PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN, LOCALIZACIÓN Y ALERTA DE AVERÍAS EN REDES DE FIBRA ÓPTICA DE PLANTA EXTERNA METROPOLITANA BASADO EN INFORMACIÓN GEOREFERENCIADA

Tesis para optar el Título de Ingeniero de Telecomunicaciones, que presenta el bachiller:

Víctor Manuel Chang Cabanillas

ASESOR: Ing. Arturo Díaz Rosemberg

Lima, junio del 2014

RESUMEN

En la presente tesis se presenta el diseño e implementación de un sistema que contribuya a agilizar los tiempos de atención de averías en enlaces de fibra óptica de una red planta externa metropolitana. Con este fin, se plantea un sistema centralizado de detección, localización y alerta de averías, el cual involucra el uso de tecnologías flexibles para visualizar y gestionar a los elementos de la red, los clientes y el despliegue de los cables de fibra óptica, así como el uso de información georeferenciada de la red para facilitar el monitoreo de la misma.

Esta tesis se estructura de la siguiente forma:

En el capítulo 1 se detalla el escenario actual del monitoreo de una red de planta externa metropolitana, se determina la problemática y los objetivos del proyecto.

En el capítulo 2 se desarrolla el marco teórico de las metodologías de detección y localización de averías en redes de fibra óptica. Se analizan las principales soluciones de monitoreo utilizadas actualmente y luego se elige la que aborda mejor la problemática y los requerimientos clave, para después plantear el modelo teórico del sistema.

En el capítulo 3 se presenta el diseño e implementación del sistema en cuestión, a través de diagramas de clase, la arquitectura del sistema y el detalle de cada funcionalidad implementada.

Finalmente, en el capítulo 4 se muestran las interfaces del prototipo implementado y las pruebas realizadas a cada funcionalidad, Así como el detalle de los resultados obtenidos.

Dedicatoria

A mis papás, Víctor y Berenice, y a mi hermano Sergio por su apoyo incondicional en todo momento

A mi novia Vanneza, por su constante apoyo, confianza y por aconsejarme, escucharme y preocuparse por mí siempre

Agradecimientos

A todos los profesores y compañeros de estudios de la especialidad de Ingeniería de las Telecomunicaciones que contribuyeron en mi crecimiento académico y como persona.

A mi asesor el Ing. Arturo Díaz Rosemberg, por apoyarme y confiar en mí para la realización de este proyecto.

A todos las personas que de alguna forma me apoyaron en la finalización de este trabajo.

INDICE

LISTA DE FIGURAS

GLOSARIO

OTDR	OPTICAL TIME DOMAIN REFLECTOMETRY
OTDR	OPTICAL TIME DOMAIN REFLECTOMETER
FTTH	FIBER-TO-THE-HOME
i-FTTH	INTELLIGENT FIBER-TO-THE-HOME
PON	PASSIVE OPTICAL NETWORK
SWITCHING	COMMUTACIÓN
BACKUP	RESPALDO
CFDS	CENTRALIZED FAILURE DETECTION SYSTEM
ACS	ACCESS CONTROL SYSTEM
CAPU	CUSTOMER ACCESS PROCTECTION UNIT
MADS	MULTI ACCESS DETECTION SYSTEM
ATTENUATION BUDGET	PRESUPUESTO DE ATENUACIÓN / ATENUACIÓN TOTAL
SPLITTING	DIVISIÓN
CO	CENTRAL OFFICE
OLT	OPTICAL LINE TERMINAL
NMC	NETWORK MANAGEMENT CENTRE
FTTP	FIBER-TO-THE-PREMISES
ONU	OPTICAL NETWORK UNIT
M-CYCLES	MONITORING CYCLES
M-PATHS	MONITORING PATHS
M-TRAILS	MONITORING TRAILS
FLOODING	INUNDACIÓN DE UN TIPO DE INFORMACIÓN
UFL	UNAMBIGUOUS FAULT LOCALIZATION
LVM	LIMITED PERIMETER VECTOR MATCHING
PES	POTENTIAL EXECUTIVE SINK
ALV	AFFECTED LINK VECTOR
MVC	MODEL – VIEW – CONTROLLER
HTML	HYPERTEXT MARKUP LANGUAGE
DAL	DATABASE ABSTRACTION LAYER
SQL	STRUCTURED QUERY LANGUAGE
DOM	DOCUMENT OBJECT MODEL
POLLING	CONSULTA CONSTANTE
JS	JAVASCRIPT
BOILERPLATE CODE	CÓDIGO QUE SE REPITE EN MULTIPLES PARTES DE UNA APLICACIÓN
SCOPE	ÁMBITO O ENTORNO DE EJECUCCIÓN
CHILD SCOPE	OBJETO SCOPE QUE HEREDA LAS PROPIEDADES DEL SCOPE PADRE
DEPENDENCY INJECTION	INYECCIÓN DE DEPENDENCIAS
URI	UNIFORM RESOURCE IDENTIFIER
CSS	CASCADING STYLE SHEETS
API	APPLICATION PROGRAMMING INTERFACE
PROXY	INTERMEDIARIO
THREADS	HILOS DE PROCESAMIENTO

CACHING	ALMACENAMIENTO TEMPORAL
SNMP	SIMPLE NETWORK MESSAGE PROTOCOL
LOGIN	INICIO DE SESIÓN

INTRODUCCIÓN

En la actualidad, el acceso a información y el estar constantemente comunicados se ha vuelto una prioridad en muchas personas, en consecuencia el ancho de banda utilizado por una persona se ha incrementado considerablemente en los últimos años. Por ello, las telecomunicaciones han ido evolucionando, siguiendo esta tendencia, con la finalidad de crear tecnologías que brinden un mayor ancho de banda y una mejor calidad de servicio para cubrir esta demanda.

Como resultado de esta evolución tenemos a la fibra óptica, una tecnología cuya utilización está aumentando en la actualidad debido a las grandes capacidades de información que puede transportar lo que permite que múltiples aplicaciones puedan utilizarse en infraestructuras más pequeñas. Sin embargo, la fibra óptica tiene una desventaja significativa, su fragilidad; por ende la supervisión y monitoreo de la fibra óptica es crucial para la supervivencia de cualquier red basada en esta tecnología y es incluso más crítico aún si la red se encuentra en un terreno tan agresivo e impredecible como lo es la planta externa en general. Y ante esta necesidad surge la técnica reflectometría óptica en el dominio del tiempo (OTDR) que permite censar los enlaces y determinar si presentan alguna avería a lo largo de su extensión, pero, es una tecnología costosa y por ello no es rentable a gran escala como para una red de planta externa metropolitana.

El presente trabajo tiene como objetivo diseñar un sistema de detección, localización y alerta de averías basado en información georeferenciada e implementar una prueba de concepto que permita monitorear una red de planta externa metropolitana sin la necesidad de utilizar la técnica de OTDR.

CAPITULO I
PROBLEMATICA

El presente capítulo se realizará un análisis a la situación actual de una red de planta externa metropolitana y se planteará la problemática que enfrenta a la hora de afrontar la aparición de averías y en la metodología actual de resolución de las mismas.

1.1 Análisis de una red de fibra óptica de planta externa metropolitana

1.1.1 Red de Planta Externa Metropolitana

Una red de planta externa es aquella que comprende todo el conjunto de construcciones, cables, instalaciones, equipos y dispositivos que se ubican fuera de los edificios hasta el terminal de distribución. Será aérea, cuando los elementos que conforman la planta externa están fijados en postes o estructuras, y será subterránea, cuando los elementos que la conforman se instalan en canalizaciones, cámaras, ductos y conductos [19]. Y será considerada como metropolitana cuando este conjunto de elementos se encuentre dentro de la extensión de una metrópoli o ciudad.

Debido a su amplia extensión, actualmente el funcionamiento de las redes de fibra óptica de planta externa se ve afectado en gran medida por diferentes problemas causados por entes ajenos a la red, ya sea por vandalismo, obras de construcción, instalaciones ilegales, etc. y el Perú es una prueba real de este escenario. En consecuencia existe una presencia constante de averías en

diferentes partes de la red, las cuales toman un lapso muy grande en ser solucionadas. Esto se da principalmente porque no se sabe dónde se encuentran las averías ni por donde comenzar a buscar.

1.1.2 Elementos de la red

A modo de delimitar que es lo que compone una red de fibra óptica de planta externa metropolitana, a continuación se detalla los elementos que se pueden encontrar generalmente en dicha red:

1.1.2.1 Cables de fibra óptica:

La fibra óptica es una fibra flexible que consta de núcleo y revestimiento, ópticamente transparente, generalmente de vidrio o de plástico, través de la cual la luz puede ser transmitida por sucesivas reflexiones internas [13]. A nivel mundial existen dos tipos de fibra óptica:

Monomodo: Tipo de fibra óptica que posee un único modo de propagación debido a que las dimensiones de su núcleo son muy pequeñas, típicamente 8-12µm y utilizan láseres como fuente de luz. [17]

Multimodo: Tipo de fibra óptica que posee múltiples modos de propagación debido a que las dimensiones de su núcleo son más grandes en comparación a la monomodo, típicamente 50µm y utilizan leds como fuente de luz. [17]

En consecuencia, un cable de fibra óptica es aquel que contiene una o más fibras ópticas, dotándolas de las protecciones necesarias para el ambiente donde estará instalado [20]. Cada fibra está protegida por recubrimientos fabricados con polímeros:

- El recubrimiento primario que siempre está colocado directamente sobre el revestimiento de la fibra óptica.
- El recubrimiento secundario puede estar aplicado directamente sobre el primario o pueden ser mecánicamente independientes. [20]

De acuerdo al tipo de recubrimiento secundario se puede clasificar en:

- *Estructura apretada (tight buffer):* Cuando el recubrimiento secundario se encuentra adherido directamente sobre el primario o puede ser independiente a nivel mecánico. (Ver Figura 1.1)

