Arquitectura de un Sistema Operativo Web Basado en Sistemas Multiagentes

José Aguilar[♠] Niriaska Perozo Edgar Ferrer Juan Vizcarrondo[♣]

Resumen

La cantidad de sistemas, servicios y aplicaciones desarrollados para la Web ha crecido considerablemente, en algunos casos, el soporte por parte de los sistemas operativos existentes a ellos, no es el esperado. Como una solución a esta necesidad, se plantea un modelo de sistema operativo denominado SOW, el cual soporta y maneja un conjunto de servicios en un contexto heterogéneo, dinámico y adaptativo, bajo el enfoque de reconfiguración de aplicaciones. El SOW está conformado por cuatro subsistemas, donde cada una lleva a cabo una serie de funciones coordinadas, que permiten un uso eficiente de los recursos sobre Internet. En este trabajo se presenta el diseño del SOW usando Agentes, así, cada uno de los subsistemas está compuesto por un conjunto de agentes a través de los cuales se distribuyen las tareas asignadas a él.

Palabras Claves: *Sistemas Distribuidos, Computacion Web, Sistemas Operativos, Sistemas Multiagentes.*

Abstract

The amount of systems, services and applications developed for the Web has grown considerably. In some cases, the support of the existing operating systems to them, is not the awaited one. Like a solution to this necessity, we propose a model of operating system denominated SOW. It supports and handles a set of services in a heterogenous and dynamic environment like Internet. The SOW is conformed by four subsystems (resource, repositories, web object and communities manager subsystems), where each one carries out a series of coordinated functions, that allow an efficient use of the resources on Internet. In this work we present the design of the SOW based on Agents. That is, each one of the subsystem is viewed like an agent who make the tasks assigned.

Keywords: *Distributed Systems, Web Computing, Operating Systems, Multiagents, Systems*

1. Introducción

Dada la amplia variedad de servicios inimaginables que surgen cada día en la Web, resulta difícil diseñar un sistema operativo que apoye/use a cada uno de esos servicios. De éstas necesidades surge una corriente de desarrollo llamada Sistema Operativo Web (SOW), que tiene como objetivo

[♠] CEMISID, Dpto. de Computación, Facultad de Ingeniería, Av. Tulio Febres. Universidad de los Andes, Mérida 5101, Venezuela, aguilar@ula.ve

[♣] CEMISID, Dpto. de Computación, Facultad de Ingeniería, Av. Tulio Febres. Universidad de los Andes, Mérida 5101, Venezuela, {nperozo,eferrer,nvizcarrondo}@cemisid.ing.ula.ve

principal proveer una plataforma que permita a los usuarios beneficiarse del potencial computacional ofrecido en la Web, a través del compartimiento de recursos y de la resolución de los problemas de heterogeneidad y adaptabilidad dinámica presentes en la misma. Así, para alcanzar un rendimiento óptimo en un ambiente dinámico de recursos distribuidos como la Internet, el SOW debe ser configurable y capaz de adaptarse a los cambios en cuanto a la disponibilidad de recursos (de software y de hardware). Teniendo en cuenta esas consideraciones, el modelo de SOW presentado en este trabajo propone una serie de aspectos para proveer servicios que se adecuen a esos rasgos especiales de la Web. Así, el SOW presenta un diseño que cuenta con las herramientas asociadas para permitir el uso transparente e interactivo de los recursos accesibles a través de la red, en cualquier momento que un usuario lo requiera. Esos servicios pueden ser hardware, software, o una combinación de ambos. El usuario sólo necesita comprender la interfaz del SOW, sin importarle como su solicitud es satisfecha.

Existen varias propuestas para el manejo e integración de los recursos computacionales disponibles en la Web. Quizás el proyecto más general sea el WOS TM [2], ya que permite el manejo e integración de los recursos tratando el problema de la heterogeneidad y volatilidad en la Web. Ese proyecto, al igual que la presente propuesta, está basado en la idea del uso de versiones como solución a esos problemas. Otra propuesta para el manejo de recursos en la Web parecida, es la arquitectura Jini de SUNMicrosystems [6]. Jini provee servicios de localización de recursos (conocidos como *lookups*), y aplica la idea de agrupar recursos en federaciones.

Otros esfuerzos para explotar los recursos distribuidos en Internet incluyen *2k* [13], el cual es una arquitectura de un sistema operativo distribuido para el manejo de recursos en redes heterogéneas que utiliza CORBA y la configuración de aplicaciones distribuidas basada en componentes para enfrentar los problemas de heterogeneidad; ***Netsolve,*** es un sistema de cómputo distribuido que ofrece funcionalidades para aplicaciones científicas [4]; ***Globe,*** el cual es un sistema para manejo de recursos globales donde se implementan estrategias adaptativas de replicación de recursos Web [8]; ***Legión,*** el cual es un metasistema que permite interconectar un conjunto de computadores heterogéneos y geográficamente distribuidos, ofreciendo una máquina virtual coherente y simple [9]; ***Globus,*** que construye una rejilla (grid) computacional que permite el acceso a los recursos computacionales independientes de la posición geográfica [5]; y ***WebOS,*** que provee servicios de un sistema operativo básico necesitados para construir aplicaciones que están distribuidas geográficamente y deben ser reconfigurables dinámicamente [1].

Existen otras propuestas en donde estos problemas de interoperabilidad de plataformas han sido manejados exitosamente a través de Java. Entre éstas tenemos ***Charlotte,*** el cual es un sistema que provee facilidades a los programadores de aplicaciones paralelas para escribir programas en Java y ejecutarlos desde un navegador Web [3]; ***Popcorn,*** que es un sistema de metacomputación que permite que cualquier computador que disponga de una máquina virtual Java pueda participar en el metasistema [10]; y ***Javelin,*** que es un sistema que provee un ambiente para la ejecución de aplicaciones paralelas, donde el cómputo paralelo se distribuye entre nodos de Internet [7]. Esos proyectos están dirigidos principalmente a modelos de programación orientados en Java para computación paralela basada en Internet. SOW no es dependiente del modelo de programación, aunque los modelos de programación orientada a Java pudieran perfectamente ser integrados. Como se mencionó anteriormente, el modelo propuesto, al igual que el WOS TM y Jini, es diferente al resto de los proyectos mencionados, en el sentido de que no se requiere ningún catálogo global centralizado de recursos. En general, el modelo de SOW soporta y maneja un conjunto de servicios en un contexto heterogéneo, dinámico y adaptativo, bajo el enfoque de reconfiguración de las aplicaciones.

En cuanto a Sistemas Multiagentes (SMA), surgen del área de la Inteligencia Artificial Distribuida (DAI), como una solución para el diseño de sistemas muy grandes donde no se puede realizar un control centralizado por poseer ambientes heterogéneos, abiertos, que cambian dinámicamente, como por ejemplo, Internet [15]. Por estas razones, y porque además, a través del diseño de un SMA se puede distribuir las funciones y papeles a través de sus componentes, para

que a través de las interacciones y cooperación entre ellos se resuelvan los problemas, se propone realizar el diseño del SOW basado en la teoría de agentes.

2. Generalidades de la Arquitectura Propuesta

A continuación las características de nuestro SOW y su integración en un ambiente distribuido.

2.1 Características del SOW [18]

- **Distribuido y versionado:** Diferentes versiones de cada uno de los servicios que el SOW ofrece, están esparcidos sobre la red.
- **Dinámico:** La Web es una entidad que está evolucionando permanentemente, por lo tanto, el SOW debe adaptarse a eso. Es por ello que tiene incorporado cierto dinamismo en los subsistemas que lo integran. Por ejemplo, a través de la configuración dinámica de los servicios que ofrece, a través de la actualización dinámica de la información sobre los recursos locales disponibles en cada nodo, a través de la agrupación dinámica de los nodos existentes de acuerdo a ciertas características, y a través de la migración, replicación y seguimiento de objetos Web.
- **Abierto:** SOW es un sistema abierto, de tal forma que permite que diversas tecnologías sean usadas (heterogeneidad). Esto facilita la gestión de cualquier nodo de Internet por el sistema.
- **Inteligente:** Cada uno de los subsistemas del SOW tendrá algún nivel de inteligencia para el desarrollo de alguna de sus funciones.

SOW es un sistema operativo versionado e inteligente, que se autoconfigura dinámicamente para permitir un acceso fácil y transparente a los recursos distribuidos sobre la Internet. El SOW utiliza un **motor de inferencia**, como un sistema reactivo, el cual forma parte del subsistema manejador de recursos. El subsistema manejador de recursos administra los recursos del ambiente computacional heterogéneo existente en la Web. En el SOW propuesto existen dos tipos de repositorios por cada nodo: los **repositorios de recursos locales,** los cuales consisten en dispositivos de almacenamiento que guardan y actualizan la información sobre los recursos disponibles localmente y los **repositorios de recursos remotos,** los cuales consisten en dispositivos de almacenamiento de la información remota a la que se accede con frecuencia. Ambos tipos de repositorios requieren de mecanismos de actualización y coherencia de la información almacenada. Cada vez que un requerimiento llega a un nodo del SOW, es el subsistema manejador de recursos quién recibe la solicitud, y solicita la participación de los repositorios en pro de localizar los recursos que permitirán satisfacer ese requerimiento. En general, los repositorios locales son los primeros repositorios consultados por los motores deductivos, ya que la prioridad es ejecutar los servicios solicitados localmente en la medida de lo posible para ofrecer tiempos de respuesta aceptables.

