COMUNIDAD WEB CACHÉ

KEVIN MORAGA

Un Modelo de Caché Web Distribuido Sobre el protocolo HTTP y P2P

Noviembre 2014 – Version 1.0

Kevin Moraga: *Comunidad Web Caché,* Un Modelo de Caché Web Distribuido Sobre el protocolo HTTP y P2P, © Noviembre 2014

SUPERVISOR:
Isaac Ramírez

UBICACIÓN:
San José, Costa Rica

FECHA:
Noviembre 2014

Solo un tonto no tiene *miedo*.
El *valor* es ver el miedo y seguir adelante de todas formas.

— Julian Assange

Dedicado a mis padres, abuela, hermanos y hermanas que han estado siempre a mi lado.

ABSTRACT

Short summary of the contents...

PUBLICACIONES

Some ideas and figures have appeared previously in the following publications:

Put your publications from the thesis here. The packages `multibib` or `bibtopic` etc. can be used to handle multiple different bibliographies in your document.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth [4]

AGRADECIMIENTOS

Les agradezco a todas aquellas personas que han dedicado un poco de su tiempo, invirtiéndolo en mi formación y la de muchos otros alumnos de este respetado Instituto.

En torno al conocimiento y aporte de ideas, le agardezco a Nereo Campos, Herson Esquivel, Juan Carlos Brenes.
En apoyo relacionado a temas de la tesis, Wathsala Vithanage, Nuwan Gunaratne y Nishshanka Sirisena [1], quienes fueron muy amables en responder todas las dudas que tuve en relación a su proyecto y a toda LaTeX-community por el soporte, ideas y el grandioso software.

1 Miembros de diseño de Dalesa Cache

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE LISTADOS

ACRÓNIMOS

CWC Comunidad Web Caché

API Application Programming Interface

UML Unified Modeling Language

CV Comunidad Virtual

Parte I

INTRODUCCIÓN

Como parte de una introducción, se describirán las razones que impulsaron el desarrollo de ésta investigación, mostrando el problema su justificación y además cuáles son los objetivos propuestos.

1

INTRODUCCIÓN

Durante los últimos años el incremento en el uso de Internet ha sido exagerado, diariamente aparecen cientos de sitios web ofreciendo información muy variada, el acceso a Internet paso de estar en una cuantas manos a ser usado diariamente por la mayoría de las personas en el mundo. La expansión del Internet se consolido durante el aparecimiento del protocolo HTTP en la década de los noventas, este protocolo vino a establecer una forma universal para intercambiar información, el protocolo se baso en la arquitectura cliente-servidor, en esta arquitectura un cliente realiza la petición de algún recurso al servidor y el servidor satisface esta petición mediante el envió del recurso solicitado. Debido al auge del protocolo HTTP se dio la expansión del Internet, esta expansión trajo consigo que se creara una gran red de comunicación a nivel mundial, en estos días por ejemplo podemos acceder a información de último minuto en cualquier parte del mundo y la mayor cantidad de internautas, poseen su propio sitio web o alguna página personal hospedada por algún servidor gratuito. Conforme el uso Internet fue creciendo, también la tecnología fue evolucionando, llegando a que actualmente podemos acceder a gran cantidad de información instantáneamente y la mayoría de los usuarios de Internet ignoran todo lo que implica esto, cada vez que acceden a un gigabyte de información situado en el otro lado del mundo no tienen una idea de la tecnología involucrada ni mucho menos la inversión realizada para poder entregar esta información en un tiempo aceptable. Ahora nos preguntamos ¿qué justifica esta mejora en la tecnología tanto desde el punto de vista de protocolo así como dispositivos involucrados?, la respuesta es muy sencilla, al ser HTTP un protocolo tan versátil, un lenguaje universal para el intercambio de datos, se comenzó a utilizar para diferentes aplicaciones, se paso de un simple intercambio de información, a ser la piedra angular por la cual se mueve la mayoría de las actividades humanas, hoy en día tenemos muchos usos, desde la aparición de la multimedia, los requerimientos de todas las partes involucradas (cliente, servidor y medio de comunicación) se ha incrementado, hoy en día compartimos videos de gran tamaño, se transmite televisión en alta definición, se comparte música, documentos, en fin un gran número de archivos de diferentes tamaños, que hasta hace unos años con nuestra conexión de 56Kbps parecía imposible. Pero ¿acaso esta maravilla tecnológica es gratuita?, y la respuesta es no, como lo mencionamos anteriormente cada parte involucrada requiere adquirir la nueva tecnología, en el caso de un cliente imaginamos a una persona en su casa la cual

paga su conexión a Internet de banda ancha para compartir archivos, revisar su correo electrónico, escuchar música, podemos hablar de ver películas, participar de algún juego en línea, participar en video conferencias; con estos requerimientos el equipo requerido es bastante costoso, pero digamos que no inaccesible, ahora vámonos hacia el otro extremo, el servidor, supongamos un servidor que hace transmisión de videos, los cuales son aportados por los usuarios del sitio que se encuentra hospedado en este servidor, imaginemos que el sitio recibe un gran número de visitas, hablemos de más de 10000 visitas diarias, ya es un numero grande, ahora usted como lector y experto se dirá, resuelvo fácilmente este problema, compro un gran número de servidores, con los cuales utilizando una solución de balanceo de carga, puedo hacer frente a este gran número de usuarios, pensemos por un momento en nuestras palabras tomemos el tiempo e investigamos el precio de un servidor, de una solución de NLB y de una conexión de Internet lo suficientemente buena como para servir videos, ya los números son bastante grandes, lo más preocupante es que nunca es suficiente, siempre tendremos que estar actualizándonos ya que lo único constante es que la tecnología es cambiante. Debido al problema expuesto anteriormente, aparece como solución el caching, en su definición más sencilla podemos decir que es un modelo mediante el cual, objetos que se encuentran hospedados en un sitio web son almacenados en otros servidores de forma que pueden ser servidos desde otros servidores distintos al servidor web original, esto reduce el número de peticiones que llegan al servidor original lo que disminuye su carga (en la sección 3 de este documento se repasaran los modelos de web caching existentes), el caching ha sido un tema de estudio en cientos de tesis e investigaciones, pero las soluciones usadas actualmente se limitan a soluciones que deben ser adquiridas por el dueño del sitio web y como hablamos no es muy rentable, y soluciones que son adquiridas por un cliente en su beneficio. Al parecer llegamos a un callejón sin salida. Desviemos nuestra atención de nuestra reflexión hacia dos aspectos de suma importancia que fueron traídos con el auge del Internet, el primero es el establecimiento de comunidades virtuales, en las cuales un grupo de Internautas se reúnen alrededor de un sitio web que presenta información de su interés y como comunidad ayudan para mantenerlo y tienen su propio conjunto de reglas, y el segundo aspecto es el compartir archivos en Internet, tal vez el más polémico de todos y el que más nos interesa es el protocolo P2P, el cual básicamente permite que un número de usuarios que cuentan con un archivo especifico lo compartan con otros de manera simultánea, esto quiere decir que si 100 usuarios tienen un archivo de mi interés y este tiene un tamaño de 100MB, en lugar de bajarlo desde un solo servidor, bajo pequeñas partes desde cada uno de los usuarios que lo tienen, con esto el proceso de descarga es mucho más rápido y no se satura a un solo servidor. Para ir finalizando esta extensa intro-

ducción, el presente documento pretende introducir una posible solución de bajo costo para resolver el problema que se ha expuesto, esta solución la hemos llamado Community Web Caching, si está interesado en conocer como combinamos P2P, HTTP, comunidades virtuales, un servidor web y múltiples clientes para crear sitios web robustos, con pocos recursos, le recomendamos que continúe con esta lectura y cualquier comentario o recomendación estaremos sumamente complacidos en atenderlo.

1.1 COMUNIDAD WEB CACHÉ

La idea del Community Web Caching nace a partir de las siguientes premisas:

1. Los usuarios cuentan con una gran cantidad de recursos en sus computadoras, los cuales son desperdiciados en más de un 80 %.

