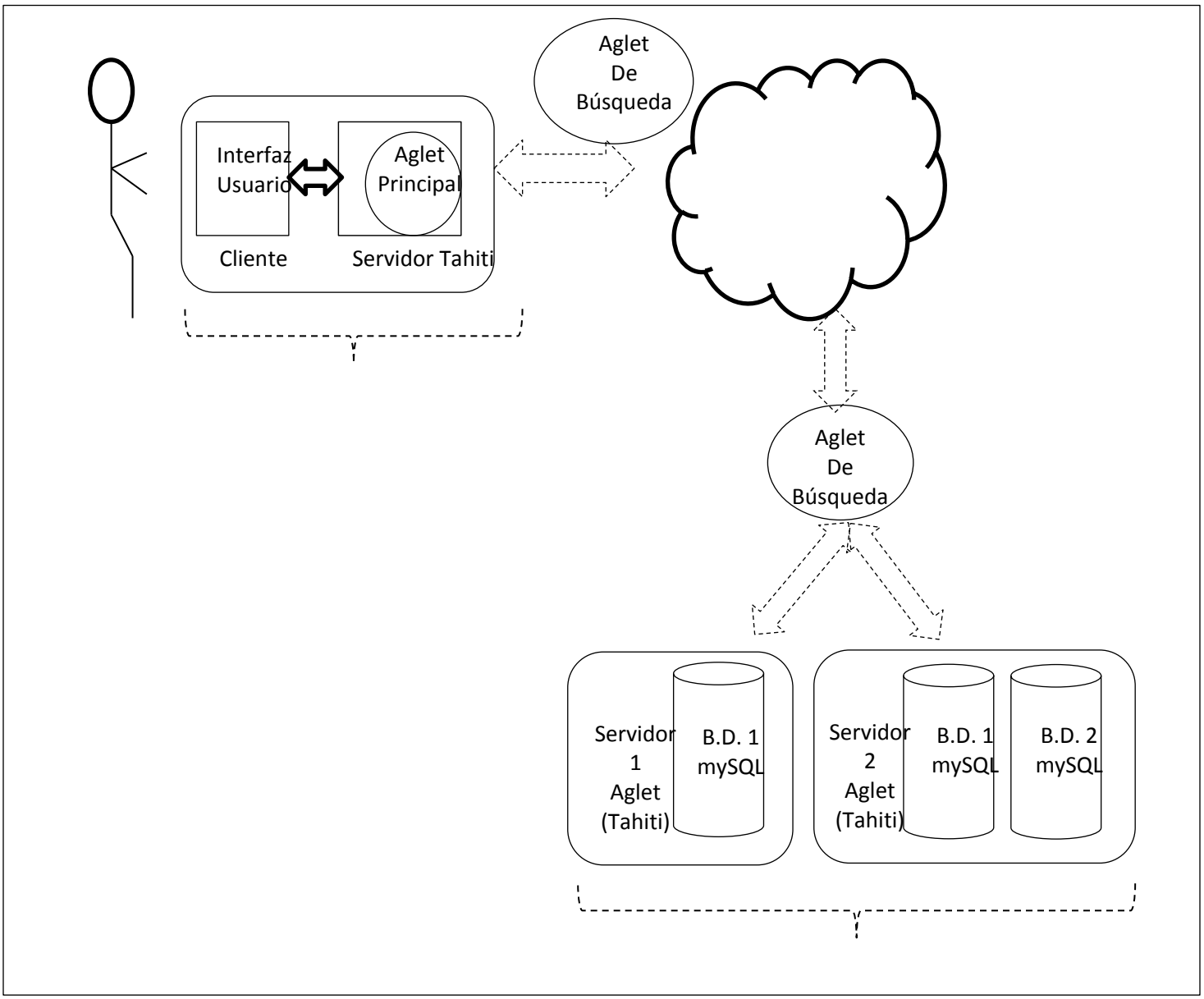


Figura 13 Arquitectura del Sistema. El sistema propuesto para este proyecto está compuesto por un Agente Estático y un Agente de Búsqueda (Móvil). El primero no tiene movilidad ni inteligencia para viajar en la red, por lo tanto será el principal, el segundo será activado por el primero (estático), este segundo es el agente que viajará por la red y hará la búsqueda en la base de datos, así como devolverá la información al primer agente.

La figura 13 muestra la forma en que operará el sistema, esto es, el usuario solicitará la consulta mediante una interfaz, esta consulta ejecutará un *aglet*, que hace las funciones de agente principal, es importante mencionar que este agente es estático (estacionario) en la propia máquina del cliente, es decir, un *Aglet* que no puede ser despachado a otros servidores y cuya misión es la de proporcionar una interfaz de comunicación entre la aplicación cliente y otros *Aglet* creados por este (a los que se le llamarán *Aglet* hijo) que son los llamados *Agentes Móviles*. El *agente principal* manda ejecutar al A*gente Móvil*, que es el *Aglet* que contiene la programación necesaria para la búsqueda de la información en las diferentes bases de datos. Esta búsqueda la hace de manera remota, en el servidor de datos, en el cual está instalado el servidor *Tahiti*, que se utiliza para la ejecución de *Agentes Móviles*. Una vez que el *Agente Móvil* encuentra o no, la información solicitada, regresa un mensaje al *agente principal* (que se encuentra en el cliente) ya sea con la información solicitada o con un mensaje de que no encontró dicha información.

3.2 Características del Hardware

No es necesario que los equipos cuenten con características especiales, en realidad la configuración tanto del cliente como del servidor puede quedar como lo muestra el cuadro 2:

	Cliente	**Servidor**
Nombre Equipo	Raguilarptc	UPJR
IP	192.168.1.70	192.168.1.252
HD	100 GB	500 GB
RAM	1 GB	3 GB
Sistema Operativo	Ubuntu Ver 10 Windows XP	Ubuntu Ver 11
Servidor instalado	Tahiti (Servidor de Aglets)	Tahiti (Servidor de Aglets)
Aglet Instalado	Aglet Principal	Aglet de Búsqueda

Cuadro 2 Características de hardware en donde se ejecutarán los Agentes

3.3 Instalación de Software

De todo el software necesario para la ejecución de agentes, solo se explicará la instalación del servidor *Tahiti*, ya que el software restante es de uso cotidiano y por lo que no es necesario que se detalle su instalación. Es importante aclarar que esta instalación es en una plataforma de Linux. Para un Sistema Operativo Windows cambia un poco esta instalación.

Para la instalación del Servidor *Tahiti* se llevaron a cabo los siguientes pasos (Manual de Agentes Móviles ,1999):

1) Descomprimir el archivo *aglet-2.0.2.jar*

   jar xvf aglet-2.0.2.jar.

2) Instalar la Plataforma.

   Para instalar esta plataforma es necesario ejecutar el *ant* de Apache. Esta es una herramienta expresamente hecha para compilar e instalar aplicaciones Java.

   Esto es, estando en la carpeta en donde se desea hacer la instalación, por ejemplo:

   root@raguilarptc-laptop:/usr/aglet202/bin

   Teclear lo siguiente:

   *root@raguilarptc-laptop:/usr/aglet202/usr/aglet202/bin#**chmod 755 ant***

   *root@raguilarptc-laptop:/usr/aglet202/usr/aglet202/bin# **./ant***

3) Configuración de políticas

   Al igual que otras aplicaciones de Java, los servidores Aglet requieren una inscripción al archivo de políticas de Java (usualmente ./java.policy) para abrir los socket, ejecutar un agente, acceder los archivos locales, entre otros.

   *root@raguilarptc-laptop:/usr/aglet202/bin# **ant install_home***

4) Configurar las variables de ambiente.

   Estando en la carpeta de *bin* de la instalación de Aglet, ejecutar:

   export AGLETS_HOME=/usr/aglet202
   export AGLETS_PATH=$AGLETS_HOME
   export PATH=$PATH:$AGLETS_HOME/bin

5) Ejecutar el Servidor de Aglet

   Una vez instalada la plataforma Aglet, se puede ejecutar el servidor para los Agentes Móviles que esta por omisión, el cual es llamado *Tahiti*. El servidor puede ser habilitado desde la terminal de Linux con el comando *agletsd* el cual inicia el servidor Tahiti, este se inicia de la siguiente forma:

   *root@raguilarptc-laptop:/usr/aglet202/bin#**agletsd –f ../cnf/aglets.props***

   Cabe mencionar que *aglets.props* es un archivo en donde se especifican las propiedades del servidor de agentes. El Servidor *Tahiti* corre en el puerto 4434, ese mismo número de puerto se aplicará al cliente y se usará el puerto 8000 para el servidor. Sin embargo se puede utilizar cualquier otro número de puerto, siempre y cuando no esté ocupado. La necesidad de especificar un número de puerto radica en que servirá como argumento para envío de agentes.

   Estando en el cliente:
   *usr/aglet202/bin#**agletsd –f ../cnf/aglets.props –port 4434***

   Estando en el Servidor
   *usr/aglet202/bin#**agletsd –f ../cnf/aglets.props –port 8000***

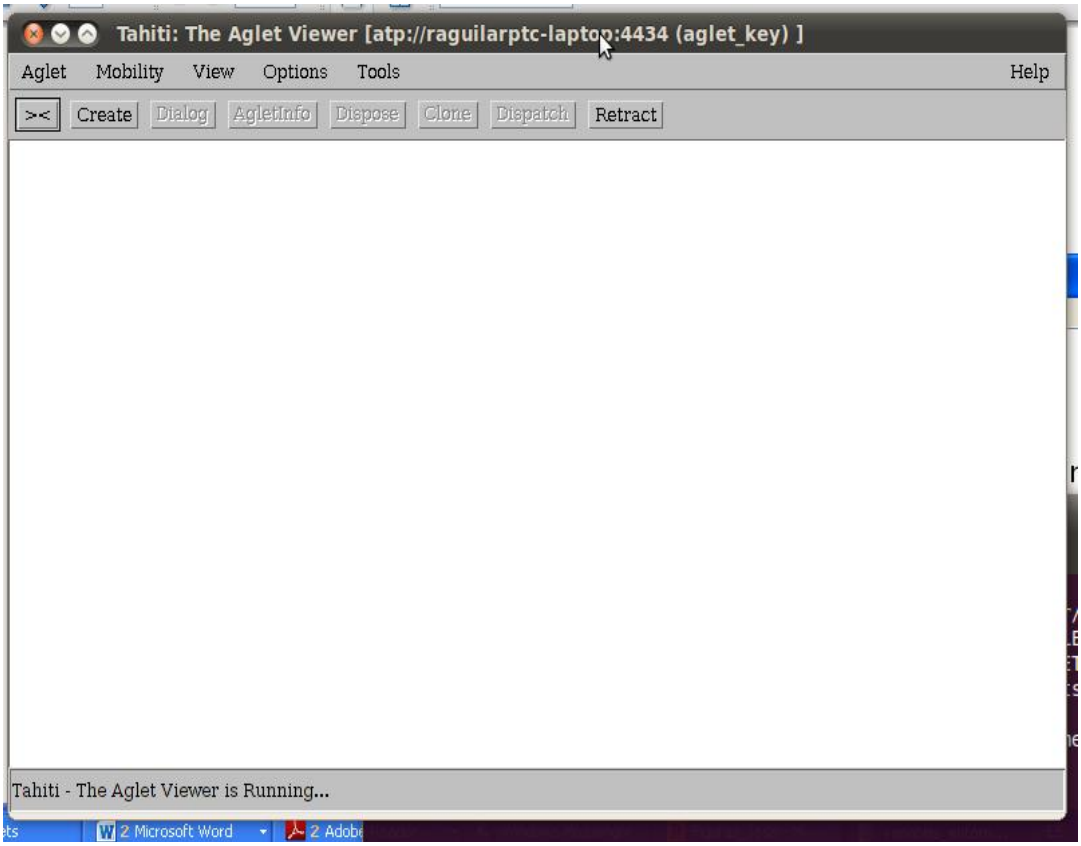


Figura 14 Servidor Tahiti, aquí es en donde se dará de alta los Agentes Móviles y Estáticos.

En la figura 14 se muestra la pantalla una vez ejecutado el servidor *Tahiti*, en el cual se dará de alta el agente principal y los agentes de búsqueda, tanto en el servidor así como en el cliente. Más información sobre las opciones de *Tahiti* se puede encontrar en la Guía de Usuario de *Tahiti* que está disponible en el sitio oficial de Aglets (The Aglets 2.0.2 User's Manual, 2009).