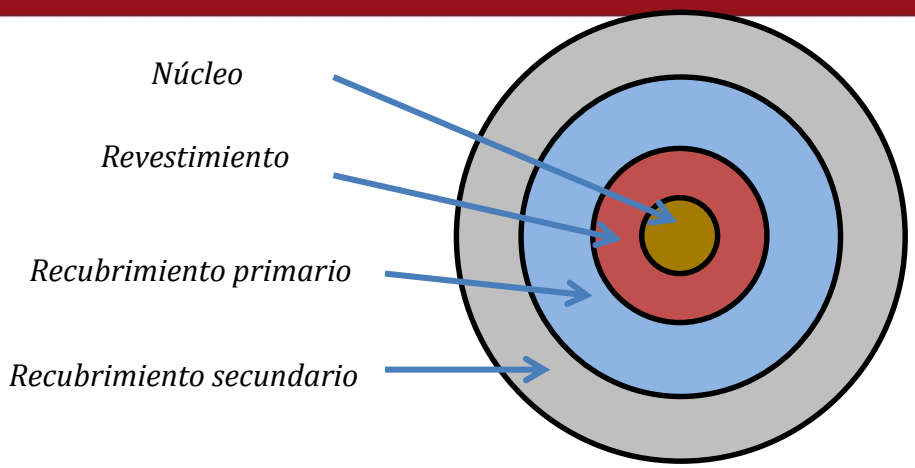


Figura 1.1. Recubrimiento de estructura apretada [20]

- *Estructura holgada (loose buffer):* Cuando el recubrimiento secundario no está adherido al primario, es decir, están desvinculados de manera mecánica. [20] (Ver Figura 1.2)

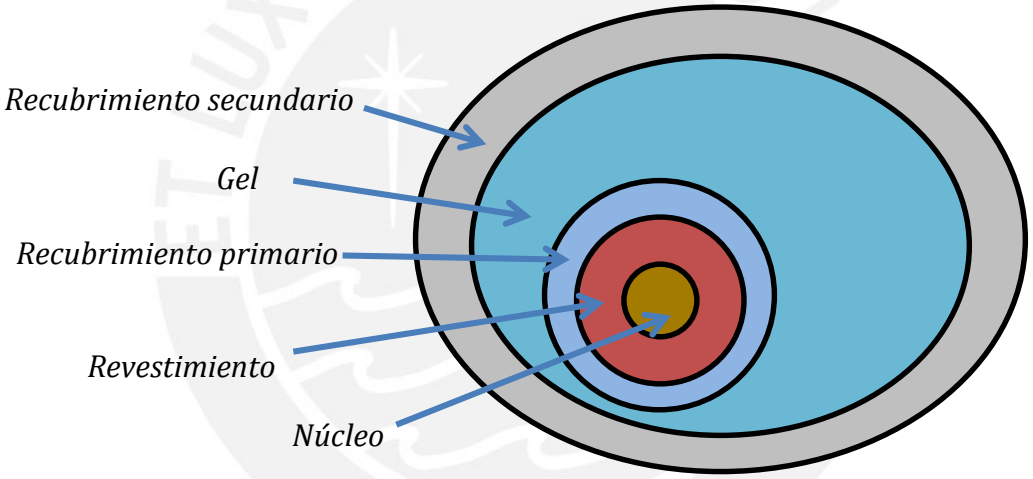


Figura 1.2. Recubrimiento de estructura holgada [20]

Luego, los elementos de tensión son aquellos que permiten soportar la tensión ejercida hacia el cable durante su instalación y/o manipulación. Son de dos tipos:

- Centrales: alambres metálicos, barras de fibra de vidrio, hilos de polímeros (nylon, kevlar).
- Envolventes: hilos de aramida. [20]

Finalmente, tal y como observamos en la Figura 1.2, el cable posee una cubierta exterior que proporciona resistencia mecánica al conjunto y protege de daños a las fibras de su interior [20]. El factor determinante del tipo de cubierta a utilizar es el medio en el que va a estar instalado el cable:

- Ligero: para instalaciones dentro de edificios, cuartos de comunicaciones, etc. (planta interna)

- Robusto: para instalaciones en buzones, postes, etc. (planta externa) [20]

1.1.2.2 Cámaras

Es la construcción a ejecutarse en el subsuelo, que albergará los empalmes, dispositivos o elementos de conexión de la red de telecomunicaciones, permitiendo además el cambio de dirección y distribución de los cables. [19] (Ver Figura 1.3)

1.1.2.3 Canalización

Es la red de ductos que sirven para enlazar: dos cámaras entre sí, una cámara y un armario, una cámara y una caja de distribución, etc. [19] (Ver Figura 1.3)

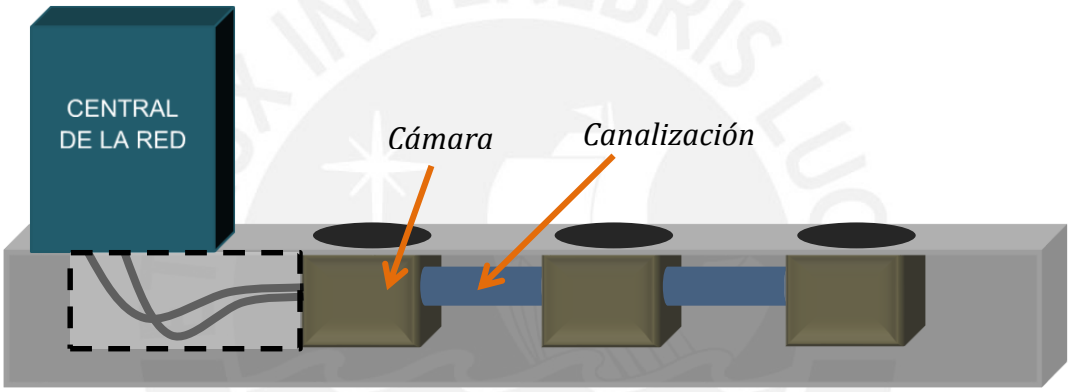


Figura 1.3. Dibujo de una cámara y canalizaciones en la planta externa [Elaboracion propia]

1.1.2.4 Cajas de empalme

Las cajas de empalme contienen de una a docenas de bandejas de empalme y proporcionan la entrada para múltiples cables fibra óptica. Todas estas se sellan por completo para la protección ambiental de los cables y empalmes, pero la mayoría pueden ser re-introducidos para pruebas y solución de problemas o re-empalme. Las cajas de empalme vienen en docenas de configuraciones, por ejemplo con entradas de cable en un extremo o ambos, para múltiples cables, o cajas de tipo vertical, horizontal, enterrado, de montaje en poste o mensajero, etc. De acuerdo con la aplicación dada, se debe elegir una caja que vaya acorde con las condiciones y necesidades de la instalación. [23]

1.1.2.5 Postes

Es una estructura considerada como una unidad principal de soporte [24] en la planta externa, por lo general puede estar conformado por concreto, madera o acero según su aplicación y sirve para el montaje de diferentes equipos ya sean o no pertenecientes a la red de telecomunicaciones.

1.1.2.6 Cajas de distribución:

Aloja los equipos y dispositivos de distribución de fibra óptica, proveyendo la seguridad y el espacio necesario para efectuar las conexiones de los enlaces [19]. En la Figura 1.4 podemos ver un ejemplo más claro del elemento mencionado.

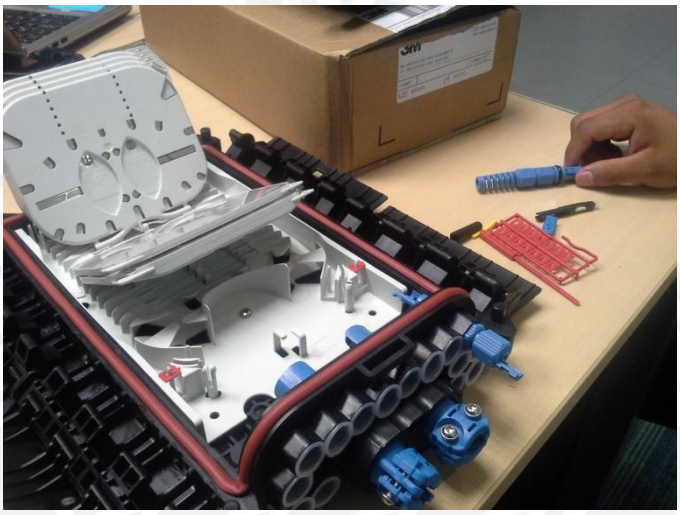


Figura 1.4. Fotografía de una caja de distribución fibra óptica de la marca 3M que cuenta con bandejas de empalme y ranuras para splitters [Elaboración propia]

1.2 Problemática del monitoreo

Actualmente la detección de averías toma demasiado tiempo, aproximadamente entre 4 a 8 horas siguiendo un procedimiento estándar que consiste en una serie de pasos que comienzan con la detección de la avería en la central de monitoreo, este primer paso puede tomar entre 5 a 30 minutos dependiendo de a qué hora se presenta la avería; una vez identificado el enlace averiado se procede a enviar al equipo técnico a buscar la posición de la avería, por lo general la localización se realiza mediante el uso de la reflectometría óptica en el dominio del tiempo (*OTDR*).

Este método consiste en conectar un equipo OTDR al enlace de fibra óptica que queremos analizar, este enviará un pulso de luz a lo largo de la fibra y de encontrarse una falla, la reflexión de este pulso ocasionada por la falla será capturada por el equipo y servirá para determinar la distancia a la que encuentra en base al tiempo que demora en viajar el pulso reflejado y la velocidad a la que se lanzó dicho pulso. Entonces, supongamos que el enlace en cuestión tiene una extensión de 10km; con el OTDR obtenemos como resultado que a 2km desde el punto de medición se encuentra la avería. Sin embargo, la información obtenida sigue siendo muy general puesto que no considera el perfil del enlace ni los elementos de la red involucrados en el mismo, lo que supone una demora aproximada de 3 horas entre ir físicamente al sitio y encontrar el problema, ya que uno debe aproximar donde se cumple la distancia de 2km en la planta externa y cuáles de los elementos dentro de esa distancia son los involucrados.