Por otro lado, debido al gran dinamismo presente en la Web y a la gran cantidad de nodos que puedan estar conectados al SOW, se crean **comunidades,** que son conjunto de nodos pertenecientes al SOW que exhiban afinidades funcionales y conductuales, los cuales son capaces de asociarse y disociarse autónomamente. La utilización de comunidades permite optimizar la búsqueda de algún servicio, ya que en lugar de buscar nodo por nodo se busca comunidad por comunidad.

Por último, SOW realiza la gestión de los objetos Web, para garantizar que la ubicación de ellos sea cerca de los sitios de mayor demanda en la Internet para ofrecer mejores tiempos de respuesta, a través de mecanismos que soportan **objetos móviles** capaces de decidir autónoma e inteligentemente a donde moverse, y de mecanismos de replicación de objetos que determinen de

manera autónoma cuando replicar o eliminar un objeto en una entidad distinta a donde actualmente se encuentra el objeto origen.

2.2. Integración del SOW en un Ambiente Distribuido

SOW está al tope de una plataforma estándar distribuida (Middleware), para aprovechar los servicios provistos por dicha plataforma, tales como los mecanismos de nombramiento y seguridad, entre otros. Además, esto le permite coexistir con aplicaciones tradicionales, y ser usado por una amplia comunidad (ver figura 1).

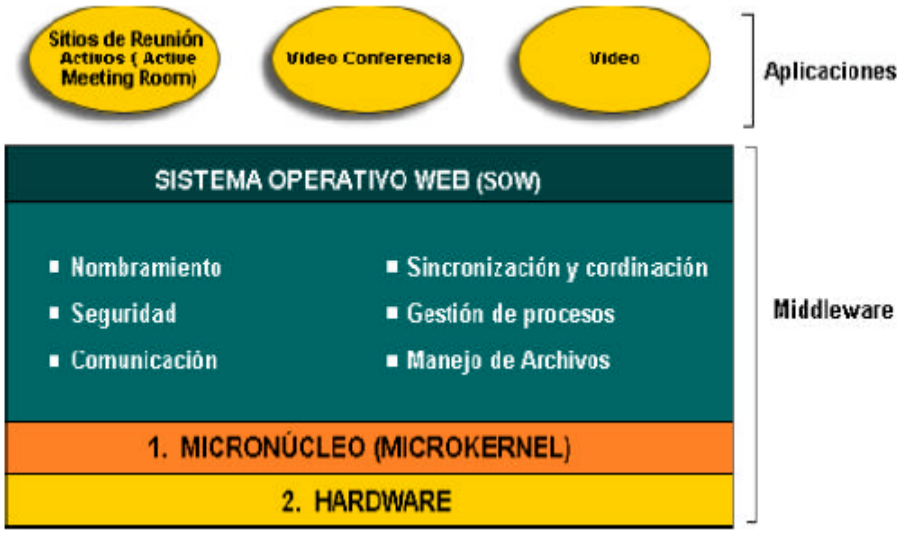

Figura 1. Esquema de Integración del SOW Propuesto en un Ambiente Distribuido

Particularmente, entre los servicios que deberán ser provistos por SOW por la plataforma estándar distribuida se encuentran [18]: un **Servicio de Nombres (páginas blancas)** para la identificación y localización de los recursos en el entorno distribuido por nombramiento; un **Servicio de Directorio (páginas amarillas)** para la identificación y localización de los recursos en el entorno distribuido a través de búsquedas por atributo; un **Servicio de Seguridad** que proporcione *confidencialidad* (protección contra usuarios no autorizados), *integridad* (protección contra alteraciones o corrupción de la información), y *disponibilidad* (protección contra interferencia en los medios para acceder los recursos); **Servicios de Sincronización y Coordinación** para el manejo del tiempo (sincronización de relojes), concurrencia, paralelismo y transacciones; un **Servicio de Gestión de Procesos** que se encarga de la asignación de los procesadores, y de la planificación interna y global de los procesos; un **Sistema de Archivos Distribuidos** que permita gestionar los distintos sistemas de archivos en los diversos nodos y, el compartimiento de información de manera transparente; **Servicios de Comunicación** que permitan la interacción entre los modelos existentes (cliente/servidor, intermediarios (Proxy, Dispatcher, Caches, etc.), comunicación en grupo (multicast), entre otros). Las interacciones se llevan a cabo a través de pase de mensajes, llamadas a procedimientos y métodos remotos (RPC, RMI, etc.), manejo de memoria compartida, entre otros.

El *micronúcleo* (ver figura 1) consta sólo de las funciones absolutamente esenciales del núcleo del sistema operativo; coordina las interacciones entre los procesos servidores (componentes del sistema operativo externos al micronúcleo) a través de mensajes distribuidos (valida los mensajes y los pasa entre los componentes, entre otras cosas), y otorga el acceso al hardware. Dado que un

sistema operativo para Internet debe ofrecer una nueva plataforma para la nueva generación de aplicaciones que son independientes de la localización y de los dispositivos, SOW maneja un alto grado de transparencia a nivel de acceso, localización, migración, concurrencia, fallas, persistencia, rendimiento, y escalabilidad [12, 13], descritas en la tabla 1. Todos son igualmente importantes, y algunos están vinculados con otros, por ejemplo migración con fallas o con localidad.

Tabla 1. Niveles de Transparencia a Considerar en el Diseño del SOW

Transparencia	Descripción
Acceso	Esconde diferencias en la representación de datos y como un recurso es accedido.
Localización	Esconde la localización del recurso.
Migración	Esconde el movimiento de un recurso a otra localización.
Concurrencia	Esconde que un recurso pueda ser compartido por varios usuarios, competidores sin interferencia entre ellos.
Fallas	Esconde la falla y recuperación de un recurso.
Persistencia	Esconde si un recurso (software) está en memoria o disco.
Rendimiento	El Sistema se reconfigurará para mejorar el rendimiento cuando la carga varía.
Escalabilidad	El Sistema y las aplicaciones se podrán expandir escalarmente sin cambiar la estructura del sistema.

Debido a que los proyectos más cercanos al SOW son Jini y WOS™, comparémoslos en cuanto al *manejo, localización y agrupamiento de los recursos, y a la migración y replicación de los objetos Web*. Así, tenemos que, **Jini** maneja los recursos a través de servicios que no son registrados al incorporarse a la red, sino que se autoregistran ellos mismos, y en el momento de ser necesitado alguno, es encontrado a través de un proceso de búsqueda y descubrimiento ("lookup/discovery service"); además, solo permite búsqueda cuando todos los atributos del recurso están representados en la búsqueda, ya que busca por contenido y no por nombres ("Lookups"), la búsqueda puede ser local y externa; y agrupa *los recursos* en grupos predefinidos, llamados federaciones; finalmente, mueve objetos a través de Java RMI. Por otro lado, **WOS™**, maneja los recursos a través de un motor eductivo ("Eductive Engine"), usa un esquema de Versiones y autoconfiguración dinámica; maneja búsqueda de recursos a nivel local y a nivel de comunidades, para ello, utiliza repositorios *Pasivos* que almacenan información que podría ser remplazada, *Activos* que almacenan información obtenida de otros repositorios, y *Adaptativos* que almacenar perfiles o información ("Profiles") de usuarios y estadísticas de acceso; realiza un manejo dinámico de federaciones y permite la migración, no permite la replicación de objetos. Nuestro **SOW** maneja los recursos a través de un motor de Inferencia, usa Versiones y autoconfiguración dinámica; realiza la búsqueda de recursos en tres niveles: a nivel local (en los repositorios locales), a nivel remoto (en los repositorios remotos) y a nivel Global (en las Comunidades); agrupa virtualmente los nodos, en grupos dinámicos y emergentes, llamados comunidades; y posee módulos que le permiten la migración y replicación de objetos Web en cualquier nodo del SOW (ambos son procesos emergente).

3. Descripción Detallada del SOW Propuesto

El SOW adopta un modelo donde cada entidad tiene su propia identidad. Está integrado por las siguientes entidades [18]: *el subsistema manejador de recursos (SMR), el subsistema manejador de repositorios locales (SMRL) y remotos (SMRR), el subsistema manejador de objetos Web (SMOW) y el subsistema manejador de comunidades (SMC)* (ver figura 2).