Algunas de las librerías de Aglets que se usarán en este proyecto son las siguientes:

com.ibm.aglet.Aglet

Es la clase principal de Aglet, la cual contiene todos los métodos fundamentales para un *Agente Móvil* (por ejemplo despachar URL), el control de movimientos, y su ciclo de vida. Todos los Agentes definidos dentro del Aglet, tienen la extensión de

esta clase abstracta. El método Aglet.dispatch(URL) nos proporcionará que un Aglet se mueva del servidor local al destino.

com.ibm.AgletProxy

Esta clase se usará para proporcionar seguridad a los Agentes. Esto es existirán varios métodos públicos que no deben ser accesados por otro *Aglet* por razones de seguridad, cualquier otro agente (Aglet) que se quiera comunicar con estos Agentes, tendrán primero que obtener el objeto *proxy* de éste Agente y después interactuar con él.

com.ibm.AgletContext

Esta clase proveerá a los Agentes de una interfaz para la rutina de desarrollo éste ocupe. Este método sirve para que cualquier Agente pueda obtener una referencia a este objeto AgletContext vía Aglet.getAgletContext() y usarla para obtener información local tal como la dirección del contexto del servidor y la enumeración del AgletProxies, o para crear un nuevo Agente (Aglet) dentro del contexto. Una vez que se despacha al Agente el objeto del contexto actualmente ocupado es separado y el contexto nuevo adherido. Esto es, el agente obtendrá el nuevo contexto en el cual se encuentre.

com.ibm.aglet.Message

El objeto Aglets se comunicará cambiando objetos de la clase *Message.* Un objeto de *Message* puede tener un objeto *String* para especificar la clase del mensaje y objetos arbitrarios como argumentos. Un mensaje puede ser enviado al Aglet llamado al objeto: *AgletProxy.sendMessage(Message msg)* y este es pasado a un argumento a *Aglet.handleMessage(Message msg)*.

void Aglet.onCreation(Object init)

Este método será llamado una sola vez durante el ciclo de vida del Agente. Este se usará para inicializar al Agente.

void Aglet.run()

Esta clase *run()* será llamado siempre que una instancia es reconstruida, esto es, cuando la instancia es creada, cuando el *Agente* es clonado, cuando llega al destino y cunado es activado.

boolean Aglet.handleMessage(Message msg)

Todos los mensajes enviados al *Aglet* serán pasados al método *handleMessage*. Aquí se podrá comprobar si el mensaje enviado es un mensaje conocido y así poder realizar al tarea según la clase de mensaje.

3.4 Caso de Estudio

Para probar el uso de los *Agentes Móviles*, se utilizará una base de datos de un sistema de control escolar de una Universidad. Esta base de datos cuenta con las siguientes tablas (solo se mencionarán las más relevantes para este caso de estudio):

a) Catálogo de alumnos.
b) Catálogo de materias.
c) Catálogo de maestros.
d) Registro de horarios por alumno.
e) Historial Académico por alumno.
f) Catálogo de Especialidades.

Estructuras de las diferentes tablas:

Catálogo de Alumnos (A)

catalumnos

Campo	Tipo	Nulo	Predeterminado	Comentarios
matricula	varchar(10)	No		
nombre	varchar(50)	No		
tutor	varchar(50)	No		
direccion	varchar(50)	No		
telef	varchar(20)	No		
eMail	varchar(50)	No		
espec	varchar(3)	No		

Índices:

Nombre de la clave	Tipo	Único	Empacado	Campo	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	matricula	0	A		

Datos

matricula	nombre	TUTOR	DIRECCION	TELEF	EMAIL	espec
309030038	PEREZ ACOSTA LUCRECIA VIRIDIANA	OBED CHIMAL				IPL
310030126	TORRES PLASCENCIA BRENDA SOLEDAD	OBED CHIMAL				IPL
309030017	ORTEGA SANLUISEÑO ISMAEL DARIO	OBED CHIMAL				IPL
310030134	GODINEZ PIRUL ALEJANDRO	OBED CHIMAL				IPL
311030130	NAVARRO DOMINGUEZ	OBED CHIMAL				IPL

	JONATHAN ANDRES					
310030121	MENDOZA GUZMAN GUADALUPE DE JESUS	OBED CHIMAL				IPL
310030113	DAMIAN SUBIAS DANIELA	OBED CHIMAL				IPL
310030120	MEJIA GARCIA JUAN GABRIEL	OBED CHIMAL				IPL
309030042	RUBIO BARRERA VICTOR MANUEL	OBED CHIMAL				IPL
310030114	JUAREZ TAPIA MARIA SOLEDAD	OBED CHIMAL				IPL
310030129	CAMPOS MOTA JESUS	OBED CHIMAL				IPL

Catálogo de Materias (B)

catmaterias

Campo	Tipo	Nulo	Predeterminado	Comentarios
codMateria	int(11)	No		
descripcion	varchar(50)	No		

Índices:

Nombre de la clave	Tipo	Único	Empacado	Campo	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	codMateria	0	A		

Datos

codMateria	Descripción
256	ALGORITMOS

254	CALCULO VECTORIAL (TE)
259	DESARROLLO HUMANO II (TE)
260	INGLES II (TE)
255	MATEMATICAS DISCRETAS (TE)
253	PROBABILIDAD Y ESTADISTICA (TE)
257	QUIMICA (TE)
258	REDES I
16	CALCULO DIFERENCIAL E INTEGRAL (PM)
28	DESARROLLO HUMANO II (PM)

Catálogo de Maestros (C)

catmaestros

Campo	Tipo	Nulo	Predeterminado	Comentarios
codMaestro	int(11)	No		
nombre	varchar(50)	No		

Índices:

Nombre de la clave	Tipo	Único	Empacado	Campo	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	codMaestro	0	A		

Datos

CveMaestro	Nombre
1	RICARDO GARCÍA FÉLIX
2	BEATRIZ DE LA VEGA
3	GEMA SUBÍAS GORDILLO
4	JOSÉ MARTÍNEZ VÁSQUEZ
5	VICTOR HUGO MANCILLA GARCÍA

Historial Académico por Alumno (E)

matricula	Sex	Carrera	idm	folio	gpo	cal	tipo_	tip	fechareg	idper	peri
307030003	0	INGENIERIA EN TELEMATICA	256	ITE02B	58	8.20	A	ORD	28/04/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	254	ITE02B	55	5.00	NA	ESP	07/05/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	259	ITE02B	61	8.20	A	ORD	06/05/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	260	ITEN2B	68	5.00	NA	ESP	07/05/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	255	ITE02B	60	7.90	A	ESP	08/05/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	253	ITE02B	57	5.00	NA	ESP	08/05/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	257	ITE02B	56	8.50	A	REG	30/04/2008	81	0108
307030003	0	INGENIERIA EN TELEMATICA	258	ITE02B	59	8.30	A	ORD	28/04/2008	81	0108
307030006	0	INGENIERIA EN PROCESOS DE MANUFACTURA	16	IPMUNI	47	7.50	A	REG	05/05/2008	81	0108
307030006	0	INGENIERIA EN PROCESOS DE MANUFACTURA	28	IPM02B	46	8.90	A	ORD	25/04/2008	81	0108

Catálogo Especialidades (F)

catespec

Campo	Tipo	Nulo	Predeterminado	Comentarios
codEsp	varchar(3)	No		
descripcion	varchar(50)	No		

Índices:

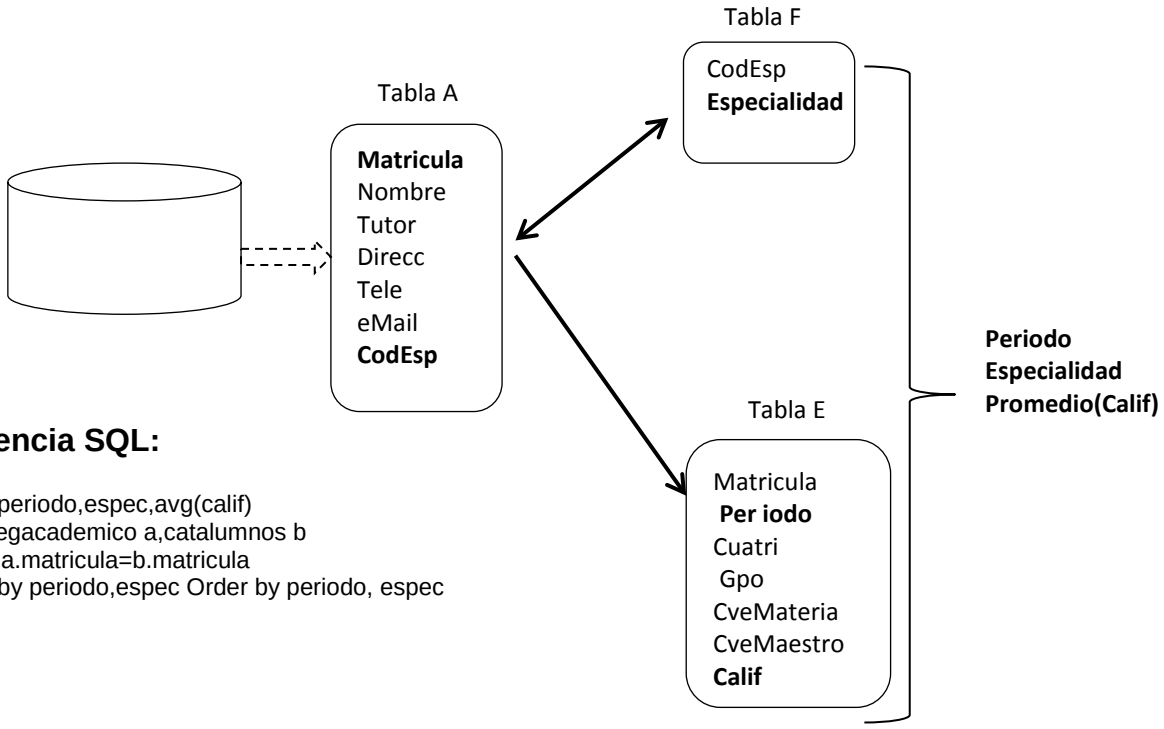
Nombre de la clave	Tipo	Único	Empacado	Campo	Cardinalidad	Cotejamiento	Nulo	Comentario
PRIMARY	BTREE	Sí	No	codEsp	0	A		

Datos

CodEspec	Especialidad
ITE	INGENIERÍA EN TELEMÁTICA
IPL	INGENIERÍA EN PLÁSTICOS
IME	INGENIERÍA EN METALÚRGICA
ISA	INGENIERÍA EN SISTEMAS AUTOMOTRICES

Para probar el acceso remoto a bases de datos mediante agentes, se realizarán consultas de la siguiente forma:

a) Mejores Promedio por Periodo, Especialidad



Sentencia SQL:

```
Select periodo,espec,avg(calif)
From regacademico a,catalumnos b
Where a.matricula=b.matricula
Group by periodo,espec Order by periodo, espec
```

Figura 15 Consulta General de Promedios por Periodo y Especialidad

La figura 15 representa las tablas que se involucrarán en la consulta solicitada, se llevará a cabo una consulta vertical tanto en la tabla A como en la E, en la cual se extraerán los datos necesarios para formar la consulta final que es la tabla creada por Tabla A U Tabla E (Tabla A unión Tabla E).

b) Cálculo de promedios Generales por Cada Alumno por Periodo y Especialidad.

