PROTOTIPO SISTEMA DE VIGILANCIA REMOTA ORIENTADO A ZONAS CERRADAS, BASADO EN LA PLATAFORMA DE DESARROLLO ECB_AT91, CÁMARA Y CONEXIÓN A INTERNET BANDA ANCHA.

JOHANNA ALEXANDRA ARIAS RIÁTIGA

CINDY MARCELA GARCÍA MARÍN

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS

ESCUELA DE INGENIERÍA ELÉCTRICA ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA

2009

PROTOTIPO SISTEMA DE VIGILANCIA REMOTA ORIENTADO A ZONAS CERRADAS, BASADO EN LA PLATAFORMA DE DESARROLLO ECB_AT91, CÁMARA Y CONEXIÓN A INTERNET BANDA ANCHA.

JOHANNA ALEXANDRA ARIAS RIÁTIGA

CINDY MARCELA GARCÍA MARÍN

Trabajo de grado para optar por el título de Ingeniero Electrónico

Director

MsC. Jorge Hernando Ramón Suarez

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS

ESCUELA DE INGENIERÍA ELÉCTRICA ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA

2009

Agradecimientos

Un agradecimiento especial a nuestros padres por su apoyo incondicional.

Agradecemos a todas las personas que contribuyeron en nuestro proceso de formación profesional y que directa o indirectamente apoyaron en la realización de este trabajo.

Agradecemos especialmente a nuestro director Jorge Hernando Ramón, por todo su apoyo, dedicación, aportes, enseñanzas, paciencia y comprensión en este desarrollo, al ingeniero Oscar Carrillo por su experiencia y sus aportes, y al profesor Elkim Roa por su desinteresada colaboración.

A Dios, por brindarme apoyo y fortaleza.

A mis padres, por su confianza, incondicionalidad y apoyo.

Y a todas las personas que siempre me acompañaron en esta meta y que directa o indirectamente participaron en la elaboración de este trabajo.

Johanna

A mi madre quien estuvo a mi lado en todo momento.

A mi padre por sus consejos y su amor incondicional.

Gracias por su apoyo, su orientación y que Dios los bendiga.

Con muchísimo aprecio les dedico esta obra fruto del esfuerzo.

Marcela

RESUMEN

TITULO: PROTOTIPO SISTEMA DE VIGILANCIA REMOTA ORIENTADO A LA VIGILANCIA DE ZONAS CERRADAS, BASADO EN LA PLATAFORMA DE DESARROLLO ECB_AT91, CÁMARA Y CONEXIÓN A INTERNET BANDA ANCHA[1].

AUTORES: Arias Riátiga Johanna Alexandra y García Marín Cindy Marcela[2].

PALABRAS CLAVES: Sistemas embebidos, sistemas de tiempo real, Linux, Kernel, Software libre, GPL, Copyleft, Ethernet, ECB_AT91, ARM.

DESCRIPCIÓN:

El presente trabajo describe la implementación de un prototipo funcional embebido para un sistema de vigilancia remoto. Es clave resaltar la filosofía Hardware – Software libre del desarrollo, por lo cual fue implementado bajo el sistema operativo Linux: Kernel versión 2.6, sobre la plataforma de desarrollo ECB_AT91, la cual cuenta con un procesador ARM de 180 MHz, 2 MB de memoria flash, 32 MB de memoria RAM, interfaz Ethernet, interfaz USB, entre otras especificaciones necesarias para el desarrollo del sistema.

A lo largo del documento se sitúa al lector en el entorno de los sistemas embebidos y de la comunidad Linux, software libre y las licencias usadas: GPL y Copyleft, se inicia con una descripción del proyecto, a continuación se presenta la plataforma de desarrollo, la selección de los componentes y la implementación de las tres etapas del sistema: Adquisición de imagen, transmisión de datos por Internet y aplicación. Finalmente se describen las pruebas y resultados obtenidos.

Con este trabajo de grado en la modalidad de investigación se logró un avance en el desarrollo de sistemas embebidos en el ámbito de los sistemas de vigilancia y sistemas de tiempo real, suministrando la información de la cámara, para ser visualizada en tiempo real en un equipo que cumpla unas especificaciones mínimas, permitiendo brindar comodidad, bienestar y seguridad a los diferentes miembros de la sociedad.

---

[1] Trabajo de grado

[2] Facultad de Ingenierías Físico Mecánicas. Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones. Director: Jorge Hernando Ramón Suarez.

ABSTRACT

TITLE: PROTOTYPE OF REMOTE MONITORING SYSTEM ORIENTED TO CLOSED AREAS MONITORING, BASED ON THE DEVELOPMENT PLATFORM ECB_AT91, CAMERA AND BROADBAND INTERNET CONNECTION[3].

AUTORS: Arias Riátiga Johanna Alexandra and García Marín Cindy Marcela[4].

KEYWORDS: Embedded Systems, Real Time Systems, Linux, Kernel, Free Software, GPL, Copyleft, Ethernet, ECB_AT91, ARM.

DESCRIPTION:

The present work describes the implementation of an embedded functional prototype for a remote system of monitoring. Is key to emphasize the philosophy free Hardware – Software development, thus was implemented under Linux operating system: Kernel version 2.6, on the development platform ECB_AT91, which counts with a 180 processor ARM of MHz, 2 MB of flash memory, 32 MB of RAM memory, Ethernet and USB interfaces, among others necessary specifications for the development of the system.

Throughout the document the reader is placed in the surroundings of the contracted systems and the Linux community, free software and the used licenses: GPL and Copyleft, begin with a project description, next it appears the development platform, the components selection and the implementation of the three stages of the system: Acquisition of image, Data Transmission by Internet and Application. Finally the tests and obtained results are described.

With this work of degree in the investigation modality it was obtained an advance in the development of embedded systems in the scope of the monitoring systems and realtime systems, providing the information of the camera, to be visualized in real time in an equipment that fulfills specifications minimum, allowing to offer comfort, well-being and security to the different members from the society.

---

[3] Degree Work

[4] Faculty of Physical - Mechanic Engineering. Electrical, Electronic and Telecommunications Engineering School. Director: Jorge Hernando Ramón Suarez.

Contenido

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción y Conceptos Básicos

1.1 Introducción

Las necesidades a las que se enfrenta diariamente el mundo ha llevado a la creación de diversas tecnologías y aplicaciones, además el rápido avance tecnológico de los últimos años ha favorecido notablemente el desarrollo de las mismas. En lo que a nosotros respecta, lo concerniente a sistemas de vigilancia, se han ideado diversas formas a lo largo de los años, el primer método de vigilancia usado es totalmente aislado de tecnología, es el método más sencillo, es vigilancia directa, realizado por un humano o un animal. Un método que revolucionó en ese tiempo los sistemas de vigilancia fue la conexión cableada de una cámara a un monitor o televisor, de esta forma no era necesaria la vigilancia directa, es decir en el mismo lugar, además ofrecía la posibilidad de vigilar diversos lugares al tiempo, utilizando varios monitores, o alternando las imágenes en un mismo monitor. Luego surgen las cámaras IP, que se conectan a través de una red local, o red LAN, y la imagen puede ser vista en Internet con la dirección IP de la misma. Actualmente se utilizan para la trasmisión de video por Internet se utilizan diferentes sistemas; por ejemplo una o varias cámaras CCTV combinada con una tarjeta DVR, cámaras IP, entre otros.

Un análisis de los sistemas de vigilancia existentes en el mercado para la vigilancia por Internet arroja que son de elevados costos para el público en general, importados, sin soporte técnico y la mayoría sin garantía, en consecuencia la problemática radica en que no existe en el mercado un equipo que cumpla con la función de vigilancia por Internet, con soporte técnico calificado, y que esté dentro de un rango de precios considerable para el usuario doméstico.

Con esta problemática en mente, este trabajo de grado en la modalidad de investigación representa un avance en el desarrollo de sistemas embebidos en el ámbito de los sistemas de vigilancia y sistemas de tiempo real, además busca brindar comodidad, bienestar y seguridad a los diferentes miembros de la sociedad, suministrando la información de la cámara, para ser visualizada en tiempo real desde cualquier lugar del mundo, en un equipo que cumpla unas especificaciones mínimas y tenga conexión a Internet banda ancha.

El documento está organizado de tal forma que el lector pueda llevar una secuencia lógica y viva la investigación paso a paso en la realización del prototipo.

En este capítulo se pretende situar al lector en el entorno de los sistemas embebidos y de la comunidad Linux, software libre y las licencias usadas como GPL y Copyleft que permiten el mejoramiento continuo de cada programa desarrollado con la exposición del código fuente a la comunidad. Seguido de esto se realiza una descripción del proyecto, los objetivos, la motivación a realizarlo y la descripción general de desarrollo del prototipo.

El capítulo dos, Sistema de vigilancia, presenta la plataforma sobre la cual se desarrollo el proyecto; las características de hardware y software, el procesador, la arquitectura, entre otras cosas relevantes para el desarrollo de proyecto, presenta la selección de los componentes; la cámara, el paquete de adquisición de imagen, el servidor Web y el lenguaje de programación de la página Web, finalmente presenta el desarrollo de cada una de las etapas del sistema; Adquisición de imagen, transmisión, y aplicación al usuario.

El capítulo tres, Pruebas y resultados, presenta los resultados obtenidos en la conexión del prototipo a la red.

El capítulo cuatro, Perspectiva de negocio, presenta el esquema general de la realización de un negocio, del plan de negocio, y da una perspectiva general del mercado del sistema.

Finalmente el capítulo cinco presenta las conclusiones y observaciones del proyecto.

1.2 Sistemas embebidos

Un sistema embebido es la combinación de software y hardware diseñado para una función específica. Por ejemplo un Router, una cámara digital, etc. Algunos de ellos pueden hacer parte de un sistema más grande, y pueden pasar desapercibidos por el usuario, como es el caso de un sistema de inyección de frenos dentro de un automóvil. Ahora, si bien un computador personal está formado por software y hardware, no está diseñado para un uso específico, por el contrario es multitarea, y generalmente son llamados de propósito general[5].

Algunas de las características principales de los sistemas embebidos son el bajo costo, el bajo consumo de potencia y la flexibilidad al momento de tener que realizar una modificación al sistema; en general basta con ingresar al código fuente y modificar las líneas que sean necesarias.

Figura 1.1: Ejemplos de sistemas embebidos en el mercado.

Fuente: http://www.lacofa.es/wp-content/uploads/2008/06/linux_embebido2.jpg

A gran escala un sistema embebido está compuesto por un microprocesador, un software, una memoria RAM[6], o ROM[7], memoria cache, un chipset, ranuras de expansión, dispositivos entrada-salida, y los demás dispositivos necesarios para desempeñar la función programada. Por

[5] Fuente: Yaghmour, K. Building embedded Linux systems [en línea]. O'Reilly, 2003
[6] Memoria de acceso aleatorio (Ramdom Acces Memory)
[7] Memoria de solo lectura (Read Only Memory)

ejemplo un sistema de vigilancia, a grandes rasgos estaría compuesto por una cámara, un procesador, un puerto para conectar la cámara, conexión a Internet, y memoria cache y RAM.

Los sistemas embebidos son únicos y son parte fundamental de nuestra vida actual, ellos están presentes en el trabajo, en el entretenimiento, en las comunicaciones, en los medios de transporte, entre otros. Algunos ejemplos se muestran en la Figura 1.1.

Finalmente el desarrollo de cada sistema embebido es particular, en consecuencia no existe una metodología de diseño generalizada para el desarrollo de aplicaciones sobre estos sistemas; no es lo mismo que desarrollar aplicaciones para un PC, porque un desarrollo implica conocer el sistema operativo embebido, la jerarquía de archivos, los controladores, la arquitectura, entre otras características del sistema, además de conocer las características propias de la aplicación.

1.3 Sistemas de tiempo real

Los sistemas de tiempo real (STR) son sistemas informáticos que dependen no sólo de la lógica e implementación, sino de su tiempo de respuesta. Es decir si las restricciones de tiempo no son respetadas se dice que el sistema ha fallado aún cuando la lógica e implementación sean correctas. Actualmente los sistemas de tiempo real se encuentran en nuestra cotidianidad, muchos de ellos sistemas embebidos y están en aumento directo, algunos ejemplos de ellos serían teléfonos móviles, semáforos, procesos de fabricación, entre otros[8].

Es claro que en el desarrollo de sistemas de tiempo real sobre sistemas embebidos, además de las restricciones impuestas por el sistema, es de suma importancia tener caracterizados los tiempos internos; como por ejemplo la tasa de transmisión de datos interna, tiempo de procesamiento, para evitar retardos.

[8] Fuente: Berger, A. Embedded systems design: an introduction to processes, tools and techniques. [en línea] CMP Books, 2001.

1.4 Proyecto GNU

GNU es un sistema de software creado por Richard Stallman publicado oficialmente en 1983 como "el manifiesto GNU", con la idea principal de crear un sistema operativo completamente libre y volver cooperativa a la comunidad programadora tal y como había sido en su comienzo[9].

Figura 1.2: Imagen oficial del proyecto GNU.

Fuente : http://www.gnu.org/home.es.html

GNU fue basado en Unix, en consecuencia es totalmente compatible con estos sistemas; sin embargo es clara la diferencia y se recalca con su nombre que significa "GNU is no Unix".

El emblema del proyecto es un ñu debido a la pronunciación de su nombre. Ver Figura **1.2**

El proyecto GNU abarca la realización de software y núcleo, actualmente está en ejecución y el desarrollo del núcleo no ha florecido aún, en consecuencia se ha integrado con el Kernel creado por Linus Torvalds. Está unión de núcleo y aplicaciones es llamada GNU/Linux el cual se presentará posteriormente.