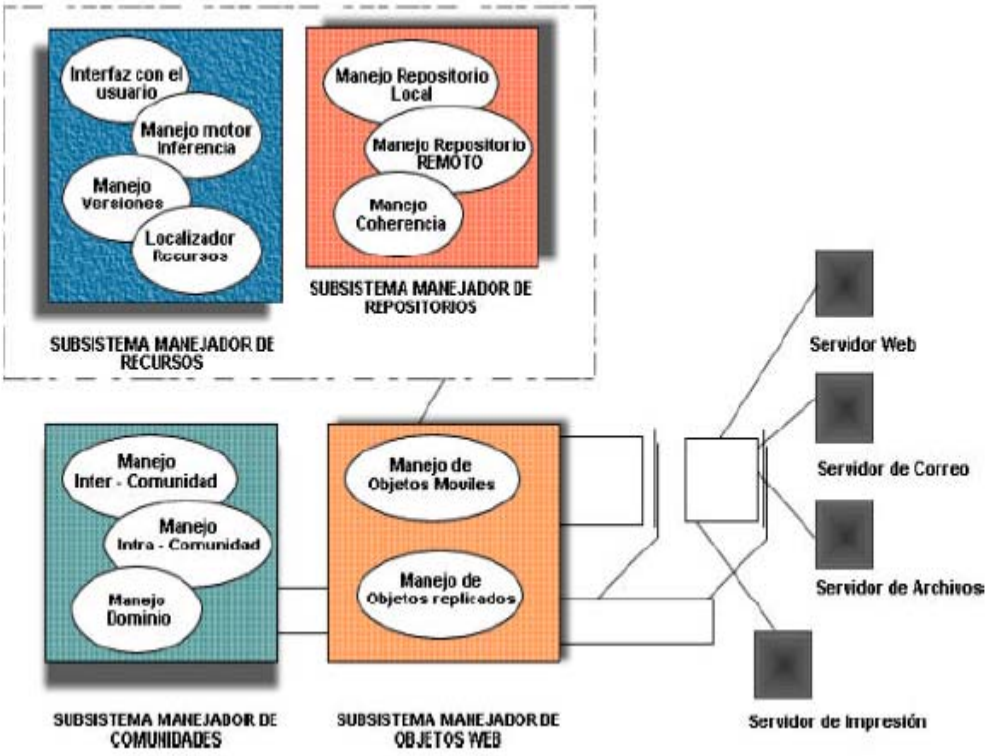

Figura 2. Subsistemas que integran el SOW

En nuestro SOW se presentan 3 niveles de búsquedas, coordinadas por el SMR para satisfacer una solicitud de servicio dada. A continuación se describen estos tres niveles de búsqueda [18].

I. Nivel Local: en primer lugar, se busca en los repositorios locales a través del SMRL. En caso de tener éxito, se ofrece tiempos de respuesta inmediatos, y en caso contrario, se pasa al siguiente nivel de búsqueda.

II. Nivel Remoto: se busca en los repositorios remotos a través del SMRR, dado que consultar el repositorio remoto, es consultar a los nodos vecinos, ya que en él se encuentran las referencias a los servicios remotos más recientemente utilizados. En caso de no conseguirlo, se pasa al tercer nivel de búsqueda.

III. Nivel de Comunidades: se busca a través del SMC en las comunidades existentes. En caso de no tener éxito, no puede ser satisfecha la solicitud.

3.1 Subsistema Manejador de Recursos (SMR)

En SOW, el Subsistema Manejador de Recursos está conformado por mecanismos que permiten administrar e integrar los recursos computacionales presentes en la Web. Se considera un recurso computacional Web a todo lo que pueda ser manipulado desde un nodo del SOW. Los recursos pueden tener representación física, como por ejemplo, un archivo o una impresora, o representaciones abstractas, como el tiempo de CPU. Debido a la naturaleza dinámica de los

recursos en la Web, y a su amplitud, es imposible desarrollar un catálogo con todos los recursos y servicios disponibles. Como consecuencia de esto no se puede desarrollar un simple Manejador de Recursos que por sí solo pueda administrar los recursos de la Web. Además, en lugar de proveer un soporte general para un recurso específico, se debe proveer soporte para un conjunto de diferentes versiones de un recurso. Así, por ejemplo, en vez de tener un sistema operativo que provea un conjunto fijo de técnicas de planificación de procesos o de caches, tendríamos un sistema operativo que provee los medios para designar a un requerimiento la técnica de planificación particular que sea necesaria en un momento dado. En esos casos se aplicará la idea del uso de versiones, las cuales podrían estar físicamente distribuidas por la red. Las versiones de los servicios en el ámbito de hardware y software son ubicuas en la Web. Sin embargo, aunque se tenga un conjunto completo de versiones para los distintos recursos, servicios o estrategias; es imprescindible poseer el medio que permita la adecuada utilización de éstas versiones, para esto se requiere de mecanismos de configuración de servicios.

Todos los requerimientos hechos al sistema son manejados por un motor de inferencia. Éste funciona como un sistema reactivo de manejo de demandas a través del cual se administran los recursos del ambiente computacional heterogéneo provisto por la Web. Particularmente, él realiza todo el proceso de búsqueda, determinación del servicio a proveer y asignación de recursos, entre otras cosas. El SOW posee varios motores de inferencia distribuidos en la red. Cuando un usuario intenta tener acceso a un recurso de la Web, el SOW recibe la petición, dicha petición es manejada por el motor de inferencia del nodo local, el cual consulta a sus repositorios para determinar si es posible satisfacer la petición localmente, de no ser así, la petición es enviada a otro motor de inferencia y se repite el proceso anterior hasta conseguir satisfacer la petición. El motor de inferencia está en capacidad de manejar peticiones provenientes desde un usuario o desde otro motor de inferencia remoto. Existe un protocolo de comunicación entre los motores de inferencia que establece con quien se debe comunicar un motor de inferencia para buscar un servicio, este protocolo pertenece al Sistema Manejador de Comunidades. En general, cada motor de inferencia tiene acceso a un repositorio de recursos locales y a un repositorio de recursos remotos. El repositorio de recursos locales siempre es consultado, en cambio el repositorio de recursos remotos sólo es consultado cuando no se puede satisfacer una petición de recursos en el ámbito local. Estos repositorios almacenan información referente a las versiones de recursos. Así, para satisfacer una petición de recurso, el motor de inferencia solicita a sus repositorios la ubicación del recurso con su respectiva versión. Si esta información está en los repositorios asociados al motor de inferencia, éste recibe como respuesta a esta solicitud la dirección del nodo que está en capacidad de satisfacer el requerimiento (si la respuesta la da el repositorio local, significa que el recurso está disponible localmente). En caso de que el motor de inferencia no pueda encontrar en sus repositorios la información requerida, el requerimiento se transfiere a un motor de inferencia de otro nodo.

3.2. Subsistema Manejador de Repositorios Locales (SMRL)

Uno de los grandes retos de SOW es implementar un algoritmo eficiente para la búsqueda y descubrimiento de servicios o recursos disponibles en la red. Los repositorios asociados con los nodos del SOW proveen la información necesaria para satisfacer las solicitudes de servicio. Así, cada nodo usa sus repositorios para almacenar y continuamente actualizar la información acerca del nodo, de los servicios y de los recursos disponibles en éste y en su entorno más cercano (entorno definido por el protocolo manejador de comunidades). Esto permite la interacción con muchos repositorios diferentes, cada uno ofreciendo por ejemplo, versiones diversas de los servicios disponibles, técnicas de manejo de recursos, aplicaciones, etc.

Los repositorios locales almacenan la información sobre los recursos locales, la cual debe ser mantenida actualizada. Al ser esa información quizás replicada en otros sitios, se deben tener mecanismos de coherencia de datos para mantener la información consistente. Además, el Subsistema Manejador de Repositorios Locales (SMRL) establecerá políticas para hacer un uso eficiente de la capacidad disponible en los dispositivos y, utilizará una estructura, como por

ejemplo tipo árbol, que facilite los procedimientos de búsqueda local a través de la implementación de algoritmos eficientes. Todo esto conlleva a establecer técnicas de optimización de almacenamiento, de planificación de acceso a disco, y de definición de patrones de acceso, entre otras cosas. Por otro lado, dado que la mayoría de las aplicaciones en Internet requieren interacciones en tiempo real y que es necesario compartir la información disponible, el SMRL podría establecer mecanismos transaccionales eficientes como la utilización de planificaciones de Entradas y Salidas paralelas, que logren ofrecer tiempos de respuesta aceptables, garantizando un uso óptimo del ancho de banda y la eliminación de conflictos.