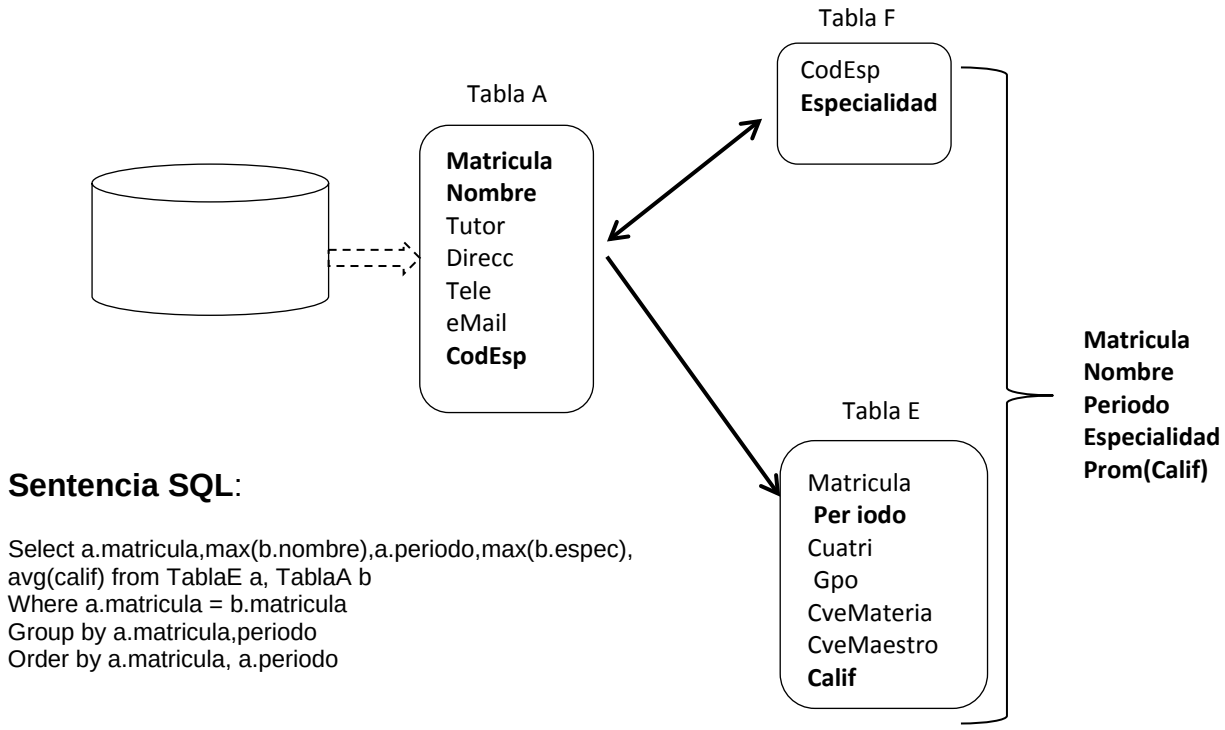
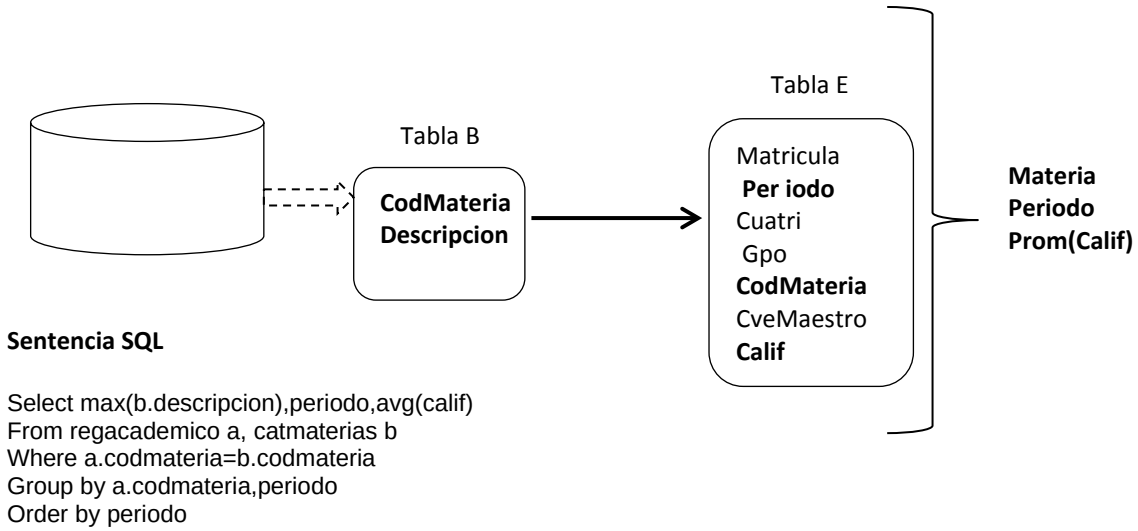


Figura 16 Consulta de Promedios por Alumno, Periodo y Especialidad

La figura 16 muestra una consulta vertical, en donde por cada alumno inscrito mostrará cuál es su promedio en cada periodo, así como mostrará su especialidad.

c) Cálculo de promedios por materia y periodo.



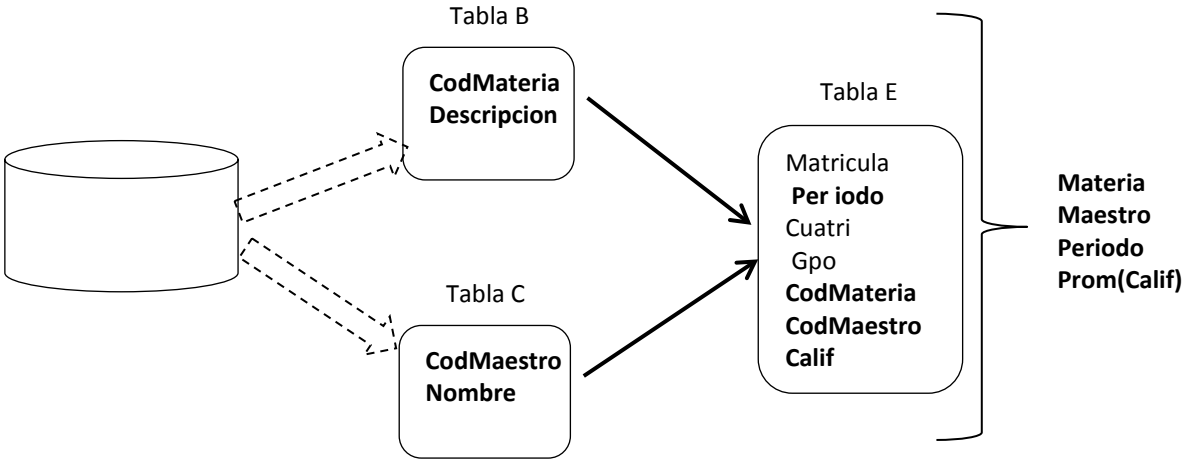
Sentencia SQL

```
Select max(b.descripcion),periodo,avg(calif)
From regacademico a, catmaterias b
Where a.codmateria=b.codmateria
Group by a.codmateria,periodo
Order by periodo
```

Figura 17 Consulta de Promedios por Materia, Periodo

La figura 17 muestra la consulta que mostrará el promedio que tiene cada materia en cada periodo cursado.

d) Cálculo de promedios por Materia, Maestro y Periodo.



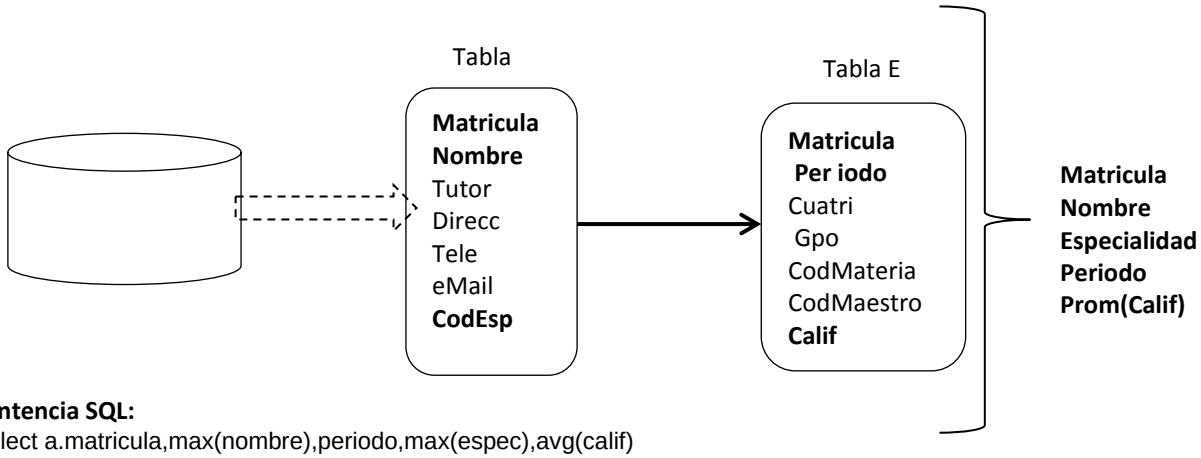
Sentencia SQL:

```
Select max(b.descripcion),max(nombre),periodo,avg(calif)
From regacademico a, catmaterias b, catmaestros c
Where a.codmateria=b.codmateria and a.codmaestro=c.codmaestro
Group by a.codmateria,a.codmaestro,periodo
```

Figura 18 Consulta de Promedios por Materia, Maestro y Periodo

En la figura 18 se hace una consulta basada en 3 tablas, esta consulta es también vertical en donde se va amostrar cuales son las materias con mayores promedios y que maestros la están impartiendo.

e) Cálculo de Alumno con promedios mayores a 9.0, para el periodo actual y por especialidad.



Sentencia SQL:

```
Select a.matricula,max(nombre),periodo,max(espec),avg(calif)
From catalumnos a, regacademico b
Where a.matricula=b.matricula and periodo='0311'
Group by a.matricula,periodo
Having avg(calif)>9
```

Figura 19 Consulta de Alumnos con promedios mayores a 9 para el periodo actual por y por especialidad

En la figura 19 se muestra una consulta tanto vertical como horizontal, ya que se están restringiendo los registros a mostrar, esto con la condición de que el promedio sea mayor a 9 y además que solo promedie las calificaciones del periodo actual (0311, septiembre a diciembre del 2011).

Es importante mencionar que este caso de estudio, las consultas se ejecutarán tanto en arquitectura Cliente/Servidor así como en la plataforma Aglets, esto para comparar el uso de ancho de banda que ambas arquitectura usan los medios de transmisión.

3.5 Diseño del Agente Móvil

A continuación se describirá el funcionamiento del sistema que generará los indicadores necesarios para Control Escolar. Este sistema está dividido en dos agentes principales los cuales se describirán en los objetivos siguientes.

Agente Estático.

Este agente tiene la función de manipular al agente de búsqueda. El nombre de este programa es *AgPrinCtrlEs*, el hecho de ser un agente estático significa que no podrá viajar a otro servidor que no sea en donde está instalado, además que como es el principal, este será el primero en ejecutarse, y es el que mostrará la interfaz del usuario, el cual podrá elegir los indicadores que desee calcular. Una vez que el usuario seleccione, el o los, indicadores deseados presionará la función de calcular, en ese momento el AgPrinCtrlEs enviará un mensaje con los parámetros necesario al agente AgBusCtrlEs para que se ejecute e inicie la búsqueda de la información deseada en la base de datos remota.

Agente Móvil

Este *Agente Móvil* llamado AgBusCtrlEs recibirá los mensajes y será despachado por el *Agente Estático*. Este Agente tendrá las características esenciales de movilidad para que este pueda ser despedido y viajar a diferentes nodos de la red.

Este Agente viajará por la red, llegará al nodo destino y buscará la información necesaria por el usuario, este agente regresará junto con la información obtenida y descargará los resultados al *Agente Estático* para que los muestre al usuario. En este caso se dirigirá al servidor que se encuentra de manera remota y hará la búsqueda en la base de datos que se encuentra en dicho servidor. Este es un *Agente Móvil* ya que tiene la característica de poder viajar a cualquier servidor que se le indique a diferencia del estático, así como procesará en el mismo servidor la

información y regresará los resultados de la información encontrada al *Agente Estático*.

Para lo anterior se mostrará la interfaz (figura 17) que utilizará el usuario final para su operación.

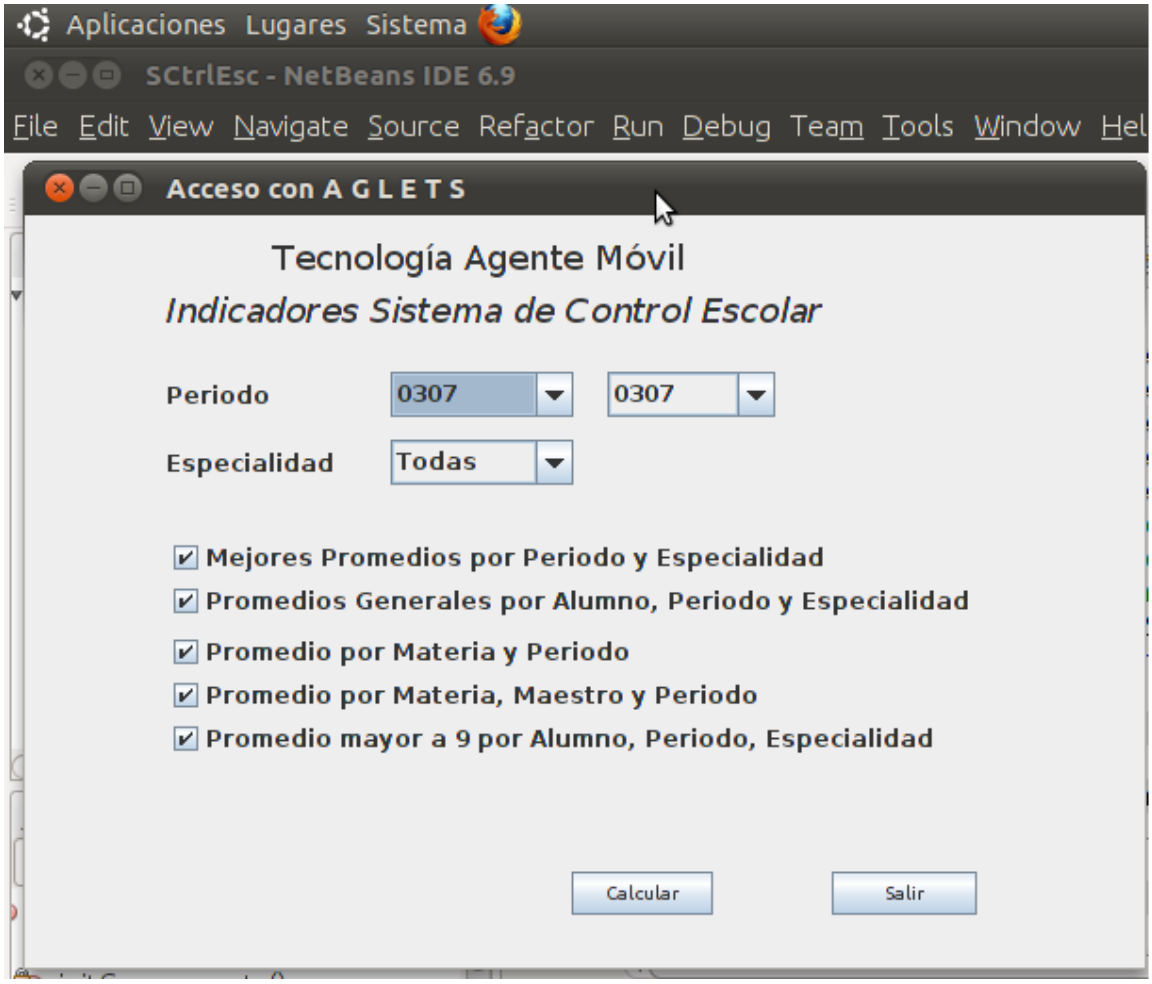


Figura 20 Pantalla Principal de Acceso a Base de Datos en plataforma Aglet

En la interfaz que muestra la figura 20 (del lado del cliente) muestra los datos que se utilizarán como criterio para la consulta, el usuario deberá seleccionar que periodos desea hacer la consulta y la especialidad de la cual se desean obtener los datos.