Pues bien, se logró encontrar la falla pero esta no puede ser solucionada con los recursos y conocimientos del equipo técnico, por ello deben escalar al especialista quien es el que tiene el conocimiento necesario para la adecuada solución de la falla. En suma, le agregamos unas 3 horas más entre lo que se contacta al especialista y le toma llegar al sitio, en el peor de los casos, asumiendo que por lo general solo se asigna 1 o 2 y no se encuentran la mayor parte de su tiempo en un solo lugar. Finalmente, toma entre 30 minutos a 3 horas más en solucionarse la falla, dependiendo de la gravedad de la misma, y realizar las pruebas necesarias para que por fin se restablezca el servicio del cliente. Un ejemplo más real de este escenario se puede observar en el Anexo 1.

Como un adicional, el uso del método de OTDR se debe realizar de manera independiente para cada enlace de fibra óptica que exista en la red, es decir, se necesita un *input* de OTDR por cada enlace que se desee censar si se quiere monitorear múltiples enlaces a la vez, lo que se traduce en costos elevados de implementación.

En síntesis, es que a medida que la red se va expandiendo el uso del OTDR se vuelve muy costoso y puede tomar más tiempo del que podría tomar solucionar la avería si se mantiene el procedimiento actual. Por ello, se planteará la implementación de una alternativa que permita detectar y localizar las averías de

manera centralizada, que brinde una localización más precisa, basado en el análisis de estado de los enlaces y la información georeferenciada de los elementos que conforman la red.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar un sistema de detección, localización y alerta de averías en una red de fibra óptica de planta externa metropolitana basado en información georeferenciada, que permita una rápida atención de las averías.

1.3.2 Objetivos específicos

- Diseñar e implementar un aplicativo web en un framework basado en JAVA, que permita la detección y localización de averías en la red y así como el envío de alarmas al personal pertinente.
- Crear una base de datos georeferenciada que almacene toda la información de los elementos que pertenecen a la red de planta externa (cámaras, cables, postes, etc.)
- Crear un registro que almacene toda la información de estado de los enlaces de la red, así como la información de las alarmas, los enlaces a los que pertenecen dichas alarmas y la fecha y hora en la que se enviaron.

1.4 Justificación

La principal razón por la que se realiza este proyecto es debido a la gran necesidad de un monitoreo constante en los enlaces de fibra óptica de la red de planta externa metropolitana, esto con la finalidad de mantener una alta disponibilidad y continuidad del servicio brindado al cliente, lo que en consecuencia se considera como una forma de brindar calidad al servicio.

1.5 Alcance

Este proyecto tiene como ideal agilizar los tiempos que toma solucionar una avería contribuyendo en la detección, localización y alerta temprana en una red de planta externa metropolitana. Sin embargo, todo sistema posee un alcance determinado, por lo que se han delimitado las funcionalidades con las que va a funcionar el sistema en cuestión:

- El diseño e implementación del sistema abarca la detección, localización y alerta de las averías en la red. Enfocándose principalmente en el posicionamiento de la avería, el envío y visualización de las alertas a través de una interfaz web.

- Para determinar el tipo de avería presente es necesario contar con un equipamiento especial que pueda obtener esta información, es por ello que en la alerta solo se presentará la avería y su localización, mas no la clase de la misma.

- Para el caso especial de una falla energética, también se necesitaría equipamiento especial en el borde del cliente. Por ello, en caso se presentara una falla de este tipo, esta aparecerá como una falla total de un segmento de la red, de modo que quedará en manos del personal asignado para el monitoreo en determinar si ese es el caso en cuestión.

CAPITULO II
MARCO TEÓRICO

2.1 Estado del Arte

El monitoreo de averías en fibra óptica siempre ha sido un punto crucial en el avance de esta tecnología. Debido a su complejidad y a los años que se viene usando esta tecnología, la localización de fallas ha sido un problema principalmente por que los equipos diseñados con este fin son costosos y cuando la red de fibra óptica se expande, este costo se eleva excesivamente. Sin embargo, se han buscado soluciones alternativas que están basadas en la lógica de funcionamiento de los métodos convencionales como el OTDR. Por eso la metodología para abordar la detección y localización de averías se divide dos grandes ramas: Metodología basada en OTDR y la metodología basada en el análisis del estado del enlace:

2.1.1 Metodología basada OTDR

Esta metodología fue la primera en brindar la información de enlaces punto a punto con gran detalle y exactitud por lo cual es la metodología de mayor confiablidad y efectividad en el mundo.

Para redes del tipo FTTH se ha creado el concepto de i-FTTH que consiste en monitorear una red PON de ocho ramas terminales a nivel de capa física, es decir mediante el censado físico de los enlaces i-FTTH consiste en el esquema que observamos en la Figura 2.1 [1]

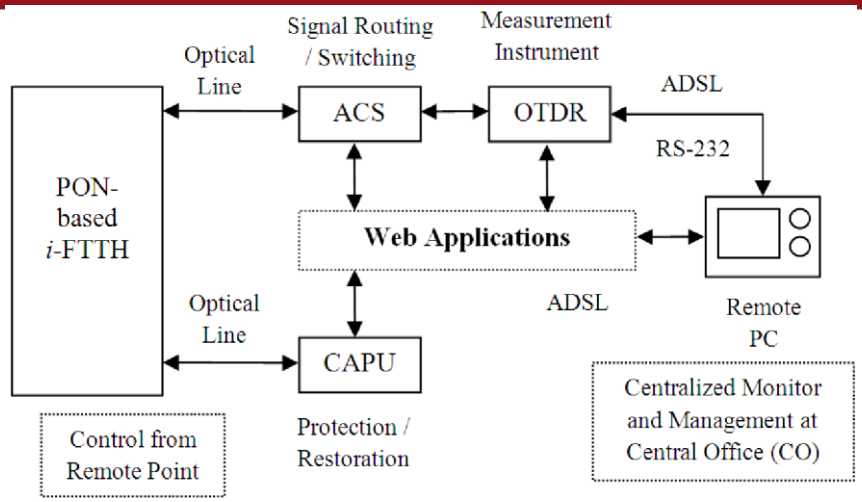


Figura 2.1. Esquema de interacción del PON-based i-FTTH [1]

- CFDS: basado en el uso de OTDR integrado con equipos ópticos adicionales de bajo costo en medio del sistema de la red para monitorear la planta externa de fibra óptica en un rango lo suficientemente amplio para cubrir la red PON.
- ACS: encargado de manejar la detección de la línea central para poder desviar el pulso del OTDR de la oficina central para omitir el filtro y conectarse a cada enlace de fibra correspondiente en la sección de caída.
- CAPU: enfocado en el *backup* de la línea del usuario final, es decir se encarga de hacer el *switching* del enlace fallido al enlace *backup* del usuario.
- MADS: enfocado en la detección del acceso múltiple a la red para sectorizar las secciones de fallas.[1]

Principalmente se abordan dos tipos de averías, las causadas por corte de fibra óptica y las causadas por falla de equipo. Para las que define que en caso de una corte de fibra las pérdidas de tráfico de datos serían bastante altas y afectarán a todos los enlaces relacionados, mientras que una falla de equipo se considera que básicamente exista una falla en el ACS por ser el elemento más importante y complicado de la red.

Entonces la detección de la falla es realizada por el CFDS que mediante OTDR permite determinar su posición en la red, el ACS es aquel que se encarga de la alerta y habilita el CAPU que es el encargado de iniciar el enlace *backup* del cliente mientras se soluciona la avería.[1]

Por otra parte, de acuerdo con [10] se definen soluciones de monitoreo para redes del tipo FTTH para los períodos de implementación y operación de la red. Para el caso de implementación, lo importante es monitorear las atenuaciones en las fibras para evitar un *attenuation budget* alto ya que esto conlleva a altas pérdidas de datos en la red, según este estudio a mayor tasa de *splitting* más corta es la distancia máxima posible entre la oficina central y el cliente ya que el *attenuation budget* es la suma de la atenuación en la fibra y las pérdidas por inserción del *splitter*. Desafortunadamente las fibras de distribución individuales pueden ser difícilmente evaluadas por una simple medida de OTDR debido a que al estar detrás de una tasa de *splitting* de 1:8 da una atenuación de 0.6dB. Por ello es necesario caracterizar la infraestructura con mediciones de extremo a extremo con una fuente de luz y medidores de potencia a diferentes longitudes de onda como podemos visualizar en la Figura 2.2.