[9] Fuente: http://www.gnu.org

1.4.1 Software Libre

La idea de software libre pretende dar a los usuarios libertad total de ejecutar, mejorar, cambiar y distribuir el software. Sin embargo esta libertad está condicionada básicamente por licencias que buscan la continuidad de este emblema, que las modificaciones realizadas sigan siendo bajo software libre[10]. A continuación se presentan las libertades planteadas:

- *La libertad de usar el programa, con cualquier propósito (libertad 0).*
- *La libertad de estudiar el funcionamiento del programa, y adaptarlo a las necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.*
- *La libertad de distribuir copias, con lo que puede ayudar a otros (libertad 2).*
- *La libertad de mejorar el programa y hacer públicas las mejoras, de modo que toda la comunidad se beneficie (libertad 3). De igual forma que la libertad 1 el acceso al código fuente es un requisito previo*[11]

Existen varios tipos de licencias que dependen específicamente de las libertades mencionadas anteriormente. Estas licencias se dividen en licencias permisivas y licencias robustas.

Las licencias permisivas se refieren a que no imponen condiciones al usuario en la redistribución del producto dando condescendencia en realizar lo que quiere con el producto en la siguiente distribución luego de alguna modificación; algunas de estas licencias son: BSD License, Artistic, Academia Free License, entre otras.

Las licencias robustas también suelen llamarse copyleft, son aquellas que defienden el derecho jurídico de utilización, modificación y distribución de un programa o sus derivativos. Algunos ejemplos de estas licencias son: GNU GPL, GNU LGPL o Mozilla Public License.

[10] Fuente: http://www.fsf.org
[11] Fuente: http://www.gnu.org/home.es.html

1.4.2 Licencia GPL

GNU GPL por sus siglas "licencia pública general", es la licencia que permite al autor conservar sus derechos de autoría conocidos con el término copyright. Esta licencia permite modificar y distribuir el producto siempre y cuando se conserven los derechos de cada autor, además obliga a que cualquier modificación sea publicada bajo esta misma licencia.

Esta Licencia fue creada por la fundación de software libre en los años ochenta y su principal objetivo es realizar un contrato donde se defienda los derechos de los usuarios y de los autores, se estipulan las libertades de los usuarios, compactibilidad con otras licencias, recalcando siempre el dominio público del producto.

1.5 Fundación de software libre

En los años siguientes a la creación del proyecto GNU, Richard Stallman creó la fundación para el software libre (FSF, free software foundation), con la que ha supuesto proveer recursos legislativos, económicos y legales para sostener el proyecto GNU[12].

El objetivo principal de la FSF es mantener funcionando el software libre bajo las licencias mencionadas anteriormente; en consecuencia fundamento que un software para ser libre, debe estar publicado bajo una licencia de software libre como GPL[13], FDL[14], LGPL[15] y vela por el cumplimiento de estas licencias para evitar las patentes de dicho software.

1.6 Linux

Linux es un sistema operativo que fue inicialmente creado como afición de Linus Torvalds, un joven estudiante de la universidad de Helsinki en Finlandia. Linus tuvo un interés en Minix, un pequeño sistema operativo basado en Unix, desarrollado por Andrew Tanenbaum, y decidió

[12] Fuente: http://www.fsf.org
[13] GENERAL PUBLIC LICENSE (Licencia Publica General)
[14] FREE DOCUMENTATION LICENSE (Licencia de documentación libre)
[15] LESSER GENERAL PUBLIC LICENSE (Licencia Publica General Reducida)

desarrollar un sistema que excediera los estándares del mismo. El empezó su trabajo en 1991 cuando liberó la versión 0.02. Trabajó constantemente hasta que en 1994 liberó la versión 1.0 del Kernel de Linux. La imagen oficial de Linux es Tux[16]. Ver Figura **1.3**.

El Kernel creado por Torvalds, como corazón de todos los sistemas Linux fue desarrollado y liberado bajo la licencia GNU (GNU GPL), y su código fuente está disponible para todos los usuarios. Es la base alrededor de la cual las distribuciones de Linux son desarrolladas, en consecuencia es necesaria una actualización constante de las versiones de Kernel, y estas actualizaciones siguen bajo la dirección de Linus.

Linux, aparte de ser distribuido libremente, presenta ventajas como mayor funcionalidad, adaptabilidad, y robustez frente a otros sistemas operativos, lo cual ha hecho que sea la principal alternativa para programas de marca registrada como Unix y sistemas operativos de Microsoft. IBM, HP y otros grandes del mundo de la computación están adoptando este sistema operativo, además están contribuyendo con el desarrollo del soporte técnico y con contribuciones monetarias.

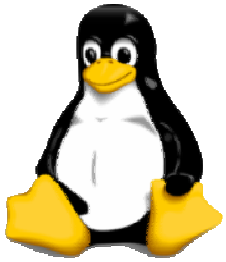

Figura 1.3: Tux, Imagen oficial de Linux.

Fuente: http://www.linux-es.org

[16] Fuente: http://www.linux-es.org

1.6.1 GNU/LINUX

El término Linux fue inicialmente usado sólo para el núcleo creado por Torvalds. Sin embargo el núcleo es usado con frecuencia con software del proyecto GNU. La unión de estos dos proyectos de software libre, la mayoría bajo licencia GPL se popularizó rápidamente ya que no existía otro núcleo libre para el proyecto GNU. Cuando la gente empezó a llamar a esta unión de proyectos "Linux", Richard Stallman[17], solicito que el nombre usado fuera "GNU/Linux" como se presenta en la Figura 1.4 para reconocer la participación de los dos proyectos. Esta solicitud generó literalmente innumerables oposiciones; mientras GNU y Debian aceptaron el nombre, la mayor parte de desarrolladores y distribuciones no lo hicieron, debido a que en muchas ocasiones el software usado por Linux no era propiedad de GNU y por otra parte, a la comodidad y simplicidad del nombre Linux. Por esta razón el nombre usado generalmente para el proyecto es simplemente "Linux".

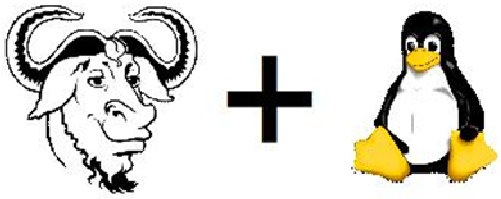

Figura 1.4: Sistema operativo GNU/LINUX.

1.6.2 Arquitectura

Linux fue diseñado bajo un núcleo monolítico híbrido, debido a las ventajas que proporcionaba tener un núcleo monolítico, como mayor velocidad, y en busca de solucionar las desventajas del mismo. La arquitectura del núcleo de Linux se mejoró de tal forma que los controladores de los dispositivos se pudieran cargar fácilmente en el sistema como módulos, cosa que no sucedía con los núcleos monolíticos los cuales eran núcleos grandes y complejos,

[17] Fundador del proyecto GNU

envolvían todos los servicios del sistema, programados de forma no modular, y no permitían la inserción de nuevos dispositivos sin tener que recompilar el núcleo completo y reiniciar el sistema.

En el núcleo monolítico híbrido, los controladores de dispositivos y las extensiones del núcleo se ejecutan en espacio conocido como ring 0 (anillo 0) aunque algunos se ejecutan en el espacio de usuario (espacio de aplicación externo al núcleo; en sistemas tipo Unix un ejemplo sería /home que es el espacio de almacenamiento de datos no críticos para el sistema). Los controladores son agregados como módulos y pueden ser prevolcados bajo ciertas condiciones, lo cual permite gestionar interrupciones de hardware.

1.6.3 Sistema de archivos

El FHS (Filesystem Hierarchy Standard) en español Estándar de jerarquía del sistema de archivos es una norma que define la jerarquía de los directorios del sistema operativo Linux (ver Figura 1.5). Se creó en 1994 para estandarizar el sistema de archivos de Linux basándose en la organización de los sistemas de archivos de Unix. En 1995 se normalizó para Linux y cualquier Unix puede adoptar esta jerarquía. Este estándar esta mantenido por FSG (Free Standards Group), una organización sin ánimo de lucro, formada por compañías de Hardware y Software como Debian, AMD, IBM, HP, MySQL, Intel, Google, Dell, Novell, Red Hat, Sun Microsystems, entre otras.

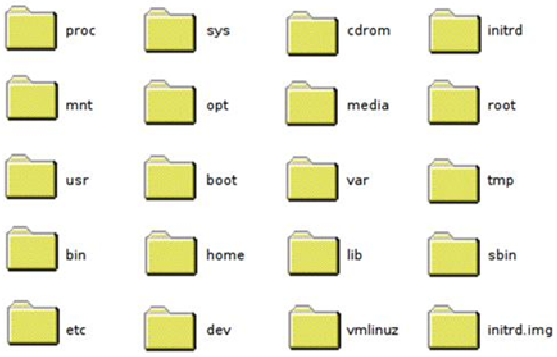

Figura 1.5: Jerarquía de directorios en Linux.

Existen varias clasificaciones de los directorios: estáticos y dinámicos, compartidos o restringidos. Los directorios estáticos no cambian sin la intervención del administrador (root), pero pueden ser leídos por los usuarios, por ejemplo, /bin, /opt, /boot. Los directorios dinámicos contienen archivos que cambian y pueden leerse y escribirse de acuerdo a los permisos designados por el usuario creador. pe: /home. Los directorios compartidos son los que se comparten entre usuarios y/u ordenadores. En contraparte se encuentran los restringidos que solo pueden ser leídos o modificados por el administrador.

Cabe resaltar que todos los archivos y directorios aparecerán bajo la raíz (/) aún los discos extraíbles, discos duros, etc.

1.6.4 Lenguaje de programación

Linux está escrito en lenguaje de programación C, y utiliza el compilador GCC. Asimismo se utilizan otros lenguajes como Perl, Python, C++, Fortan y varios lenguajes de Shell scripting para la programación de drivers y aplicaciones. Cabe resaltar que el sistema de construcción de Linux oficialmente sólo soporta GCC cómo núcleo y compilador de controlador.

1.6.5 Portabilidad

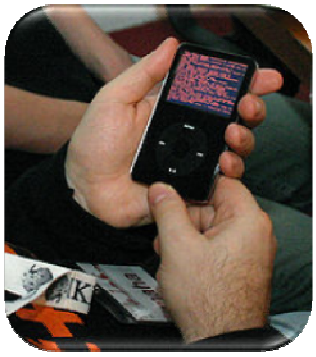

Figura 1.6: Ipod corriendo un núcleo de Linux.

Fuente: http://es.wikipedia.org/wiki/Archivo:Ipod_linux_booting_kernel.jpg

Linux no fue diseñado como sistema portable, sin embargo con el desarrollo de la tecnología ha evolucionado y apuntado en esa dirección. Actualmente es uno de los núcleos de sistema operativo más usado en sistemas portables como iPAQ, zSeries, Ipod (ver Figura 1.6), entre otros.

1.6.6 Distribuciones

Si bien se habló de Linux y sus características, existen dos formas de tenerlo instalado en nuestros computadores; bien mediante la compilación de un Kernel, o mediante la instalación de una distribución.

Una distribución es una recopilación de programas y ficheros, organizados y preparados para su instalación. Se pueden obtener a través de Internet descargando la imagen de instalación vía torrent o ftp, o comprando los CDs de las mismas en los sitios que las fabrican.

Figura 1.7: Distribuciones de Linux.

Fuente: http://fherx.files.wordpress.com/2008/07/logos_linux.png

Existen muchas y variadas distribuciones creadas por diferentes empresas y organizaciones, en la Figura 1.7 se presentan algunas de ellas, así mismo se encuentran algunas distribuciones en versiones LiveCD que permite al usuario correr el sistema operativo desde el CD con el fin de probar la distribución sin ser instalada.

1.6.7 Linux frente a otros sistemas operativos

Finalmente se presenta Linux como una excelente alternativa frente a los demás sistemas operativos, por las evidentes ventajas de costo, versatilidad, confiabilidad, velocidad, bajos requisitos de hardware, etc. Además, Linux es superior en el sistema multitarea, multiusuario, administración de memoria, y conectividad. Las únicas desventajas que presenta es la disponibilidad de software, específicamente de controladores de hardware, pero éste problema está desapareciendo debido a la adopción del sistema y la múltiple programación en el proyecto GNU.

1.7 Red y protocolo sobre IP

1.7.1 Red

Una red es un conjunto de dispositivos conectados entre si por algún medio físico o virtual, que comparten información y recursos.

Una red se puede construir a partir de los protocolos TCP/IP para conectarse con la red mundial "Internet". Los servicios prestados a partir de estos protocolos son de varios tipos entre los cuales están los más importantes los protocolos HTTP, FTP, POP3, DNS entre otros, dando la posibilidad de comunicar remotamente dos dispositivos de diferentes características cumpliendo los estándares establecidos por las organizaciones de telecomunicaciones[18].

1.7.2 Protocolo sobre IP

Los protocolos de Internet para comunicar dispositivos que componen una red están basados en el modelo TCP/IP, el cual se divide en capas, donde cada capa tiene un propósito específico, las capas son: física, enlace, interred, transporte, y aplicación.

[18] Fuente: S. Feit. TCP/IP, Arqutectura y protocolos. Mc Graw Hill.

La capa física tiene como propósito la descripción del medio físico que se utilizara para la comunicación de los dispositivos como son la fibra óptica, el cable coaxial, como también dar información de la potencias de señal, longitudes de onda, sincronización y temporización.

La capa de enlace o acceso a la red tiene como función especifica el control de flujo de datos, donde se marca la delimitan el tamaño de los paquetes y se realiza el control de errores en la tecnología de red utilizada como puede ser Ethernet , Token Ring, Wifi.

La capa de interred es la capa encargada del enrutamiento de los datos desde su punto de origen hasta su destino, ubicándolo por medio de la dirección IP y utilizando datagramas IP, tabla de enrutamiento por medio de algunos protocolos específicos para estas funciones como son los RIP o los protocolos USPF.

La capa de transporte tiene como finalidad controlar la conexión en estaciones finales, donde se encarga de manejar los datos proporcionando la fiabilidad y seguridad necesaria en el transporte de los mismos, los protocolos utilizados en esta capa pueden ser TCP o UDP, dependiendo si son o no orientados a conexión. También tienen como función determinar a cual aplicación van destinados los datos que están transportando.

La capa aplicación es la encargada de proporcionar un servicio al usuario , donde también realiza la tarea de la presentación de los datos ante el usuario o realizar también el inicio de sesión ante un programa especifico, como ejemplo HTTP, FTP, TELNET, IMAP, entre otros.