Una vez que el usuario selecciona los criterios correctos, deberá presionar el botón “Calcular”, este botón deberá ejecutar el *Agente Móvil* que buscará la información en el servidor remoto indicado. Esta información se refiere a las operaciones de búsqueda que deberá realizar sobre las diferentes tablas de la base de datos para obtener cada uno de los indicadores. Todo el proceso de búsqueda lo hará sobre el servidor y la información encontrada la regresará al cliente para que la muestre en la interfaz de la figura 20.

El método que ocurre cuando se ejecuta el botón “Calcular”, es enviar un mensaje a AgPrinCtrlEs para solicitar el envío del Aglet de búsqueda, éste es handleMessage(Message m). Todos los mensajes que se envían a un Aglet son tratados por el método handleMessage. Los programadores de aglets pueden comprobar si el mensaje recibido es un mensaje conocido para así poder realizar la tarea que corresponda con el tipo del mensaje. En la figura 21 muestra el código para el método handleMessage(Message m) para AgPrinCtrlEs.

Código del método handleMessage(Message m) en AgPrinCtrlEs

```
public boolean handleMessage(Message m) {

// m es la respuesta que envía el Aglet de Busqueda sobre la información que encontró en el servidor

        return handleRequest(m); // Recibe el mensaje y ejecuta el método de respuesta

    }
```

Figura 21 Código del método handleMessage(Message m) en AgPrinCtrlEs

Dentro del método handleMessage(Message m) se realiza una llamada al método handleHttpRequest(m, pw) que es la parte esencial de AgPrinCtrlEs. Este método se encarga dentro de otras cosas de crear al Aglet de búsqueda AgBusCtrlEs y enviarlo al servidor de datos. En la figura 22 se puede ver parte del código de handleHttpRequest en donde se realizan estas acciones.

Creación de AgBusCtrlEs y su envío al sitio remoto

```
//Creando AgBusCtrlEs

try{

    proxy=getAgletContext().createAglet(null,"AgBusCtrlEs",null);
```

```
}catch(Exception e){

    System.out.println("Problema creando Aglet de Busqueda:"+
e.getMessage());

}



//Enviar el Aglet al sitio remoto, servidor corriendose en el puerto 8000.

AgletProxy remoteProxy = null;

   URL destination=new URL("atp://192.168.1.64:8000");  //Es el servidor
remoto

    remoteProxy=proxy.dispatch(destination);

}
```

Figura 22 Creación de AgBusCtrlEs y su envío al sitio remoto

4. Análisis de Resultados

A continuación se presentan los resultados obtenidos en esta investigación, el estudio realizado a las dos tecnologías para la recuperación de información de una base de datos remota.

4.1 Tiempo de respuesta de una conexión remota utilizando Cliente/Servidor.

A continuación se muestra la interfaz del sistema una vez hecha la búsqueda en Cliente/Servidor, figura 23.

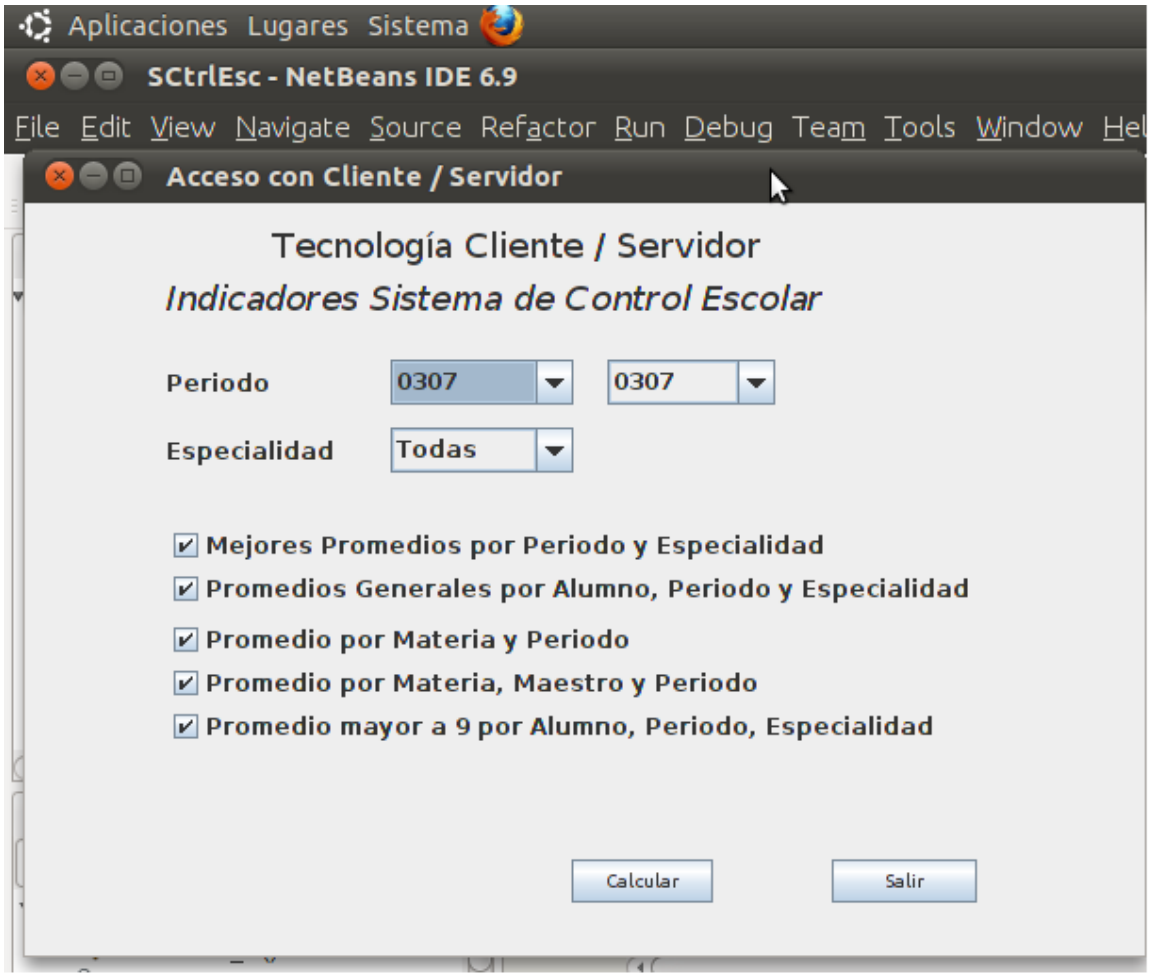


Figura 23 Pantalla principal para elegir los indicadores a calcular

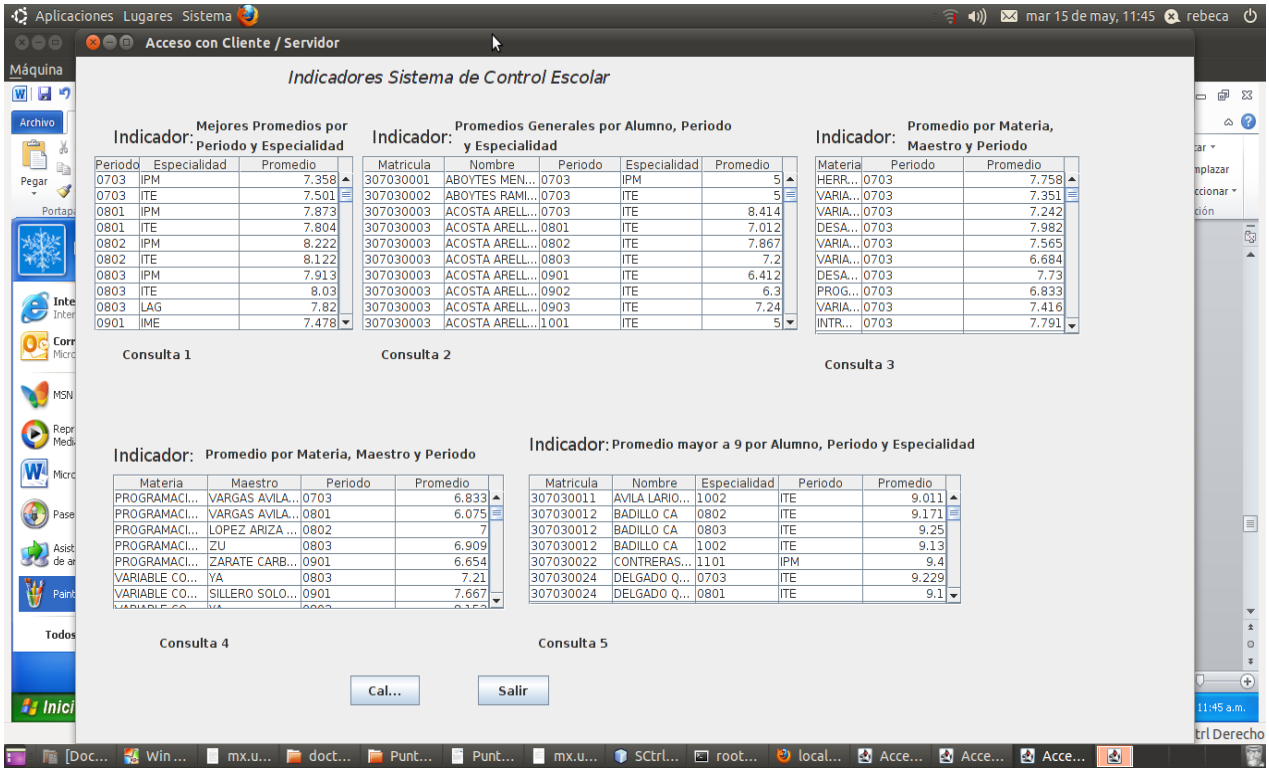


Figura 24 Interfaz de usuario utilizando conexión remota con Cliente/Servidor

La información de la figura 24, muestra los resultados de los indicadores seleccionados, estos mediante el control de checkbox, así como de los criterios seleccionados en los controles de combobox, esto es, el usuario podrá seleccionar los indicadores a calcular, así como el rango de periodos de los cuales desea obtener los dato y la especialidad.

Esta prueba se obtuvo al momento de ejecutar el sistema en dos equipos, un equipo portátil y el otro un servidor que presta servicio a la Universidad elegida. Cabe mencionar que la aplicación se ejecutó en un sistema operativo Linux con distribución Ubunto versión 10. Estos equipos fueron conectados mediante red inalámbrica, el cliente es raguilarptc-laptop o 192.168.1.70 (donde se encuentra instalada la aplicación) y el servidor llamado upjr o 192.168.1.252 ,en donde se encuentra instalada la base de datos. La conexión se hizo mediante el driver JDBC de java para base de datos remota.

jdbc:mysql://192.168.1.252/ctrlescolar","aglet", ""

En donde 192.168.1.252 es la IP del servidor, “ctrlescolar” es el nombre de la base de datos, “aglet” es el usuario de la base de datos y por ultimo las comillas que se encuentran al final es la contraseña del usuario, para este caso el usuario Aglet no tiene contraseña.