2. La mayoría del contenido en Internet es de interés común y no solo de la persona que lo publica, entonces si es de interés común porque solo unos cuantos deben mantenerlo. Porque no lo podemos mantener entre todos.

3. Quien publica contenido en Internet muy pocas veces tienen la capacidad para mantener grandes servidores que puedan servir a grandes cantidades de usuarios.

4. Así como existen comunidades en Internet que se conforman por un interés común, por ejemplo hacer amigos, deportes, etc. Porque no crear una comunidad alrededor de un sitio web.

Comunidad Web Caché Version 1.0

La propuesta que se quiere implementar es dejar de lado el modelo tradicional de servir archivos mediante HTTP y dejar de lado también los modelos tradicionales de Web Caching, para utilizar las ventajas de ambos mas la creación de comunidades de caches y el uso de los recursos de los miembros de las comunidades caches y de esta manera utilizar algún protocolo, en este caso P2P para servir archivos desde diferentes localizaciones.

En este caso supongamos el siguiente ejemplo, contamos con el sitio www.cwc.org que sirve archivos de video de más de 30Mb, con un modelo tradicional HTTP, cliente-servidor, un cliente obtendría el archivo desde el servidor imaginemos 1000 clientes haciendo esto, sería demasiado para el servidor si tiene recursos limitados, entonces tendríamos que pensar en aumentar los recursos del servidor, entre otras cosas. Nuestra propuesta es crear una comunidad alrededor de www.cwc.org supongamos que 30 de esos 1000 clientes forman parte de la comunidad y que tienen los recursos suficientes para

servir archivos, entonces con esto podemos servir los archivos desde 31 lugares en lugar que desde uno y es evidente las ventajas que nos puede dar un modelo como este. Ahora que pasa si miembros de la comunidad no tienen suficientes recursos, bueno podemos hacer que en lugar de servir un archivo desde una sola localización se haga desde múltiples que queremos decir con esto, que en lugar de servir archivos grandes de 30MB o mas que sirvan solo una porción de estos, como en una comunidad en la cual cada miembro aporta lo que puede en beneficio de todos y para esto se utilizaría un protocolo como P2P, otra idea podría ser que los clientes con menos recursos sirvan archivos pequeños como por ejemplo imágenes. Esto le quitaría gran cantidad de trabajo al servidor ya que una imagen significa un thread más que se debe crear para servir un archivo.

La idea es que estas comunidades se conformen de manera automática y voluntaria, esto quiere decir que un usuario instala el plugin en su navegador web (Firefox) y el sitio al que trata de acceder puede usar el protocolo CWC entonces los contenidos accedidos quedan automáticamente en la cache y pueden ser servidos.

La cache podrá tener dos modos de operación, el primero sería que sea administrada por el usuario, esto quiere decir que cuando el usuario accede a un contenido este queda automáticamente en la cache y el usuario puede decidir si dejarlo o eliminarlo, el segundo seria manejado por el servidor esto quiere decir que cuando entramos en este modo el servidor tendrá derecho a utilizar los recursos asignados a la cache, entonces el servidor podrá enviar archivos que considere deberían de estar en la cache o borrar los que no considere necesarios.

Parte II

MARCO TEÓRICO

En los siguente capítulos se aclararán temas relacionados a los protocolos utilizados por CWC, como lo son HTTP y P2P. Además establecerán todas aquellas definiciones necesarias para el desarrollo de la presente investigación.

2 PROTOCOLO HTTP

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2.1 ¿CÓMO TRABAJA?

El protocolo HTTP es un protocolo petición/respuesta. Un cliente envía una petición al servidor en la forma de un método de petición, URI, versión de protocolo, seguido de un mensaje estilo- MIME conteniendo los modificadores de la petición, información del cliente y posiblemente el contenido del cuerpo sobre una conexión con el servidor, el servidor resuelve con un mensaje de estatus, incluyendo la versión del protocolo del mensaje y el código de error o éxito, seguido de un mensaje estilo-MIME que contiene la información del servidor, la meta-información de la entidad y posiblemente el contenido del cuerpo de la entidad. La conexión más simple seria en la cual el agente de usuario realice una conexión directa con el servidor de origen, entonces la conexión de una petición seria:

Figura 1: Conexión HTTP

En conexiones más complejas pueden intervenir uno o más intermediarios, como lo pueden ser:

- Proxies que son agentes de reenvío, este recibe peticiones para un URI en su forma absoluta, sobre escribe toda o parte del mensaje y reenvía las peticiones re-formateadas hacia el servidor identificado por la URI.
- Gateway es un agente de recepción, actuando como una capa sobre otros servidores y si es necesario traduce la petición al protocolo del servidor subyacente.
- Túnel: Actúa como un punto de transmisión entre dos conexiones sin cambiar el mensaje, los túneles son usados cuando la conexión pasa a través de un intermediario (como un firewall) aun cuando el intermediario no entiende los contenidos del mensaje.

2.2 MENSAJES HTTP

TIPOS DE MENSAJES: Los tipos de mensajes HTTP son de petición de un cliente hacia el servidor y de respuesta de un servidor hacia el cliente.

```
"HTTP-message = Request | Response ; HTTP/1.1 messages"
```

Los mensajes utilizan la forma genérica de mensajes, definido en RFC822 para transferencia de mensajes. Ambos mensajes consisten una línea de inicio, uno o más valores de encabezado, una línea vacía que indica el final de los valores de encabezado y posiblemente un cuerpo del mensaje.

ENCABEZADOS DEL MENSAJE: Los valores HTTP, los cuales incluyen general-header, request-header, response-header y entityheader, siguen el mismo formato genérico del RFC822. Cada campo del encabezado consiste de un nombre seguido por ":" y un valor. Los nombres de campos son case-insensitive y el valor se puede extender múltiples líneas.

CUERPO DEL MENSAJE: El cuerpo del mensaje de un mensaje HTTP es utilizado para trasportar el cuerpo de la entidad asociado con la respuesta o la petición. El cuerpo del mensaje difiere del cuerpo de la entidad solo cuando una codificación de transferencia ha sido aplicada. "message-body = entity-body | entity-body codificado" El campo Transfer-Encoding debe ser utilizado para indicar cualquier codificación de transferencia aplicada por una aplicación para asegurar la transferencia adecuada del mensaje. Transfer-Encoding es una propiedad del mensaje no de la entidad.

LARGO DEL MENSAJE: El transfer-length de un mensaje es el largo del cuerpo del mensaje a como este aparece en el mensaje, que es, sin que se le allá aplicado ningún filtro. Cuando el cuerpo del mensaje es incluido con un mensaje, el transfer-length de ese cuerpo es determinado por uno de:

- Cualquier mensaje de respuesta que no incluya cuerpo del mensaje como las repuestas estatus 204 y 304 son siempre terminadas con la primer línea vacía después de los campos de encabezado presentes en el mensaje.
- Si un campo de Transfer-Encoding se encuentra especificado y tiene cualquier valor diferente a "Identity", entonces el transfer-length está determinado por el uso de "chunked" transfer-coding, a menos que el mensaje se terminara por que se cerró la conexión.
- Si un campo Content-Length se encuentra presente, su valor decimal en octetos representa el entity-length y transfer-length. Content-Length no se debe enviar si los dos anteriores no son iguales.
- Si el mensaje utiliza el tipo de media "multipart/byteranges" y el largo de trasferencia no está especificado, entonces este tipo de media determina el transfer-length. Este tipo de media no debe ser utilizado a menos que quien envía conozca que el recipiente lo puede parsear.
- Por un cierre de conexión por parte del servidor

2.3 PETICIÓN

Un mensaje de petición de un cliente hacia un servidor incluye, en la primera línea de ese mensaje, el método a ser aplicado al recurso, el identificador del recurso y la versión del protocolo en uso.

```
"Request = Request-Line *(( general-header | request-header | \\
 entity-header ) CRLF) CRLF[message-body ]"
```

LÍNEA DE PETICIÓN: La línea de petición inicia con un token de método, seguido por la URI que se está pidiendo y la versión del protocolo y terminando con CRLF (retorno de carro y cambio de línea). Los elementos son separados por caracteres SP.

```
"Request-Line = Método SP URI-pedida SP HTTP-Version CRLF"
```

MÉTODO El token de método indica el método a ser aplicado al recurso identificado por la URI-pedida. El método es case-sensitive.