1.8 Servidores Web

El término servidor Web puede referirse principalmente a la máquina que almacena y maneja los sitios Web o al software que funciona en la máquina y maneja el servicio de la página Web de acuerdo a las peticiones de los clientes. En este caso nos referimos al software que permitirá el acceso remoto a páginas Web[19].

[19] Fuente: S. Feit. TCP/IP, Arquitectura y protocolos. Mc Graw Hill.

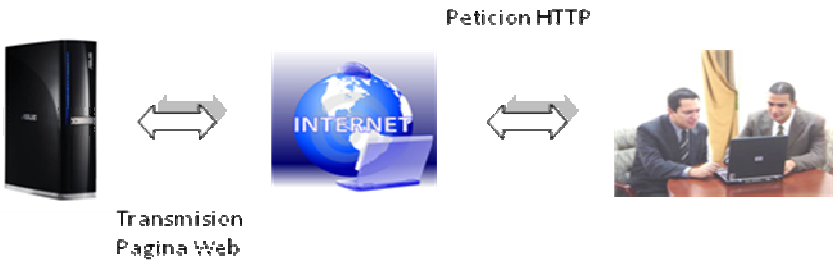

Figura 1.8: Funcionamiento de un Servidor Web.

Un servidor Web es el encargado de proporcionar acceso a los archivos y/o servicios almacenados que los usuarios soliciten. Generalmente usan el protocolo HTTP (Protocolo de transferencia de Hipertexto) para enviar las páginas Web y sirve contenido estático a un navegador. El servidor siempre está a la espera de peticiones HTTP por parte de los clientes HTTP normalmente conocidos como usuarios.

En la Figura 1.8 se presenta el funcionamiento de un servidor; el cliente envía una petición HTTP a través del navegador usado, el servidor la interpreta y le envía en respuesta la página o contenido solicitado. El cliente es el encargado de la interpretación del contenido, el servidor únicamente transfiere el contenido.

1.9 Página Web

La Web es una Red o telaraña en la cual se alojan y se vinculan los sitios Web; los sitios Web son espacios virtuales que contienen documentos organizados jerárquicamente, a estos documentos se les llama página Web[20].

Las páginas Web son fuentes de información de gráficos, texto, audio, video, dependiendo del lenguaje en el que hayan sido escritas. Estas son accedidas por medio de un navegador Web por ejemplo: Mozilla, a través de una petición HTML enviada por el usuario. Cada sitio Web

[20] Fuente: S. Feit. TCP/IP, Arquitectura y protocolos. Mc Graw Hill.

tiene una página principal, sin embargo en la mayoría de los casos éstas páginas están interconectadas a otras con hipervínculos.

Existen varios tipos de páginas Web; Estáticas, animadas, Dinámicas, portales, tiendas virtuales y gestores de contenido. Está clasificación se basa en las características de las mismas y el lenguaje de programación usado; por ejemplo una página Web animada con las que usan tecnología Flash.

Para crear una página Web es necesario un editor de texto, o en preferencia un editor de HTML. El diseño es personal aunque existen plantillas básicas para crear una página. La publicación se puede realizar por medio de un servidor Web o un servicio de hosting, algunos gratuitos.

1.10 Descripción del proyecto

1.10.1 Objetivo del proyecto

1.10.1.1 Objetivo general

Implementar un prototipo de sistema de vigilancia remota orientado a la vigilancia de zonas cerradas: microempresas, hogares, oficinas, habitaciones, basado en la plataforma de desarrollo ECB_AT91, cámara y conexión a Internet banda ancha.

1.10.1.2 Objetivos específicos

- Implementar la adquisición de la imagen
- Implementar la transmisión de datos a través de Internet
- Realizar la programación de una página Web, a través de la cual sea accesible en tiempo real la imagen.

1.10.2Descripción del sistema

El prototipo está compuesto principalmente por tres etapas; la etapa de adquisición, etapa de transporte y etapa de aplicación como se presenta en la Figura 1.9.

La etapa de adquisición consiste en la captura de la imagen proveniente de la cámara, ésta depende en gran medida del hardware utilizado, en éste caso cámara Web, entonces dependerá de la cámara y los controladores en primera medida, seguido de un software que permita la captura singular de imagen a través del puerto. Se escogió la captura de imagen y no de video para no limitar el ancho de banda con el uso del dispositivo, es decir, al transmitir frames por segundo se necesitará un ancho de banda más bajo comparado con la transmisión de video.

La etapa de transporte consiste en la transmisión de los frames por segundo a internet sobre el protocolo IP.

Finalmente se encuentra la etapa de aplicación que es la presentación final al usuario. En éste caso corresponde a una página Web que presenta la captura en frames por segundo simulando un video.

En analogía con el modelo TCP/IP se encuentran las etapas del sistema. A continuación se presenta el diagrama de bloques del sistema. Ver Figura 1.9

Figura 1.9: Diagrama de bloques del prototipo de sistema de vigilancia.

Capítulo 2

Sistema de vigilancia

2.1 Tarjeta de desarrollo ECB_AT91

En primera instancia la tarjeta ECB_AT91 es una plataforma para el desarrollo de sistemas embebidos. De acuerdo a la definición presentada en el capítulo 1, está compuesta tanto por software como por hardware y está delimitada por los mismos.

La tarjeta se presenta en la Figura 2.1. Fue diseñada Copyleft bajo licencia GNU GPL. En consecuencia se tiene el código fuente, los esquemáticos y la documentación abierta para el desarrollo de la misma.

La tarjeta está basada en el procesador AT91RM9200 de Atmel, soporta hasta 64MB de memoria SDRAM. Soporta las distribuciones Debian de sistema operativo Linux y openembedded y sus dimensiones son 85mm x 77 mm.

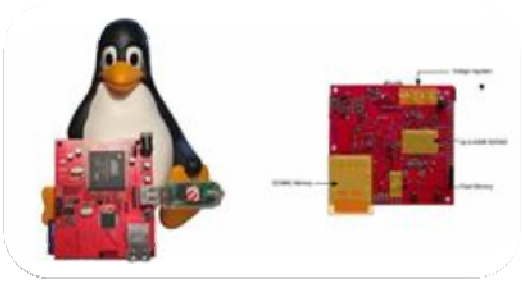

Figura 2.1: Tarjeta de desarrollo ECB_AT91.

Fuente: www.emqbit.com

2.1.1 Especificaciones de hardware

- Procesador 180 MHz ARM9 (Atmel AT91RM9200)
- 2 MB de Serial Flash
- 32 MB de SDRAM (Soporta 8M/16M/32M/64M)
- 1 SD/MMC slot (Soporta hasta 8GB)
- USB 2.0 host
- I^2C port
- 1 interfaz Ethernet 10/100
- 1 interfaz de alta velocidad USB 2.0
- 4 interfaces SPI
- 2 interfaces seriales (RS232)
- Soporte JTAG[21]

2.1.2 Organización

Como se mencionó en el capítulo 1 los sistemas embebidos están compuestos por hardware y software, la tarjeta está compuesta por el procesador, la memoria y algunos periféricos. La Figura 2.2 muestra la organización de la tarjeta en un diagrama de bloques.

[21] Fuente: http://www.emqbit.com

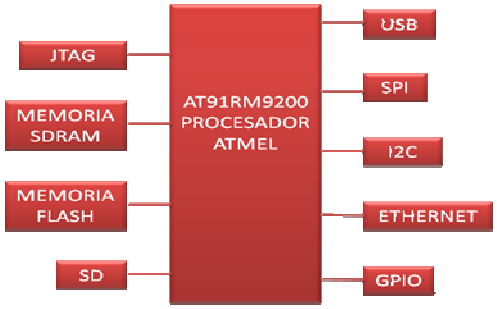

Figura 2.2: Diagrama de bloques de ECB_AT91.

2.1.3 Procesador ARM920T

2.1.3.1 Arquitectura

El AT91RM9200 es un SoC[22] de alto desempeño basado en el procesador ARM920T de Atmel, el incorpora un excelente arquitectura, múltiples periféricos e interfaces estándares para proveer una solución para el amplio rango de aplicaciones que requieren máxima funcionalidad con el mínimo consumo de potencia y bajo costo[23].

Las principales características son:

Procesador ARM920T de 180MHz

16KB de memoria de datos.

16KB de memoria de instrucciones.

Bus del sistema a 120MHz.

Interfaz JTAG para depuración.

Periféricos integrados

[22] System on chip.
[23] Fuente: Atmel. AT91RM9200, Datasheet

MAC Ethernet 10/100

2 puertos USB

MCI (multimedia card interface)

Puerto SPI

Periféricos internos

2 Timers de propósito general (PWM incorporado)

20 canales DMA para transferencia de datos

Interfaces

I²S

Pines de entrada y salida de propósito general

16 GPIOS con uso de interrupciones

8 GPIOS adicionales para manejo de periféricos

La Figura 2.3 muestra la organización del AT91RM200.

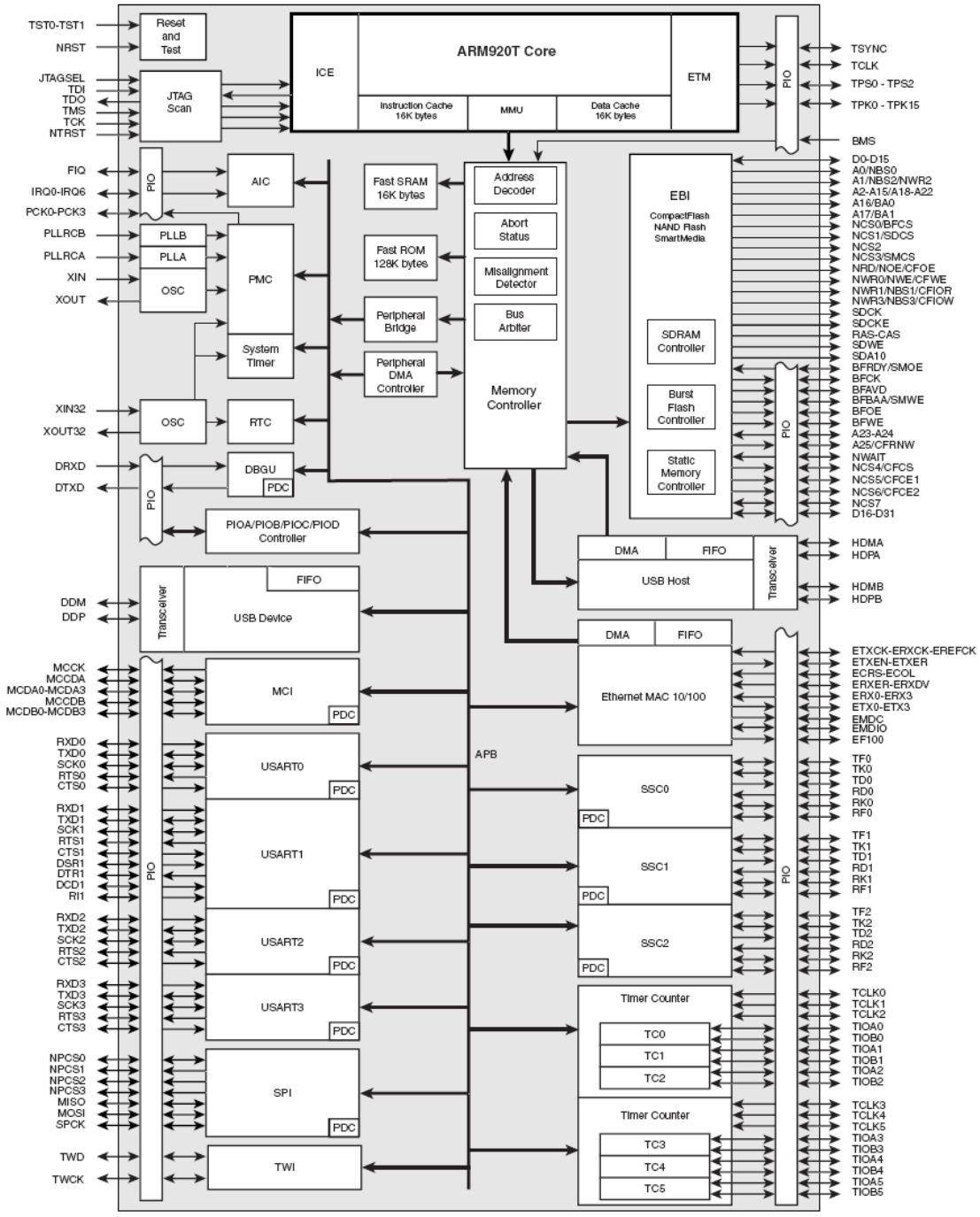

Figura 2.3: Organización del AT91RM9200.

Fuente: Atmel. AT91RM9200, Datasheet.

2.2 Selección de componentes

2.2.1 Selección de la cámara

La cámara fue un factor fundamental en los resultados y aceptación del sistema de vigilancia final. El dispositivo seleccionado deberá contar con unos requerimientos mínimos como son: resolución, iluminación, tamaño, costo, tipo de conexión y flexibilidad. A continuación se da una breve descripción de cada uno de ellos y la importancia en la selección.

- Resolución**:** Se refiere a la calidad y nitidez de la imagen y movimiento, es directamente proporcional a estas características. Se especifica en pixeles. Es importante porque en un sistema de vigilancia la imagen debe tener una nitidez y calidad aceptable.

- Iluminación: Está relacionada con la cantidad de luz que existe en el espacio de trabajo. Se especifica en Lux, a menor valor de lux se obtendrán mejores imágenes en espacios con poca luz. Ésta característica es importante pero no fundamental; el sistema está orientado a zonas cerradas, sin embargo se requieren zonas con iluminación aceptable, preferiblemente de día.

- Tamaño: Se refiere al tamaño volumétrico de la misma. En este caso el tamaño no juega un papel determinante para el prototipo. Lo será para el dispositivo final.

- Costo: Valor monetario. El costo es influyente debido a que el sistema está dirigido a la comunidad, se busca un dispositivo de bajo costo y de disponibilidad en el mercado.