3.3. Subsistema Manejador de Repositorios Remotos (SMRR)

Un problema que debe atacar un SOW es el no contar con un ancho de banda adecuado y permanente para transportar los objetos Web. Una de las maneras de afrontar esta deficiencia, es la de tratar de tener más cerca de los usuarios la información que requieren. En nuestra propuesta consideraremos los repositorios remotos como una caché en la Web, que tiene la capacidad de almacenar los objetos remotos más frecuentemente usados cerca de los usuarios, para que no sea necesario acceder a un objeto desde su ubicación original cada vez que se requiera. El aspecto más crítico en el diseño de estas cachés en la Web es que el tamaño de almacenamiento es finito. Por ello, se debe hacer uso de una política de reemplazo de objetos viejos, por nuevos que garantice el uso optimo de este espacio reducido por parte de los objetos más usados. La forma como se organizan las cachés en grupos para compartir objetos entre sí, también es importante para obtener una respuesta rápida a los requerimientos realizados al SOW. Otro elemento fundamental son los mecanismos para mantener coherente la información de estos repositorios.

Entre las ventajas de contar con una caché en la Web están la de reducir los tiempos de respuesta (al descargar objetos Web), eliminar el tráfico en la red (al tener la información requerida por el usuario cerca de él), aumentar la escalabilidad y disponibilidad de los servidores y otros recursos de la red, entre otros. El funcionamiento de la caché en la Web es el siguiente: si un usuario solicita un objeto, la caché realiza una copia local de ésta; cuando se requiera de nuevo el mismo objeto, la caché muestra su copia disponible, de ésta manera no es necesario requerirlo nuevamente al servidor de origen.

3.4. Subsistema Manejador de Objetos Web (SMOW)

Este subsistema maneja todos los tipos de objetos Web, pero particularmente los móviles y aquellos que deben ser replicados. Los objetos móviles son aquellos que se van migrando a través de los sitios de Internet según ciertos criterios. Por ejemplo, se ubican en los sitios de mayor demanda y ancho de banda en Internet, haciéndolo de manera autónoma e inteligente. Entre las ventajas que ofrece la migración de objetos en Internet tenemos, reducir los costos de comunicación, permitir interacciones en tiempo real, y hacer más tolerante a fallas al sistema. Por otro lado, los objetos Web pueden ser replicados, por lo cual se deben establecer políticas de gestión que garanticen la accesibilidad y la ubicación óptima de las réplicas en los repositorios remotos de la Web.

El diseño de las políticas de ubicación de los objetos móviles es basado en trazas. El objeto al moverse va dejando el rastro de sus movimientos, lo que podría utilizarse para encontrarlo al ser requerido por un usuario. Es decir, la estela que va dejando un objeto Web al desplazarse por distintos puntos de Internet da una forma de seguirle la pista. Por otro lado, el rastro dejado por los objetos al desplazarse debe ir desapareciendo a medida que transcurre el tiempo, al igual que ocurre, por ejemplo, con el feromona dejado por las hormigas.

En esta propuesta de un Sistema Operativo Web se plantea el diseño de estos objetos Web (móviles y replicados) como agentes de software, esto implica darle a los objetos Web una noción de dinamismo, autonomía e inteligencia para que puedan trasladarse por distintos nodos del SOW según los requerimientos de los usuarios, o definir cuando deben replicarse o autodestruirse.

3.5. Subsistema Manejador de Comunidades (SMC)

Los nodos interactuarán en el SOW a través de mecanismos de solicitud/respuesta y negociaciones. Aquellos nodos que exhiban afinidades funcionales y conductuales pueden dinámicamente asociarse para formar comunidades. Un nodo en el SOW está definido según sus habilidades y comportamiento. Son esos elementos los que se consideran como características del nodo. Así, un nodo según sus características se afilia como un todo (con sus recursos y elementos, entre otros) a una comunidad dada.

SOW trata a la comunidades desde un punto de vista emergente, es decir, las comunidades emergen y desaparecen por sí solas, se adaptan a su entorno y se autoorganizan de acuerdo a los requerimientos que surjan; careciendo de un control centralizado y mostrando un orden global (todo el grupo de nodos de una comunidad trabaja coherentemente, compartiendo sus informaciones y recursos entre ellos como si fueran un individuo) [14].

El Subsistema Manejador de Comunidades (SMC) tiene protocolos que se encargan de la comunicación entre los nodos de una comunidad, y entre las diversas comunidades. Además, tiene mecanismos para manejar los elementos pertenecientes a una comunidad dada. En otras palabras, hay protocolos que gestionan la creación, modificación y eliminación de comunidades y, en general, la incorporación, desincorporación, traslado o migración de nodos entre las diversas comunidades existentes.

4. Diseño de nuestro SOW como Sistema Multiagentes

El diseñar un Sistema Operativo Web conlleva tratar con el problema de heterogeneidad que están en los múltiples servicios que se ofrecen en la Web, a nivel del hardware y del software presentes en los diferentes sitios, y con el problema de adaptabilidad dinámica debido a la volatilidad de los recursos.

Realizar el diseño del SOW basado en SMA facilita el desarrollo de cada uno de los subsistemas del SOW por separado, ya que los SMA usan la estrategia “divide y vencerás”, logrando reducir la complejidad requerida para solucionar el problema de gestión a nivel global. Así, al diseñar nuestro SOW como un sistema multiagentes, hemos definido varios niveles de abstracción: un **primer nivel de abstracción**, denotado por $\mathbf{L_0}$, donde el SOW puede ser visto como un solo componente S, y donde además, la información interna y los procesos en él no son especificados (se ocultan las comunicaciones y los procesos internos en el SMA); un **segundo nivel de abstracción**, denotado por $\mathbf{L_1}$, donde el componente S puede ser visto como una colección de Subsistemas (agentes), que cumplen con los objetivos del componente S (SOW) (ver tabla 2); y por último, un **tercer nivel de abstracción**, denotado por $\mathbf{L_2}$, donde cada subsistema del SOW puede ser visto como una colección de agentes, los cuales intercambian información entre ellos y con el entorno, además de tener el control sobre sus tareas. Así, cada subsistema del SOW es descompuesto en un SMA, que cumple con sus objetivos y coordina las interacciones establecidas para cada uno de sus agentes, logrando de esta manera, satisfacer y cumplir con los objetivos de los subsistemas, y por consiguiente del SOW, a través de un eficiente trabajo en equipo. A continuación se describen la arquitectura, actores y casos de uso, agentes y tareas involucradas en cada uno los subsistemas del SOW. Para esa descripción usaremos la metodología MASINA [17].

Tabla 2. Descomposición del SOW en los Niveles de Abstracción L_0 y L_1

Sistema Multiagentes (Nivel L_0)	**S Descompuesto en Subsistemas(Nivel L_1)**	**Satisface Objetivos**	**Interrelacionado con**
SOW es S	**SMR**	Administrar recursos computacionales presentes en el SOW	**Usuarios, SMRL, SMRR y SMC**
	SMRL	Buscar información o recursos en los Repositorios locales	**SMR, SMRR y SMOW**
		Mantener coherente la información existente en los Repositorios locales	
		Administrar el uso eficiente de los dispositivos locales	
	SMRR	Buscar Información en los RR	**SMR, SMRL, SMOW**
		Mantener actualizada y coherente la información en los Repositorios Remotos	
	SMOW	Replicar objetos en los Repositorios Remotos	**SMRL, SMRR y SMC**
		Migrar objetos hacia nodos de mayor demanda	
	SMC	Buscar servicios en las comunidades existentes Agrupar nodos virtualmente con características similares	**SMR, SMOW**

4.1. Diseño del Subsistema Manejador de Recursos

4.1.1. Arquitectura

En el sistema manejador de recursos las operaciones son coordinadas por cuatro componentes fundamentales. Estos cuatro componentes constituyen las unidades funcionales básicas de la arquitectura del SMR. A continuación se presenta una breve descripción de cada una de estas unidades (ver figura 3).

Interfaz con el usuario: La unidad de interfaz con el usuario se encarga de recibir solicitudes de los usuarios, enviarla al sistema de razonamiento, y enviar los resultados de las solicitudes a los usuarios. En esta unidad se validan las entradas al sistema, y se analiza la sintaxis y la semántica de la información ingresada.

Sistema de razonamiento: Es la unidad corazón del sistema. En esta unidad se procesan los requerimientos recibidos de la interfaz con el usuario, con el fin de dar una respuesta adecuada a dichos requerimientos. Esta unidad posee un motor de inferencia capaz de coordinar inteligentemente los mecanismos de localización, configuración y asignación de servicios requeridos por los usuarios.