Este sistema utiliza una conexión mediante cliente/servidor para obtener los datos remotos tal como lo muestra la siguiente figura 25:

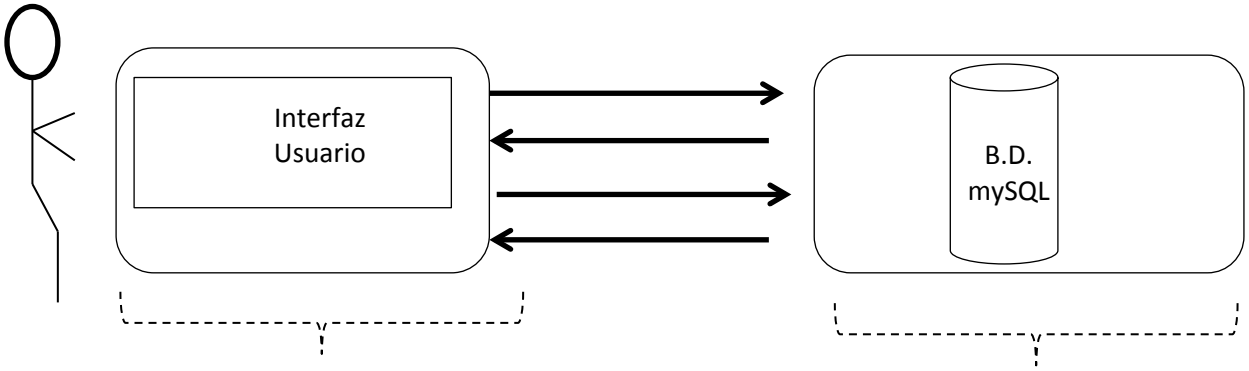


Figura 25 Uso de Tecnología Cliente/Servidor

La figura 25 representa el tipo de conexión que se hace para la consulta de la información de base de datos en el servidor, la cual tiene la necesidad de una comunicación constante entre cliente y servidor, para mostrar la información al cliente, lo cual tendrá como consecuencia un tiempo de respuesta lento y generando una carga de datos en el medio de transmisión traducida como “tráfico en la red”.

Los tiempos que arrojó esta prueba se muestran en el cuadro 3.

Indicador	**Tamaño (bytes)**	**8:00 – 10:00 (seg)**	**12:00 – 14:00**	**15:00 – 17:00**
Consulta 1	1,133	9.49	10.48	9.51
Consulta 2	3,638	11.81	12.16	11.65

Consulta 3	990	1.15	1.29	1.16
Consulta 4	283,764	27.88	30.18	28.42
Consulta 5	72,512	11.23	12.58	12.1

Cuadro 3 Tiempos de Acceso a Datos con Cliente-Servidor

Estos resultados fueron obtenidos mediante 3 pruebas en diferentes horarios por cada una de las consultas, esto es con el fin de mostrar los tiempos de acceso en los horarios en donde existen mas demanda de los usuarios y donde hay menos demanda.

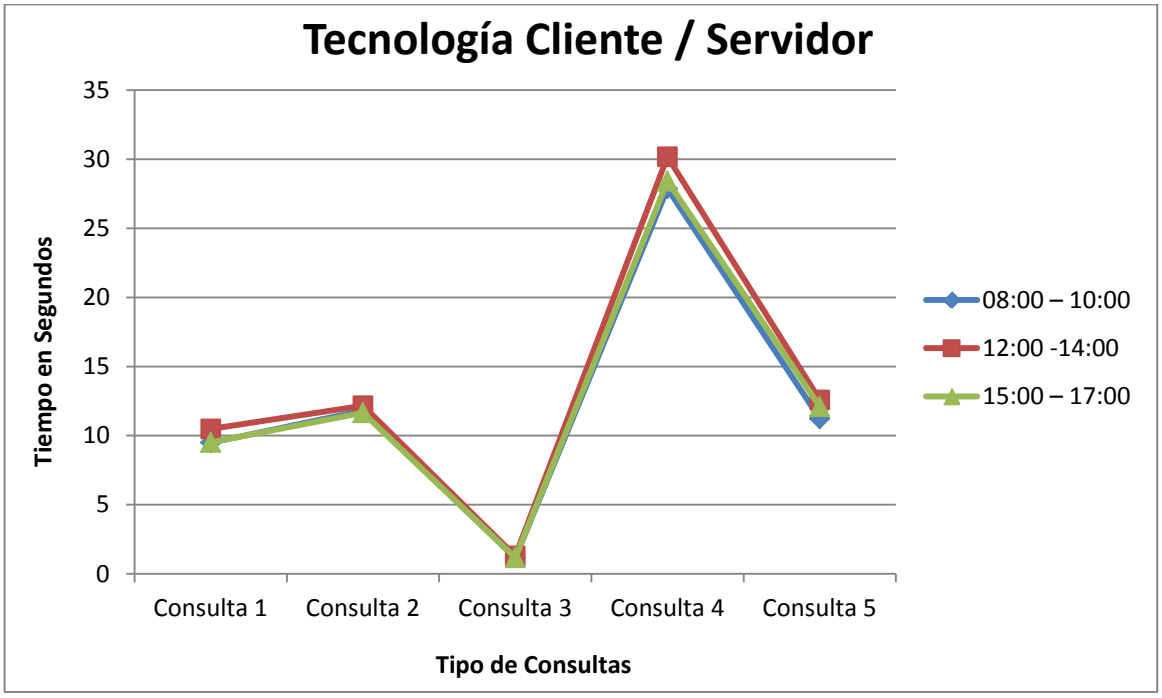


Figura 26 Gráfica de tiempos de respuesta de Cliente/Servidor

En la figura 26, se muestras los tiempos de respuesta que se obtuvieron al acceder a la información de la base de datos remota mediante la tecnología de Cliente/Servidor, estos serán comparados posteriormente con la tecnología de Agentes Móviles.

4.2 Tiempo de respuesta mediante una conexión remota utilizando un algoritmo de Agentes Móviles.

A continuación se mostrará la interfaz para la tecnología de *Agentes Móviles*, que en realidad es igual a la de Cliente/Servidor, la diferencia radica en el acceso a los datos remotos, este lo hace mediante un agente que viajará al servidor para generar todas las consultas y regresará con los resultados al cliente.

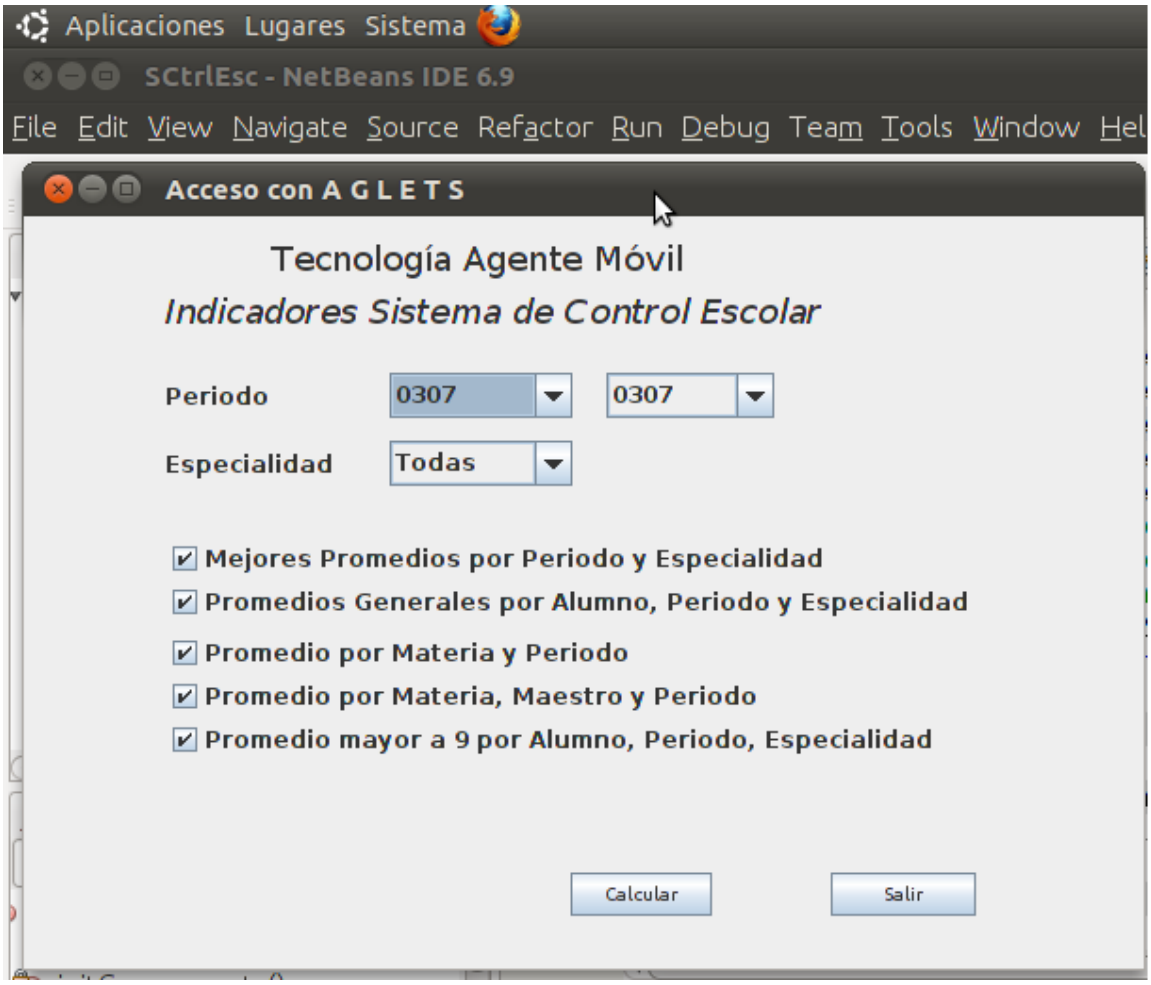


Figura 27 Pantalla principal para elegir los indicadores a calcular

En la figura 27 se muestra la pantalla principal en donde el usuario podrá elegir los indicadores que desea calcular.

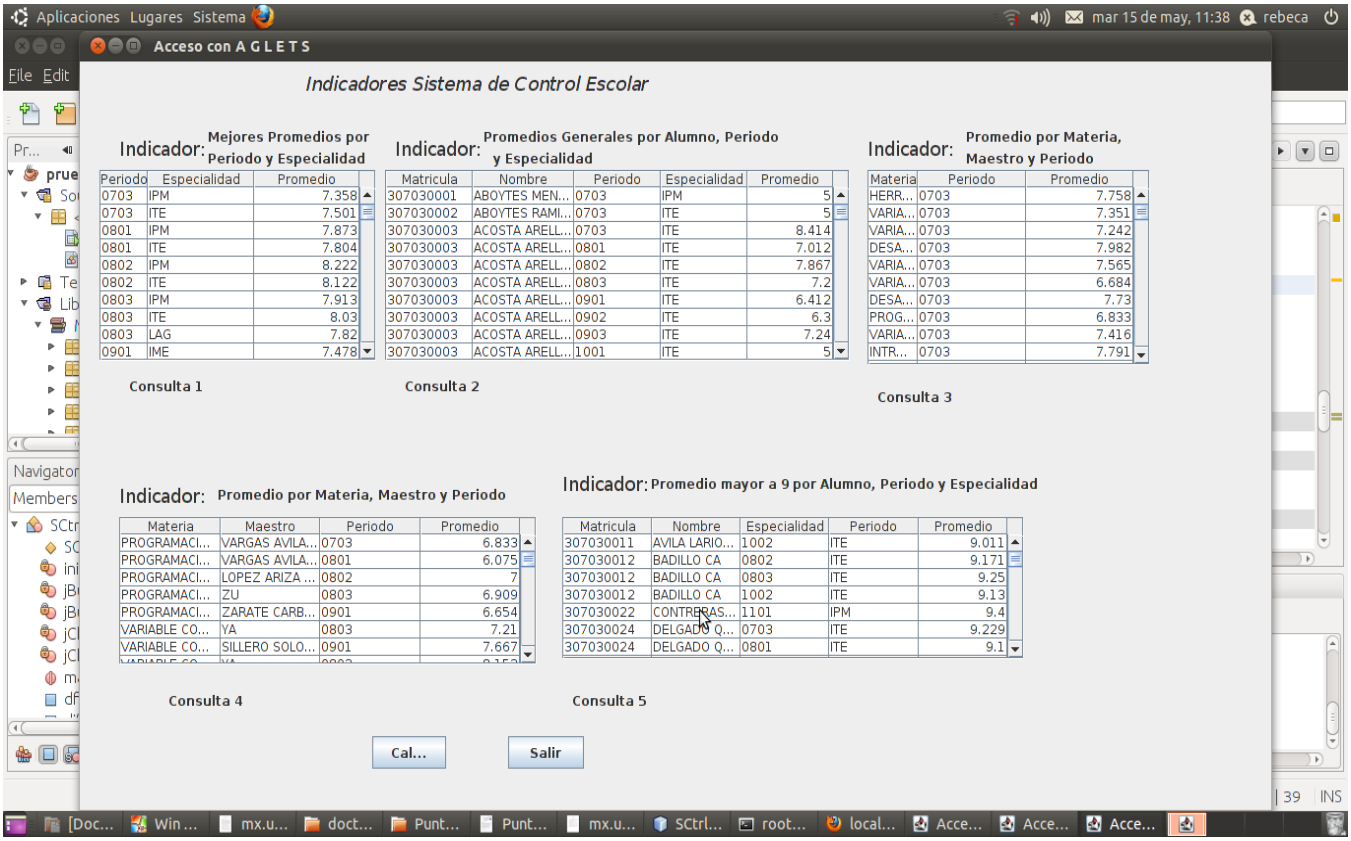


Figura 28 Información obtenida mediante conexión con Agentes Móviles

En la figura 28 se muestran los resultados de las consultas hechas mediante *Agente Móvil*. Para esta consulta se utilizó un *Agente Estático* llamado *AgPrinCtrlEs* el cual llamó al agente móvil, de nombre *AgBusCtrlEs*, el cual se encargó de viajar hacia el servidor para hacer la búsqueda de la información requerida. Esta información fue generada bajo los siguientes requerimientos seleccionados por el usuario, tal como, el periodo seleccionado fue 0211 y 0311 (que es el cuatrimestre 2 y 3 del año 2011), así como también el usuario indicó que requería la información de todas las especialidades y por último seleccionó los indicadores que deseaba calcular. Con estos datos al momento de dar el botón de “Calcular” el agente estático lanza al agente móvil para la búsqueda.