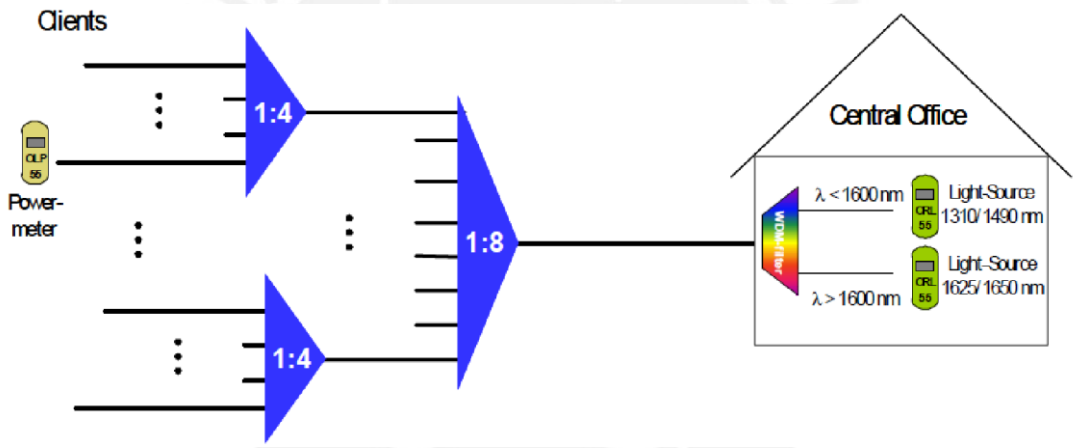


Figura 2.2. Esquema de medición de extremo a extremo [10]

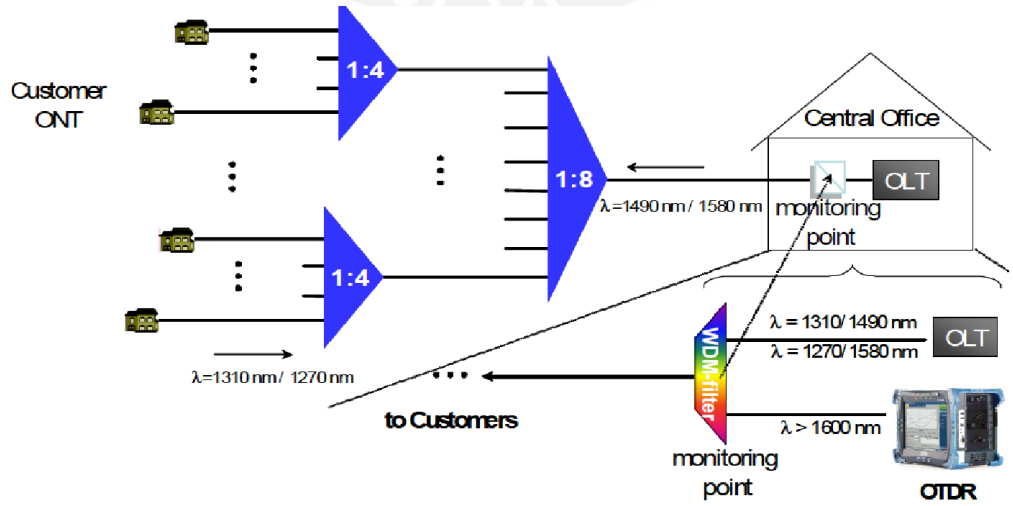


Figura 2.3. Esquema de monitoreo en la oficina central (CO) [10]

Para el caso de la red en operación, definen que solo es necesario poner un punto de monitoreo dentro de la oficina central (CO) puesto que las mediciones deben realizarse sin afectar el tráfico de datos. Por eso tener el punto de monitoreo dentro de la CO es bastante útil, ya que consiste en poner un OTDR conectado al filtro de borde del terminal de línea óptico (*OLT*) como podemos observar en la Figura 2.3Figura 2.3 y puesto que la longitud de onda reflejada de los elementos activos del PON está en el rango de 1600nm en adelante, basta con configurar el OTDR en ese rango de longitud de onda para poder monitorear los enlaces sin influenciar en el tráfico del usuario. Además este método nos da una gran ventaja ya que las pérdidas por inserción causadas por el OTDR son bastante bajas.[10]

Consecuentemente todo el sistema de localización de fallas debe ser controlado desde un punto centralizado llamado NMC siguiendo el esquema de la Figura 2.4

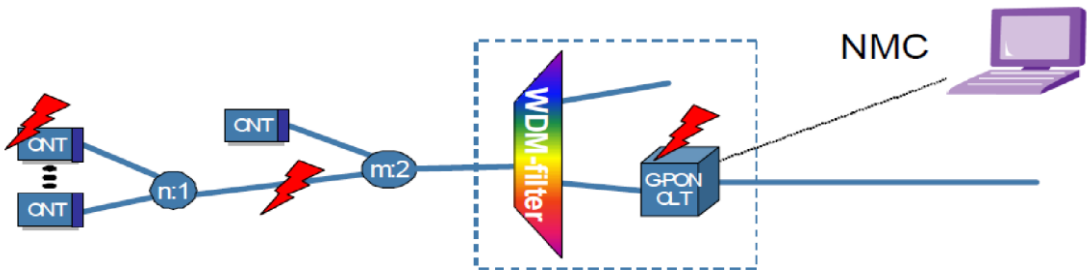


Figura 2.4. Esquema de monitoreo del Network Management Centre [10]

Sin embargo, una técnica de costo-beneficio basada en la localización de fallas usando la correlación entre las medición del trazado del OTDR contra un conjunto de trazados de referencia pre-medidos, es la más adecuada para redes del tipo FTTP. [2]

Los autores explican que la metodología de la localización se basa en realizar las medidas con un OTDR y correlacionar los trazados medidos contra el conjunto de trazados hipotéticos que son obtenidos de la combinación de trazados referenciales pre-medidos para ramas de fibras individuales bajo diferentes condiciones. De esta manera, la fibra defectuosa puede ser fácilmente identificada seleccionando el trazado hipotético con el valor de correlación más alto contra el trazado medido en el OLT. Tal y como se muestra a continuación en la Figura 2.5, la medición es para cada fibra individual y al aplicar el método a toda la red se

obtiene lo que observamos en la Figura 2.6, donde se puede apreciar el uso de la correlación para ubicar la falla.[2]

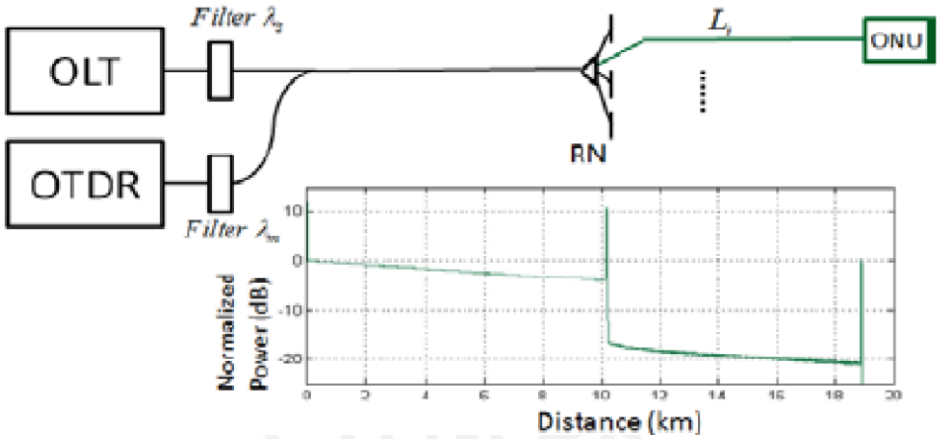


Figura 2.5. Esquema de medición del trazado del OTDR [2]

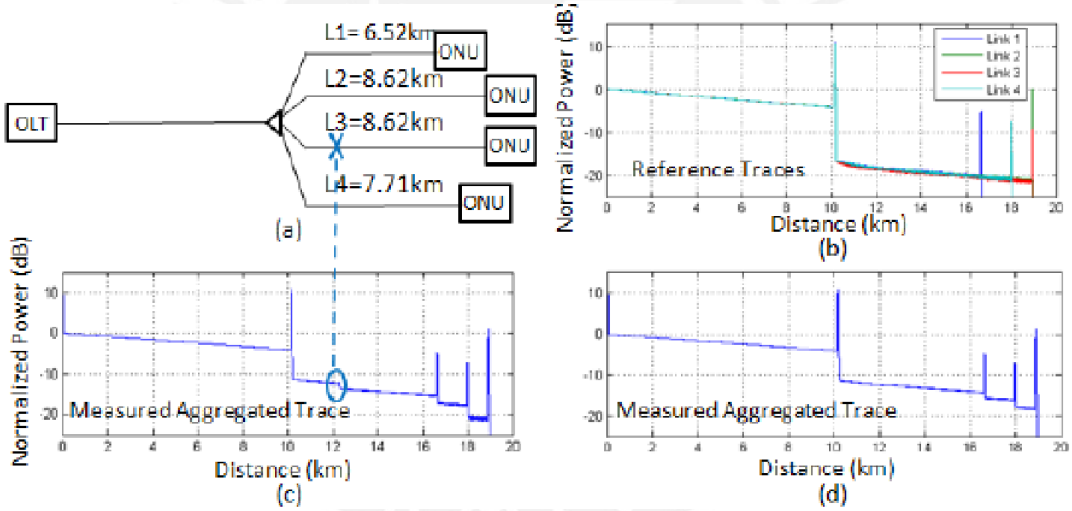


Figura 2.6. Localización de la falla en la red FTTP [2]

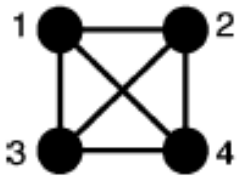
2.1.2 Metodología basada en el análisis del estado del enlace

Esta nueva metodología ha sido puesta en práctica desde hace unos cuatro años con el objetivo de conseguir alternativas de localización de averías menos costosas y que pueden ser tan altamente precisas como la reflectometría óptica en el dominio del tiempo. Entre las técnicas más comúnmente usadas tenemos los *ciclos de monitoreo (**M-Cycles**), las rutas de monitoreo (**M-Paths**), trazados de monitoreo (**M-Trails**) y el protocolo Limited Perimeter Vector Matching (**LVM Protocol**).*

La técnica utilizada y desarrollada por [6] utiliza tanto el concepto de ***Monitoring Cycles (M-Cycles)*** como el de ***Monitoring Paths (M-Paths)*** en conjunto, para ello define dos formas de localizar la avería: Con una zona de monitoreo y con múltiples zonas de monitoreo.