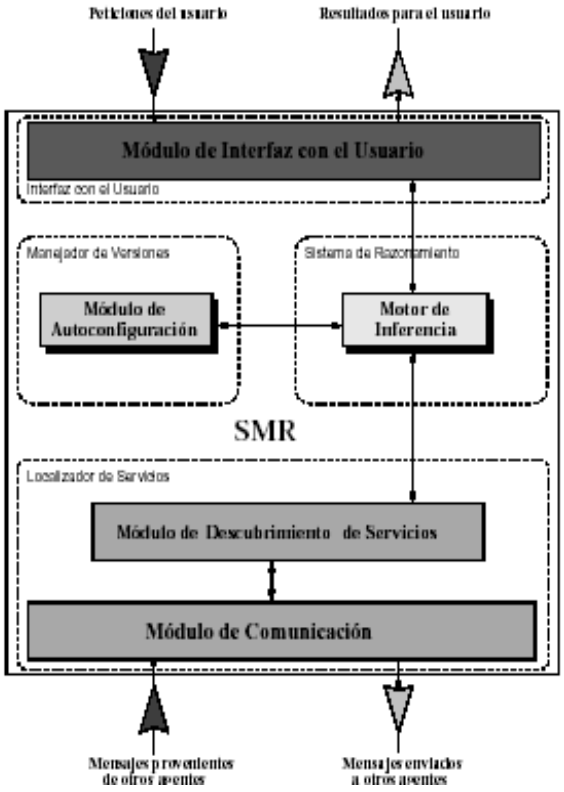

Figura 3. Arquitectura del SMR.

Localizador de servicios: La unidad localizador de servicio tiene como objetivo la búsqueda de los servicios y versiones que son requeridas por el motor de inferencia. A través del *módulo de descubrimiento de servicios* se obtiene la información relacionada con la ubicación física de los servicios y de las versiones requeridas. Para ello, la unidad localizadora de servicio hace uso del *módulo de comunicaciones*, el cual se encarga de enviar y recibir mensajes hacia y desde otros componentes externos del SMR.

Manejador de versiones: Esta unidad posee un módulo de autoconfiguración, el cual se encarga del manejo de las versiones de los servicios en función de la información provista por el motor de inferencia. Esta unidad tiene como resultado final la configuración de un servicio determinado de acuerdo con el requerimiento de los usuarios.

4.1.2. Agentes y Tareas Involucradas

Los agentes del SMR propuestos mantienen los papeles e interacciones desempeñadas por el SMR en el SOW [16], para esto, se proponen los siguiente agentes: *Solicitante*, *Administrador*, *Localizador y Configurador*. Las tareas que ejecutará el SMR se derivan también de los papeles desempeñados por el subsistema en el SOW. Las tareas identificadas se muestran en la tabla 3.

Tabla 3. Agentes y Tareas del SMR

Agente	Tareas	Descripción
Solicitante	Procesar Solicitud	A través de esta tarea el SMR recibe la información dada por el usuario al momento de realizar un requerimiento al SOW, esta información es validada y estructurada para luego ser enviada al agente Administrador como una solicitud de búsqueda del recurso requerido.
	Armar Respuesta	A través de esta tarea el agente solicitante recibe la información del agente administrador como respuesta a un requerimiento realizado por el usuario. Esta información es estructurada para luego ser presentada en una manera legible para el usuario.
Administrador	Coordinar Solicitud	Se reciben los parámetros de la petición y se construye la estructura de datos que será usada para iniciar la tarea de Descubrir Servicio.
	Caracterizar Versiones	Con esta tarea se inicia el proceso de análisis de la información suministrada de una petición de búsqueda de servicio, esto con el fin de definir los datos y las directivas que le serán suministradas al agente Configurador.
	Asignar Recurso/Servicio	Esta tarea Asigna y Desasigna el hardware y el software que ser usado para ejecutar el servicio requerido por el usuario.
	Ejecutar Servicio	Una vez que el agente tiene el servicio configurado según el requerimiento del usuario, se realiza la llamada al servicio para lograr su ejecución.
Localizador	Coordinar Búsqueda	Se reciben los parámetros de la petición y se construye el mensaje que ser enviado al SMRL, SMRR y SMC.
	Descubrir Servicio	A través de esta tarea el agente Localizador se comunica con el SMRL, SMRR y SMC con el fin de obtener información útil para la ubicación de un servicio y de la versión requerida.
Configurador	Configurar Servicio	Se construye la versión de servicio requerido a partir de la información suministrada por el agente Administrador.

4.2. Diseño del Subsistema Manejador de Repositorios Locales

4.2.1. Arquitectura

En el Sistema Manejador de Repositorios Locales (SMRL) del SOW las operaciones son coordinadas por cinco componentes fundamentales [16]. A continuación se presenta una descripción de cada una de éstas unidades (en la figura 4, se muestra la arquitectura del SMRL).

Coordinador de Búsqueda: es la unidad encargada de recibir las solicitudes de búsqueda de servicios de otros subsistemas como el SMR, el SMRR y el SMC. Además, por medio de su *módulo de coordinación* se encarga, por un lado, de controlar la descomposición, clasificación y

distribución de las solicitudes de búsqueda de servicios locales que llegan; y por otro lado, es el responsable de la integración de los resultados a emitir al subsistema solicitante.

Manejador de Recursos Locales: es la unidad encargada de gestionar eficientemente los recursos de hardware y software existentes en el nodo, para ello efectúa los procedimientos de búsqueda necesarios para proporcionar los recursos locales requeridos por el Coordinador de Búsqueda a través de su *módulo de búsqueda de Recursos Locales*; se encarga también de la actualización (inclusión, eliminación) del catálogo de recursos locales a través de su *módulo de actualización*.

Manejador de Información Local: es la unidad encargada de gestionar la información local existente en cada nodo, para ello efectúa los procedimientos de búsqueda necesarios para proporcionar la información local requerida por el Coordinador de Búsqueda a través de su *módulo de búsqueda de Información Local*; por otro lado, se encarga de la actualización (inclusión, modificación, eliminación) de la información local, y asimismo, de mantener la información consistente después de cualquier proceso de actualización, esto es logrado a través del trabajo conjunto de su *módulo de actualización* y su *módulo de coherencia de datos.*

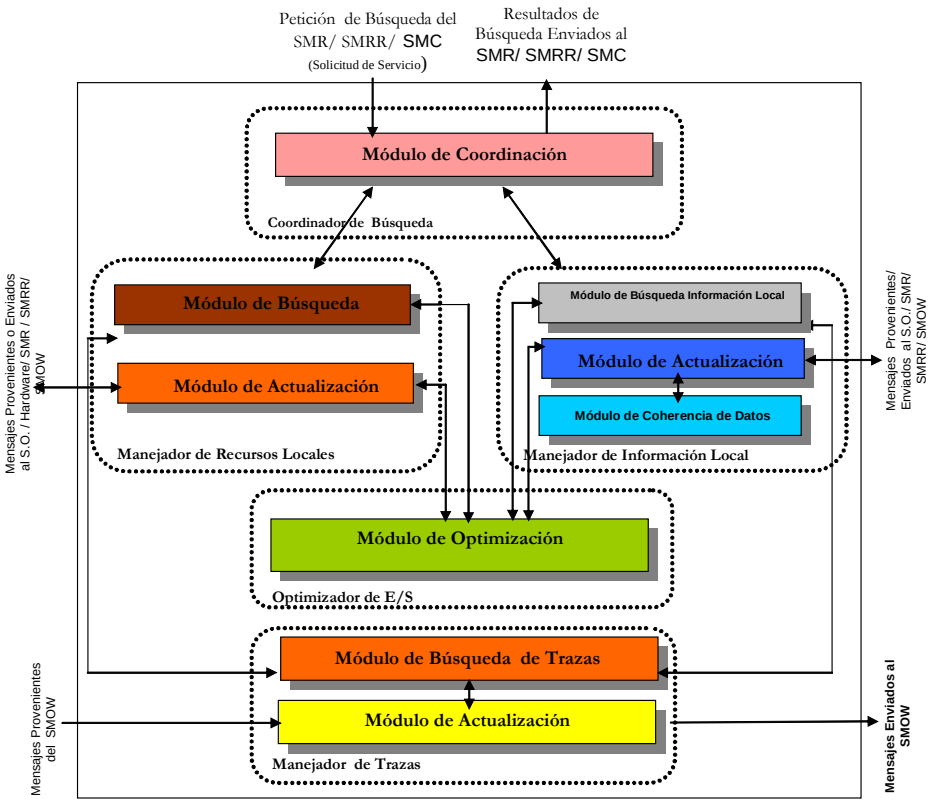

Figura 4. Arquitectura del SMRL

Optimizador de E/S: es la unidad encargada, a través de su *módulo de optimización*, de establecer políticas para hacer un uso eficiente de la capacidad disponible en los dispositivos, y de establecer técnicas de optimización y de acceso a éstos dispositivos eficientes a través de la planificación de acceso a disco y la definición de patrones de acceso, entre otras. Además, éste *módulo de optimización* utiliza mecanismos transaccionales para ofrecer tiempos de respuesta

aceptables, garantizar un uso óptimo del ancho de banda, del espacio de almacenamiento y evitar conflictos.

Manejador de Trazas: es la unidad encargada del manejo de las trazas a nivel local a través de su *módulo de actualización*. Esta unidad permite al SMRL participar en el proceso de creación, reforzamiento, evaporación y eliminación de trazas (actualización), en conjunto con el SMRR y el SMOW. Además, por medio de su *módulo de búsqueda de trazas* permite la localización de alguna traza en particular que sea solicitada o necesitada por el SMRR.