URI-PEDIDA La URI-Pedida es un identificador uniforme de recurso (URI) e identifica el recurso sobre el cual se va a aplicar el mensaje de petición.

MÉTODO	DESCRIPCIÓN
OPTIONS	Representa una solicitud de información acerca de las opciones de comunicación disponibles en la cadena de Petición/Respuesta identificada por el URI-pedida
GET	Significa que se recupere cualquier información (en la forma de entidad) que es identificada por la URI-pedida
HEAD	Es idéntica a GET con la excepción de que el servidor no retornara en cuerpo del mensaje en la respuesta.
POST	Se utiliza para solicitar que el servidor original acepte la entidad envuelta en la petición como un nuevo subordinado del recurso identificado por el URIpedida en la línea de petición.
PUT	Solicita que la entidad envuelta sea almacenada bajo el URI-perdida suplida.
DELETE	Solicita que el servidor original borre el recurso identificado por el URI-pedida
TRACE	Es utilizada para invocar un remoto, de nivel de aplicación loop- back del mensaje de petición.
CONNECT	Se utiliza con proxies que pueden dinámicamente cambiarse para ser un túnel.
extension-method	Definidos por alguna aplicación.

Tabla 1: Métodos de HTTP. Tanenbaum and Wetherall

```
“Request-URI = "*" | absoluteURI | abs_path | authority”
```

Las opciones del URI-pedida dependen de la naturaleza de la petición. El asterisco significa que la petición no aplica a ningún recurso en particular, pero al servidor en sí mismo y solo es permitido cuando el método utilizado no necesariamente aplica a un recurso.

EL RECURSO IDENTIFICADO POR UNA PETICIÓN Para extraer el identificador de recurso por una petición en Internet es determinado mediante el proceso de examinar ambos URI-pedida y el campo del encabezado HOST. Un servidor de origen que no diferencie recursos de acuerdo en el host solicitado debe utilizar las siguientes reglas para identificar el recurso pedido:

1. Si URI-pedida es un URI absoluta, el HOST es parte de URI-pedida. Cualquier campo de encabezado HOST debe ser ignorado.
2. Si URI-pedida no es una URI absoluta y la petición incluye un campo de encabezado HOST, el HOST es determinado por el campo HOST de la petición.
3. Si el HOST identificado por las reglas 1 y 2 no es un HOST valido en el servidor entonces el servidor deberá devolver la respuesta 400 Bad Request.

CAMPOS DE ENCABEZADO DE UNA PETICIÓN El encabezado de la petición permite a el cliente pasar información adicional acerca de la petición y acerca del cliente al servidor. Estos campos actúan como modificadores de la petición, con semántica equivalente a los parámetros en la invocación de métodos en los lenguajes de programación. Entre las más importantes se encuentran:

2.4 RESPUESTA

Después de recibir el interpretar un mensaje de petición, el servidor responde con un mensaje de respuesta HTTP.

```
"Response = Status-Line *(( general-header | response-header | \\
entity-header ) CRLF) CRLF [message-body ]"
```

LÍNEA DE ESTADO: La primera línea de un mensaje de respuesta es Status-Line, la cual se encuentra conformada por una versión de protocolo seguida por un código numérico de estatus y su frase de texto asociada, con cada elemento separado por caracteres SP.

```
"Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF"
```

MÉTODO	DESCRIPCIÓN
ACCEPT	Se utiliza para especificar ciertos tipos de media que son aceptables por la respuesta.
ACCEPT-CHARSET	Se utiliza para indicar cuales CHARSETS son aceptables en la respuesta.
ACCEPT-ENCODING	Similar a ACCEPT pero limita los content-encodings aceptables en la respuesta.
ACCEPT-LANGUAGE	Similar a ACCEPT pero con la diferencia que restringe el conjunto de lenguajes que son preferidos como respuesta a la petición.
ACCEPT-RANGES	Permite al servidor indicar su aceptación de una petición de rango por un recurso.
AGE	Transmite la estimación de tiempo desde que la respuesta fue generada en el servidor original.
ALLOW	Lista los métodos disponibles para un recurso identificado por un URI-pedida.
AUTHORIZATION	Se utiliza cuando el agente de usuario quiere identificarse con él con el servidor. Se envían las credenciales.
CACHE-CONTROL	Se utiliza para especificar directivas que deben ser obedecidas por todos los mecanismos de cache a lo largo de la cadena de Petición/Respuesta.

Tabla 2: Métodos de Petición. Tanenbaum and Wetherall

CÓDIGO DE ESTADO Y FRASE: El código de estado es un código de resultado de 3 dígitos que indica el intento de entender y satisfacer la petición. Esos códigos ya se encuentran definidos. La frase intenta dar una corta descripción textual del Código de estado.

ENCABEZADO DE RESPUESTA: Los campos del encabezado de respuesta permiten al servidor pasar información adicional acerca de la respuesta que no pueden ser colocadas en la Status-Line. Estos campos del encabezado dan información acerca del servidor y acerca de futuros accesos al recurso identificado por la URI-Pedida.

3

PROTOLOCO P2P

Una red peer-to-peer (P2P) o red de pares, es una red de computadoras en la que todos o algunos aspectos de ésta funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre sí. Es decir, actúan simultáneamente como clientes y servidores respecto a los demás nodos de la red.[11]

Las redes peer-to-peer aprovechan, administran y optimizan el uso del ancho de banda de los demás usuarios de la red por medio de la conectividad entre los mismos, obteniendo más rendimiento en las conexiones y transferencias que con algunos métodos centralizados convencionales, donde una cantidad relativamente pequeña de servidores provee el total del ancho de banda y recursos compartidos para un servicio o aplicación. Alfredo F. nos dice que de forma simple puede verse como la comunicación entre pares o iguales utilizando un sistema de intercambio [2].

Los sistemas compañero a compañero (o P2P por sus siglas en inglés) permiten que computadoras de usuario final se conecten directamente para formar comunidades, cuya finalidad sea el compartir recursos y servicios computacionales. En este modelo, se toma ventaja de recursos existentes en los extremos de la red, tales como tiempo de CPU y espacio de almacenamiento. Las primeras aplicaciones emergentes se orientaban a compartir archivos y a la mensajería [2].

3.1 CLASIFICACIÓN

3.1.1 *Directorio Centralizado*

Esta primera visión se puso en práctica para el año 1997 con la red Napster. Bajo este modelo los clientes o peers realizan el descubrimiento y búsqueda de los otros miembros mediante un servidor central, luego el mismo servidor se encarga de negocia la conexión entre ambos clientes.

Este tipo de red P2P se basa en una arquitectura monolítica en la que todas las transacciones se hacen a través de un único servidor que sirve de punto de enlace entre dos nodos y que, a la vez, almacena y distribuye los nodos donde se almacenan los contenidos. Poseen una administración muy dinámica y una disposición más permanente de contenido. Sin embargo, está muy limitada en la privacidad de los usuarios y en la falta de escalabilidad de un sólo servidor, además de ofrecer problemas en puntos únicos de fallo, situaciones legales y

enormes costos en el mantenimiento así como el consumo de ancho de banda. [11]

El principal problema presentado en este modelo es que el servidor P2P Central es un posible punto de quiebre que pueda atentar contra la estabilidad de la red.