Para el primer caso plantea el uso únicamente de *M-Cycles* ya que solo existe un única zona de monitoreo, para este caso el objetivo poder construir un conjunto de ciclos que permitan localizar la avería. Para ello es necesario reservar una longitud de onda por cada ciclo a utilizar de modo que la falla en un ciclo puede ser detectada usando un monitor dedicado por ciclo o por un conjunto de monitores de tiempo compartido disponible en la zona de monitoreo.[6]

Un ejemplo de este caso lo observamos en la Figura 2.7 según los autores una solución factible es que todos los enlaces estén presentes al menos en un ciclo y por lo tanto una falla de enlace resulta en la falla de al menos un ciclo. Entonces la solución factible está dada por los ciclos C_1 (1 – 2 – 4 – 1), C_2 (1 – 3 – 2 – 4 – 1) y C_3 (1 – 2 – 4 – 3 – 1) donde todos los enlaces están en dos de los ciclos, además explica que se necesario y suficiente que exista conectividad entre al menos tres nodos de borde para pueda existir una solución factible para este escenario. [6]



Links	c_1	c_2	c_3	Links	c_1	c_2	c_3
(1, 2)	×		×	(4, 1)	×	×	
(2, 3)		×		(4, 2)	×	×	×
(3, 4)			×	(1, 3)		×	×

Figura 2.7. Ejemplo de topología de una red donde el nodo 1 es la zona de monitoreo, sus ciclos de monitoreo y los casos asociados [6]

Entonces para una red con pares de enlaces (L, L') la detección de la avería se realiza construyendo un conjunto de ciclos que contienen L pero no L', que contienen L' pero no L y que contienen ambos, de modo que como en el ejemplo si falla el enlace L=1 esto resultará en la falla de los ciclos C_1 y C_3 mientras que una falla de L'=3 resultará en una falla del ciclo C_2 únicamente. Por lo que fácilmente se puede detectar las fallas para los enlaces L y L'.

Para el escenario con múltiples zonas de monitoreo es necesario definir el modo de operación de dichas zonas:

Con intercambio de información: Las zonas de monitoreo trabajan de manera colaborativa para encontrar la falla, lo cual agrega un retardo al tiempo de localización pero utiliza menos recursos de red.

Sin intercambio de información: Las zonas de monitoreo trabajan independientemente, aquí la localización toma menos tiempo pero es posible que requiera mayor cantidad de recursos de red.

Después el funcionamiento del mecanismo consistirá en dos pasos:

1. ***Cloud Formation:*** Consiste en asociar cada enlace a la nube de zonas de monitoreo más cercana a dicho enlace, ya que al estar más cerca es más probable que el *M-Cycle* que pase por ese enlace sea más corto, haciendo más fácil la localización.
2. ***Cycle Formation:*** Consiste en formar un conjunto de ciclos que pasen por cada zona de monitoreo de modo que para cada falla de un enlace se obtenga como resultado la falla de una combinación única de ciclos de monitoreo: en la red (si el intercambio de información está habilitado) y en la zona de monitoreo (si el intercambio está deshabilitado).[6]

Sin embargo al tener varias zonas de monitoreo es necesario utilizar los ***M-Paths*** que se originan desde una zona de monitoreo y terminan en otra. Entonces si ocurre una falla a lo largo del *M-Path* esta será detectada por la zona de monitoreo final, de modo que si la zona de monitoreo final es la misma que la inicial, el *M-Path* se reduce a un *M-Cycle* y se aplican las condiciones del primer caso con la diferencia de ahora es necesario tener un nodo central de monitoreo para todas las zonas de modo que pueda manejarse la detección de averías de manera centralizada.[6]

Otra técnica desarrollada utiliza el concepto de ***Monitoring Trails (M-Trails)*** que consiste en construir un camino de luz no simple mediante un transmisor y un receptor, lo siguiente es usar una señal óptica supervisora que se envía a través del *M-Trail* de modo que si un enlace atravesado por el *M-Trail* falla, la señal óptica supervisora es alterada lo que en consecuencia desencadena un *flooding*

de alarmas en el dominio de control de la red óptica. Más adelante, en la Figura 2.8 observamos la definición de un *M-Trail*. [4]

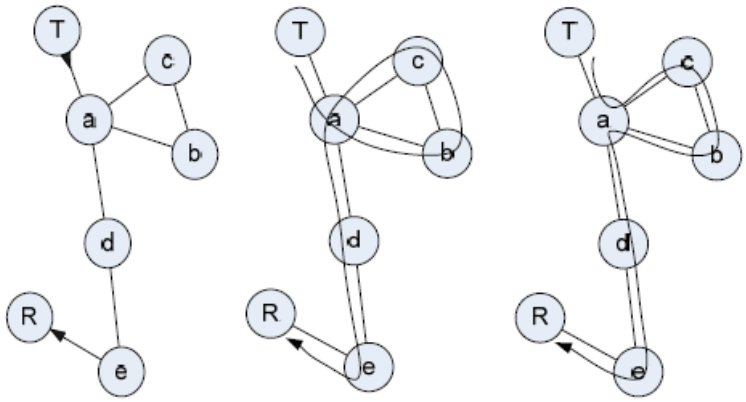


Figura 2.8. Un M-Trail está formado por un Transmisor (T) y un Receptor (R) donde este último posee un monitor para supervisar el camino de luz [4]

Se plantea el uso de un punto medio geográfico para optimizar creación de *M-Trails* ya que en principio la detección de averías se realiza utilizando un conjunto suficiente de *M-Trails* para que un sistema de administración de red pueda detectar la avería recolectando las alarmas lanzadas por los monitores de los *M-Trails* asociados y los caminos de luz desplegados, de manera oportuna. Si observamos la Figura 2.9 y vemos como se crean los *M-Trails* siguiendo los pasos del punto medio geográfico en una gráfica de red G(N, A):

1. Identificar todos los N nodos que representan el final del arco de los nodos que constituyen los bordes sólidos en G(N, A).
2. Calcular el Punto Medio Geográfico (también conocido como centro de masa o centro de gravedad)
3. Seleccionar los dos arcos que estén lo más cerca posible al punto medio calculado.
4. Encontrar el par de caminos más corto entre esos nodos de arco en la gráfica utilizando el algoritmo de Dijkstra.
5. Conectar los dos arcos con el camino más corto entre sus nodos finales para formar un nuevo arco en la gráfica.
6. Remover los nodos intermedios del nuevo arco formado desde los nodos N (calculados en el paso 1)

7. Repetir del paso 2 al 6 hasta que conectemos todos los arcos que formaran el *M-Trail* y obtendremos el diagrama que observamos en la Figura 2.10.[4]

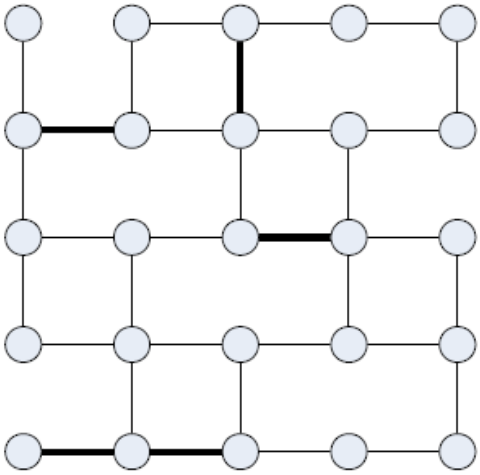


Figura 2.9. Formación de los arcos más cercanos al punto medio geográfico [4]

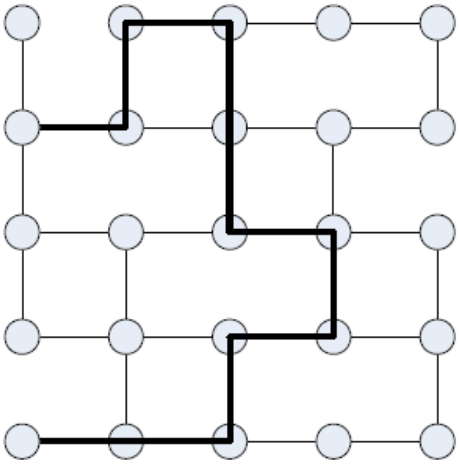


Figura 2.10. M-Trail formado con la técnica del punto medio geográfico.[4]

Pero para el uso de *M-Trails* puede ser utilizado junto con un modelo matemático con restricciones [5], es decir delimitar condiciones de forma y longitud para el *M-Trail* con la finalidad de obtener y agilizar la localización inequívoca de fallas (*UFL*) bajo el concepto anteriormente explicado. Para ello plantean que para una gráfica de red G (E, V) con |*E*| enlaces y |*V*| nodos, un límite de longitud $k \geq 1$ y para un valor mínimo *b,* se establecen las siguientes restricciones:

- **Restricción UFL:** Cada enlace debe ser asignado con un único código de alarma binario diferente de cero, donde *b* es la longitud del código.

- **Restricción de forma del *Trail*:** Los enlaces con el valor igual a '1' al inicio del código de alarma deben formar el Trail para una posición *L=1,…, b*. Esto es necesario para que un *M-Trail* sea capaz de atravesar cada enlace con un solo camino de luz con bit 1 en la posición *L* correspondiente del código de alarma.
- **Restricción de longitud:** Cada *M-Trail* puede atravesar a lo más *k* enlaces.[5]

Por otra parte, se presenta una técnica que utiliza el **protocolo LVM**, este es un protocolo de localización de fallas en redes ópticas totales. El protocolo asume que no está disponible un monitoreo de potencia óptica en los nodos intermedios y que solo los nodos de borde son capaces de detectar las pérdidas de potencia o la degradación de la calidad de las señales ópticas [7]. En consecuencia, la detección de la avería se realiza a través de los siguientes pasos:

1. Una vez que falla un enlace, todos los caminos de luz que pasan a través del enlace averiado se interrumpieron. Como resultado, el destino de cada uno de los caminos de luz interrumpidos detectará la pérdida de la señal del camino de luz y de la falla.
2. Luego de que un nodo de destino detecta la falla, se emite un mensaje de ALARM a todos los nodos dentro de su dominio, este mensaje contiene su identificador de nodo y la longitud del camino de luz más corta afectada que termina en él.
3. Cada nodo receptor, también llamado **potencial sumidero ejecutivo (PES),** tiene una copia de los mensajes ALARM enviados por los otros potenciales sumideros ejecutivos. Cada PES compara el camino de luz que termina en él con los otros caminos de luz recibidos por las alarmas originadas por otro PES. El PES que tiene el camino de luz más corto es elegido como el sumidero ejecutivo.
4. El sumidero ejecutivo a continuación, crea un vector de enlace en su camino de luz afectado más corto, llamado **vector de enlace afectado (ALV)**, define un perímetro limitado que consta de todos los enlaces ALV y sus nodos vecinos, luego hace *multicast* del ALV a todos los nodos dentro del perímetro limitado.
5. Cada sumidero que recibe el ALV dentro del perímetro limitado crea su propio vector de enlace y compara el vector con el ALV. Si su vector de enlace tiene

un vínculo común con el ALV, el sumidero envía un vector binario de regreso al sumidero ejecutivo. El vector binario se compone de un vector coincidente y el estado del camino de luz correspondiente. El vector coincidente contiene los mismos enlaces dentro del ALV. Si el camino de luz se encuentra en estado de funcionamiento, su estado se pone a "1". De lo contrario, el estado se pone a "0". Para un camino operativo, un elemento en el vector coincidente se establece en "0" si el enlace correspondiente se encuentra en el vector de enlace del sumidero. De lo contrario, se establece en "1". Para un camino de luz fallido, un elemento en el vector coincidente se establece en "1" si el enlace correspondiente se encuentra en el vector de enlace del sumidero. De lo contrario, se establece en "0".