- Conexión: tipo de conexión de la cámara; por ejemplo I2C, USB, Serial, entre otros. Es de suma importancia ya que de esto dependerá la etapa de adquisición de imagen y en

gran medida si es o no necesario realizar procesamiento antes de la transmisión. Así mismo está ligada con la flexibilidad o adaptabilidad del sistema final.

- Flexibilidad o adaptabilidad: se refiere a la capacidad del sistema de adaptarse a otras cámaras sin mayores inconvenientes. En el caso del tipo de conexión, se refiere a que pueda cambiar la cámara sin tener que reprogramar todo el sistema.

Los tipos de cámara que se encuentran en el mercado son cámaras Web con sensor cmos[24]. De igual forma se podrían utilizar los sensores cmos en compañía de un sistema adicional de adaptación para la captura de la imagen.

Figura 2.4: Cámara Web.

El sistema debe ser portable y flexible, en consecuencia se decidió usar una cámara Web para el prototipo, ya que existen literalmente cientos de cámaras Web en el mercado, con las características que el usuario desee.

Finalmente usar una cámara Web beneficia el sistema ya que se puede adaptar fácilmente a los requerimientos del usuario final sin tener que reprogramar el sistema por el tipo de conexión o características de la misma[25].

[24] Los sensores CMOS son utilizados en sistemas de vigilancia ya que son más rápidos, más económicos y con menor consumo de potencia que los sensores CCD. Sin embargo necesitan más luz.
[25] El único requerimiento para usar una determinada cámara Web es que esté soportada bajo Linux, ya sea por la versión del Kernel que se está usando o que sea fácilmente configurable como módulo.

2.2.2 Selección de paquete de captura de imagen

Existen variedad de paquetes que permiten la captura de imágenes o video de la cámara Web; en el sistema por simplicidad y manejo de recursos de ancho de banda tal como se describió en el capítulo 2 se realizará captura de imágenes cada determinado intervalo de tiempo[26].

El paquete seleccionado deberá contar con unas características básicas para el funcionamiento en el sistema embebido; en primera instancia deberá ser liviano, con bajo procesamiento, es decir bajo uso de procesador, veloz y sobra decir que sea bajo licencia GPL continuando con la filosofía del proyecto.

En la comunidad se encontraron paquetes que no requerían el sistema xwindow activo para la captura y manipulación de imágenes, como Streamer, camE, motion, Webcam, y otros que por el contrario si lo requerían y debido a ésto no eran de utilidad para este sistema. Cabe resaltar que en la compilación del Kernel debe estar activo V4L[27] para el funcionamiento de video. A continuación se da una descripción de los paquetes encontrados:

- Streamer[28]: Es una alternativa de fácil uso, permite la captura de imágenes ejecutando una sola línea. Permite la captura de imagen en formato jpg o de video en formato avi. Es una opción útil para tomar imágenes cada determinado intervalo de tiempo.

- camE[29]: al igual que Streamer es una alternativa de fácil uso, permite la captura de imágenes ejecutando una sola línea.

Para el sistema no existe una diferencia clave entre estos dos paquetes de captura de imagen.

[26] Fuente: http://www.linux.com/base/ldp/howto/Webcam-HOWTO/framegrabbers.html
[27] V4L por sus siglas en ingles Video For Linux.
[28] Fuente: http://linux.about.com/library/cmd/blcmdl1_streamer.htm
[29] Ibid, www.linux.com

- Webcam[30]: Se podría decir que es una versión automatizada de camE, permite la configuración y programación del mismo a través de un script de configuración y permite agregar funciones integradas con un servidor Web. Es claro que son paquetes GPL, luego cualquier modificación realizada quedara bajo la misma licencia.

- Motion[31]: Al igual que Webcam es un sistema automatizado que permite la captura de imágenes o videos de acuerdo a la programación que se haya guardado en el script de configuración; de igual forma es un paquete GPL luego los cambios realizados también quedara bajo la misma licencia. Como valor agregado presenta la característica de que puede ser programado para la captura de imagen cuando detecta movimiento. Permite guardar las imágenes en formato jpg y el video en mpg y avi.

Se realizaron pruebas de cada uno de los paquetes de captura de imagen para verificar velocidad y ajuste sobre la tarjeta de desarrollo. De acuerdo a los resultados obtenidos y las características de los paquetes presentados, se seleccionó el paquete motion ya que se ajusta a las necesidades del prototipo y presenta un valor agregado que es la detección de movimiento para la captura de imagen.

2.2.3 Selección servidor Web

Acorde con la definición presentada en el capítulo 1, el servidor Web es el encargado de permitir el acceso a la página Web donde se visualizará en tiempo real los datos suministrados por la cámara Web.

De acuerdo a ésta definición y los requerimientos del sistema, se necesita un servidor Web que permita páginas Webs dinámicas pues el contenido estará en continuo cambio. No es recomendable el uso de servidor externo ya que se tendrían mayores retardos en la transmisión y en consecuencia en la visualización final, de igual forma el sistema no pretende soportar gran

[30] Fuente: http://webcamserver.sourceforge.net/
[31] Fuente:: http://www.lavrsen.dk/twiki/bin/view/Motion/WebHome

cantidad de usuarios simultáneos por lo que el servidor no requiere ser de gran capacidad en cuanto a soporte a usuarios.

En consecuencia con lo expuesto anteriormente, el servidor será la tarjeta de desarrollo, pues permite la transmisión directa de los datos al usuario.

Como el servidor se encontrará en la tarjeta, es necesario seleccionar un servidor Web que pueda ser instalado en un sistema embebido bajo Linux, con requisitos de hardware limitados, y conservando la línea del proyecto que sea bajo licencia GPL.

En busca de servidores Web libres para Linux se encontraron literalmente gran cantidad de servidores Web algunos de ellos: AOL Server, Apache, Boa, GNOME personal Web server, Hidra, Zeus, entre otros. Sin embargo la documentación y soporte para cada uno de ellos escasa: aún cuando pueden ser de alto rendimiento, seguros, estables, se eligió para el sistema un servidor Apache que cuenta con múltiples usuarios vinculados a proyecto, documentación, y las características y funciones necesarias para este sistema.

A continuación se da una descripción de Apache[32].

El servidor Apache es uno de los proyectos de "Apache Software Foundation", en actual desarrollo dentro del proyecto "HTTP Server (httpd)". Es software libre para todas la plataformas sobre las que está soportado. Es altamente aceptado en la red y es uno de los más usados. Presenta dentro de sus características fácil configuración[33], autenticación, modular y con amplio soporte de la comunidad. Entre los módulos configurables se encuentra SSL para la seguridad. Así mismo permite la integración con PHP, Perl, Python, entre otros.

2.2.4 Selección lenguaje programación página Web

Las páginas Webs son consideradas hoy día como fuentes de información y aprendizaje, ya que los usuarios pueden interactuar en ellas, por consiguiente es necesario optar por un lenguaje de programación que permita realizar cambios rápidos de información, y le brinde

[32] Fuente: www.apache.org/
[33] Modo texto

tanto como al programador como a los usuarios cada vez mas aplicaciones, facilidades ya sea por el manejo de su lenguaje o su estructura.

A continuación se presentan los lenguajes más comunes de programación para la Web.

HTML[34] (por sus siglas en ingles Hyper Text Markup Language) es el lenguaje de marcas de texto utilizado normalmente en la WWW (World Wide Web). Fue creado por Tim Berners-Lee en 1986. HTML no es propiamente un lenguaje de programación como C,C++, Visual Basic, etc, sino un sistema de etiquetas o tags donde sus notaciones son encerradas dentro de una instrucción (<y>). HTML no presenta ningún compilador, por lo tanto algún error de sintaxis que se presente éste no lo detectará y se visualizara en la forma como éste lo entienda. El entorno de trabajo de HTML es un procesador de texto en cualquier sistema operativo como Windows o Linux por ejemplo, el conjunto de etiquetas que se creen, se deben guardar con la extensión .htm o .html y podrán ser visualizadas por un navegador Web.

PHP[35] (es el acrónimo de Hypertext Preprocessor), es un lenguaje de programación de código abierto , distribuido bajo la licencia GNU, el cual puede ser un lenguaje embebido del código HTML, está pensado para funcionar como una interfaz de vía de acceso común. Éste lenguaje posee una sintaxis similar a C, Java**,** el cual es un lenguaje interpretado, no necesita ser compilado por lo que permite realizar cambios rápidos a diferencia de C. PHP cuenta con un gran número de funciones entre las que se encuentra la creación de archivos PDF, la creación y codificación de imágenes, accesos a ftp, creación de sockets, etc.

JSP[36] por sus siglas en ingles "Java Server Pages". Es un lenguaje para la creación de sitios Web dinámicos. Está orientado a desarrollar páginas Web en Java. Es un lenguaje multiplataforma. Creado para ejecutarse del lado del servidor. JSP fue desarrollado por Sun Microsystems. Comparte ventajas similares a las de ASP.NET, desarrollado para la creación de aplicaciones Web potentes. Posee un motor de páginas basado en los servlets de Java. Para su funcionamiento se necesita tener instalado un servidor Tomcat.

[34] Fuente: S. Feit. TCP/IP, Arqutectura y protocolos. Mc Graw Hill.
[35] Fuente: http://www.php.net/
[36] Fuente: http://java.sun.com/products/jsp/

Python[37]: Éste lenguaje de programación permite la creación de todo tipo de programas incluyendo los sitios Web. Su código no necesita ser compilado, del mismo modo que PHP es un lenguaje interpretado.

De acuerdo a la descripción de los lenguajes de programación se presenta una tabla comparativa entre ellos. Ver Tabla 2.1

	HTML	PHP	JSP	Python
Ventajas	Sencillo Estructurado Archivos pequeños Fácil aprendizaje. Multiplataforma	fácil aprendizaje Orientación a objetos Multiplataforma Conexión con SQL Modular Documentación Es libre	Ejecución rápida del servlets. Crear páginas del lado del servidor. Multiplataforma. Código bien estructurado. Integridad con los módulos de Java.	Libre Lenguaje de propósito general. Sencillo Multiplataforma. (Opensource). Orientado a Objetos. Portable.
Desventajas	Estático Limitado	Se necesita instalar un servidor Web Todo el trabajo lo realiza el servidor	Aprendizaje complejo	Lentitud

Tabla 2.1: Comparación entre lenguajes de programación de páginas Web.

Requerimientos del sistema de vigilancia para a página Web:

En el sistema de vigilancia el paquete de captura de imagen tiene dos posibilidades de salida para la transmisión Web de acuerdo a lo presentado más adelante en el capítulo 5. El sistema de captura se programó de tal forma que el Streaming saliera por un puerto virtual, luego con el lenguaje HTML se puede visualizar sin mayor inconveniente en cualquier navegador sin la necesidad de codecs, ni características especificas. La otra forma es visualizando las fotos que

[37] Fuente: www.**python**.org/

son grabadas cada vez que se detecta el movimiento, de igual forma con el lenguaje HTML se puede realizar un refresh[38] automático a la página Web cada determinado tiempo y se programa para evitar que las imágenes sean guardadas en la memoria cache de los usuarios.

En consecuencia, acorde con la comparación de los lenguaje presentada en la Tabla 2.1 y a las necesidades del sistema, se eligio el lenguaje HTML, que es un lenguaje sencillo, estructurado, multiplataforma, y de fácil aprendizaje. Sin embargo debido a la necesidad de autenticación de usuario, es necesario usar la combinación del lenguaje de programación HTML, con PHP.

2.3 Sistema de vigilancia

Los sistemas de vigilancia remota han sido creados para la facilidad del usuario en varios campos de su vida, debido a su capacidad de brindar seguridad y confianza, en aplicaciones comerciales, industriales o familiares, evitando así múltiples delitos, como robos , maltratos a infantes, a personas mayores de edad, entre otros.

2.3.1 Etapa de adquisición de imagen

La captura de imágenes se realiza por medio de una cámara USB. En primera instancia es fundamental tener el soporte bajo Linux para este dispositivo. Tanto en el PC como en la tarjeta de desarrollo es necesario insertar un driver o controlador para la cámara usada.

Para realizar la inserción de dicho controlador, primero se debe conocer que driver le sirve a la cámara, o mejor aún que drivers se tienen disponibles para seleccionar la cámara Web de acuerdo a esta limitación. El proceso de instalación del driver de la cámara tanto en el PC como en a tarjeta se encuentra en el anexo C.1.

Una vez se tiene soportada la cámara se procede con el acople del dispositivo con el programa de captura de video seleccionado, en este caso motion. Se debe conocer en que puerto

[38] Refrescar

quedo instalada la cámara Web para poder enlazarlo en el script de configuración del programa de captura. En el anexo C.2 se encuentra explicado detalladamente el proceso de instalación y configuración del programa de captura.

Seguido de esto, se tendrá captura de imágenes en formato JPEG, cada determinado inérvalo de tiempo, es decir se tendrán frames por segundo, lo que será semejante a un clip de "video" en tiempo real. Finalizada la captura, viene la etapa de transmisión.

2.3.2 Etapa de Transmisión

La transmisión digital de "video", tomado como secuencias sucesivas de imágenes es en estos días muy conocida por su utilización en videoconferencias, distribución de videos o TV, este sistema de transmisión presenta algunas limitaciones de velocidad, ancho de banda o pérdida de datos en la red.

La transmisión de estos frames por segundo se realizará a través de Internet. Cabe resaltar que el "alcance" de transmisión en el prototipo está limitado por la dirección IP privada que posee la tarjeta, es decir, la transmisión únicamente podrá realizarse dentro de la red LAN de la universidad. Para un desarrollo final es necesario tener una IP pública visible al exterior, visible por todas las redes externas de Internet con el fin de poder ser accedida desde cualquier lugar del mundo.

Para la transmisión se seleccionó un servidor Web en la tarjeta de desarrollo, tal y como se presentó en el capítulo de selección de elementos. De acuerdo a la definición presentada en el capítulo 1, el servidor funciona con un esquema cliente-servidor con el cual se realiza el enlace entre los datos trasmitidos por la cámara Web y la página de aplicación. Para conocer en detalle la instalación y configuración del servidor, remitirse al anexo C.3.

2.3.3 Etapa de aplicación: Página Web

Finalmente se encuentra la capa de aplicación, la página Web donde se presenta el contenido del Streaming.