4.2.2. Agentes y Tareas Involucradas

El SMRL provee las siguientes funcionalidades de acuerdo con los actores y casos de uso definidos: *búsqueda de información y recursos locales, actualización de información y recursos locales, optimización en la realización de las dos primeras funciones y por último, la actualización de las trazas existentes*. Estas funciones son llevadas a cabo por los agentes: Coordinador, Administrador de Recursos Locales, Administrador de Información Local, Optimizador, y el Administrador de Trazas.

Tabla 4. Agentes y Tareas Involucradas en el SMRL

Agente	Tareas	Descripción
Coordinador	Tareas de coordinación	Recibir Solicitud de Servicio, Clasificar subpeticiones de la solicitud de servicio, Integrar Respuesta
Administrador de recursos locales	Tareas de administración de recurso local	Localizar Recurso Local, Almacenar Recurso Local, Eliminar Recurso Local, Recibir solicitud para asignación de Recurso Local, Recibir solicitud para desasignación de Recurso Local, Ejecutar Recurso Local
Administrador de informacion local	Tareas de administración de información local	Localizar Información Local, Almacenar Información Local, Eliminar Información Local, Manejar coherencia de datos
Optimizador	Tareas de optimización	Optimizar ejecución de Recurso Local, Optimizar capacidad de almacenamiento, Planificar acceso a disco, Definir patrones de acceso
Administrador de trazas	Tareas de administración de trazas	Crear Traza, Reforzar traza, Evaporar traza, Eliminar traza, Localizar traza

4.3. Diseñador del Subsistema de Repositorios Remotos

4.3.1. Arquitectura

En el sistema manejador de Repositorios Remotos (SMRR) del SOW las operaciones son coordinadas por dos componentes fundamentales (Figura 5), estos son: el Localizador de Objetos Web, y el administrador del Repositorio Remoto. Estos dos componentes constituyen las unidades funcionales básicas de la arquitectura del SMRR.

Localizador: Esta unidad tiene como objetivo la búsqueda de las réplicas y objetos referenciables que se encuentran en el SMRR y que son requeridos por el SMR. Para esto, utiliza el módulo “procesar solicitud”, el cual determina a qué tipo de objeto se refiere la búsqueda.

Además, a través del módulo "búsqueda de objetos" es posible dar a conocer al resto de componentes del SOW los objetos que se encuentran localizados en el SMRR.

Administrador: Esta unidad tiene como objetivo el mantenimiento del Repositorio del SMRR, para esto es necesario que este componente realice la eliminación, almacenamiento y mantenga la coherencia de las réplicas almacenadas en el RR. Esta unidad cuenta con tres módulos funcionales que le permiten realizar estas operaciones:

- **Módulo de Reemplazo:** Cuando el SMRR necesita almacenar una nueva réplica o un objeto referenciable, no siempre existe el espacio necesario en el RR, por ello este módulo elimina los objetos menos usados cuando el espacio sea requerido.
- **Módulo de Actualización:** Ingresa, elimina y actualiza las réplicas y los objetos referenciables en el RR, además de generar la solicitudes de réplicas al SMOW.
- **Módulo de Coherencia:** Su tarea es verificar la consistencia de la información que se encuentra en el RR.

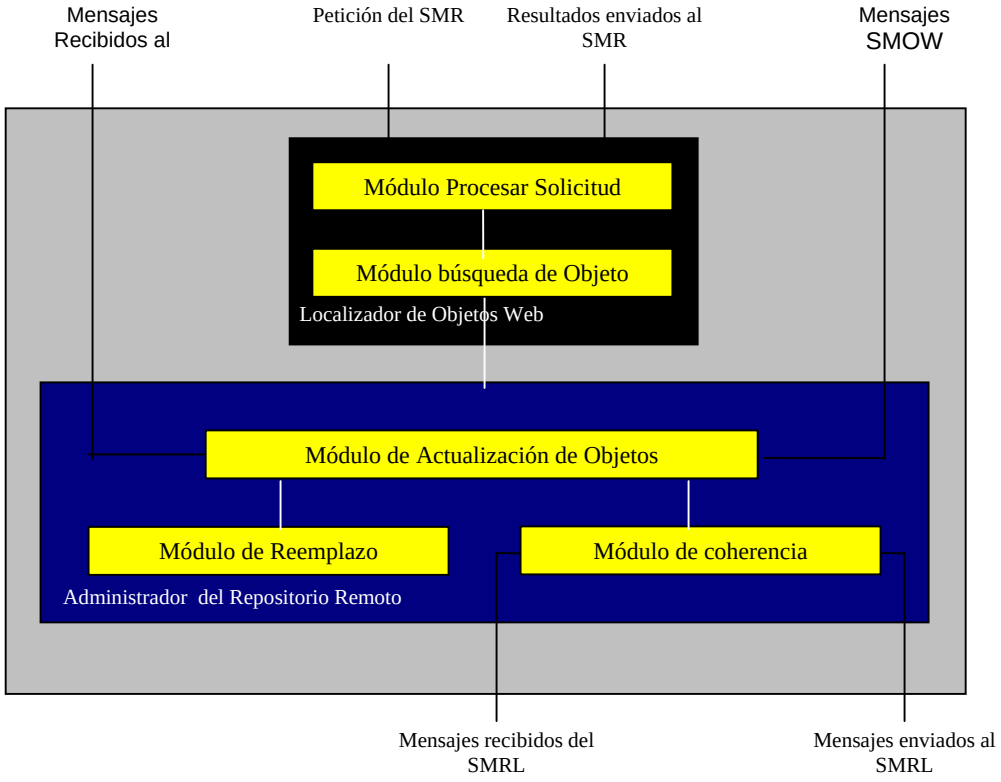

Figura 5. Arquitectura del SMRR

4.3.2. Agentes y Tareas Involucradas

Los agentes del SMRR propuestos mantienen los papeles e interacciones desempeñadas por el SMRR en el SOW [16], así, el SMRR esta compuesto por los siguientes agentes: Localizador de Objetos y el Administrador del RR. Las tareas identificadas se muestran en la tabla 5.

Tabla 5. Agentes y Tareas Involucradas en el SMRR.

Agente	Tareas	Descripción
Localizador	Localizar Réplica	El agente Localizador de Objetos realiza la búsqueda de réplicas en el interior del RR.
	Localizar Objeto Referenciable	El agente Localizador de Objetos realiza la búsqueda de objetos referenciables en el interior del RR.
	Actualizar Ubicación de Objeto	El agente Localizador necesita proveer los mecanismos que le permitan actualizar la ubicación de los objetos referenciables en el RR.
Administrador del RR	Almacenar Nueva Réplica	El Repositorio Remoto necesita renovar el inventario de objetos para ajustarlos a los nuevos requerimientos, para esto, almacena nuevas réplicas por solicitud del SMR.
	Almacenar Nuevo Objeto Referenciable	El Repositorio Remoto necesita almacenar los objetos referenciables en el RR por solicitud del SMR.
	Reestablecer Coherencia de Réplica	Con esta tarea el agente Administrador actualiza la réplica cuando se ha vuelto inconsistente.
	Actualizar Objeto fuente en RL	El Repositorio Remoto necesita proveer los mecanismos que le permitan actualizar aquellos objetos Web incoherentes que se encuentran en el RL y han sido modificados en el RR.
	Asignación de Réplica	Con esta tarea el agente concede el permiso de uso de los objetos que se encuentran en el interior del RR.
	Desasignación de Réplica	Con esta tarea el agente recibe la solicitud de conclusión del permiso de uso de las réplicas que se encuentran en el interior del RR.

4.4. Diseño del Subsistema Manejador de Objetos Web

4.4.1. Arquitectura

En el sistema manejador de Objetos Web (SMOW) del SOW las operaciones son coordinadas por dos componentes fundamentales (Figura 6), estos son: el Replicador de Objetos y el Migrador de Objetos. Estos dos componentes constituyen las unidades funcionales básicas de la arquitectura del SMOW.

Replicador de Objetos: Esta unidad tiene como objetivo el traslado de las réplicas de los objetos en el SOW. Para esto utiliza dos módulos, el *Módulo de Replicación*, que le permite realizar las réplicas; y el *Módulo de Comunicación*, que le permite la coordinación con los SMRL fuente y SMRR.

Administrador de Traslado de Objetos: Esta unidad tiene como objetivo la migración de objetos web en el SOW y la creación de trazas. Para esto utiliza tres módulos, el *Módulo de Migración de Objetos*, que le permite realizar el traslado de Objetos de un SMRL a otro SMRL, el *módulo de trazas* que permite la creación de trazas cuando un objeto es migrado; y *el Módulo de*

Comunicación que le permite la interacción con el SMRL donde se encuentra el objeto en el momento de iniciar la migración (fuente).