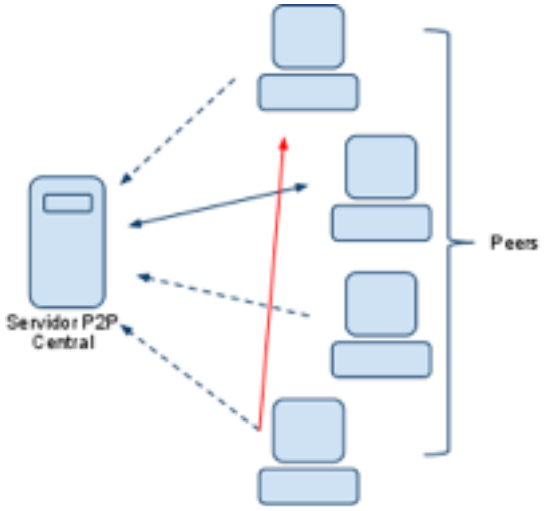

Figura 2: Directorio Centralizado

3.1.2 *Solución Distribuida*

Las redes P2P de este tipo son las más comunes, siendo las más versátiles al no requerir de un gestiona miento central de ningún tipo, lo que permite una reducción de la necesidad de usar un servidor central, por lo que se opta por los mismos usuarios como nodos de esas conexiones y también como almacenistas de esa información. En otras palabras, todas las comunicaciones son directamente de usuario a usuario con ayuda de un nodo (que es otro usuario) quien permite enlazar esas comunicaciones [11].

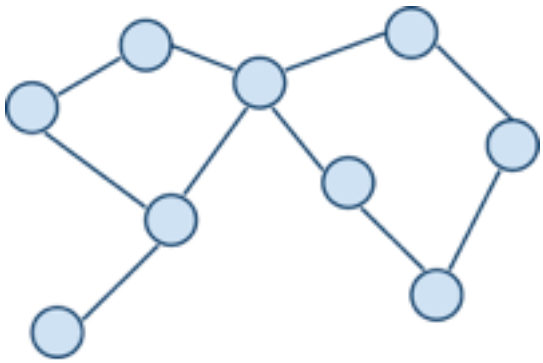

Figura 3: Solución Distribuida

3.1.3 *Directorio Descentralizado*

En este tipo de red, se puede observar la interacción entre un servidor central que sirve como hub y administra los recursos de banda ancha, enrutamientos y comunicación entre nodos pero sin saber la identidad de cada nodo y sin almacenar información alguna, por lo que el servidor no comparte archivos de ningún tipo a ningún nodo. Tiene la peculiaridad de funcionar (en algunos casos como en Torrent) de ambas maneras, es decir, puede incorporar más de un servidor que gestione los recursos compartidos, pero también en caso de que el o los servidores que gestionan todo caigan, el grupo de nodos sigue en contacto a través de una conexión directa entre ellos mismos con lo que es posible seguir compartiendo y descargando más información en ausencia de los servidores [11].

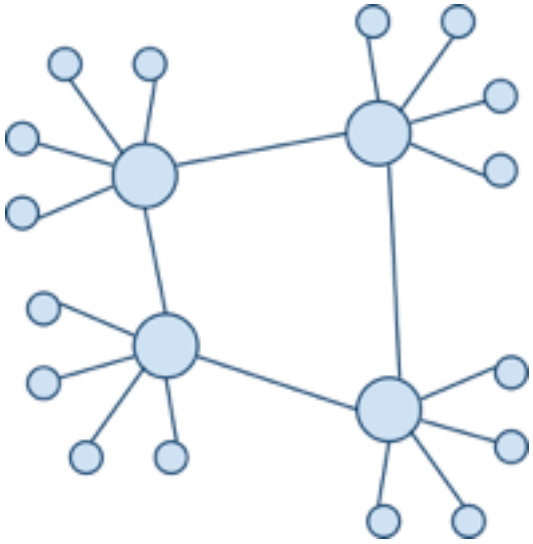

Figura 4: Directorio Descentralizado

3.2 MECANISMOS DE BÚSQUEDA EN REDES P2P

CENTRALIZADOS: Un repositorio central almacena un índice de todos los recursos en la red, junto con la localización de dichos recursos (qué nodos los ofrecen). Todos los miembros de la red registran sus recursos y dirigen sus búsquedas a este repositorio. Un ejemplo es la red Napster. En realidad, a los sistemas de este tipo no se les considera como P2P puros, ya que ese repositorio es un servicio centralizado, con los inconvenientes típicos: representa un punto único de fallo, y compromete la escalabilidad del sistema.

DESCENTRALIZADOS: No existe ningún repositorio centralizado. Para encontrar un recurso, los nodos lanzan mensajes de búsqueda que son encaminados por la red. Las redes descentralizadas son

clasificadas a su vez en: Redes no estructuradas y Redes estructuradas.

REDES ESTRUCTURADAS: Las redes estructuradas tienen importantes ventajas. Proporcionan un mecanismo determinista de localización, de tal forma que se asegura que un recurso será encontrado siempre que esté en la red. Además, los mensajes de búsqueda son encaminados de forma eficiente, de tal forma que el recurso es encontrado en $O(\log n)$ saltos, donde n es el tamaño de la red.

4

MODELOS DE WEB CACHING

El web caching no es un concepto nuevo. Este tema ya se ha estudiado y también ha sido objeto de numerosos papers, artículos y tesis. Por otro lado el concepto de cache web distribuida es un poco más nuevo pero también suma bastantes proyectos entorno a él.

Para adentrarnos un poco es necesario definir el concepto más sencillo el de caché web (también llamado proxy):

"Se llama caché web a la caché que almacena documentos web (es decir, páginas, imágenes, etcétera) para reducir el ancho de banda consumido, la carga de los servidores y el retardo en la descarga. Un caché web almacena copias de los documentos que pasan por él, de forma que subsiguientes peticiones pueden ser respondidas por el propio caché, si se cumplen ciertas condiciones." [9]

Ya con una definición clara acerca de caché web podemos encaminarnos a definir un concepto un poco más complicado: caché web cooperativo. Según [1] mediante esta técnica, el almacenamiento dedicado a la cache de una serie de servidores-proxy adyacentes se gestiona conjuntamente como si se tratase de una única cache distribuida. Las políticas de gestión de esta cache se realiza teniendo en cuenta el estado y los contenidos de todos los servidores-proxy que la componen. Por ejemplo, las políticas de reemplazo en la cache cooperativa tendrán en cuenta las estadísticas de acceso en todos los servidores-proxy cooperantes.

Esta técnica permite incrementar considerablemente el tamaño de la cache, pero a costa de reducir la probabilidad de acierto local. Bajo el esquema de cache cooperativo, un único servidor-proxy no dispone de los contenidos más populares, provocando un aumento del porcentaje de peticiones que se tienen que servir remotamente desde los otros servidores que componen la cache ó desde el servidor principal.

Hemos logrado identificar proyectos que giran alrededor de nuestra tesis. Entre ellos están: CoralCDN [3], Transparent Distributed Web Caching [6], Dalesa [8], Squirrel [1], entre otros. Más adelante ampliaremos con más información acerca de ellos.

4.1 WEB CACHING MEDIANTE PROXY

4.1.1 *Proxy Simple*

Para iniciar vamos a mostrar el modelo tradicional de cliente servidor. Bajo este modelo el cliente tiene una conexión directa con el servidor y no existe cache en medio, por lo que dicha conexión se obtienen los

servicios bajo demanda, haciendo uso únicamente del caché local del explorador web.

Figura 5: Conexión Directa

El siguiente modelo muestra el típico modelo de proxy, donde existe un servidor central por el cual los clientes pasan y le solicitan el recurso deseado. Este servidor descargar el recurso solicitado y lo almacena en su caché y además lo reenvía al solicitante. Si otro cliente solicitara el mismo recurso, el servidor le respondería con el recurso almacenado localmente (al no ser que la caché haya caducado).