La página Web fue desarrollada en lenguaje HTML, en conjunto con PHP, el cual permite el manejo de autenticación de usuarios. La página inicialmente solicita datos de usuario y contraseña, seguido de esto ingresa a ver el Streaming sin necesidad de descarga de codecs, o programas para la visualización.

La página Web debe ser alojada en la carpeta de servidor Web, de tal forma que pueda ser accedida por los usuarios. El código de la página Web y la configuración se encuentran en el anexo C.4.

Es clave mencionar que el acceso a la página Web en el prototipo se realiza por medio de la dirección IP, seguida del nombre de la página por defecto, sin embargo para un desarrollo final es excelente alternativa contratar los servicios del Sistema de Nombres de Dominio (DNS) de Internet con el fin de facilitar al usuario el ingreso al portal Web.

Capítulo 3

Pruebas y Resultados

Las pruebas realizadas al prototipo primero se evaluaron en equipo de cómputo personal para implementar después los resultados obtenidos en la tarjeta de desarrollo.

Las pruebas en el equipo de cómputo se dividieron en tres etapas, la etapa de adquisición de la imagen, la etapa de transmisión de datos y la última etapa la visualización de la página Web dentro de la red local.

3.1 Etapa de Adquisición de la imagen

En primera instancia se comprobó que la cámara estuviera soportada bajo Linux:

Para comprobar la compatibilidad con la plataforma se conectó la cámara directamente al equipo de trabajo y se ejecuta la instrucción "lsusb" en la consola. La anterior instrucción arroja la información necesaria para encontrar el controlador y se instala de acuerdo al anexo C. Con el comando lsmod se verifica que esté activo el módulo de controlador y con cat /proc/devices que se encuentre activo el puerto de video.

Una vez verificado el funcionamiento de la cámara Web, se comprueba el funcionamiento del paquete de captura de imagen. Es necesaria realizar la configuración del paquete de acuerdo al puerto de video.

Tal y como se presentó en el capítulo 2.2.2, se realizó la prueba de varios paquetes, para seleccionar la mejor alternativa, en cuanto a funcionamiento y características técnicas.

Se experimentó primero el comando de la librería de Linux, el comando Streamer, con la que se utiliza una instrucción para determinar el formato de imagen o video que se desea a la salida como los frames por segundos, el proceso se ejecuta mientras no se le dé la orden de

parar con las teclas Ctrl.+C. Los resultados fueron exitosos, obteniendo la captura de imagen correspondiente.

```
$streamer –o nombre.jpeg

#instrucción si se desea una imagen

$streamer –o nombre.avi –r 20 –t 200 –f yuy2 –f stereo

#en la instrucción  se da  la cantidad de frames (200) y los fps (20).
```

El segundo paquete fue camE. Primero debe descargar el paquete de camE y la librería giblib, luego de instalarlo se ejecuta de forma similar al Streamer, en una sola línea se configura lo necesario como los frames por segundo y el formato de imagen o video. En esta prueba se obtuvo la captura de imagen deseada, sin embargo el proceso es lento alrededor de 1 segundo para la captura de una imagen. En consecuencia este paquete para una aplicación en tiempo real no es adecuado.

La tercera forma examinada es el paquete Webcam Server, donde se debe descargar e instalar desde la consola del equipo de cómputo, luego en una línea se especifica resolución y formato de salida. Cabe mencionar que este paquete obtiene una secuencia de fotos en formato JPEG.

```
$webcam-server –v –g 352x288

#coloca a andar el programa.

$webcam-server –h

#cuando se necesita ayuda en la modificación de la configuración.
```

El último paquete examinado en el equipo fue motion, este paquete se instala de forma manual y luego se programa en el script de configuración las características requeridas; como

definición del dispositivo de video, resolución de imagen, formato de salida, número de frames por segundo, el puerto de control, puerto de configuración, etc. (ver anexo C.2). Los resultados fueron exitosos, se consiguió la captura de imagen en tiempo real de acuerdo a la detección de movimiento.

La adquisición en el computador se considera aceptable, maneja hasta 25 frames por segundo, los cuales permiten apreciar el movimiento. El valor máximo de fps está limitado por la cámara, el procesador y la velocidad de transmisión de datos manejada por el DMA. De acuerdo a las características técnicas de las cámaras Web, gran parte de ellas soporta máximo esta velocidad en fps.

Las pruebas realizadas en la tarjeta no fueron satisfactorias, se presentan varios inconvenientes; primero, la tarjeta de desarrollo utiliza un sistema operativo Linux embebido, el cual maneja limitaciones en cuanto a utilidades del sistema. En el caso del procedimiento aplicado para la captura de imagen, la tarjeta no soporta la mayor parte de las librerías requeridas para el manejo de video: opencv, xfree86, jpeg, entre otras, imposibilitando la captura o adquisición de imagen por medio de paquetes de Streaming o códigos de adquisición basados en librerías de manejo de imágenes. Segundo, Linux, en general, presenta limitación con los controladores de las cámaras Web y el manejo de video; los controladores existentes en el mercado no satisfacen las necesidades de los usuarios. Tercero, Actualmente se encuentra en una transición de video V4L1 a V4L2, lo cual trae incompatibilidades entre hardware y software. Es decir incompatibilidad entre los controladores de hardware y el soporte de software.

3.2 Etapa de transmisión de datos

En esta etapa se implementa el servidor Web Apache, para realizar la transmisión de la tarjeta de desarrollo ECB_AT91 y un equipo de cómputo conectados en la misma red local.

Para comprobar el funcionamiento del servidor lo que se hizo primero fue instalar el servidor Web en el equipo de cómputo como se ve en el anexo C.3.1.

Después se activa en el equipo que funciona como servidor con la instrucción:

```
$etc/init.d/apache2 start
```

Luego desde el equipo cliente, se buscara en el navegador Web la dirección IP de equipo y se espera que presente la página de inicio de Apache.

```
http://192.168.46.45
```

El resultado de la etapa de transmisión fue exitoso tanto en el equipo de cómputo como en la tarjeta de desarrollo, de lo cual se deduce que cuando se aplique el prototipo fuera de una red interna su comportamiento va a ser optimo.

3.3 Etapa de visualización de la página Web.

En esta etapa lo que se realizó fue la programación de 2 páginas Web, una solo en lenguaje Html y la segunda combinando lenguaje PHP y HTML.

Para las páginas realizadas con PHP, se instala en primera medida el programa de PHP de la página principal con librerías para aplicaciones adicionales como el manejo de imágenes y se configura el archivo apache2.conf, de apache para que pueda ejecutar este lenguaje de programación (ver anexo C.3.2).

Las páginas Web se guardan en el directorio "var/www/" y después se abren en el navegador del equipo de cómputo que funciona como cliente de la siguiente forma:

```
http://192.168.46.45/prueba.html

http://192.168.46.45/prueba2.php
```

Después de abrir las páginas con el navegador Web se revisa el hipervínculo que comunica con el puerto donde se ve el video capturado.

El resultado de la etapa es exitoso en el equipo de cómputo donde deja ver a los usuarios conectado en la red local el Streaming de video en tiempo real. En la tarjeta de desarrollo presenta limitación la programación en PHP, debido al hecho que no soporta las librerías necesarias, en consecuencia, no es posible realizar la autenticación de usuario.

Capítulo 4

Perspectiva de Negocio

En primera instancia se presenta un esquema general del proceso de desarrollo de un negocio; la Figura 4.1 muestra cada una de las fases de este proceso. Seguido de esto nos centraremos en un bosquejo de lo que podría ser el inicio de un plan de negocio.

La primera fase es la descripción de un producto o servicio. Con la idea descrita se procede a realizar un análisis superficial de posible nicho de mercado con el fin de tener una idea del futuro éxito o fracaso del negocio. Si el análisis preliminar arroja buenos resultados, se elabora un plan de negocios para estructurar detalladamente la empresa ó negocio. Una vez realizado el plan, se presenta a los inversionistas para conseguir financiación, si es aprobada se procederá con el montaje y puesta en funcionamiento del negocio, paso a paso se irá aplicando el plan de negocio desarrollado, para el éxito del negocio es fundamental tenerlo siempre presente, sin llegar a ser inflexibles ante las variaciones del mercado. Una vez establecida la empresa se debe rehacer el modelo continuamente, con el fin de buscar encontrar la excelencia en el sistema de gestión de calidad.

Definido el proceso de desarrollo de un negocio, se presenta el plan de negocio como una de las etapas de más importantes ya que este plan constituirá los pilares de la empresa. En gran parte el éxito depende de un plan de negocios desarrollado contemplando la mayor cantidad de variables posibles.

Un plan de negocios está conformado por ciertos aspectos relevantes en la estructura del negocio, cada uno de ellos actúa en forma conjunta con el otro, es decir, en caso de que uno de

ellos sea modificado, será necesaria la adecuación de los demás. La Figura 4.2 presenta los componentes de un plan de negocios.

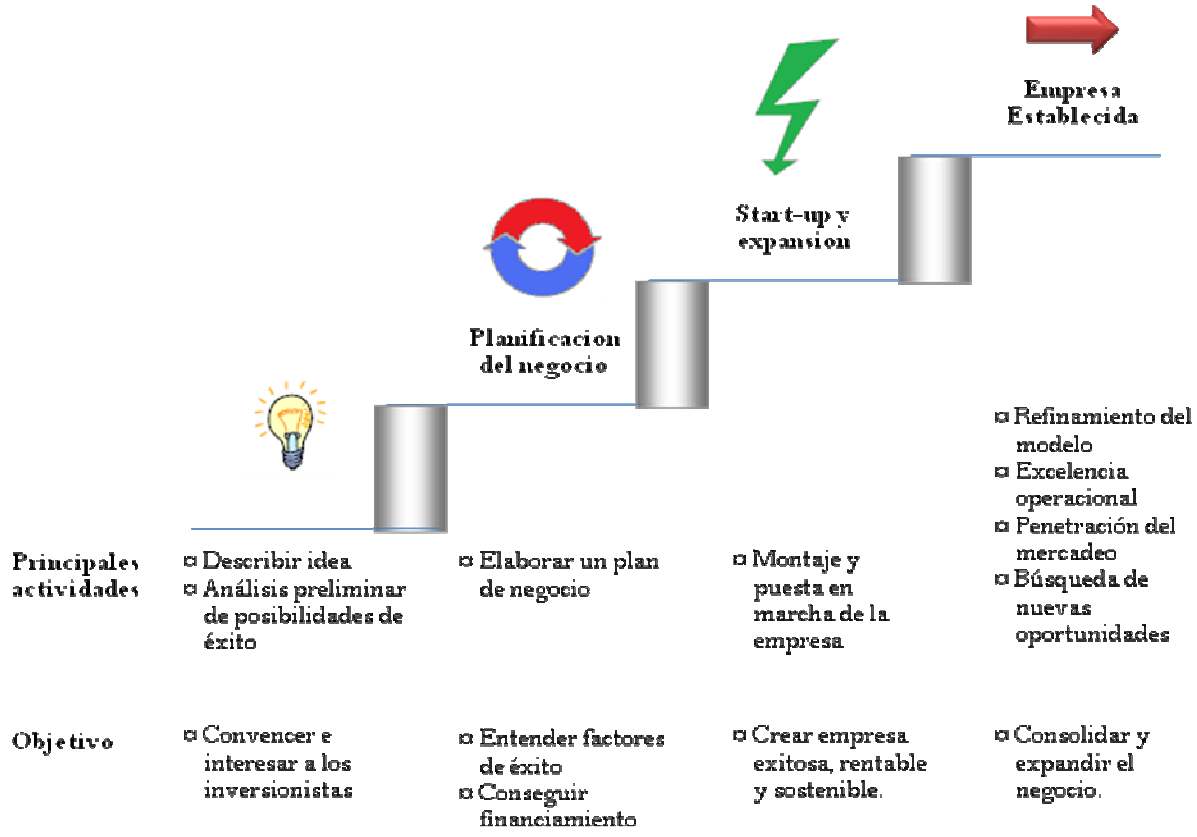

Figura 4.1: Proceso de desarrollo de un negocio.

Figura 4.2: Estructura de un plan de negocios.

A continuación se presenta una perspectiva de la oportunidad de negocio.

4.1 Resumen

Empresa:	Empresa de servicios de ingeniería, apoyada por ingenieros de sistemas y electrónicos, que desarrollan y comercializan sistemas de vigilancia.

Productos y servicios:	¤ Sistema de vigilancia por Internet.
	  ¤ El sistema es de bajo consumo de potencia.
	  ¤ Fácil instalación y uso.
	  ¤ Requiere de conexión a Internet.
	  ¤ Cuenta con un año de garantía y soporte técnico.
	  ¤ Es un sistema de software y hardware libre.
	  ¤ Es de bajo costo.
	¤ Servicio de mantenimiento y soporte técnico

Mercado y competencia:	¤ Clientes: Hogares, empresarios, investigadores, logistica. ¤ Mercado: Bucaramanga y el área metropolitana, aproximadamente 1 millón de habitantes. Fuente: DANE ¤ Competencia: Cámaras IP.
Mercadeo :	Campaña publicitaria para dar a conocer el producto.
Sistema de negocio:	¤ Venta de sistemas de vigilancia ¤ Servicio de soporte técnico
Oportunidades y riesgo:	¤ Aprobación del sistema

- Prueba de eficiencia operacional
- Certificación de calidad

Equipo de trabajo:

- Equipo de ingeniería: investigación y desarrollo, producción, y soporte técnico
- Equipo de ventas y marketing
- Equipo administrativo

Cabe resaltar que cada integrante de la empresa puede pertenecer a diferentes equipos de trabajo.

4.2 Producto

El sistema de vigilancia ofrecido es un servidor de Streaming que permite a los usuarios ver el video en tiempo real desde cualquier lugar del mundo a través de una conexión a internet y un explorador web. El video Streaming podrá ser visto desde cualquier explorador web comercial sin la necesidad de descarga de codecs o programas de visualización de video.

El sistema transmitirá alrededor de 20 fotogramas por segundo en su estado de mayor movimiento, en caso de no detectar cambio en la escena, no transmitirá nada, disminuyendo así el ancho de banda usado.