6. Después que el sumidero ejecutivo recoge los vectores binarios de todos los sumideros dentro del perímetro limitado, realiza una operación AND lógica en todos los vectores binarios recogidos para determinar la ubicación de la falla. El enlace con "1" en el vector binario resultado se determina como el enlace fallido. Si hay más de "1" en el vector binario resultado, el sumidero ejecutivo puede extender el perímetro limitado mediante la inclusión de los vecinos de los nodos perimetrales originales y ejecutar una segunda ronda del proceso anterior.
7. Una vez que el enlace averiado se localiza, el sumidero ejecutivo hace *broadcast* de un mensaje FAULT_LOCATION a todos los nodos de la red, les notificará el lugar del fallo.[7]

2.2 Modelo Teórico

El modelo teórico que se plantea es un sistema de supervisión centralizado de una de red de fibra óptica de planta externa metropolitana, que requiera de la utilización de una nueva metodología de detección, localización y alerta de averías conocida como análisis del estado de los enlaces y una aplicación web en un framework basado en JAVA conectada a una base de datos con información georeferenciada de la red de planta externa metropolitana cuyo objetivo es permitir el envío y visualización de las alertas de averías, así como una localización más precisa y eficiente de las averías presentes en la red.

Los componentes necesarios para la implementación del sistema son los siguientes:

- **Infraestructura de red de transporte entre los nodos principales:**

Como paso inicial se debe contar con una infraestructura de red de transporte previamente implementada que permita la comunicación entre los diferentes nodos principales de la red de planta de externa metropolitana y el sistema de supervisión, esto es necesario para que el sistema pueda recopilar la información de estado de los enlace de fibra óptica y sea capaz de detectar y localizar las averías. Esta infraestructura deberá ser de alta disponibilidad y también será censada por el sistema, por encontrarse también en la planta externa, con la finalidad de asegurar que la información llegue siempre al sistema; además de permitir una supervisión total de la planta externa.

- **Diseño de la aplicación web y base de datos con información georeferenciada:**

El siguiente paso consistirá en el diseño de una aplicación web en un framework basado en JAVA y una base de datos que almacene la información georeferenciada de la red de planta externa, la aplicación tendrá tres funciones principales: primero, la de detectar una avería bajo criterios de identificación que descarten el reinicio de equipos o la desconexión física de los enlaces de fibra óptica con el fin de evitar detecciones erróneas; segundo, la de localización de la avería en base a un procedimiento de descarte de casos de averías junto con el uso de la base de datos con la información georeferenciada de la red de planta externa metropolitana, que consiste en datos de latitud y longitud de cada elemento presente en la red (cables de fibra óptica, cámaras, postes, etc.), además de información que permita identificar y diferenciar entre sí cada tipo de elemento, con la finalidad de localizar las averías lo más preciso posible; tercero, el envío de alertas o alarmas por correo y la visualización en pantalla de la alarma y la localización de la avería en un navegador web, para la visualización de la avería se utilizará el servicio web gratuito de GoogleMaps® en el cual se mostrará la posición donde se encuentra la avería en una capa sobre el mapa de calles de la planta externa metropolitana, razón principal por la que el aplicativo debe ser

web además de que brinda facilidades de acceso a la información a múltiples usuarios.

- **Implementación de los servidores web y de base de datos:**

El tercer paso consistirá en la implementación de los servidores sobre los que se montarán el aplicativo web y la base de datos georeferenciada, cada uno sobre un servidor para obtener el máximo rendimiento para las operaciones definidas en cada procedimiento del sistema. La idea de tener cada servidor de manera independiente es de brindar la mayor cantidad de recursos de procesamiento de datos y evitar estancamientos o sobrecarga de procesos, por falta de los mismos en los servidores, debido a la gran cantidad de operaciones y/o conexiones que realizará el sistema, tanto en la captura y almacenamiento de información por parte de la base de datos como en la detección y localización de averías por parte de la aplicación web.

- **Uso de una central de monitoreo de la red:**

El último paso será la participación de un grupo humano que controle y monitoree el sistema y la red, puesto que a pesar que las alertas se envían automáticamente debe existir una corroboración de que estas alertas no sean falsas y una comunicación con el personal de mantenimiento para realizar las operaciones necesarias; por ende, es necesario contar con un grupo de personal que monitoree las veinticuatro horas del día y los siete días de la semana, tanto el sistema de supervisión como la red en sí y para ello debe también existir una infraestructura o central de monitoreo donde se pueda colocar todos los equipos necesarios para el monitoreo, además de los propios servidores del sistema que serán controlados desde este punto para lograr la centralización requerida, sin dejar de lado que debe tener acceso a la red de transporte para que el sistema pueda funcionar correctamente.

Tal y como se expuso anteriormente, todos estos componentes trabajan en conjunto para lograr el objetivo de optimizar la atención de averías en redes de planta externa metropolitana, brindando de manera más eficiente y rápida la información de las averías que se presenten en las diferentes redes donde se implemente el sistema. De esta forma se conseguirá reducir los tiempos que toma

la detección y localización de averías y en consecuencia mejorar la calidad de servicio brindado al cliente, así como el aumento en su satisfacción respecto al uso de los servicios contratados.

A continuación, se presenta una ilustración gráfica del modelo teórico en la Figura 2.11 que nos brinda una visión de una red de planta externa metropolitana con una red de transporte y muestra el funcionamiento del sistema de supervisión implementado en dicha red. El modelo consta de los siguientes elementos y funciona de la siguiente forma:

Elementos:

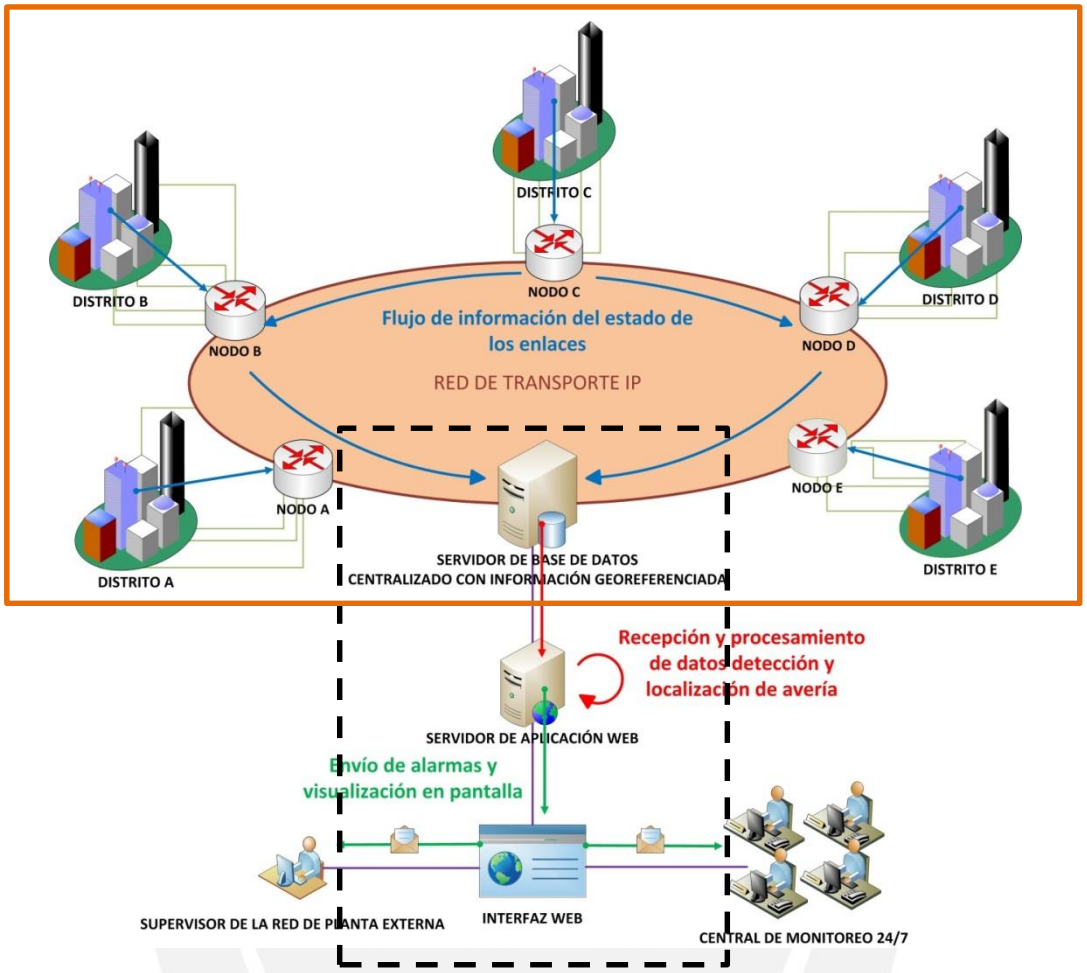
- Red de transporte de fibra óptica que interconecta los nodos principales de la red de planta externa metropolitana
- Servidor Centralizado de Base de Datos con información georeferenciada de la red de planta externa metropolitana
- Servidor de Aplicación Web donde se montará el aplicativo para el monitoreo de la red de planta externa metropolitana
- Aplicación Web para monitoreo de la red de planta externa metropolitana
- Central de monitoreo de la red que contiene al grupo humano que trabajará con el aplicativo web para la atención y solución de averías.