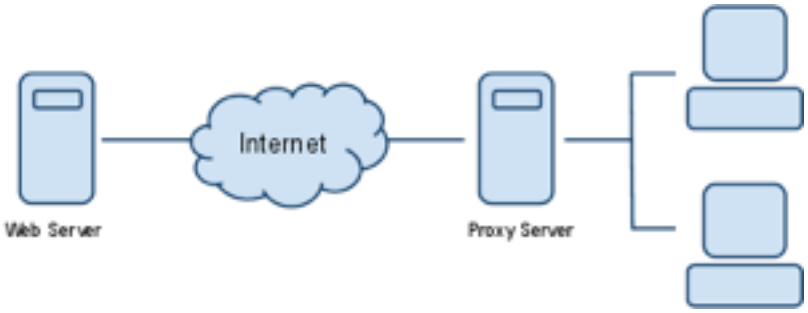

Figura 6: Proxy Simple

4.1.2 *Proxy Jerárquico*

La figura siguiente muestra el modelo de proxy jerárquico o también conocido como web caché cooperativo. En este modelo se tiene servidores proxy ubicados de una manera jerárquica en una Red de área local. Cuando un cliente solicita un recurso, éste lo hace a su proxy jerárquicamente más cercano. Si dicho servidor posee los recursos éste se lo reenvía al cliente. En caso que no posee el recurso, el proxy le reenvía la solicitud de su cliente al servidor proxy de nivel superior. Si el servidor de nivel superior posee el recurso lo reenvía al servidor proxy que solicitó el recurso, éste último se deja una copia en su caché local y reenvía el recurso al cliente original.

4.1.3 *Proxy Descentralizado*

Bajo este modelo se encuentran los proyectos de Squirrel y Dalesa. Estos proyectos intentan utilizar los conceptos de P2P para implementar y compartir una cache distribuida a través de una LAN. Como

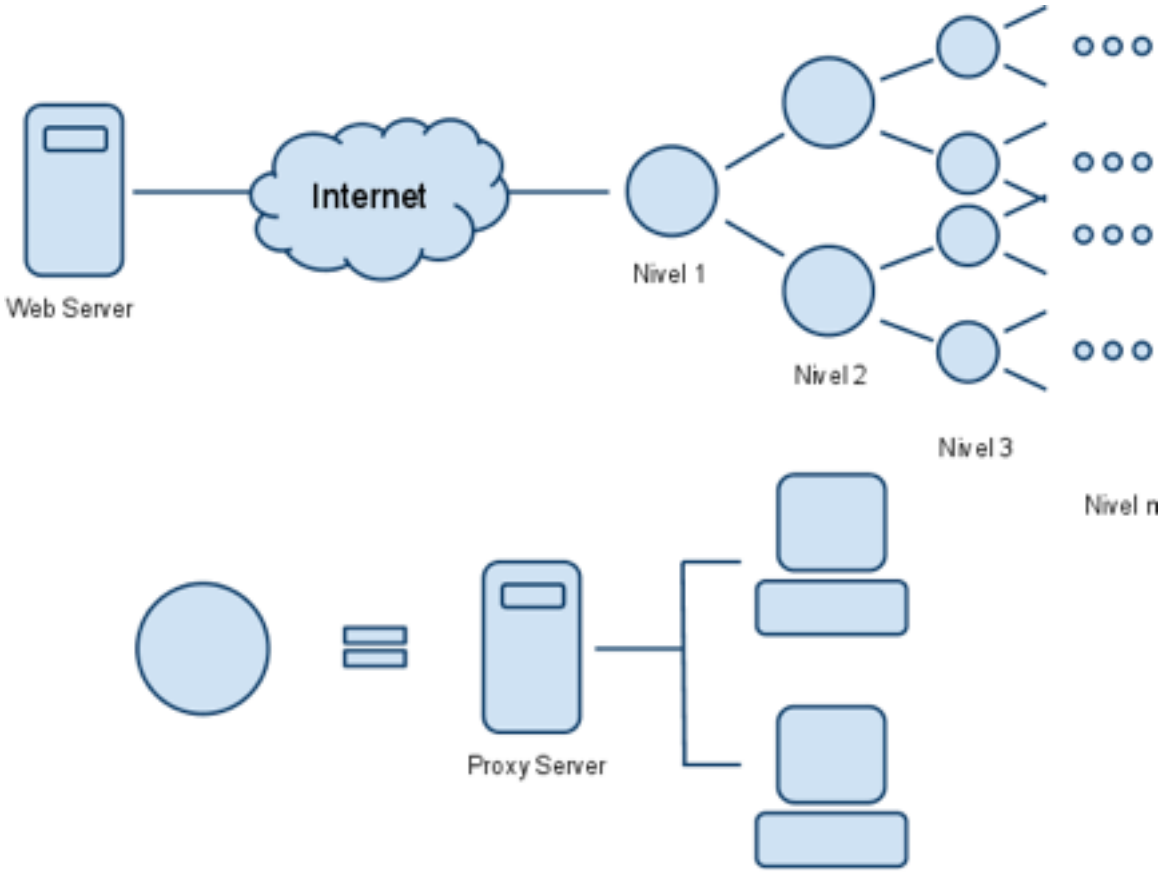

Figura 7: Proxy Jerárquico

se muestra en la figura. Cada computadora forma parte del proxy descentralizado, y utilizando un software creados por los respectivos participantes del proyecto, pueden compartir la cache. Este modelo es el que más se acerca al modelo que proponemos como tesis aunque todavía dista en ciertas características.

4.2 REDES CDN

Una red de distribución de contenidos es una red de equipos que opera de forma transparente para entregar contenido de sus clientes cumpliendo principalmente alguno de los siguientes objetivos:

1. Escalabilidad
2. Coste
3. Eficiencia

Haciendo una definición más amplia una Red CDN es un sistema de computadoras el cual contiene copias de datos, ubicados en varios puntos de la red para así maximizar el ancho de banda para el acceso de los datos desde los clientes a través de la red. Un cliente accede

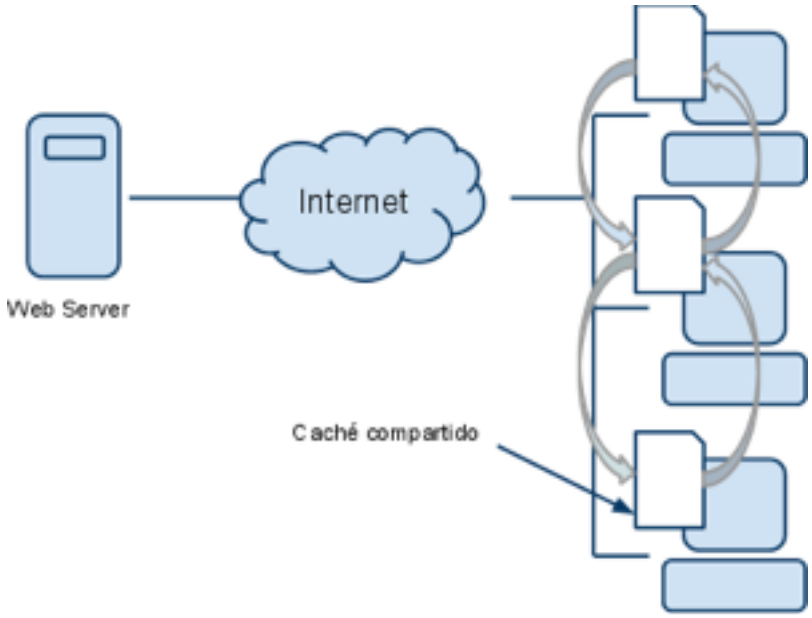

Figura 8: Proxy Descentralizado

una copia de los datos, la más cercana, contrario al concepto de que todos los clientes accedan al mismo servidor central provocando un cuello de botella en el servidor [10].

El contenido puede incluir objetos web, objetos descargables (archivos de medios, software, documentos, etc.), aplicaciones, media streams en tiempo real, y otros componentes.

En la figura siguiente se puede observar el funcionamiento de una Red CDN distribuida a través de Internet.