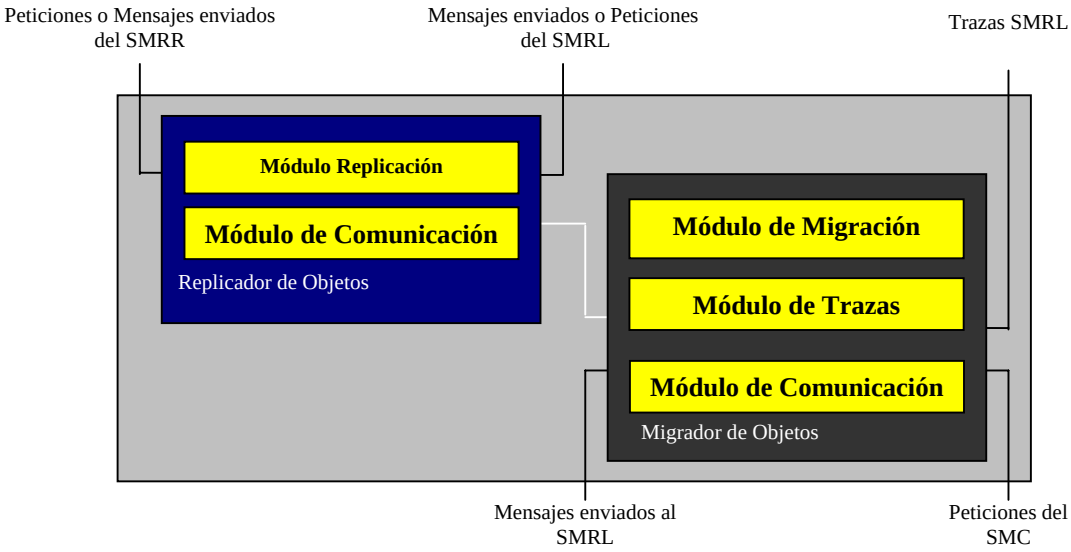

Figura 6. Arquitectura del SMOW

4.4.2. Agentes y Tareas Involucradas

El SMOW esta compuesto por los siguientes agentes: Administrador de Replicación de Objetos (ARO) y el Administrador de Migración de objetos (AMO) [16]. Las tareas identificadas se muestran en la tabla 6.

Tabla 6. Agentes y Tareas Involucradas en el SMOW

Agente	Tareas	Descripción
Administrador de replicación de Objetos (ARO)	Replicar Objetos	El agente ARO recibe la solicitud de replicar un objeto por parte de SMRR, para esto crea una réplica del objeto en el SMRL y luego la traslada al SMRR solicitante.
Administrador de Migración de Objetos AMO	Migrar Objetos	El agente AMO recibe peticiones de mover un objeto hacia otro nodo, este decide si migrar el objeto hacia el destino sugerido en base a las demandas de los dos nodos y si el nodo origen presenta problemas de conexión. Si el agente AMO decide migrar el objeto, este mueve el objeto hacia el SMRL destino.
	Crear Traza	El agente AMO necesita crear trazas cada vez que realiza la migración de un objeto.

4.5. Diseño del Subsistema Manejador de Comunidades

4.5.1. Arquitectura

En el Sistema Manejador de Comunidades (SMC) del SOW, las operaciones son coordinadas por dos componentes fundamentales, éstos son: el manejador de comunidades y el manejador de búsqueda. Estos dos componentes constituyen las unidades funcionales básicas de la arquitectura del SMC. A continuación se presenta una descripción de cada una de éstas unidades (ver figura 7).

Manejador de Comunidades: Es la unidad principal del SMC, ya que está encargada de gestionar las comunidades existentes en el SOW. Caracteriza una nueva comunidad que emerge antes de ser agregada al SOW a través de su *módulo de caracterización de comunidades*; con su *módulo de actualización* se encarga de la creación, eliminación, y modificación de una comunidad dada; además, permite localizar alguna(s) comunidad(es) con ciertas características o perfil a través de su *módulo de búsqueda de comunidades*, y por último, por medio de su *módulo de caracterización de nodos* se encarga de describir un nodo para así conocer explícitamente sus características antes de que pueda ser incorporado a una comunidad dada.

Manejador de Búsqueda: Es la unidad encargada de recibir peticiones de búsqueda de servicios de un SMR o de otro SMC, y procesarlas por medio de su *módulo de procesamiento de servicios.* Realiza la búsqueda de un servicio dado, en primer lugar, a través de su *módulo de búsqueda interna,* y en caso de no tener éxito, a través de su *módulo de búsqueda externa*.

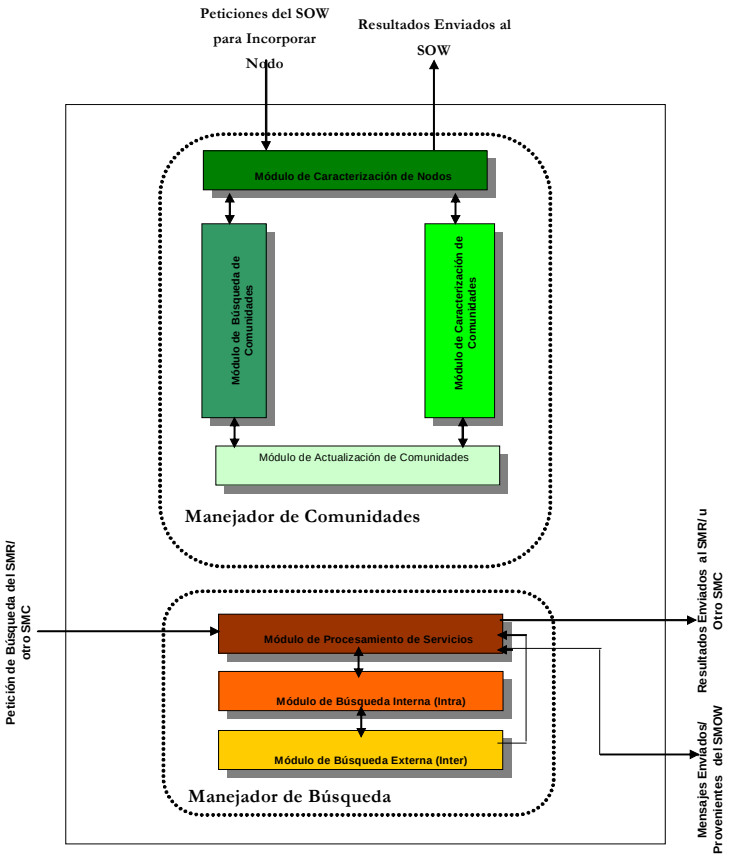

Figura 7. Arquitectura del SMC

4.5.2. Agentes y Tareas Involucradas

Según los actores y los papeles definidos, tenemos que el SMC provee las siguientes funcionalidades: *actualización de comunidades y la localización de servicios a nivel de comunidades*. Manteniendo a todos los actores como agentes de nuestro sistema, podemos identificar dos agentes: Agente Administrador de Comunidades y Agente Coordinador de Búsqueda.

Tabla 7. Agentes y Tareas Involucradas en el SMC

Agente	Tareas	Descripción
Administrador de comunidades	Tareas de administración	Recibir Solicitud para Incorporar Nodo, Caracterizar Nodo, Determinar Tipo de Comunidad, Crear Tipo de Comunidad, Localizar Comunidad, Incorporar Nodo a una Comunidad, Crear Comunidad, Caracterizar Comunidad, Eliminar Nodo de una Comunidad, Eliminar Comunidad, Transferir Nodo (es virtual en la comunidad), Inhibir Nodo, Reactivar Nodo
Coordinador de búsqueda	**T**areas de localización	Procesar Solicitud de Búsqueda de Servicio, Localizar Servicio Internamente, Localizar Servicio Externamente, Emitir Respuesta, Registrar Búsquedas Exitosas Internas, Registrar Búsquedas Exitosas Externas, Registrar Búsquedas Exitosas de Objetos Migrables 2.8 Solicitar Migración de un Objeto

4.6. Modelo de Coordinación, Comunicación e Inteligencia

El diseño de cada uno de los subsistemas del SOW según la metodología MASINA[17], permite mostrar las comunicaciones existentes (actos de habla) entre los agentes involucrados. Estas se pueden distinguir a través de un *diagrama de interacción de UML*, donde se representan las interacciones entre los agentes del sistema, por ejemplo, la conversación: *Actualizar comunidad* [16], del agente administrador de comunidades (AC) del SMC, es mostrada en la figura 8.