La página Web cuenta con un sistema de autenticación de usuario con el fin de garantizar la privacidad del usuario. Además cuenta con una aplicación de configuración que permite ajustar brillo, intensidad, cantidad de fotogramas, alarmas, entre otros.

- El sistema es de bajo consumo de potencia.
- Fácil instalación y uso.
- Requiere de conexión a Internet.

- Cuenta con un año de garantía y soporte técnico.
- Es un sistema de software y hardware libre.
- Es de bajo costo.

4.3 Costo

A continuación se presentan los costos de producción del prototipo. Ver Tabla 4.1.

	Costo
Cámara Web	$30000
Tarjeta de Desarrollo	$250000
Costo del prototipo	$280000

Tabla 4.1: Relación de costos del prototipo[39].

Finalmente el costo de producción del prototipo es $280.000 sin embargo es importante aclarar que la investigación y desarrollo de un prototipo tiene un costo elevado respecto al costo de producción en grandes volúmenes, además que los elementos utilizados en el prototipo pueden ser desarrollados para esta única función; por ejemplo en el diseño final del sistema de vigilancia, la tarjeta de desarrollo sería cambiada por una tarjeta que ejecute exclusivamente esa función, luego necesitaría sólo algunos de los elementos encontrados en ella y disminuiría los costos. La ganancia sería a producción en escala.

4.4 Mercado

Mercado total: El producto está dirigido a Bucaramanga y su área metropolitana.

[39] Este costo no contiene valor de montaje y puesta en servicio.

Nicho de mercado:	Personas naturales o jurídicas que requieran vigilancia de sus hogares, de sus empresas, sitios de trabajo, etc. Colegios, salones de belleza, salas de Internet, empresas de calzado, empresas de confección textil, almacenes, micro mercados, edificios, logística de eventos, seguridad industrial, investigación criminalística, joyerías, etc.
Tamaño y crecimiento del mercado:	Crecimiento promedio de la población y crecimiento empresarial 1.2 millones de habitantes en Bucaramanga y el área metropolitana, y 2.1 millones de habitantes en Santander. Crecimiento 2015 Según DANE. En la medida de crecimiento de la empresa se puede ampliar el mercado a todo el departamento de Santander. Visión a 5 años expansión a varios departamentos del país.
Competencia:	La principal competencia son las cámaras IP, en la mayoría de los casos, son importadas y vendidas en centros de tecnología o vía Web. No ofrecen instalación ni soporte técnico garantizado. Son de costos superiores, aunque algunas de ellas manejan Voz.
Posicionamiento:	Las cámaras IP son excelente opción de vigilancia, sin embargo este sistema de vigilancia no pretende ser su remplazo, ni está en busca de su nicho de mercado. Las cámaras IP manejan otras funcionalidades; por ejemplo voz, rotación de la cámara entre otras. Lo cual representa mayor

funcionalidad y mayor costo. El producto presenta las características mencionadas anteriormente y su fuerte es el bajo consumo de ancho de banda, fácil configuración, soporte técnico, y bajo costo.

4.5 Sistema de negocio

4.5.1 Cadena de valor

Figura 4.3: Cadena de Valor.

La primera franja indica el proceso de investigación y desarrollo del sistema de vigilancia, y cada una de sus innovaciones, estará a cargo del equipo de Ingeniería de la empresa.

La siguiente es la producción, se realizará en serie, teniendo en cuenta que la producción a escala arroja mayores ganancias, sin embargo es fundamental para la empresa tener un flujo de dinero constante por lo cual no se producirá en grandes cantidades, ya que representaría tener capital invertido en materia prima y en bodega, y tener una bodega. Estará a cargo del equipo de ingeniería de la empresa.

La franja de mercadeo y ventas se refiere básicamente a las ventas directas y ventas via Internet, y el marketing realizado. En el siguiente ítem se presentara el marketing. Estará a cargo del equipo de ventas y marketing de la empresa.

La distribución, abarca entrega de productos a distribuidores y clientes. Y envio de ventas por Web. Estará a cargo del equipo administrativo y una compañía de transporte.

Por último la franja de servicio comprende la instalación, asesoramiento post-venta y servicio al cliente. Estará a cargo del equipo de ingeniería y administración depediendo del requerimiento del cliente.

4.6 Marketing

4.6.1 Análisis DOFA

DOFA	
Debilidades	Fortalezas
Empresa nueva Sistema no reconocido - certificado	Capacidad de servicio Equipo solido y competitivo Tecnología actual Orientación al servicio y al cliente
Amenazas	Oportunidades
Irrupción de competidores	Crecimiento de demanda Ausencia de competencia de calidad

Tabla 4.2: Análisis DOFA.

4.6.2 Política de producto y servicio

Tal y como se presentó el producto es un sistema de vigilancia con garantía de un año , servicio de mantenimiento y servicio soporte técnico.

4.6.3 Política de precios

La comercialización del producto se orientará por tres líneas:

- Venta directa.

- ¤ Venta Web. A menudo se realizaran promociones en lapsos de tiempo establecidos en las ventas Web.
- ¤ Servicio de mantenimiento y soporte técnico

4.6.4 Estrategia de penetración en el mercado

El marketing. Durante los dos primeros años, especialmente el primero se realizará un esfuerzo publicitario con el fin de posicionar la marca, dar reconocimiento a la empresa y al producto.

Los medios publicitarios deben generar los contactos, el equipo de mercadeo convierte esos contactos en citas y potenciales clientes, el equipo de ventas concretará los clientes.

4.6.4.1 Publicidad

La publicidad se realizará en tres líneas específicas:

- ¤ Publicidad en prensa y revista
- ¤ Marketing directo (mails)
- ¤ Publicidad por Internet.

Capítulo 5

CONCLUSIONES

La motivación principal para la realización de este trabajo de grado en modalidad de investigación, fue avanzar en la búsqueda por encontrar un mecanismo para el desarrollo e implementación de sistemas embebidos en el ámbito de los sistemas de vigilancia y sistemas de tiempo real. Además, de acuerdo al propósito fundamental de la ingeniería, continuar en la búsqueda de brindar comodidad, bienestar y seguridad a los diferentes miembros de la sociedad, eliminando la necesidad de establecer una vigilancia directa permanente, permitiendo realizarla de forma remota a través de una cámara e Internet.

A continuación se citan en primer lugar conclusiones generales acerca del sistema de vigilancia, posteriormente los resultados de la investigación, y finalmente las líneas de desarrollo de esta investigación.

5.1 Conclusiones generales del sistema de vigilancia

Este sistema de vigilancia representa un avance en el desarrollo e implementación de sistemas de sistemas embebidos en la que el beneficio social predomina, tal y como se presentó a lo largo del documento, en busca de bienestar, comodidad, funcionalidad, y bajo costo. Cabe resaltar que la filosofía del proyecto es software libre, al igual que hardware libre.

En relación con la adquisición de imagen:

En primera instancia, en la selección de la cámara Web fue un factor fundamental el controlador de la misma, de acuerdo a esto, la selección estuvo limitada únicamente a las cámaras soportadas por el Kernel a utilizar en la tarjeta de desarrollo.

En la etapa de adquisición de imagen en el computador; la adquisición se considera aceptable, maneja hasta 25 frames por segundo, los cuales permiten apreciar el movimiento. El valor máximo de fps está limitado por la cámara, el procesador y la velocidad de transmisión de datos manejada por el DMA. De acuerdo a las características técnicas de las cámaras Web, gran parte de ellas soporta máximo esta velocidad en fps.

En la tarjeta de desarrollo no se alcanzaron los resultados esperados, el sistema operativo Linux embebido maneja limitaciones en cuanto a utilidades del sistema. En el caso del procedimiento aplicado para la captura de imagen, la tarjeta no soporta la mayor parte de las librerías requeridas para el manejo de video, imposibilitando la captura o adquisición de imagen por medio de paquetes de Streaming o códigos de adquisición basados en librerías de manejo de imágenes. Es importante mencionar que este procedimiento fue implementado inicialmente en un computador, en donde se obtuvo exitosamente la captura de imagen.

En relación con la transmisión de datos sobre IP:

Se implementó un servidor Web en la tarjeta de desarrollo, de tal forma que los usuarios puedan acceder a la aplicación desde la red externa. En el desarrollo de este prototipo, el acceso está limitado a la red LAN de la universidad debido a que se posee una IP privada. Las pruebas realizadas demuestran que el servidor es óptimo para la aplicación, cabe aclarar que para esta aplicación no es necesario el soporte de gran cantidad de usuarios por el servidor.

Como valor agregado del servidor, Apache permite la inserción de diversos módulos y es de fácil configuración en modo texto, esto permitió la conjugación con PHP necesaria para la autenticación de usuarios

En relación con la aplicación página Web:

La página Web fue desarrollada en lenguaje HTML por su simplicidad, facilidad de programación y por ser multiplataforma. Además fue complementada con el lenguaje PHP para realizar la autenticación de usuario.

En el computador fue completamente funcional, sin embargo la instalación de PHP en la tarjeta no fue posible debido a que no soporta la instalación de librerías de manejo de imágenes, en este caso JPEG. En consecuencia la página web de la tarjeta no maneja autenticación de usuario.

La página permite a los usuarios acceder desde la red externa y visualizar el Streaming de video en tiempo real. Es de gran importancia resaltar que no se necesitan ningún códec o programa especial para su reproducción.

5.2 Resultados de la investigación

Sistemas embebidos:

El desarrollo de cada sistema embebido es particular, en consecuencia no existe una metodología de diseño generalizada para el desarrollo de aplicaciones sobre estos sistemas; no es lo mismo que desarrollar aplicaciones para un computador, porque un desarrollo implica conocer el sistema operativo embebido, la jerarquía de archivos, los controladores, la arquitectura, entre otras características del sistema, además de conocer las características propias de la aplicación.

Sistema de vigilancia:

En la implementación del sistema de vigilancia, la inserción de drivers externos al sistema operativo embebido es compleja, en primera instancia se debe realizar una compilación cruzada de ellos, además de esto es necesario aplicar parches que lo acomoden a la arquitectura. Esto presenta una gran limitación y que el fabricante proporciona estos parches y en su gran mayoría no están disponibles o desarrollados.

Sistema operativo Linux:

Linux es excelente entorno para el desarrollo de aplicaciones, pues permite conocer hasta el mínimo detalle de su arquitectura, facilitando así, el desarrollo de las mismas. Sin embargo es claro que es un sistema en actual desarrollo y tiene grandes limitaciones en cuanto a los controladores de hardware, y las aplicaciones disponibles.

En lo estrictamente relacionado con el sistema de vigilancia, Linux, presenta limitación con los controladores de las cámaras Web y el manejo de video. Actualmente se encuentra en una transición de video V4L1 a V4L2, lo cual trae ciertas desventajas: la transición fue adoptada por los desarrolladores de Kernels, es decir las versiones iguales o superiores a 2.6 vienen con soporte del nuevo estándar de video, los drivers en consecuencia son compatibles con el nuevo estándar, sin embargo el desarrollo de aplicaciones está atrás y continua con el soporte de V4L1. Esta incompatibilidad no hace posible el acople de hardware y software por el momento.

Proyecto GNU:

GNU, ponen a disposición de los usuarios aplicaciones con código abierto desarrolladas por cientos o miles de desarrolladores en el mundo, permitiendo dos cosas: la primera conocer a fondo la aplicaciones, el código fuente, y la estructura por parte de los usuarios, y segundo, permite la contribución de los usuarios, de tal forma que las aplicaciones siempre están en continuo desarrollo.

Fundación de Software Libre:

Es la única forma legal de proteger estas aplicaciones desarrolladas, de tal forma que no sean patentadas.

Comunicación sobre IP y acceso al servidor:

Es de suma importancia contar con una dirección IP pública de tal forma que sea accesible desde una Red externa. De igual forma es recomendable contar con el servicio de DNS, facilitando así el acceso a los usuarios.

Perspectiva de negocio:

En el desarrollo de un plan de negocio es de suma importancia seguir los pasos mencionados en el capítulo 4, en este caso de posee la idea y una estructura básica de la visión del proyecto, el paso a seguir es hacer un plan de negocio teniendo en cuenta cada uno de los pasos para realizarlo, con el fin de conseguir financiación y aprobación del proyecto y procedes al montaje y puesta en funcionamiento. Es claro que el plan de negocio es una guía para el desarrollo y montaje de la empresa, y se debe ser flexible ante las variaciones constantes de la economía y el mercado.

5.3 Líneas de desarrollo de esta investigación

A partir de la investigación realizada y de los resultados obtenidos en este trabajo, se proponen las siguientes líneas de continuación de la labor efectuada:

- Continuar con la implementación del prototipo (interfaces, manejo de grandes volúmenes de datos, etc.).
- Incluir un módulo de seguridad al servidor web, de encriptación de datos de tal forma que no puedan ser descifrados por los programas capturadores o sniffers en la red y se garantice la privacidad del usuario.
- Diseño de un PCB exclusivo para la aplicación
- Realizar un plan de negocio.

Referencias

[1] M. Erazo y S. Banguero. *Plataforma para el desarrollo de sistemas embebidos basada en el gameboy advance*. UIS, 2007.

[2] W. Salamanca. y O. Carrillo. *Diseño de una plataforma para el desarrollo de sistema embebidos*. UIS, 2008

[3] T. FLOYD. *Fundamentos de sistemas digitales* 7 Ed. Madrid: Prentice Hall, 2000.

[4] Atmel. AT91RM9200, Datasheet

[5] W. Stallings. *Comunicaciones y redes de computadores*. Prentice Hall 7 Ed.

[6] A. Tanenbaum. *Redes de computadores*. Prentice Hall 3 ed.

[7] Yaghmour, K. Building embedded Linux systems [en línea limitado]. O'Reilly, 2003

[8] J. Lombardo. Embedded Linux [en línea limitado]. New Riders, 2002

[9] Berger, A. Embedded systems design : an introduction to processes, tools and techniques. [en línea] CMP Books, 2001.

[10] S. Feit. TCP/IP, Arqutectura y protocolos. Mc Graw Hill.