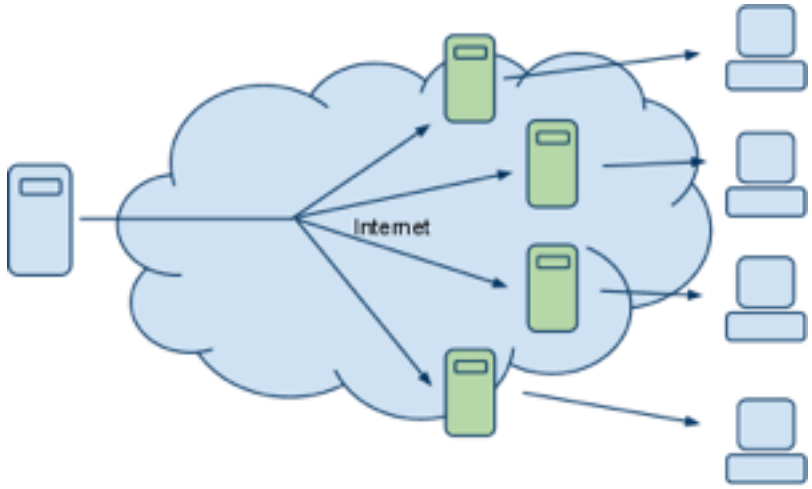

Figura 9: Red CDN

5 MODELOS DE COSISTENCIA

Un requerimiento importante en los sistemas distribuidos es la replicación de datos, lo datos por lo general se replican para mejorar fiabilidad y incrementar el rendimiento, uno de los problemas más importantes es, mantener todas las replicas consistentes, esto quiere decir que cuando una copia se actualiza, necesitamos asegurar que las otras copias se actualizan también.

5.1 RAZONES PARA REPLICAR

1. Los datos se replican para incrementar la fiabilidad del sistema, esto quiere decir que si el sistema ha sido replicado este puede seguir trabajando después de alguno de sus nodos falle, mediante el cambio a uno que si funcione. Al mantener muchas copias es posible proporcionar una mejor protección contra datos corruptos.

2. Los datos se replican para incrementar el rendimiento, es importante cuando un sistema distribuido necesita ser escalado en números y áreas geográficas. En el caso de números, cuando un sistema necesita procesar mucha información, este se escala agregando otro nodo al sistema y replicando los datos, para dividir el trabajo. En el caso de área geográfica se coloca una copia cerca del proceso que lo necesita, para reducir el tiempo de acceso a los datos.

5.2 MODELOS DE CONSISTENCIA SIN SINCRONIZACIÓN

LINEARIZABILITY O CONSISTENCIA ESTRICTA: Linearizability es la condición correcta para objetos concurrentes. Las operaciones write que se ejecutan en cualquiera de los sitios de un sistema, se ejecutan simultáneamente en todas las copias de los datos, ose es la condición ideal siempre las copias son coherentes. En pocas palabras es la consistencia más estricta, esta establece que cualquier operación de lectura en el sistema, debe devolver el valor con la última actualización antes de realizar la lectura. Lo cual quiere decir que el tiempo máximo para propagar una actualización al resto de sitios en el sistema es cero.

CONSISTENCIA SECUENCIAL: Es un modelo de consistencia un poco más débil, que la consistencia estricta. Básicamente Lamoprt lo define mediante:

"El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos fueran ejecutadas en algún orden secuencial, y las operaciones de cada proceso individual aparecen en esta secuencia en el orden especificado por su programa"[5]

La consistencia secuencial no garantiza que una operación de lectura devuelva el valor escrito por otro proceso. Lo único que asegura es que todos los sitios vean todas las operaciones en el mismo orden.

CONSISTENCIA CAUSAL: En un sistema se da consistencia causal si las escrituras parcialmente relacionadas de forma causal (sobre el mismo objeto concurrente) son vistas por todos los procesos en el mismo orden, las escrituras concurrentes pueden ser vistas en un orden diferente en maquinas diferentes.

Este modelo es aun más débil que la consistencia secuencial, la cual requiere que todos sitios vean las escrituras en el mismo orden.

Cuando un sitio realiza una lectura seguida de una escritura, aun en diferentes objetos concurrentes, se dice que la primera operación esta causalmente ordenada antes que la segunda, debido a que el valor almacenado por la escritura puede haber sido dependiente del resultado de la lectura. De manera similar una operación de lectura puede estar causalmente ordenada después de una escritura previa en el mismo objeto concurrente que almacena lo que fue leído. De la misma manera dos operaciones de escritura en el mismo sitio se encuentran causalmente ordenadas en el mismo orden que fueron realizadas.

Si existen operaciones que no están causalmente relacionadas se dice que son concurrentes.

CONSISTENCIA FIFO: Este modelo es aun más débil que la consistencia causal, esta consiste en eliminar el requisito de que las operaciones relacionadas se vean en orden por parte de todos los sitios en un sistema. Para esto las escrituras por un sitio son recibidas por otros procesos en el orden en que son realizadas, pero las escrituras de sitios diferentes pueden verse en un orden distinto en sitios diferentes.

Este tipo de consistencia es fácil de implementar, ya que no existe garantía del orden en que los diferentes sitios vean las escrituras, más que dos o más escrituras por parte de un proceso llegan en orden, como si estuvieran entubadas. Remontándonos a la definición de consistencia causal, podemos decir que en consistencia FIFO todas las escrituras y lecturas locales (realizadas en el mismo sitio) están causalmente relacionadas, mientras que todas las que suceden en sitios diferentes son concurrentes.

5.3 MODELOS DE CONSISTENCIA CON SINCRONIZACIÓN

CONSISTENCIA DÉBIL: La consistencia débil utiliza variables de sincronización para propagar las escrituras a todo el sistema, mediante esto una operación de escritura en un sitio, tomara un candado el cual evitara que otros puedan leer o escribir hasta que la operación termine y se propague al resto de los sitios que conforman el sistema. A este mecanismo se le llama exclusión mutua.

El modelo tiene las siguientes condiciones:

- Los accesos a las variables de sincronización son secuencialmente consistentes.
- No se permite ingresar a una variable de sincronización hasta que todas las operaciones de escritura hayan terminado en el resto de los sitios.
- No se permite realizar un acceso a los datos (operaciones de lectura y escritura) hasta realizar todos los accesos a las variables de sincronización.

CONSISTENCIA RELEASE: Es muy parecida a la consistencia débil, lo que hace es dar solución a un problema de rendimientos que tiene esta, cuando se acceda a una variable de sincronización, el objeto concurrente no sabe si se debe a que el sitio a terminado de escribir en el objeto concurrente o está a punto de iniciar su lectura. Para una implementación más eficiente de la consistencia débil se crean dos variables de sincronización que diferencien estos casos. Estas operaciones se llaman release y acquire, antes de realizar una operación de escritura a un objeto concurrente un sitio debe adquirir el objeto mediante la operación acquire y más tarde liberarlo mediante release.

Para contar con consistencia release se debe cumplir:

- Antes de realizar una operación a un objeto concurrente, deben terminar con éxito todas las adquisiciones anteriores del sitio en cuestión.
- Antes de realizar un release, se deben terminar todas las escrituras y lectura del sitio.
- Los accesos a release y acquire deben de ser consistentes con el procesador.

CONSISTENCIA DE ENTRADA: Utiliza las mismas operaciones de sincronización mencionadas anteriormente acquire y release. La diferencia radica en que requiere que cada objeto concurrente se asocie con alguna variable de sincronización, como en un modelo de exclusión barrier o tree.

La consistencia de entrada debe cumplir:

- No se permite a unos sitios realizar un acquire a un objeto concurrente con respecto a un sitio hasta que se realicen todas las actualizaciones de los datos compartidos protegidos con respecto a este sitio.
- Antes de permitir el acceso de modo exclusivo a una variable de sincronización por un sitio, ningún otro sitio debe poseer la variable de sincronización, ni siquiera en modo no exclusivo.
- Después de realizar un acceso en modo exclusivo a una variable de sincronización, no se puede realizar el siguiente acceso en modo no exclusivo de otro sitio a esta variable de sincronización hasta haber sido realizado con respecto al propietario de esa variable.