Es importante mencionar que en este caso, el *Agente Móvil* es el que hace la conexión a la base de datos una vez que ésta en el servidor, esta conexión la hace mediante el driver JDBC de Java:

jdbc:mysql://192.168.1.252/ctrlescolar","aglet", ""

En donde la IP 192.168.1.252 es la del servidor en al cual se encuentra alojada la base de datos llamada “ctrlescolar” que se accedida con el usuario “aglet” sin contraseña.

A continuación la figura 29 muestra una representación de la tecnología *Agentes Móviles*.

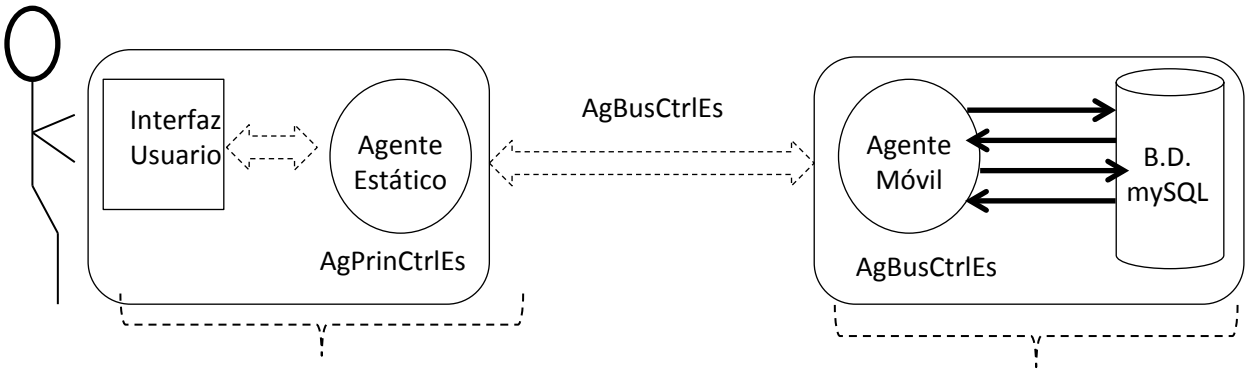


Figura 29 Tecnología de Agente Móvil

En la figura 29 se muestra la forma en que la tecnología de *Agente Móvil* accede a la información de manera remota, esto es, el usuario lanza la consulta que desea hacer, el *Agente Estático* manda ejecutar (Grimshaw D., 2001) al *Agente Móvil* (*AgBusCtrlEs)* el cual viajará por la red hacia el servidor, una vez estando en el servidor hará todas las consultas necesarias, por lo que habrá constante comunicación entre el *Agente Móvil* y la base de datos, pero la ventaja de éste, es que esta comunicación se hará de forma local. Una vez que el *Agente Móvil* termina

de hacer todas las consultas requeridas por el cliente, regresa por la red hacia donde está el agente estático, le entrega la información y el *Agente Estático* se encargará de mostrarla al usuario mediante la interfaz preparada para este.

Se espera que con esta tecnología se reduzca el tiempo de respuesta entre cliente y servidor. Para comprobar lo anterior se muestra a continuación, en el cuadro 4, los tiempos que se obtuvieron al acceder la información con *Agentes Móviles*.

Indicador	Tamaño (bytes)	8:00 – 12:00	13:00 – 15:00	16:00 – 17:00
Consulta 1	72,512	9.16	10.35	10.1
Consulta 2	129,162	11.1	10.63	9.35
Consulta 3	990	1.2	1.3	1.16
Consulta 4	283,764	24.25	26.17	22.36
Consulta 5	23,556	10.17	11.45	11.18

Cuadro 4 Tiempos de respuesta con tecnología de Agentes Móviles.

Estos resultados también se obtuvieron en tres horarios diferentes para tratar de abarcar los mejores y peores tiempos del algoritmo de *Agentes Móviles*.

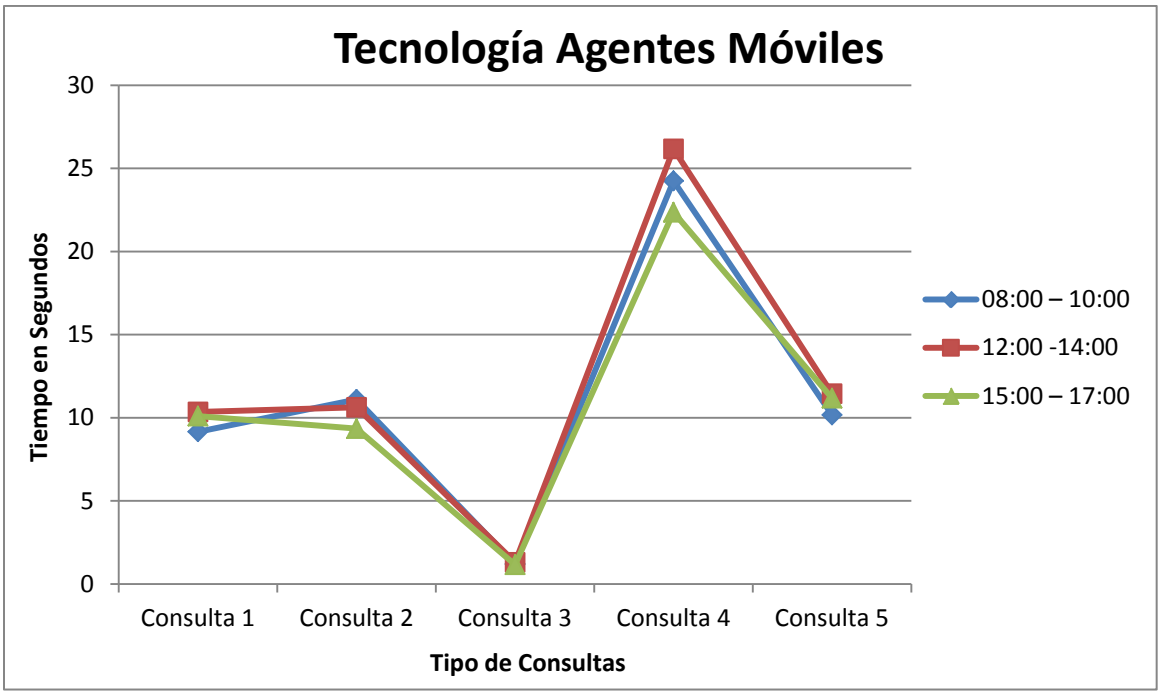


Figura 30 Gráfica de tiempos de respuesta de Agentes Móviles.

En la figura 30 muestra los tiempos que se obtuvieron al acceder a la información de la base de datos remota mediante la tecnología de *Agentes Móviles*. En el siguiente objetivo se compararán estos tiempos con los obtenidos con la tecnología cliente/servidor.

4.3 Comparación entre Tecnología Cliente/Servidor y Agentes Móviles

A continuación se muestra una comparación de los resultados obtenidos con las dos tecnologías.

El promedio de tiempos entre C/S y Agentes Móviles se muestra a continuación:

Tecnología	Tiempo
C/S	12.73
AM	11.32

Cuadro 5 Tiempos de respuesta promedio por tecnología.

El cuadro 5 representa un resumen de los tiempos que se obtuvieron como promedio para las dos tecnologías que se están comparando, en donde se aprecia

que Agentes Móviles (AM) da un tiempo de respuesta menor que Cliente/Servidor (C/S).

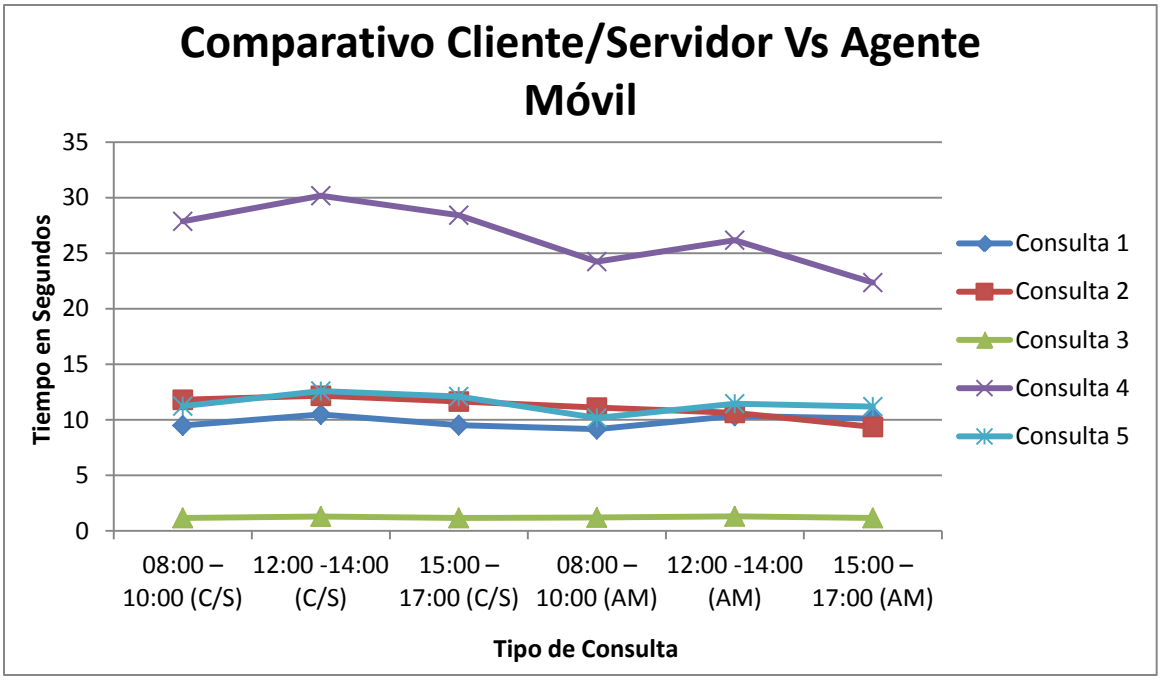


Figura 31 Comparativo Tiempos de respuesta entre Cliente/Servidor y Tecnología de Agente Móvil

El la figura 31, se muestran los tiempos obtenidos al acceder a la base de datos de control escolar mediante cliente/servidor contra los tiempos obtenidos al acceder a la misma base de datos pero con *Agentes Móviles*, está claro que la diferencia de tiempos no es significativa, dado este resultado la pregunta sería ¿y entonces que ventaja se tiene el utilizar *Agentes Móviles*?, y la respuesta es que el beneficio del uso de *Agentes Móviles* no radica realmente en el tiempo de acceder a la información, más bien, está diseñado para no generar tráfico de datos en el medio de transmisión, es decir, utilizar el menor ancho de banda posible de la red. Así como también se tienen un sinnúmero de aplicaciones, tal es el caso de aquellos procesos que se lanzan sin que se requiera de manera inmediata el resultado, o sin necesidad de que este encendido el cliente para poder obtener el resultado. Lo que pretende el *Agente Móvil* es que no sea obligatorio que tanto el cliente así como el servidor estén encendidos, bien puede lanzar el proceso el

cliente y apagar el equipo, ya que al momento de volverse a conectar el *Agente Móvil* regresará el resultado.

4.4 Evaluación de las dos tecnologías mediante métricas de software orientadas a puntos de función.

En este objetivo se procederá a obtener las métricas de software aplicadas a los sistemas desarrollados de cálculo de indicadores.