[11] http://www.emqbit.com/

[12] http://www.linux.org

[13] http://www.linux-es.org/

[14] http://www.debian.org/

[15] http://www.kernel.org/

[16] http://mxhaard.free.fr/

[17] http://www.lavrsen.dk/twiki/bin/view/Motion/

[18] http://laWebdelprogramador.com

[19] http://www.apache.org/

[20] http://www.linux.com/base/ldp/howto/Webcam-HOWTO/framegrabbers.html

[21] http://linux.about.com/library/cmd/blcmdl1_streamer.htm

[22] http://www.php.net/

[23] http://java.sun.com/products/jsp/

[24] http://www.**python**.org

[25] http://www.gnu.org

[26] http://www.fsf.org

ANEXOS

A. Instalación del sistema operativo y programas necesarios para la interacción con la tarjeta

A.1 Instalación del sistema operativo

Tal y como se menciono en el capítulo 1, para instalar Linux es necesaria la compilación de un Kernel o la instalación de una distribución; para el desarrollo de aplicaciones e interacción con la tarjeta, es posible utilizar Debian o Ubuntu, en este caso se utilizo la distribución de Debian. La instalación es sencilla, se deben seguir los pasos indicados en el CD, también es posible instalarlo desde Windows usando una réplica en Red cuyo proyecto surgió para dar mayor facilidad al usuario Windows. El proyecto se llama "goodbye Microsoft" [40]

A.2 Configuración de los repositorios

Los repositorios son los que permiten acceder a las actualizaciones de la distribución, así como programas y paquetes que se deseen descargar.

Se edita el archivo /etc/apt/sources.list de acuerdo a:

[40] http://goodbye-microsoft.com/

```
.deb http://security.debian.org etch updates main contrib non-free

.deb http://ftp2.fr.debian.org/debian/ etch main contrib non-free
```

A.3 Instalación de Minicom

En primera instancia, minicom es un modem en base texto emulador de terminal para sistemas operativos Unix, Linux. Fue escrito por Miquel Van Smoorenburg. Éste servirá para la comunicación con la tarjeta, para poder acceder a través del PC al terminal de la tarjeta.

En un computador con Debian o Ubuntu se instala la aplicación de minicom de acuerdo a la siguiente línea.

```
aptitude install lrzsz minicom
```

Ahora se presentará la configuración del mismo:

Primero ejecuta minicom con la siguiente instrucción minicom –s

Ahora se procede con la configuración del mismo; si usted tiene cable serial escoje el dispositivo correspondiente; en este caso ttys0. Si usa cable USB con un conversor a serial deberá escoger ttyusb0. O el correspondiente a su sistema.

```
A - Serial Device          : /dev/ttyS0
B - Lockfile Location      : /var/lock
C - Callin Program         :
D - Callout Program        :
E - Bps/Par/Bits           : 115200 8N1
F - Hardware Flow Control  : No
G - Software Flow Control  : No
```

Luego guarda la configuración como predeterminada para el minicom. Y podrá accederlo y conectarse a la consola de a tarjeta.

A.4 Instalación de u-boot

U-boot es un cargador de arranque, con él a través de Darrell, se cargará el Kernel a la tarjeta. Se descarga u-boot y el patch correspondiente para el uso con la tarjeta:

El u-boot se encuentra en la dirección http://sourceforge.net/projects/u-boot, la versión actual es la u-boot-1.1.6. Se descarga en el PC para luego ser cargado en la tarjeta a través del minicom.

```
wget    http://svn.arhuaco.org/svn/src/emqbit/ECB_AT91_V2/u-boot/u-boot-1.1.6-ecbat91.patch

tar xjf u-boot-1.1.6.tar.bz2

cd u-boot-1.1.6
```

Se procede con la aplicación del parche

```
$ cat ../u-boot-1.1.6-ecbat91.patch | patch -p1
```

Se configuran y se construyen las herramientas de acuerdo a la configuración de la tarjeta:

```
$ make ecb_at91_config

Configuring for ecb_at91 board...

$ make tools
```

Se compila,

```
$ crossmake
```

Y se obtiene el binario u-boot.bin. Éste binario debe ser cargado en la tarjeta a través del cargador principal[41].

Cuando se enciende la tarjeta aparecerá un menú como el que se presenta a continuación:

```
Darrell's loader - Thanks to the u-boot project
Version 1.0
RAM:32MB
1: Upload Darrell's loader to Dataflash
2: Upload u-boot to Dataflash
3: Upload Kernel to Dataflash
4: Start u-boot
5: Upload Filesystem image
6: Memory test
```

Teclea la opción 3 para cargar el binario el u-boot. Al teclearla aparecerá el siguiente cuadro:

```
Please transfer Linux via Xmodem
Receiving Xmodem Transfer
```

Luego teclea Ctrl+a s, para acceder a las opción Xmodem. Estando en el menú de Xmodem se selecciona el binario que se quiere cargar (u-boot.bin). Al terminar la transferencia ya se cuenta con U-boot en el ECB_AT91.

[41] El cargador principal Darrel se encuentra preinstalado en la ECB_AT91.

```
U-Boot 1.1.6 (Jun 11 2007 - 20:06:23)

DRAM:  32 MB


Atmel: Flash:  0 kB


DataFlash:AT45DB161

Nb pages:   4096

Page Size:    528

Size= 2162688 bytes

Logical address: 0xC0000000

Area 0: C0000000 to C000317F (RO) Darrell loader

Area 1: C0003180 to C001F73F (RO) U-boot

Area 2: C001F740 to C002183F      Environment

Area 3: C0021840 to C01ACFFF      Kernel

Area 4: C01AD000 to C020FFFF (RO) Filesystem

In:    serial

Out:   serial

Err:   serial

Hit any key to stop autoboot:  0
```

Listo. Tendrá u-boot funcionado en la tarjeta.

A.5 Instalación de Crosstool

Descargue el crosstool en el PC:

```
wget http://kegel.com/crosstool/crosstool-0.42.tar.gz

tar xzvf crosstool-0.42.tar.gz
```

Para la instalación de Crosstool se necesitan los paquetes adicionales Bison, Flex y M4. (Apt-get install bison flex m4).

Una vez instalados los paquetes necesarios, y descomprimido el Crosstool, se crea el script para la compilación del ARM (build.sh). se debe crear dentro de la carpeta del crosstool.

```
#!/bin/sh
set -ex
TARBALLS_DIR=$HOME/downloads
RESULT_TOP=$HOME/crosstool
GCC_LANGUAGES="c,c++"
PARALLELMFLAGS="-j2"
export GCC_LANGUAGES PARALLELMFLAGS TARBALLS_DIR  RESULT_TOP
mkdir -p $RESULT_TOP
# Crear el toolchain.  Toma un montón de tiempo y de gigabytes
eval `cat arm-softfloat.dat gcc-3.4.5-glibc-2.3.6.dat`  sh all.sh --notest
echo Done.
```

La variable $HOME, es la ruta donde fue descargado el crosstool. El script se ejecuta con bash –nombre del archivo. La instalación de éste compilador es un proceso demorado y por tanto se debe tener paciencia. Al final de la ejecución del script el toolchain estará en la ruta $HOME/crosstool. Se debe definir la ruta del crossmake de acuerdo a las siguientes líneas cuando se desee compilar usando el crosstool.

```
$ export PATH=$HOME/crosstool/gcc-3.4.5-glibc-2.3.6/arm-softfloat-linux-gnu/bin/:$PATH

$ alias crossmake='make ARCH=arm CROSS_COMPILE=arm-softfloat-linux-gnu- '
```

Recuerde que la variable $HOME es la ruta donde descargo e instaló el Crosstool.

B. Instalación del sistema operativo en la tarjeta.

B.1 Compilación del Kernel

La instalación del sistema operativo de la tarjeta se realiza por medio de una compilación de un Kernel.

En el computador, se descarga la imagen del Kernel. Actualmente se está trabajando con la versión 2.6.21.6.

```
$ wget http://ftp.kernel.org/pub/linux/kernel/v2.6/linux-2.6.21.6.tar.bz2
 $ tar xjf linux-2.6.21.6.tar.bz2
 $ cd linux-2.6.21.6
```

Se descargan los patch para la configuración del procesador y la tarjeta y se aplican respectivamente:

```
$ wget http://maxim.org.za/AT91RM9200/2.6/2.6.21-at91.patch.gz

$ wget http://svn.arhuaco.org/svn/src/emqbit/ECB_AT91_V2/linux-
kernel/2.6.21/ecb_at91_2.6.21.patch

$ zcat 2.6.21-at91.patch.gz | patch -p1

$ cat ecb_at91_2.6.21.patch | patch -p1
```

Luego se debe realizar una compilación cruzada con el toolchain. En este caso crosscompiler:

```
$ export PATH=$PATH:$HOME/crosstool/gcc-3.4.5-glibc-2.3.6/arm-softfloat-
linux-gnu/bin/

 $ alias crossmake='make ARCH=arm CROSS_COMPILE=arm-softfloat-linux-gnu- '
```

La variable $PATH permanece igual puesto que es allí donde se encuentran los archivos binarios. Por el contrario $HOME debe ser la misma especificada en el script creado para la compilación del ARM (build.sh).

Una vez direccionado el crosstool se empieza la configuración del Kernel de acuerdo a las necesidades del trabajo a realizar en la tarjeta. Se debe tener instalado libncurses5-dev en distribuciones Debian (apt-get install libncurses5-dev).

```
$ crossmake ecbat91_defconfig
```

Ahora se crea el kernel,

```
$ crossmake menuconfig
```

Se crea la imagen,

```
$ crossmake -j2 vmlinux
```

Se prepara la imagen para cargarla a la tarjeta,

```
$ arm-softfloat-linux-gnu-objcopy -O binary -R .note -R .comment -S vmlinux
linux.bin

 $ gzip -c -9 linux.bin > linux.bin.gz
```

Se crea el binario del Kernel

```
$HOME/u-boot-1.1.4.patched/tools/mkimage \

      -A arm -O linux -T kernel -C gzip  -a 0x20008000 -e 0x20008000 \

      -n "Linux Kernel Image" -d linux.bin.gz ecb_at91.img
```

Listo. Se tiene la imagen lista del Kernel para ser cargada.

En caso de haber seleccionado módulos duracte la configuración del Kernel, estos deben ser creados.

```
crossmake modules

INSTALL_MOD_PATH=/tmp/tmp-rootfs  crossmake modules_install
```

/tmp/tmp-rootfs es la carpeta donde quedaran creados los módulos; es necesario crearla, y una vez estén creados, se debe copiar el árbol en la tarjeta, es decir en: /lib/modules/uname -r

B.2 Carga de la imagen del Kernel en la tarjeta

Ahora la carga de la imagen del Kernel a la tarjeta por Xmodem. Se ingresa a minicom en el PC, se enciende la ECB_AT91 y se espera el primer menú. Se teclea la opción **3**, hacerlo rápido para que no cargue todo el sistema. Luego iniciar la transferencia por Xmoden, al teclear Ctrl+a s. Seleccionar la imagen creada (ecb_at91.img) e iniciar la carga. Finalizada la carga

espere que el sistema verifique la imagen, seguido de ésto seleccione la opción 4 en el menú principal para arrancar la tarjeta con u-boot.

B.3 Configuración de la red

Verificar las interfaces existentes con ifconfig y route en la consola. La configuración de Ethernet se presenta a continuación:

Se desactiva la interfaz Ethernet para realizar la configuración:

```
% ifdown eth0
```

Se edita el archivo /etc/network/interfaces de acuerdo a:

```
auto lo eth0
iface lo inet loopback
iface eth0 inet static
        hwaddress ether aa:bb:cc:00:22:22
        address 192.168.46.102
        netmask 255.255.255.0
        gateway 192.168.46.1
        broadcast       192.168.46.255
        dns-name servers 192.168.19.2  192.168.19.1
        dns-search      uis.edu.co
```

Especificando los datos respectivos de su conexión; como son dirección IP, máscara, puerta de enlace, entre otras. Se guardan los cambios realizados.

Finalmente se habilita de nuevo la interfaz y se tendrá configurada la red para acceso a internet. Cabe resaltar que esta configuración es para el manejo de IP fija.

```
% ifup eth0
```

Para que el servidor DNS pueda ser utilizado, se debe editar el archivo /etc/resolv.conf tal y como se presenta a continuación:

```
search uis.edu.co
nameserver 192.168.19.2
nameserver 192.168.19.1
```

Editando los datos de su servidor DNS respectivamente. Se guardan los cambios.

C. Configuración de componentes del sistema de vigilancia

C.1 Configuración de la cámara Web

C.1.1 Especificaciones fundamentales en la configuración del Kernel para funcionamiento de cámara Web en a tarjeta de desarrollo

Requerimientos para habilitar el video: en la configuración del Kernel es necesario habilitar los elementos mostrados en la Figura C.1.

Figura C.1: Requisitos para uso de video.
Fuente: autores.

Para el reconocimiento de la cámara Web es necesario habilitar los elementos que se muestran a continuación:

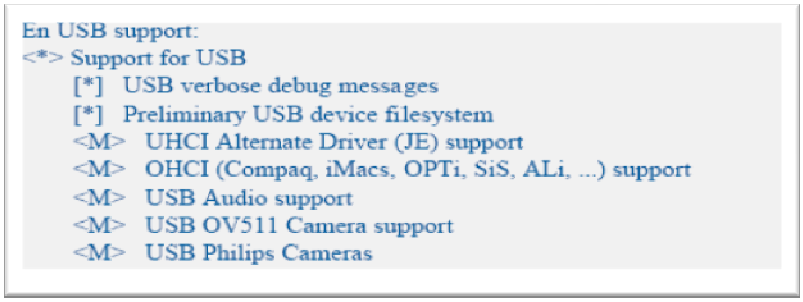

Figura C.2: Requisitos para reconocimiento de cámara Web.

Fuente: autores.

Las demás características de configuración del Kernel se dejan por defecto al aplicar el patch dado por los fabricantes de la tarjeta para el desarrollo de este prototipo. Los drivers de la cámara fueron creados como módulos en consecuencia en la compilación de la imagen del Kernel es necesario crearlos, y copiaros en el sistema e archivos respectivamente.