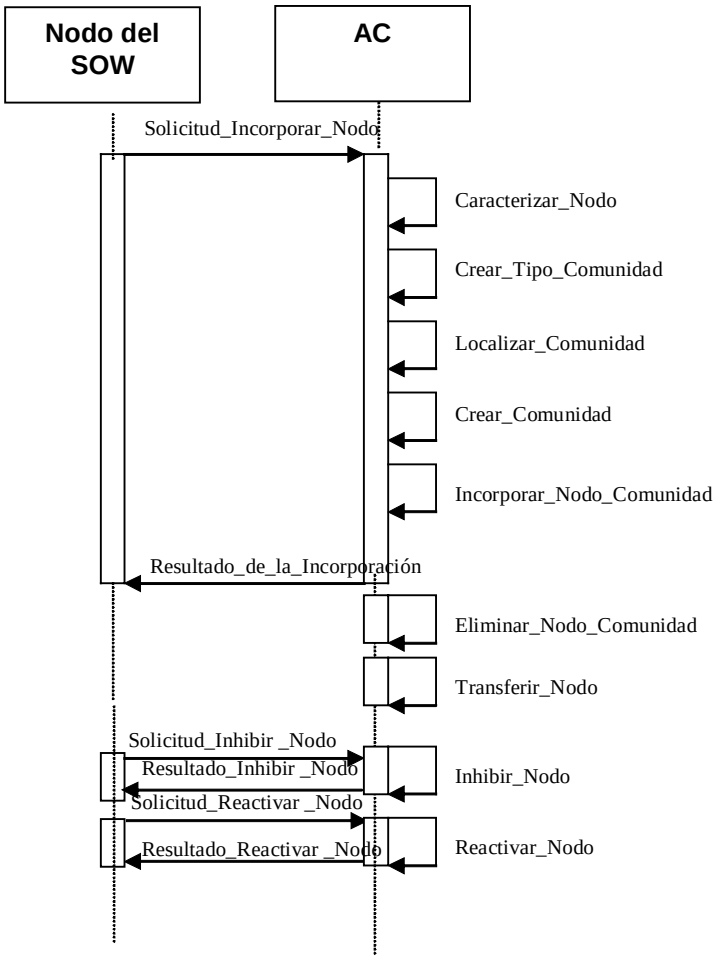

Figura 8. Conversación *Actualizar Comunidad*.

Además, MASINA permite describir la inteligencia presente en alguno de los agentes a través del modelo de inteligencia, en otras palabras, permite plasmar las capacidades de razonamiento de algún agente. Por ejemplo, a continuación se presenta el modelo de inteligencia para el agente administrador de Comunidades (AC) del SMC [16].

Tabla 8. Mecanismo de Razonamiento del agente AC

	MECANISMO DE RAZONAMIENTO
FUENTE DE INFORMACIÓN	Resultados obtenidos del agrupamiento de nodos.
FUENTE DE ACTIVACIÓN	Solicitud para incorporar nodo; transferir nodo.
TIPO DE INFERENCIA	Deductiva, Inductiva.
ESTRATEGIA DE RAZONAMIENTO	Coordinar actividades relacionadas al agrupamiento de nodos, actualización y creación de comunidades, a través de herramientas de reconocimiento de patrones basadas en redes neuronales y lógica difusa (Motor de reglas difuso).

Tabla 9. Mecanismo de Aprendizaje del Agente AC

	MECANISMO DE APRENDIZAJE
NOMBRE	Sistema de Agrupamiento
TIPO	Adaptativo
TÉCNICA DE REPRESENTACIÓN	Reglas
FUENTE DE APRENDIZAJE	Información histórica, proveniente de los procesos de agrupamiento de nodos efectuados.
MECANISMO DE ACTUALIZACIÓN	Modificación de reglas. El conocimiento es actualizado considerando las experiencias previas.

Tabla 10. Experiencia del Agente AC

	EXPERIENCIA
REPRESENTACIÓN	Reglas
TIPO	Basada en casos.
GRADO DE CONFIABILIDAD	Moderado

Tabla 11. Conocimiento Estratégico del Agente AC

	CONOCIMIENTO ESTRATEGICO
VALOR DEL CONOCIMIENTO	Agrupamiento de Nodos
AGENTE QUE LO PRODUCE	Agente Administrador de Comunidades
PROCESO QUE LO GENERÓ	Incorporación, eliminación o transferencia de un nodo.
GRADO DE CONFIABILIDAD	Alto

El conjunto de modelos de comunicación y coordinación de las diferentes conversaciones y actos de habla del sistema, así como los modelos de inteligencia del resto de los agentes, se encuentran descritos en [16]

5. Conclusiones

A pesar de los grandes esfuerzos realizados hasta ahora, las arquitecturas de sistemas operativos y de Middleware tradicionales no están preparadas para proveer un manejo eficiente de recursos para la Web. Se propone un modelo de SOW que intenta dar respuesta a éstas necesidades. El modelo fundamental propuesto está conformado por cuatro subsistemas que llevan a cabo una serie de funciones coordinadas que permiten un uso eficiente de los recursos sobre Internet, a pesar de sus características dinámicas y de heterogeneidad. El subsistema manejador de recursos ofrece los medios para la localización y asignación de recursos en el SOW, y además, es el responsable de actividades como el manejo de versiones. El subsistema manejador de repositorios proveerá

técnicas para organizar la información almacenada en los repositorios locales y remotos, y también, ayuda a coordinar el acceso a los mismos. El subsistema manejador de comunidades establece mecanismos que permiten agrupar eficientemente los recursos de la Web en comunidades organizadas. Finalmente, el subsistema manejador de objetos Web provee mecanismos para gestionar la migración y replicación de los objetos Web sobre la Web.

Referencias

[1] Vahdat A., Belani E., Eastham P., Yoshikawa C. "WebOS: Operating System Services for Wide Area Applications". In Seventh IEEE Symposium on High Performance Distributed Systems, pp. 52-63, 1998.

[2] Kropf P. "Overview of the WOS Project". In *Advanced Simulation Technologies Conferences, High Performance Computing*, pp. 350-356, 1999

[3] Baratloo, M., Karaul, M., Kedem, Z., Wykoff, P. "Charlotte: Metacomputing on the Web". *Journal on Future Generation Computer Systems*, 15 (5-6):559-570, 1999.

[4] Casanova, H., Dongarra, J. "NetSolve: A Network-Enabled Server for Solving Computational Science Problems". In *International Journal of Supercomputer Applications and High Performance Computing*, 3 (11):212-223, 1997.

[5] Foster, I., Kesselman, C. "Globus: A Metacomputing Infraestructure Toolkit". *Supercomputer Applications*, 11 (2):115-128, 1998.

[6] Morgan, S. "Jini to the Rescue". *IEEE Spectrum*, 37(4):44-49, 2000.

[7] Christiansen, B., Cappello, P., Ionescu, M., Neary, M., Schauser, K., and Wu, D. "Javelin: Internet-Based Parallel Computing Using Java". *Concurrency: Practice and Experience*, 9(11):1139-1160, 1997

[8] Van Steen, M., Homburg, P., Tanenbaum, A. S., "The Architectural Design of GLOBE: A Wide-Area Distributed System", *Technical Report IR-422, Vrije Universiteit Amsterdam*, 1997.

[9] Grimshaw, A.,Wulf, W., French, J., Weaver, A., and Reynolds,P. "A Synopsis of the Legion Project," *Technical Report No. CS-94-20, University of Virginia*, 1994.

[10] Camiel, N., Nisan,N., London, S., ans Regev, O. "Globally Distributed Computations over the Internet – the Popcorn Project". In *Proceedings of the International Conference on Distributed Computing Systems*, pp.592-601, 1998.

[11] O'Reilly, T. "Building the Internet Operating System". In *O'Reilly Emerging Technology Conference*, pp. 1-2, 2002.

[12] Coulouris G., Dollimore J., Kindberg T. "Distributed Systems, Concepts and Design". *Addison Wesley*, 2001.

[13] Kon F., Campbell R., Ballesteros F. "2k: A Distributed Operating System For Dinamic Heterogeneous Environments" In *Ninth IEEE International Symposium on High Performance Distributed Computing (HPDC'00)*, pp. 201-207, Pittsburgh, USA, 2000.

[14] Schut, M., Meijer J. "Self Organising Systems – Lecture Notes". *Vrije Universiteit Amsterdam*, 2001.

[15] Weiss Gerard. "Multiagent Systems". *Massachusetts Institute of Technology*, 1999.

[16] Aguilar, J., Ferrer, E., Perozo, N., Vizcarrondo, J. "Especificación Detallada de un Sistema Operativo Web usando Agentes", *Informe Técnico del CEMISID, Universidad de los Andes*, Diciembre 2003.

[17] Aguilar, J. "Especificación Detallada de los Agentes del SCDIA – MASINA CommonKADS". *Reporte Técnico del 3er. Año, Proyecto Agenda Petróleo* Nro. 97003817, 2003.

[18] Aguilar, J., Ferrer, E., Perozo, N., Vizcarrondo, J. "Propuesta de un Sistema Operativo WEB", *Proceedings of the TECNOCOM 20003, 3era Feria y Seminario de Informática, Electrónica y Telecomunicaciones*, (12 páginas, CD), Medellín, Colombia, Mayo 2003.