Las pruebas fueron realizadas mediante métricas de software orientadas a la función y a continuación se mostrarán los resultados que se obtuvieron para la tecnología *Agentes Móviles*.

0	1	2	3	4	5
Sin influencia	Incidental	Moderado	Medio	Significativo	Esencial

1.- ¿Requiere el sistema copias de seguridad y recuperación fiables?	5
2.- ¿Se requieren comunicaciones de datos?	5
3.- ¿Existen funciones de procesamiento distribuido?	5
4.-¿Es crítico el rendimiento?	5
5.- ¿Será ejecutado el sistema en un entorno operativo existente y fuertemente utilizado?	5
6.- Requiere el sistema entrada de datos interactiva?	2
7.- ¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas o variadas operaciones?	4
8.- ¿Se actualizan los archivos maestros de forma interactiva?	0
9.- ¿Son complejas las entradas, las salidas los archivos o las peticiones?	2
10.- ¿Es complejo el procesamiento interno?	2
11.- ¿Se ha diseñado el código para ser reutilizable?	5
12.- ¿Están incluidas en el software la conversión y la instalación?	5
13.- ¿Se ha soportado el sistema para soportar múltiples instalaciones en diferentes organizaciones?	5
14.- Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizadas por el usuario?	5
Resultado ----------→	51

Cuadro 6 Cálculo de ajuste de complejidad para tecnología Agente Móvil

Los pesos otorgados a cada punto del cuadro 6, se describen brevemente de la siguiente manera:

- Pregunta uno: Se define un peso de 5 ya que el sistema crea copias de seguridad de manera esencial para el manejo de aplicaciones posteriores, creando las carpetas *logs* y *tmp*.

- Pregunta dos: Se da un peso de 5 ya que el sistema hace peticiones a la base de datos y recupera información de forma esencial.
- Pregunta tres: Se da un peso de 5 ya que es esencial el procesamiento distribuido al momento en que el *Agente Móvil* viaja para la recuperación de la información.
- Pregunta cuatro: Se define un peso de 5 ya que es el rendimiento precisamente el que se desea medir en este proyecto.
- Pregunta cinco: Se otorga un peso de ya que el estudio sobre sistema operativo Linux es esencial para este proyecto.
- Pregunta seis: Se da un peso de 2 ya que, aunque los datos de entrada son importantes para el procesamiento de la información, existe muy poca interacción entre sistema y usuario.
- Pregunta siete: Se proporciona un valor de 4 porque en realidad existe una cantidad significativa de pantallas y de operaciones.
- Pregunta ocho: Se da un peso de 0 ya que en este sistema no se hace actualización de archivos, solo es consulta.
- Pregunta nueve: Se da un peso de 2 ya que la complejidad de entradas, salidas y procesamiento es moderado.
- Pregunta diez: Se otorga un peso de 4 ya que el procesamiento interno es significativo.
- Pregunta once: se otorga un valor de 5 ya que el código es completamente reutilizable, ya que se crean clases por separado para su futura reutilización.
- Pregunta doce: Se otorga un valor de 5 ya que es esencial la instalación y conversión, dado que java es un software abierto.
- Pregunta trece: Se otorga el valor de 5 ya que es esencial el soporte sobre la instalación de diferentes organizaciones.
- Pregunta catorce: Se da el valor de 5 ya que es sencillo de operar por el usuario.

Parámetro de medida	Cuenta	Factor de Peso			Resultado
		Simple	Medio	Complejo	
Número de entradas de Usuario	7	***3**	4	6	21
Número de salidas de usuario	5	4	*5	7	25
Número de peticiones al usuario	1	*3	4	6	3
Número de archivos	7	*7	10	15	49
Número de interfaces externas	0	5	7	10	0
Cuenta – total					**98**

Cuadro 7 Cálculo para punto de función tecnología Agente Móvil

A continuación se obtiene el punto de función para esta tecnología con los resultados del cuadro 7, para ello se aplica la fórmula de punto de función descrita en el capítulo 2.

PF = Cuenta-total * (0.65+0.01*Fi)

PF = 98 * (0.65+0.01 * 51)

PF = 98*1.16 = **113.68** Para Tecnología de Agentes Móviles

Productividad = PF * personas/mes

Productividad = 113.68 * 1

Productividad = 113.68 Tecnología de Agentes Móviles

Para obtener resultados sobre la tecnología Cliente / Servidor, al igual que *Agentes Móviles*, se aplicó la Métrica Orientada a la Función obteniendo los resultados siguientes:

0	1	2	3	4	5
Sin influencia	Incidental	Moderado	Medio	Significativo	Esencial

1.- ¿Requiere el sistema copias de seguridad y recuperación fiables?	5
2.- ¿Se requieren comunicaciones de datos?	5
3.- ¿Existen funciones de procesamiento distribuido?	0
4.-¿Es crítico el rendimiento?	5
5.- ¿Será ejecutado el sistema en un entorno operativo existente y fuertemente utilizado?	5
6.- Requiere el sistema entrada de datos interactiva?	2

7.- ¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas o variadas operaciones?	4
8.- ¿Se actualizan los archivos maestros de forma interactiva?	0
9.- ¿Son complejas las entradas, las salidas los archivos o las peticiones?	2
10.- ¿Es complejo el procesamiento interno?	4
11.- ¿Se ha diseñado el código para ser reutilizable?	5
12.- ¿Están incluidas en el software la conversión y la instalación?	5
13.- ¿Se ha soportado el sistema para soportar múltiples instalaciones en diferentes organizaciones?	5
14.- Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizadas por el usuario?	5
Resultado ----------→	52

Cuadro 8 Cálculo de ajuste de complejidad para tecnología Cliente / Servidor

Los pesos otorgados a cada punto del cuadro 8, se describen brevemente de la siguiente manera:

- Pregunta uno: Se define un peso de 5 ya que el sistema crea copias de seguridad de manera esencial para el manejo de aplicaciones posteriores, creando las carpetas *logs* y *tmp*.
- Pregunta dos: Se da un peso de 5 ya que el sistema hace peticiones a la base de datos y recupera información de forma esencial.
- Pregunta tres: Se dá un peso de 0 ya que es no se utiliza el procesamiento distribuido, todo el trabajo lo hace el cliente.
- Pregunta cuatro: Se define un peso de 5 ya que es el rendimiento precisamente el que se desea medir en este proyecto.
- Pregunta cinco: Se otorga un peso de ya que el estudio sobre sistema operativo Linux es esencial para este proyecto.
- Pregunta seis: Se da un peso de 2 ya que, aunque los datos de entrada son importantes para el procesamiento de la información, existe muy poca interacción entre sistema y usuario.
- Pregunta siete: Se proporciona un valor de 4 porque en realidad existe una cantidad significativa de pantallas y de operaciones.
- Pregunta ocho: Se da un peso de 0 ya que en este sistema no se hace actualización de archivos, solo es consulta.

- Pregunta nueve: Se da un peso de 2 ya que la complejidad de entradas, salidas y procesamiento es moderado.
- Pregunta diez: Se otorga un peso de 3 ya que el procesamiento interno es moderado.
- Pregunta once: se otorga un valor de 5 ya que el código es completamente reutilizable, ya que se crean clases por separado para su futura reutilización.
- Pregunta doce: Se otorga un valor de 5 ya que es esencial la instalación y conversión, dado que java es un software abierto.
- Pregunta trece: Se otorga el valor de 5 ya que es esencial el soporte sobre la instalación de diferentes organizaciones.

Pregunta catorce: Se da el valor de 5 ya que es sencillo de operar por el usuario.

Parámetro de medida	**Cuenta**	**Factor de peso**			**Resultado**
		Simple	**Medio**	**Complejo**	
Número de entradas de Usuario	7	***3**	4	6	21
Número de salidas de usuario	5	4	*5	7	25
Número de peticiones al usuario	1	*3	4	6	3
Número de archivos	7	*7	10	15	49
Número de interfaces externas	0	5	7	10	0
Cuenta – total					**98**

Cuadro 9 Cálculo para punto de función tecnología Cliente / Servidor

Una vez obtenidos los valores necesarios para aplicar la fórmula de punto de función, cuadro 9, se obtiene el resultado final de Punto de Función y Productividad con sus fórmulas correspondientes:

PF = Cuenta-total * (0.65+0.01*Fi)

PF = 98 * (0.65+0.01 * 52)

PF = 98*1.17 = **114.66** Para Tecnología Cliente/Servidor

Productividad = **PF** * persona / mes

Productividad = 114.66 * 1

Productividad = 114.66 Tecnología Cliente/Servidor

Existen dentro de ésta métrica de software fórmulas para calcular los costos, la documentación, etc., todas basadas en el punto de función. Aquí solo se presentó la productividad como una manera de la producción del software de ambas tecnologías con un solo programador.

Los datos obtenidos en todas las pruebas hechas al software y a las tecnologías como tal reflejan una respuesta de aceptación para el uso de la tecnología *Agentes Móviles* en la recuperación en tiempo de información de base de datos.

Como último punto a evaluar se muestra la siguiente tabla de una matriz de decisión, en la cual se llevó a cabo el análisis de los diferentes puntos importantes evaluados cuantitativamente para el estudio de esta tesis, logrando el resultado de la tecnología que alcanzó el mejor desempeño y la que llevó a cabo la conclusión final de la tecnología recomendada en esta tesis para la recuperación de información de una base de datos remota.

Los valores otorgados a las calificaciones de Cliente / Servidor y Agentes Móviles según la característica de cada uno de ellos son evaluados en una escala de 0 a 9, cuadro 10, y como se puede observar va de aumento según la aplicación que ofrece cada una de las dos tecnologías a las características evaluadas en la matriz de decisión.

Ninguno	Bajo	Medio	Alto
0	3	6	9

Cuadro 10 Valores definidos para matriz de decisión

Estos valores se obtuvieron empíricamente para poder dar un valor cuantitativo a la influencia que tiene cada característica aplicada a cada una de las tecnologías estudiadas en este trabajo.

Características	Pesos	Calificación Cliente/Sevidor (C/S)	Calificación Agente Móvil (AM)	Puntos (C/S)	Puntos (AM)
Tiempo en recuperación	30%	3	6	0.9	1.8

de información					
Puntos funcionales aplicados al software de cada tecnología	10%	6	9	0.6	0.9
Adaptación ante ambientes cambiantes	10%	9	9	0.9	0.9
Capacidad para trabajar sin conexión de red	15%	0	9	0	1.35
Reducción del tráfico de red	15%	3	6	0.45	0.9
Autonomía	20%	0	9	0	1.8
Resultados Totales	100%			2.85	7.38

Cuadro 11 Matriz de decisión

Se definen los siguientes términos utilizados en la matriz del cuadro 11:

a) Características: Son todas aquellas características importantes comparativas de cada tecnología estudiada en este trabajo para la obtención de resultados cuantitativos.
b) Pesos: Son los valores otorgados según la importancia que tiene cada característica en el estudio de esta tesis.
c) Calificación Cliente / Servidor: Es la calificación obtenida para la tecnología C/S según el desempeño que ofrece en cada una de sus características comparativas.
d) Calificación de *Agentes Móviles*: Es la calificación obtenida para la tecnología Agentes Móviles según el desempeño que ofrece en cada una de sus características comparativas.
e) Puntos de Cliente / Servidor: Es la puntuación obtenida basada en el peso otorgado a cada característica comparativa y la calificación que obtuvo la tecnología (C/S) por su desempeño e cada característica.
f) Puntos *Agentes Móviles:* Es la puntuación obtenida basada en el peso otorgado a cada característica comparativa y la calificación que obtuvo la tecnología *Agentes Móviles* por su desempeño e cada característica.

En seguida se definen los puntos comparativos utilizados en esta matriz para dar una mejor comprensión de su significado:

- Tiempo de recuperación de información: Este punto se refiere al tiempo que tardó la aplicación de cada una de las tecnologías estudiadas en este trabajo para obtener la información personalizada requerida por el usuario. Los valores de cada tecnología presentados en la sección de calificación se obtuvieron mediante una media de los tiempos obtenidos de cada una de las tecnologías mostradas en los cuadros 3, 4 y 5.
- Puntos funcionales aplicados al software de cada tecnología: son los resultados de las pruebas obtenidas al software de cada tecnología mediante el uso de métricas orientadas a la función.
- Adaptación ante ambientes cambiantes: Se refiere a como se adapta Cliente / Servidor y *Agentes Móviles* cuando existen cambios dentro de un contexto de trabajo ya sea cambio de lugar de la información, eliminación de la información que pudiera ser útil, etc.
- Capacidad para trabajar sin conexión de red: Se refiere a todas aquellas capacidades que pueden tener cada una de las dos tecnologías para lograr seguir operando aún sin conexión de red, ya sea que puedan estar en un estado de espera de conexión o simplemente termine su ciclo de ejecución al no poder trabajar sin conexión.
- Reducción de tráfico en la red: Es la ventaja que pueden ofrecer cada una de las dos tecnologías para evitar y/o reducir el tráfico en la red, tanto en la búsqueda así como en la transferencia de información.
- Autonomía: Se refiere al hecho de control sobre sus propias acciones, es decir, ellos deciden cuándo, cómo y qué hacer ante la situación que se les presente.