6 COMUNIDADES VIRTUALES

6.1 DEFINICIÓN DE UNA COMUNIDAD VIRTUAL

Bueno ya hemos hablado un poco de historia pero en realidad ¿qué son las comunidades virtuales? A pesar de que también se les designa como "congregaciones electrónicas" "comunidades en línea" "comunidades electrónicas"; el término más usado es el de comunidad virtual y está compuesto por dos nociones: la de "comunidad" y "virtual". De allí que, en aras de aproximarnos a su definición es necesario analizar por separado el concepto de estos dos términos. Definiremos primero el término Comunidad. Etimológicamente, Foster (1997) afirma que el término comunidad tiene un linaje directo con la palabra comunicación y a su vez, Merril y Loewenstein (1979) plantean que este último "proviene del latín communis (común) o communicare (el establecimiento de una comunidad o comunalidad)" (Foster, 1997 pag. 24). En este respecto, el autor advierte que aún cuando la comunicación es la base de la comunidad, ambos términos no deben confundirse, ya que un individuo puede comunicarse con otro sin que formen parte de una misma comunidad. Por otro lado, tenemos el término virtual. Desde el punto de vista histórico Wilbur (1997) afirma que la palabra virtual data de la edad media y se originó a partir de la palabra "virtud". Durante esta era, se usaba el término virtual para calificar el poder divino, porque tenía la "virtud" de ser real aun cuando no se pudiera observar en el mundo material. Esta es la primera vertiente semántica del término: lo virtual es algo " que tiene virtud para producir un efecto" (Sopena, Diccionario Enciclopédico Ilustrado 1965, pag. 3697). Las nociones de "comunidad" y "virtual", plantean la tentativa de definir una comunidad virtual como "una congregación de cibernautas que integran una comunidad que aparenta ser real al simular los efectos de las congregaciones sociales humanas reales o tradicionales, pero sin llenar todas las características de estas". Por otro lado, también tenemos la siguiente definición: Se denomina comunidad virtual a aquella comunidad cuyos vínculos, interacciones y relaciones tienen lugar no en un espacio físico sino en un espacio virtual como Internet.

6.2 CARACTERÍSTICAS DE UNA COMUNIDAD VIRTUAL

Las Comunidades Virtuales, sólo existen y funcionan en la medida en que sean fruto de la actividad de los ciudadanos, entendidos estos como individuos, colectivos formales o informales, empresas, organi-

zaciones, etc. Así se han creado espacios artificiales (virtuales) nuevos, dotados de una serie de características no siempre comprensibles desde los parámetros del "mundo real".

LA INFORMACIÓN ES DE LOS USUARIOS: Es decir, la Red, en principio, está "vacía" y son los usuarios quienes deciden qué información van a almacenar, mostrar e intercambiar. Por tanto, cada usuario decide por dónde empieza a ver la Red, para qué y con quiénes.

ACCESO A LA RED: Un punto importante que debe de considerar una CV es precisamente, establecer principios de acceso a la red.

- *Universal*: basta acceder a un ordenador de la red, para acceder a toda la red o "ver" toda la Red (otra cosa es que, una vez dentro de la Red, haya lugares donde se pida el registro para acceder a la información que contienen)
- *Simultáneo*: todos estamos en la Red al mismo tiempo, pues existimos en cuanto información (ceros y unos). En realidad, la red es desde sus orígenes el primer contestador automático que se puso en funcionamiento. Nadie sabe si estamos conectados o no, pero nos relacionamos entre todos como si lo estuviéramos a través de nuestra presencia numérica, de la información que "movemos" y de las interacciones que promovemos; Independiente del tiempo (24h./365d.) y de la distancia. Es el primer espacio abierto constantemente a la actividad del ser humano independientemente de donde se encuentre. Sólo necesita acceder a un ordenador de la Red para que todo lo expuesto anteriormente funcione.

ORGANIZACIÓN DE LA RED: Finalmente, los otros dos rasgos que cierran este comprimido código genético es que la red crece de manera descentralizada y desjerarquizada. Basta seguir añadiendo ordenadores (servidores) para que se esparza física y virtualmente y no hay ordenadores que desempeñen tareas de "comando y control" sobre los otros ordenadores de la Red.

6.3 TIPOS DE COMUNIDADES VITUALES

A grandes rasgos podemos clasificar las CV en tres grandes categorías: de ocio, profesionales y de aprendizaje. Aunque algunos autores como Polo (1998) nos indica que pueden darse:

- Centrada en las personas: Las personas se reúnen fundamentalmente para disfrutar del placer de la mutua compañía.
- Centrada en el tema: Las personas comparten un tema en común.

- Centrada en un acontecimiento: Personas centradas en acontecimientos externos.

Para Hagel y Armstrong (1997) hay dos tipos claramente diferenciados, las orientadas hacia el usuario y las orientadas hacia la organización.

ORIENTADAS A LOS USUARIOS: En las orientadas a los usuarios, son ellos los que definen el tema de la Comunidad y se pueden dividir en:

- *Geográficas*: agrupan personas que viven o que están interesadas en intercambiar Información sobre una misma área geográfica.
- *Temáticas*: orientadas a la discusión de un tema de interés para los usuarios.
- *Demográficas*: reúnen usuarios de características demográficas similares.
- *De ocio y entretenimiento*: dirigidas a aquellos cibernautas que ocupan su tiempo libre en juegos en red. Se crean por tipos de juegos como estratégicos, de simulación, etc.
- *Profesionales*: para aquellos expertos en una materia que desarrollan su actividad concreta en un área profesional definida, generalmente asociada a una formación superior. Especialmente en el caso de las profesiones liberales, cuando se trabaja de manera independiente.
- *Gubernamentales*: Los organismos gubernamentales han creado Comunidades Virtuales a las que puede acudir el ciudadano para informarse y/o discutir.
- *Eclécticas*: son aquellas Comunidades Virtuales mixtas, que intentan un poco de todo: zona de ocio, una vía de transmisión y comportamiento cultural, etc.

ORIENTADAS A LA ORGANIZACIÓN: Por el contrario en las orientadas hacia la organización, el tema es definido según los objetivos y áreas de trabajo de la organización donde reside la comunidad, y podemos dividirlas en:

- *Verticales*: que aglutinan a usuarios de empresas de diferentes ramas de actividad económica o a organizaciones institucionales.
- *Funcionales*: referidas a un área específica del funcionamiento de la organización, por ejemplo: mercadeo, producción, relaciones públicas.
- *Geográficas*: que se concentran en una zona geográfica cubierta por la organización.

En una línea similar a la anterior, Salinas (2003) nos llama la atención respecto a dentro de las CV se pueden distinguir una serie de grupos en función de:

MODO DE ASIGNACIÓN: Dado el modo de asignación se pueden encontrar:

- Comunidades de asignación libre por parte de los miembros.
- Comunidades de asignación voluntaria.
- Comunidades de asignación obligatoria.

FUNCIÓN PRIMARIA DE LA COMUNIDAD: Las comunidades también se pueden clasificar por su función:

- *Distribución*: Cuando la principal función de la comunidad radica en la distribución de información o mensajes, entre los miembros.
- *Compartir*: Se trata de comunidades donde prima el intercambio de experiencias y recursos.
- *Creación*: Cuando se generan procesos de trabajo colaborativo con el objeto de lograr materiales, documentos, proyectos compartidos.

GESTIÓN DE LA COMUNIDAD: Por otro lado un punto importante que nos puede ayudar a clasificar una CV es el tipo de gestión que se utilice:

- *Abiertas*: Cuando el acceso (independientemente de la asignación) es abierto y los recursos de la comunidad están a disposición tanto de los miembros como de personas ajenas a la comunidad.
- *Cerradas*: Cuando existe algún procedimiento que impide a las personas ajenas a la comunidad el acceso, de tal forma que los recursos, materiales, producciones, histórico, etc., sólo son accesibles para los miembros de la comunidad.

EL OBJETO DE LA COMUNIDAD: En la línea de la clasificación descrita anteriormente, las comunidades virtuales de aprendizaje podemos clasificarlas en función del objeto que persiguen, en:

- Comunidades de aprendizaje propiamente dichas, cuando han sido creadas para que el grupo humano que se incorpora a la comunidad desarrolle procesos de aprendizaje en programas diseñados al efecto.
- Comunidades de práctica, ya definidos anteriormente

- Comunidades de investigación, cuando se trata de comunidades que desarrollando actividades de aprendizaje, el objeto principal es poner en marcha proyectos de investigación conjunta de acuerdo con la filosofía del trabajo cooperativo a través de redes.
- Comunidades de innovación. Similares a las anteriores que buscan compartir, intercambiar y generar procesos de innovación en distintos campos.