C.1.2 Configuración de la cámara en el PC

Conecta la cámara al PC:

```
lsusb
```

Al dar este comando aparecerá la información técnica de la cámara, se debe buscar en internet que driver sirve para esta cámara. En este caso sirve el spca5xx.

```
apt-get install linux-headers-`uname -r`

apt-get install module-assistant

apt-get install gspca5xx-source

m-a prepare

m-a a-I spca5xx-source

modprobe spca5xx
```

Listo, con modprobe se habilita el driver de la cámara Web.

C.2 Instalación y configuración del paquete de captura de imagen

La instalación del paquete motion es sencilla, se puede realizar descargando el paquete e instalándolo de acuerdo a las instrucciones dadas en el mismo, o con un simple apt-get install motion, sin embargo la versión instalada es vieja respecto a las últimas lanzadas. Este procedimiento es igual en la tarjeta y en el PC. En consecuencia se realizó una instalación manual:

```
wget http://sourceforge.net/projects/motion/motion-3.2.11.tar.gz

tar -xvvzf motion-3.2.11.tar.gz

cd motion-3.2.11

./config_build
```

```
make

make install
```

Después de haber dado la instrucción de instalación se realiza la configuración del paquete. Se debe ingresar al script de configuración ubicado generalmente en la carpeta /etc/motion.

Por ser un script de configuración es posible agregarle todas las funciones que creamos necesarias; por ejemplo: trasmitir las imágenes a un servidor externo via ftp.

A continuación se presenta el script del prototipo:

```
# Configuración de motion para un prototipo de sistema de vigilancia


daemon off
# está función permite correr el motion en modo normal o en modo demonio;
# si está en off esta desactivado el modo demonio.


quiet on
# traduce algo como quedate en silencio, no envíe sonidos mientras haya
# movimiento, solo funciona si no está en modo demonio.


videodevice /dev/video0
# define el dispositivo de video. El video0 es por defecto.


width 160
height 120
```

```
# Tamaño de la imagen; depende de la resolución de la cámara


framerate 20
minimum_frame_time 0
# framerate indica el número de fotogramas que capturará cuando detecte
# movimiento; mínimum_fame_rate indica el tiempo que espera para volver
# a detectar


quality 85
auto_brightness off
# Regula el brillo de la imagen; en este caso no se activa porque provoca una
# variación molesta del brillo en la imagen.


threshold 1500
# es el umbral para la detección de movimiento; es decir si la cantidad de
# pixeles diferentes entre dos imágenes supera este umbral, se empieza la
# captura


noise_level 64
# nivel de ruido que acepta en la variación de imagen


brightness 0
contrast 0
saturation 0
hue 0
```

```
# Brillo, contraste, hue (NTSC) y saturación inicial. Rango entre 0 y 255
# si el valor es cero esta deshabilitado.

Webcam_quality 50
# calidad de la imagen en porcentaje

Webcam_maxrate 20
# Limita el número de frames por segundo de la cámara

minimum_motion_frames 2
pre_capture 3
post_capture 1
# fotogramas mínimos para disparar, pre y post capture indican el número
# de fotogramas que serán guardados antes y después que se detecto movimiento

output_normal on
# para guardar las imágenes; las opciones pueden ser:
# on: al detectar movimiento se guardan las imágenes
# off: al detectar movimiento no se guardan imágenes
# first: al detectar movimiento guarda solo la primera imagen
# best: al detectar movimiento guarda la imagen que más movimiento muestre

#snapshot_interval 0
# Ademas de capturar imágenes al detectar movimiento, Motion puede realizar
# capturas periódicas independientemente de que haya ó no movimiento.
```

```
# Indica aquí el intervalo en segundos entre capturas.
#snapshot_filename %d
# Nombre que recibirá el archivo generado por “snapshot_interval“
# Están desactivados porque no es de interés para el sistema.

jpeg_filename %d
# indica el nombre con el que serán guardadas las imágenes. En este caso
# se está sobre escribiendo la imagen ya que no interesa que quede guardada

Webcam_port 8081
# Puerto desde el cual se accede a las imágenes tomadas por la Webcam.

control_port 8080
# Puerto desde el cual se accede al script de configuración por red.

control_localhost off
# La opción por defecto (on) impide el acceso al servidor de control de
# Motion desde cualquier otro ordenador que no sea localhost.
# Si lo coloca en OFF se puede acceder desde cualquier ordenador.

control_authentication e3t:e3t
# permite asignarle autenticación de usuario al script de configuración en
# red.

Webcam_localhost off
```

```
# permite acceder al puerto de captura de Webcam desde otros ordenadores.
# si está en on sólo podrá accederse desde el mismo ordenador.

target_dir /var/www/Webcam
#target_dir /home/cam/
# Indica la ubicación donde serán guardadas las imágenes capturadas en este
# caso se direccionó a la carpeta del servidor Apache para que puedan ser
# compartidas.
```

C.3 Instalación y configuración del servidor Web con compatibilidad HTML, PHP.

La instalación se realiza de igual forma en el PC y en la tarjeta de desarrollo.

C.3.1 Instalación y configuración del servidor Web

La instalación de apache en la tarjeta de desarrollo se puede realizar de dos formas, la primera es la más sencilla solo con la instrucción " apt-get install apache2" , la segunda es la instalación manual; en ambos casos es necesario realizar la configuración del servidor. A continuación se presenta la instalación manual.

```
wget http://www.uniontransit.com/apache/httpd/httpd-2.2.10.tar.gz
tar xvvzf httpd-2.2.10.tar.gz
cd httpd-2.2.10
```

```
./configure --prefix=/etc/apache2

make

make install
```

Después de la instalación, se accede a los scripts de configuración (config.d[42] , sites-avalible[43], apache2.conf o httpd.conf[44]) ubicados en la carpeta /etc/apache2.

Para la comprobación del funcionamiento del servidor apache, se ejecuta la siguiente instrucción para activar apache:

```
/etc/init.d/apache2 start
```

Desde un equipo conectado en la red se busca en el navegador Web la dirección IP de la tarjeta(http://192.168.46.102/index), donde aparecerá la pagina por defecto de apache o de Emqbit, ubicado en la carpeta la carpeta del servidor(/var/www/).

C.3.2 Configuración e instalación de PHP con compactibilidad con Apache.

La instalación de PHP se realiza manualmente debido al requerimiento de modificaciones en su configuración.

Primero se instala las librerías de algunas aplicaciones que se requieren:

[42] Script donde se realiza la configuración de aplicaciones Web cuando son instaladas con la instrucción apt-get install
[43] Directorio donde se encuentra el script para realizar la configuración del virtualhost
[44] Script que contiene la mayor parte de la configuración de apache, la más importante en este caso los módulos.

```
apt-get install libxml2-dev curl libcurl3-dev libjpeg62-dev flex zlib1g-dev libpng12-dev

apt-get libapache2-mod-php5
```

Segundo se detiene el servidor:

```
/etc/init.d/apache2 stop
```

Tercero se realiza la instalación de PHP, descargando el paquete, descomprimiendo, ejecutando el script de configuración, compilando e instalando.

```
wget http://es.php.net/get/php-5.2.8.tar.gz/from/a/mirror

tar xvvzf php-5.2.8.tar.gz

cd php-5.2.8

./configure --prefix=/usr/local/php5 --enable-mbstring --with-apxs2=/etc/apache2/bin/apxs --with-curl=/usr/include/curl --with-jpeg-dir=/usr/include --with-zlib-dir=/usr/include --with-gd --with-xml --enable-ftp --enable-bcmath

make

make install
```

Cuarto se cambia de extension el script php.init-dist:

```
cp php.ini-dist /etc/local/lib/php.ini
```

Quinto se modifica el script "apache2.conf" en la carpeta de apache (/etc/apache2) con las siguientes instrucciones que deben ser colocadas respectivamente donde se encuentra el módulo <IfModule mime_module>

```
<IfModule mime_module>

AddType application/x-httpd-php.php.phtml

AddType application/x-httpd-php-source.phps
```

Sexto se modifica la linea DirectoryIndex en el script apache2.conf.

```
DirectoryIndex index.php index.html
```

Por último se activa el servidor Web.

```
/etc/init.d/apache2 start
```

Listo! Se tendrá funcionado Apache y PHP.

C.4 Página Web

El sitio Web se programa en lenguaje HTML. El desarrollo principalmente costa de una página Web inicial, con la función de realiza la autenticación del usuario enlazada hacia la página Web principal, despues de pasar por 2 páginas que no vera el usuario para realizar el control de nombre , contraseña y la seguridad de la página Web principal.

Estas páginas son almacenadas en la carpeta del servidor Web (/var/www/) para poder ser visualizadas desde la Red.

La página inicial se ve continuación :

```
<HTML>

<HEAD>

<TITLE>pagina inicial</TITLE>

</HEAD>

<BODY>

<IMG SRC="image001.png"   alt="titulo" HEIGHT = "125" aling="center"  >

<br>

<font color="#FF0000" face="Comic Sans MS" SIZE="5" ><p align="center" ><i>
<b>

<form action="control1.php" method="POST">

<?php

if($_GET["error"]=="si"){

ECHO "VOLVER A INTENTARLO" ;

}else{

}

?>

</b></i></p></font>

<font color="#0000FF" face="Comic Sans MS" SIZE="4" ><p align="center" ><i>
<b>

Nombre de usuario: <input type="text" name="nombre" maxlength="50"/>

<br>
```

```
Contrase&ntilde;a: <input  name="clave" maxlength="50" type="password" />

<br>

<input type="submit" value="INGRESAR">

<br>

<br>

</form>

</b></i></p></font><br><br><br>



<img SRC="image003.gif" border="1"  HEIGHT = "100" aling="left" hspace="150"
>

<IMG SRC="image006.jpg" border="2"  HEIGHT = "100"   aling="right"  >

</BODY>

</HTML>
```

La página de control de la autenticación de usuario:

```
<?php

ob_start();

?>

<HTML>

<BODY><?php
```

```
if (($_POST['nombre'] ==e3t)and ($_POST['clave'] ==e3t) ) {

session_start();

$_SESSION['AUTENTIFICACION']="SI";

header ("Location: prueba1.php");

}else{

$_SESSION['AUTENTIFICACION']="NO";

header("Location: index.php?error=si");

}?></BODY></HTML>
```

La página de seguridad ,para la protección de la página Web principal:

```
<HTML> <HEAD> <TITLE>seguridad</TITLE>

</HEAD>

<BODY>

<?php

 session_start();

 if ($_SESSION['AUTENTIFICACION']== "NO") {

 header("Location: index.php");

 exit();

} else {

  $_SESSION['AUTENTIFICACION']= "NO";
```

```
 }?></BODY> </HTML>
```

La página principal se divide en 5 frames, como se ve continuación:

```
<HTML> <HEAD> <TITLE>seguridad</TITLE>
<?php
ob_start();
include "seguridad.php";
$_SESSION['AUTENTIFICATION']="NO"
?>
<html><head>
<title>Aplicación segura prueba1.php</title></head>
<FRAMESET  rows="25%,43%,*" frameborder="0" framespacing="0" scrolling="no">
<frame src="titulo.html" NAME="TITULO">
<frameset cols="33%,*,34%" frameborder="0" framespacing="0" >
<frame src="ladoderecho1.html" name="derecho">
<frame src="http://192.168.46.102:8081" name="central" >
<frame src="ladoizquierdo.html" name="final">
</frameset>
<frame src="down1.html" name="abajo">
```

```
</frameset>

</html>

<?php

ob_end_flush();

?>
```

El primer frame lleva el nombre del servicio de Streaming, en este caso "WEBCAM LIFE ONLINE".

Frame superior:

```
<HTML><HEAD><TITLE>titulo</TITLE></HEAD>

<BODY bgcolor="#000000" topmargin=0 leftmargin=0 marginheight=0
marginwidth=0> <br> <br>

<IMG SRC="image001.png"  alt="titulo" HEIGHT = "125" aling="center"  >

</BODY></HTML>
```

El segundo un frame presenta un enlace para realizar modificaciones en la configuración del programa de Streaming.

Frame derecho:

```
<HTML><HEAD> <TITLE>webcam life online</TITLE></HEAD>

<BODY  bgcolor="#000000"  link="#ffff33"  alink="#ffffcc"  alink="ffff00"
topmargin=0 leftmargin=0 marginheight=0 marginwidth=0> <br> <br>

<font color="#FFFFFF" face="Comic Sans MS" size="4" ><p align="center" ><i>
```

```
 <b>Si quieres configurar el video dar click <a
href="http://192.168.46.102:8080" target="_blank">aqui </a> </b>
</i></p></font>

</BODY></HTML>
```

El tercer frame contiene el hipervínculo al puerto del video, el cual permitirá la visualización del Streaming.

El cuarto frame contiene el hipervínculo para comunicarse con los autores del proyecto en caso de tener solicitudes, inconvenientes o sugerencias.

Frame izquierdo

```
<HTML><HEAD> <TITLE>webcam life online</TITLE></HEAD>

<BODY bgcolor="#000000" link="###8080FF" alink="#ffffcc" alink="ffff00"
topmargin=0 leftmargin=0 marginheight=0 marginwidth=0>

 <font color="#ffffff" face="Comic Sans MS" SIZE="4" ><p align="center" ><i>

      <b>Inconvenientes,dudas sugerencias comunicarse con:

      <a href="mailto:johanitaj2@hotmail.com">Johanna Arias</a>

      <a href="mailto:marcela_gama@hotmail.com">Marcela Garc&iacute;a</a>
</b></i></p></font>

  </BODY></HTML>
```

El quinto frame contiene las imágenes del proyecto GNU, Linux y Apache.

```
<HTML><HEAD> <TITLE>webcam life online</TITLE></HEAD>
```

```
<BODY   bgcolor="#000000"   link="###8080FF"   alink="#ffffcc"   alink="ffff00"
topmargin=0 leftmargin=0 marginheight=0 marginwidth=0>

<img SRC="image003.gif" border="1"  HEIGHT = "100" aling="left" hspace="100"
>

<IMG SRC="image006.jpg" border="2"  HEIGHT = "100"   aling="right"  >

<IMG SRC="image007.jpg" border="2"  HEIGHT = "100"   aling="right"  >

 </BODY></HTML>
```