5. Conclusiones y Trabajo a Futuro

Este trabajo fue dedicado al estudio de dos tecnologías de acceso a información en base de datos distribuida, tal como *Cliente / Servidor* (C/S) y *Agentes Móviles* (*AM*).

El objetivo de este trabajo fue el de demostrar que *AM* era una tecnología que nos ofrecía menor tiempo de respuesta al acceder a la información en una base de datos, aunque también se aprendió que tiene otras aplicaciones importantes en el ámbito de la programación.

Como tecnología nueva, el uso de *AM* poco a poco crecerá en cuanto a base de datos distribuida y/o remota se refiere, ya que es ofrece grandes ventajas en el procesamiento de la información.

Se puede apreciar que en realidad estas dos tecnologías son muy parecidas, en cuanto a su aplicación, ya que a lo largo de este trabajo se demostró que pueden manejar las información de la misma forma, solo que su forma de conectarse a la base de datos distribuida es distinta, así como la forma de ejecutar el proceso que accederá a la información.

Algunas compañías desarrolladoras de lenguajes de programación para la creación de *AM*, entre las cuales se encuentra Aglets, están evolucionando cada vez más en sus aplicaciones para lograr proporcionarlas como una multiplataforma, y de esta manera no tener que activar siempre su servidor de agentes y ofrecer manualmente los permisos para acceso de información e identificación de los agentes que son despachados y recibidos.

Tomando esto en cuenta, mientras estas compañías no lleguen a sus metas fijadas, el envío y recibimiento de agentes debe hacerse de la forma convencional, en donde cada servidor especifica en que parte del sistema de archivos o que información de una base de datos puede acceder el *AM*.

En el caso de este proyecto, en donde se utilizó la plataforma de *Aglets* para la construcción de *Agentes Móviles*, es necesario crear un servidor de agentes, en

este caso se utilizó el *Tahiti* que trae el *ASDK* y que es el que ayudará a despachar y recibir agentes.

Los resultados obtenidos a estas tecnologías fueron solo basados dentro de este contexto de investigación. Reiterando que las dos tecnologías son fuertemente confiables en la recuperación de información distribuida.

La tecnología que obtuvo resultados reflejados en el objetivo planteado en un inicio de este trabajo fue el de *AM*, a continuación se presenta la información por la cual se decidió que ésta era la tecnología adecuada al tipo de aplicación en recuperación de información en base de datos distribuida.

Los datos reflejados en los cuadros, 3, 4 y 5 del capítulo 4, muestran las comparaciones de tiempos de respuesta al usuario en su petición de los 5 indicadores, dentro de los cuales se observa variaciones menores de tiempos de respuesta utilizando la tecnología de *Agentes Móviles*. Para una visualización más clara, se pueden observar estos mismos resultados en las figuras 26, 30 y 31, en donde se muestran las gráficas de los tiempos de respuesta de Cliente / Servidor, *Agentes Móviles* y la gráfica comparativa de las dos tecnologías anteriores respectivamente, que se encuentran en el capítulo 4.

Otro de los puntos para seleccionar la mejor tecnología en este trabajo, fue por la obtención de la Métrica Orientada a la Función. Esta métrica nos proporciona un punto de función (PF), el cual representa valores que reflejan la funcionalidad y complejidad del software. En el capítulo 4 se aplicó este tipo de métricas obteniendo los siguientes resultados:

Para el sistema basado en tecnología Cliente / Servidor : PF = 114.66

Para el sistema basado en tecnología Agentes Móviles : PF = 113.68

Teniendo en cuenta que, entre menor sea el PF, menores serán los costos, productividad, documentación, calidad. Por lo tanto los resultados obtenidos en esta métrica califican a los sistemas en recuperación de información de base de datos

remota con mejor desempeño en cuanto a complejidad para la tecnología de *Agentes Móviles*.

Por último para poder llegar a los resultados requeridos para este trabajo, se analizaron los datos obtenidos del cuadro 9, en la que muestra la matriz de decisión, la cual fue una guía a la toma de decisión de la tecnología que ofrece mejores resultados en la obtención de datos de una base de datos remota. Este cuadro muestra los siguientes puntos obtenidos por los pesos y calificaciones aplicados a las dos tecnologías en cuestión:

- Cliente / Servidor : 2.85
- Agentes Móviles : 7.38

Logrando con ello elegir a la tecnología de *Agentes Móviles* para el desarrollo de aplicaciones con acceso a base de datos remota y/o distribuida.

A continuación se presenta un cuadro 12 comparativo con las principales características y sus valores de ambas tecnologías:

Puntos de Conclusión	Agente Móvil	Cliente / Servidor
Rendimiento de tecnologías ante ambientes cambiantes.	Ofrece buen rendimiento, ya que el Agente Móvil por su autonomía decide las acciones a realizar ante su ambiente.	Ofrece un rendimiento medio, ya que, aunque puede aplicar para casi cualquier plataforma, es incapaz de reaccionar de forma autónoma a dichos cambios de tecnología.
Rendimiento de Tecnología ante desconexión de red.	Mantiene un buen rendimiento ya que el Agente Móvil espera dentro del servidor remoto hasta que haya nuevamente conexión de	No ofrece rendimiento, no puede hacer nada ante la desconexión de la red, sencillamente no responde a la petición del usuario.

	red.	
Rendimiento de tráfico en red.	Ofrece buen rendimiento ya que realiza las tareas dentro del servidor destino, o lo que es lo mismo, el procesamiento lo lleva al servidor y al cliente solo le regresa el resultado por lo que no hace constantes viajes al cliente.	No ofrece rendimiento, ya que para hacer consulta al servidor tiene que hacer constantes viajes entre el cliente y el servidor y el procesamiento lo ejecuta en el cliente.
Calificación de tecnologías mediante puntos de función.	PF = 113.68	PF = 114.66
Resultado óptimo de uso de tecnologías de acuerdo a la aplicación de la tabla de decisión.	7.38	2.85
Característica de autonomía ofrecida por la tecnología	Tiene alto grado de autonomía, el agente toma decisiones ante diferentes circunstancias.	No tiene autonomía.
Tiempo de recuperación de información ofrecida por la tecnología	Su desempeño es menor al tratarse de pocos datos, pero cuando estos son considerables su desempeño aumenta.	Ofrece buenos tiempos en recuperación de información de base de datos remota.

Cuadro 12 Conclusiones de Tecnologías Cliente / Servidor y Agentes Móviles

El beneficio del uso de *Agentes Móviles* no radica realmente en el tiempo de acceder a la información, más bien, está diseñado para no generar tráfico de datos en el medio de transmisión, es decir, utilizar el menor ancho de banda posible de la red

El uso de *Agentes Móviles* es una opción aceptable para el acceso a al información de datos remotos y/o distribuidos. Esta tecnología resulta ser sencilla de implementar y podrían obtener tiempos de acceso que permitieran agilizar la consulta de la información.

También se alcanzó a notar que esta tecnología puede resolver perfectamente los problemas de heterogeneidad de redes, ya que juegan el papel de cliente y servidor, equipos con diferentes características de hardware y software. Este conjunto de equipos bien pueden representar el escenario de una red común de computadoras, por lo se puede considerar que el caso de estudio que se expuso en este trabajo, puede ser adaptable a cualquier red heterogénea, sin embargo, habrá que hacer las pruebas necesarias.

Así como también, se pueden hacer las respectivas investigaciones para demostrar que esta tecnología puede jugar muy bien la función de programación distribuida, compitiendo con los middleware tales como RMI y CORBA, incluso su configuración es mas sencilla que las plataformas antes mencionadas.

Durante esta investigación se pudo notar que no existe mucha información respecto a esta tecnología con aplicación a programación distribuida, sin embarga bien puede ser un siguiente tema de investigación.

Otro aspecto importante de este trabajo es el hecho de utilizar la herramienta Java como lenguaje de programación, el cual tiene muchas ventajas sobre otros lenguajes de programación. Una de ellas es el de poderse ejecutar en casi todos las plataformas de sistemas operativos, sin necesidad de recompilarse. Así como también soportan cualquier gestor de bases de datos lo que le hace ser un sistema abierto. Este acceso a cualquier base de datos lo hace mediante la tecnología JDBC, que es la librería propia de Java para conexión a bases de datos. Otra

ventaja de Java es que no tienen ningún costo, es totalmente libre, pudiéndose descargar de internet de manera gratuita. Así como también cuenta con sus propias librerías para la creación de agentes móviles (Aglets).

Como se puede notar, en este proyecto no se consideró el concepto de la seguridad de la transferencia y ejecución de Agentes Móviles, ya que se analizó y se llegó a la conclusión de que bien puede considerarse como otro extenso tema de investigación así como de un trabajo a futuro.

Referencias Bibliográficas

Acebal C., Cueva-Lovelle J.M., 2000, Acceso a bases de datos distribuidas mediante el uso de agentes móviles, Universidad de Oviedo.

Agentes Móviles y CORBA 1999. www.informatica.us.es/~ramon/tesis/CORBA/Seminario-MASIF/. . (Última consulta febrero del 2012).

Barba-Matrtí A., Hesselbach-Serra X. 2002. Inteligencia de Red, Ediciones UPC, 2002.236.

Boger M. 2001. Java Distributed Systems. Editorial Wiley.

Boquera E. 2003. Servicios Avanzados de Telecomunicación, Ediciones Díaz de Santos, 537.

Braun P., Rossak W.R., Kaufmann M. 2007. Mobile Agents: Basic Concepts, Mobility Models, and the Tracy Toolkit. ISBN:1558608176.

Caballe S, Xhafa F. 2006. Aplicaciones Distribuidas en Java con Tecnología RMI . Delta publicaciones.

Castillo C., F. Sergio, León Ch., A. Luis, Gómez G., G. Janeth. 2004. Arquitectura basada en Agentes Móviles para comercio Electrónico, Energía y Computación vol 12, no. 2.

De Ockham G. 2005. Aplicación de agentes móviles. Revista Científica, Vol 3, No. 1, ISSN: 1794-192X.

Fonnegra M. N., Russi A. M. 2003. Plataforma de Servicios Distribuidos Basados en Agentes Móviles. Pontifica Universidad Javeriana.

Franco-Borré D.A., Moya-Villa Y., Quintana-Álvarez M., García-Bolaños M.A., Rodríguez-Ribón J.C. 2008. ABADAM: Un modelo de Seguridad para el Acceso a bases de datos utilizando agentes móviles, Ciencia Investigación Academia Desarrollo vol 13, no. 1.

Grimshaw D. 2001. Artificial Intelligence Topics with Agents. CPS 720. www.ryerson.ca. (Última consulta en septiembre del 2011).

Lange B.D., Oshima M., 1999. Programming and Deploying Java Mobile Agents with Aglets. Addison Wesley, 2-7, 33-35, 51-89, ISBN 0-201-32582-9. 2-7,

Lange B.D., Oshima M. 2000. Seven Good Reasons for Mobile Agents, Communications of de ACM, Volume 42, Match 2000, ACM Press, ISSN:0001-0782.

León S.S., Mancilla-Espinoza L.E. 2005. Agente Aglet: Un nuevo paradigma para el acceso a base de datos distribuidas. Encuentro de Investigación en Ingeniería Eléctrica, Zacatecas.

Manual de Agentes Móviles 1999. http://es.sourceforge.jp/projects/sfnet_aglets. (Última consulta mayo del 2012).

Mohamed A. T., King F. 2007. Increasing Mobile Agent Performance by Using Free Areas Mechanism vol. 6 no. 4, University, KSA.

Papastavrou S., Samaras G., Pitoura E. 2000. Mobile Agents for World Wide Web Distributed Databases Access ,IEEE Transactions on Knowledge and data engineering, vol 2 no. 5.

Pérez L. 1998. Agentes Móviles en Bibliotecas Digitales. Tesis de Maestría en Ciencias. Universidad de las Américas Puebla.

Pressman R.S. 2002. Ingeniería de Software, un enfoque práctico, Mc Graw Hill, Quinta edición.

Solve real problems with aglet, type of mobile agent 1997. http://www.javaworld.com/jw-05-1997/jw-05-hood.html. (Última consulta febrero del 2012).

Sommerville I. 2004. Ingeniería de Software, Pearson Educacion.

The Aglets 2.0.2 User's Manual 2009. http://jaist.dl.sourceforge.net. (Última consulta junio del 2012).

UMBC 2000. www.cs.umbc.edu/agentslist/archive/current. (Última consulta febrero del 2012).

Venners B. 1999. Java 2 Virtual Machine, McGraw Hill. ISBN 0071350934, 9780071350938.