Abordando más expresamente el último grupo planteado por el profesor Salinas, nos encontramos con la propuesta de Jonassen, Pech, y Wilson (1998) que establecen cuatro tipos de comunidades virtuales:

DE DISCURSO: El ser humano es una criatura social y puede hablar cara a cara sobre intereses comunes, pero también puede compartir estos intereses con otros semejantes más lejanos mediante los medios de comunicación. Las redes de ordenadores proporcionan numerosas y potentes herramientas para el desarrollo de este tipo de comunidades.

DE PRÁCTICA: Cuando en la vida real alguien necesita aprender algo, normalmente no abandona su situación normal y dedica su esfuerzo en clases convencionales, sino que puede formar grupos de trabajo (comunidades de práctica), asigna roles, enseña y apoya a otros y desarrolla identidades que son definidas por los roles que desempeña en el apoyo al grupo.

DE CONSTRUCCIÓN DE CONOCIMIENTO: El objetivo de este tipo de comunidades es apoyar a los estudiantes a perseguir estratégica y activamente el aprendizaje como una meta.

DE APRENDIZAJE: Si una comunidad es una organización social de personas que comparten conocimiento, valores y metas, las clases como las conocemos no son comunidades ya que los estudiantes están desconectados o están compitiendo unos con otros. Las clases son comunidades sociales, pero su propósito no es aprender juntos o unos de otros, antes parece que estos grupos buscan reforzar socialmente sus propias identidades por exclusión de los otros.

Parte III

DISEÑO DEL CWC

En este apartado de la investigación, se describirá el diseño propuesto para el CWC, reuniendo un conjunto de requerimientos basados en los objetivos de la investigación; además se propondrá un modelo que solucione y sea de ayuda para la comprobación de la hipótesis de ésta investigación.

7 CHAPTER TITLE

7.1 SECTION TITLE

Content

7.1.1 *Subsection Title*

Content

7.1.2 *Subsection Title*

Content

7.2 SECTION TITLE

Content

8 CHAPTER TITLE

8.1 SECTION TITLE

Content

8.1.1 *Subsection Title*

Content

8.1.2 *Subsection Title*

Content

8.2 SECTION TITLE

Content

Parte IV

ANÁLISIS Y RESULTADOS

En esta sección se mostrará el análisis realizado producto de los datos recolectados a lo largo de la investigación.

9 CHAPTER TITLE

9.1 SECTION TITLE

Content

9.1.1 *Subsection Title*

Content

9.1.2 *Subsection Title*

Content

9.2 SECTION TITLE

Content

10 CHAPTER TITLE

10.1 SECTION TITLE

Content

10.1.1 *Subsection Title*

Content

10.1.2 *Subsection Title*

Content

10.2 SECTION TITLE

Content

Parte V

CONCLUSIONES

11 CHAPTER TITLE

11.1 SECTION TITLE

Content

11.1.1 *Subsection Title*

Content

11.1.2 *Subsection Title*

Content

11.2 SECTION TITLE

Content

Parte VI

APPENDIX

A

APPENDIX TEST

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

Etiam ac leo a risus tristique nonummy. Donec dignissim tincidunt nulla. Vestibulum rhoncus molestie odio. Sed lobortis, justo et pretium lobortis, mauris turpis condimentum augue, nec ultricies nibh arcu pretium enim. Nunc purus neque, placerat id, imperdiet sed, pellentesque nec, nisl. Vestibulum imperdiet neque non sem accumsan laoreet. In hac habitasse platea dictumst. Etiam condimentum facilisis libero. Suspendisse in elit quis nisl aliquam dapibus. Pellentesque auctor sapien. Sed egestas sapien nec lectus. Pellentesque vel dui vel neque bibendum viverra. Aliquam porttitor nisl nec pede. Proin mattis libero vel turpis. Donec rutrum mauris et libero. Proin euismod porta felis. Nam lobortis, metus quis elementum commodo, nunc lectus elementum mauris, eget vulputate ligula tellus eu neque. Vivamus eu dolor.

A.1 APPENDIX SECTION TEST

Nulla in ipsum. Praesent eros nulla, congue vitae, euismod ut, commodo a, wisi. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nonummy magna non leo. Sed felis erat, ullamcorper in, dictum non, ultricies ut, lectus. Proin vel arcu a odio lobortis euismod. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Proin ut est. Aliquam odio. Pellentesque massa turpis, cursus eu, euismod nec, tempor congue, nulla. Duis viverra gravida mauris. Cras tincidunt. Curabitur eros ligula, varius ut, pulvinar in, cursus faucibus, augue.

More dummy text

Nulla mattis luctus nulla. Duis commodo velit at leo. Aliquam vulputate magna et leo. Nam vestibulum ullamcorper leo. Vestibulum condimentum rutrum mauris. Donec id mauris. Morbi molestie justo et pede. Vivamus eget turpis sed nisl cursus tempor. Curabitur mollis sapien condimentum nunc. In wisi nisl, malesuada at, dignissim sit amet, lobortis in, odio. Aenean consequat arcu a ante. Pellentesque porta elit sit amet orci. Etiam at turpis nec elit ultricies imperdiet. Nulla facilisi. In hac habitasse platea dictumst. Suspendisse viverra

LABITUR BONORUM PRI NO	QUE VISTA	HUMAN
fastidii ea ius	germano	demonstratea
suscipit instructior	titulo	personas
quaestio philosophia	facto	demonstrated

Tabla 3: Autem usu id.

Listado 1: A floating example

```
for i:=maxint to 0 do
begin
{ do nothing }
end;
```

aliquam risus. Nullam pede justo, molestie nonummy, scelerisque eu, facilisis vel, arcu.

A.2 ANOTHER APPENDIX SECTION TEST

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

BIBLIOGRAFÍA

[1] Sitaram Iyer Antony Rowstron, Peter Druschel. Squirrel: A decentralized peer-to-peer web cache. *21th ACM Symposium on Principles of Distributed Computing*, 2001.

[2] Fernando Raúl Alfredo Bordignon. Diseño de una plataforma de computación distribuida cooperativa, utilizando servicios de una red compañero a compañero. *Universidad Nacional de La Plata*, 2005.

[3] Michael J. Freedman, Eric Freudenthal, and David Maziéres. Democratizing content publication with coral. *New York University*, 2004.

[4] Donald E. Knuth. Computer Programming as an Art. *Communications of the ACM*, 17(12):667–673, December 1974.

[5] Leslie Lamport. How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Transactionson Computers*, pages 690–691, September 1979.

[6] Zhengang Liang, Hossam Hassanein, and Patrick Martin. Transparent distributed web caching. *26th Annual IEEE International Conference on Local Computer Networks*, page 225, 2001.

[7] Andrew S. Tanenbaum and D Wetherall. *Computer networks*. Pearson Prentice Hall, Boston, 5th ed edition, 2011. ISBN 9780132126953 (alk. paper).

[8] Wathsala Vithanage, Nuwan Gunaratne, and Nishshanka Sirisena. Dalesa cache: A peer – to – peer web cache. *Lanka Software Fundation*, 2009.

[9] Wikipedia. Caché web. http://es.wikipedia.org/wiki/Cach%C3%A9_web, 2013. URL http://es.wikipedia.org/wiki/Cach%C3%A9_web.

[10] Wikipedia. Content delivery network. http://en.wikipedia.org/wiki/Content_delivery_network, 2013. URL http://en.wikipedia.org/wiki/Content_delivery_network.

[11] Wikipedia. P2p. http://es.wikipedia.org/wiki/P2P#Redes_P2P_centralizadas, 2013. URL http://es.wikipedia.org/wiki/P2P#Redes_P2P_centralizadas.

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/

Final Version as of 25 de agosto de 2014 (`classicthesis` Version 1.0).

DECLARACIÓN

Put your declaration here.

San José, Costa Rica, Noviembre 2014

Kevin Moraga, 25 de agosto
de 2014