Funcionamiento:

1. El servidor de base de datos captura y almacena la información de estado de los enlaces de la red de planta externa a petición de la aplicación web en la etapa de detección de averías, esta recopilación de datos se realiza periódicamente en intervalos cortos (orden de los milisegundos) según lo requerido para la red en particular.
2. El aplicativo web revisará la información capturada y verificará si existe algún enlace con un estado de *online* u *offline*. En caso de detectar un estado *offline*, pasara a la etapa de descarte de averías, donde se evaluará, durante un tiempo definido según los requerimientos de los elementos de la red, si se reinició el equipo al que va dicho enlace o si se desconectó y reconectó el enlace, con la finalidad de evitar el envío de falsas alarmas.

3. Si luego de haber terminado el tiempo de descarte se sigue obteniendo un estado de enlace *offline*, se procederá a la etapa de localización de la avería, donde se determinará la posición de la avería con un descarte de escenarios posibles de localización dentro de un área geográfica determinada (por ejemplo, un distrito) en base a la información georeferenciada de los elementos de la red involucrados, a su vez se enviará la alarma respectiva a los equipos encargados de realizar la supervisión, esta alarma tendrá un enlace que apunta hacia la interfaz web que mostrará la información detallada de la avería.
4. Una vez localizada la avería, se procederá a mostrar la información de la misma en la interfaz web del aplicativo que mediante un mapa de calles de Google Maps® mostrará la posición y el detalle completo de la avería, el cual podrá ser visualizado desde cualquier navegador que soporte HTML5 (preferentemente Google Chrome® o Mozilla Firefox®). Adicionalmente, se tomará registro en la base de datos de la incidencia ocurrida, es decir, se guardará todo el detalle de la falla, incluido fecha y hora, para tener información sobre la cual realizar estadísticas y analizar futuras optimizaciones para la red.
5. Luego de completarse la visualización, el sistema reinicia el censado de los enlaces repitiendo el ciclo de funcionamiento.

Red de Planta Externa Metropolitana



Sistema de detección, localización y alerta de averías

Figura 2.11.Modelo teórico del sistema de detección, localización y alerta de averías
[Elaboracion propia]

CAPITULO III
DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

En el presente capítulo se presenta el proceso del diseño e implementación del sistema de detección, localización y alerta de averías enfocado a la problemática anteriormente planteada. Como punto de partida, se presentará el planteamiento de los casos de uso de la aplicación con la finalidad de definir las funcionalidades principales del sistema, luego se explicará la arquitectura que conforma el sistema y sus partes. Para finalmente concluir con el detalle del proceso de implementación de las funcionalidades del sistema.

3.1 Diagrama de Casos de Uso

El diagrama de casos de uso es una herramienta útil que sirve para describir de una manera sencilla la funcionalidad del sistema, presentando la participación de los actores involucrados en cada uno de los procesos del mismo.

3.1.1 Actores

En el diseño se consideran a los siguientes participantes en el funcionamiento del sistema:

- Administrador: Es el usuario asignado al manejo y mantenimiento del sistema y sus componentes, tiene acceso a todas las funcionalidades y a la configuración de la aplicación.

- Usuario Operador: Es el usuario que atenderá las averías en primera instancia, y tendrá a acceso a las funcionalidades de visualización de los mapas y elementos de la red, sus clientes y las alertas.
- Usuario Especialista: Es el usuario que atenderá las averías en segunda instancia, cuando se requiera conocimientos más especializados de la red. Este tendrá acceso a todas funcionalidades de visualización elementos y clientes en la red y a las alertas, así como también a la creación de nuevos elementos de la red y clientes.

3.1.2 Diagrama de Caso de Uso: Registrar usuarios

El administrador deberá iniciar sesión para acceder a la interfaz de administración y poder seleccionar la función de Registrar Usuario. Después de acceder, ingresará los datos del usuario en el formulario, creará su usuario, contraseña y se asignará un nivel de permiso Operador o Especialista. Una vez creada la cuenta, el usuario podrá iniciar sesión y acceder a las funcionalidades del sistema según sus permisos. (Ver Figura 3.1)

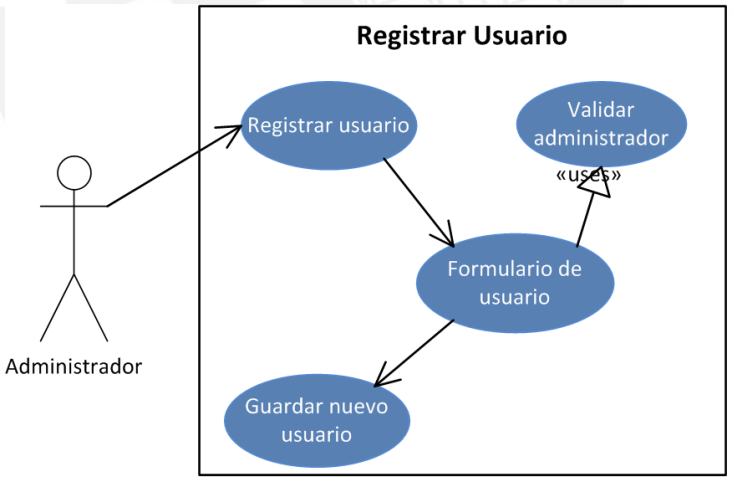


Figura 3.1. Diagrama de Caso de Uso: Registrar usuarios [Elaboracion propia]

3.1.3 Diagrama de Caso de Usos: Visualizar menú principal

El usuario operador o especialista se le solicitará iniciar sesión al intentar acceder al menú principal, luego de validar sus credenciales se mostrará en pantalla el menú con las opciones acordes a su rol (ver Figura 3.2).

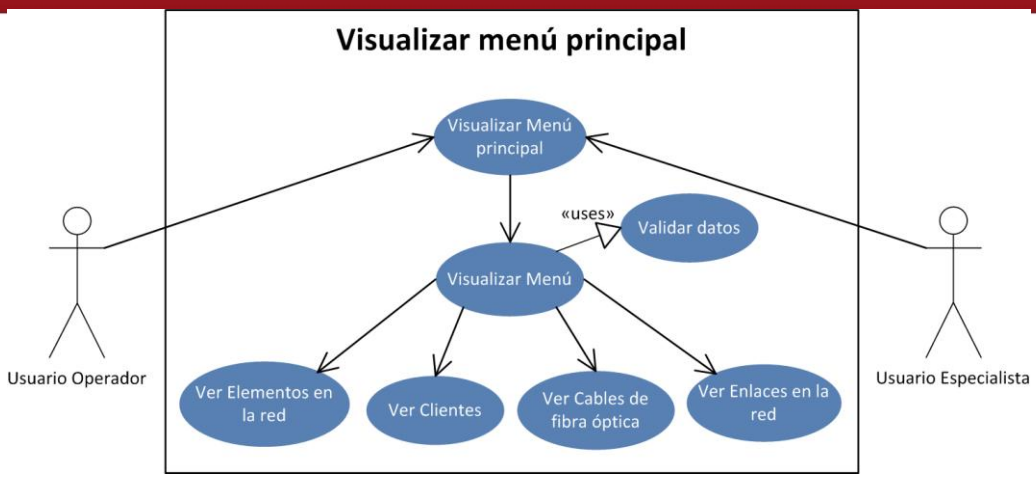


Figura 3.2. Diagrama de Caso de Uso: Ver menú principal
[Elaboracion propia]

3.1.4 Diagrama de Caso de Uso: Ver elementos en la red

El usuario operador o especialista deberá iniciar sesión para poder validar sus permisos de ingresar a la funcionalidad de los elementos de red, una vez validado podrá seleccionar la opción y visualizar los elementos disponibles en la red en un mapa en pantalla, el usuario podrá agregar nuevos elementos haciendo clic en la posición donde se ubica el elemento y enviando la información a la base de datos, de esta forma el nuevo elemento queda guardado en el sistema.(Ver Figura 3.3)

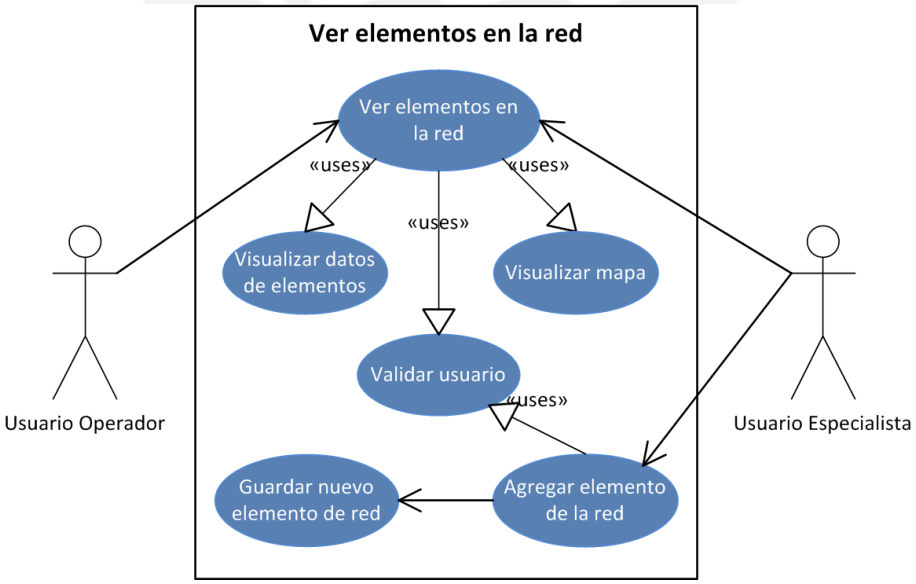


Figura 3.3. Diagrama de caso de uso: Ver elementos en la red
[Elaboracion propia]

3.1.5 Diagrama de Caso de Uso: Ver enlaces de la red

Los usuarios operador y especialista podrán visualizar inicialmente los elementos y clientes asociados a los enlaces disponibles en la red y también tendrán la opción de ejecutar la función que muestra la ruta que sigue el enlace de cada cliente. (Ver Figura 3.4)

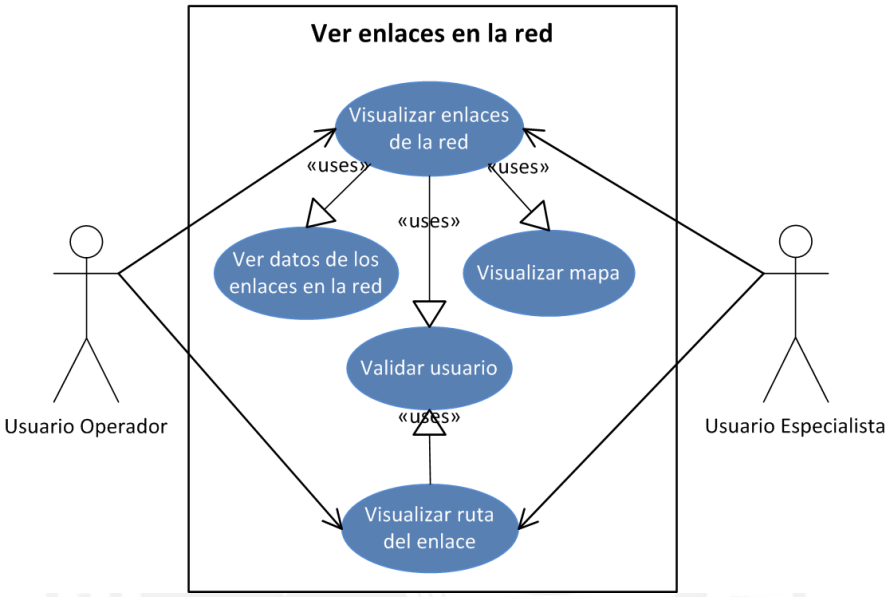


Figura 3.4. Diagrama de Caso de Uso: Ver enlaces en la red [Elaboracion propia]

3.1.6 Diagrama de Caso de Uso: Ver clientes

Los usuarios operador y especialista serán capaces de visualizar los clientes de la red en un mapa en pantalla, junto con un campo donde se detalla su información y su historial de alertas. Sin embargo, el usuario especialista podrá agregar nuevos clientes a la cartera actual a diferencia del operador. (Ver Figura 3.5)

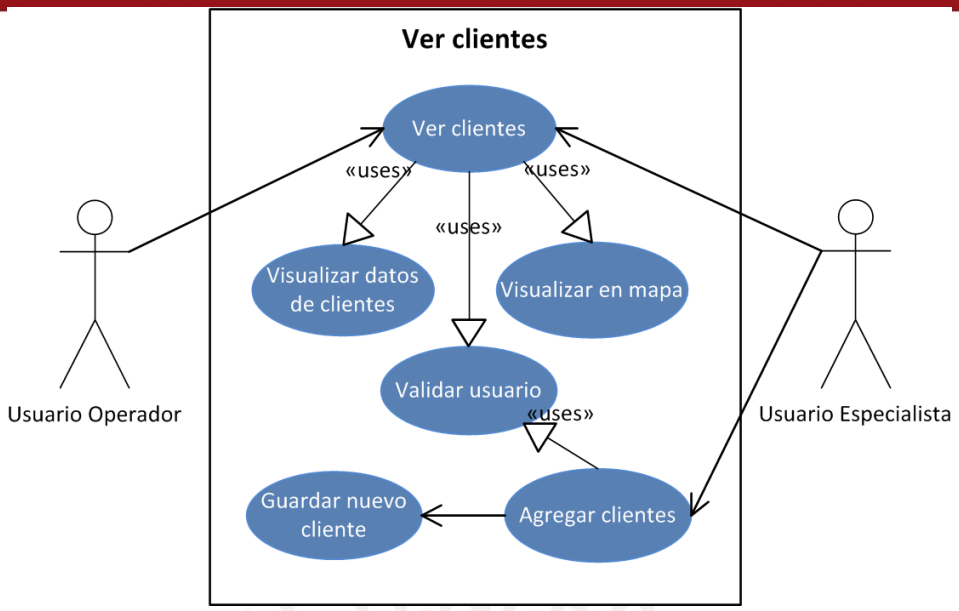


Figura 3.5. Diagrama de Caso de Uso: Ver clientes
[Elaboracion propia]

3.1.7 Diagrama de Caso de Uso: Ver cables de fibra óptica

Los usuario operador y especialista será capaz de visualizar en pantalla los cables de fibra óptica actualmente operativos en la red a través de un mapa, así como visualizar la ruta que sigue el cable y los elementos asociados al mismo. (Ver Figura 3.6).

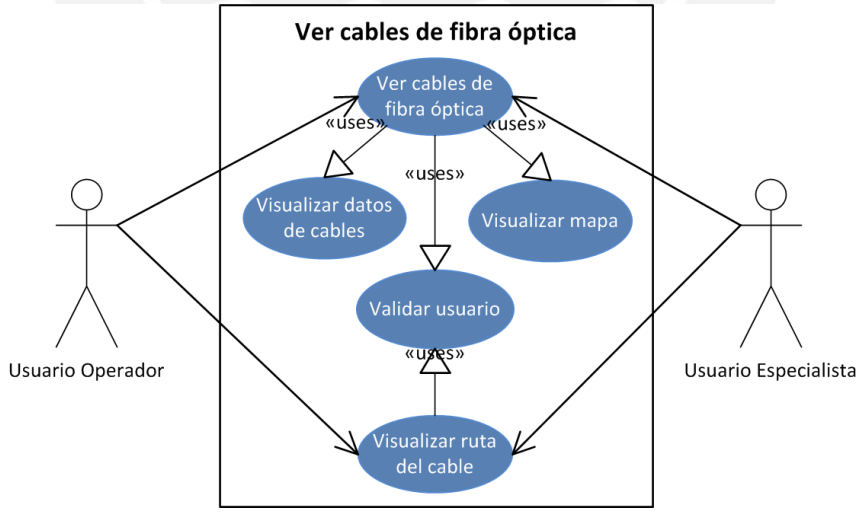


Figura 3.6. Diagrama de Caso de Uso: Ver cables de fibra óptica
[Elaboracion propia]

3.1.8 Diagrama de Caso de Uso: Ver alertas

Los usuarios operador y especialista serán capaces de visualizar las alertas de las averías, ni bien se detecten, desde cualquier funcionalidad que esté utilizando.

Basta con que este haya iniciado sesión en el sistema y una vez lanzada la alerta podrá acceder a la vista detallada de la alerta, donde se mostrara el mapa con el enlace afectado y la localización de la avería (ver Figura 3.7).

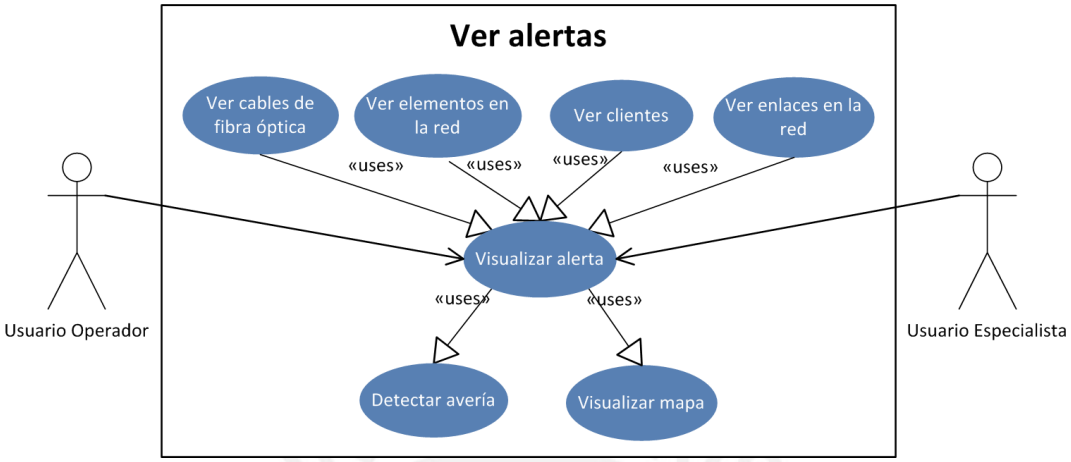


Figura 3.7. Diagrama de Caso de Uso: Ver alertas [Elaboracion propia]

3.2 Arquitectura del sistema

El sistema fue diseñado utilizando diferentes componentes para realizar cada función de manera eficiente y de modo que pueda aprovechar los recursos brindados por las computadoras de los usuarios. Tal y como podemos observar en la Figura 3.8, se ha utilizado en conjunto el *web framework web2py* como núcleo sobre el cual será implementada la aplicación web, así como el uso de *MySQL Server* para el almacenamiento de la información utilizada por la misma, esto por el lado del servidor (*backend*) sobre el cual se desplegará la aplicación. Para el lado del usuario (*frontend*), es decir, para la parte que interactúa con el mismo, se optó por utilizar la librería de *Angular JS* con la finalidad de lograr que la visualización de la información en pantalla sea en tiempo real, además de permitir el uso de Google Maps ® de una forma más interactiva.

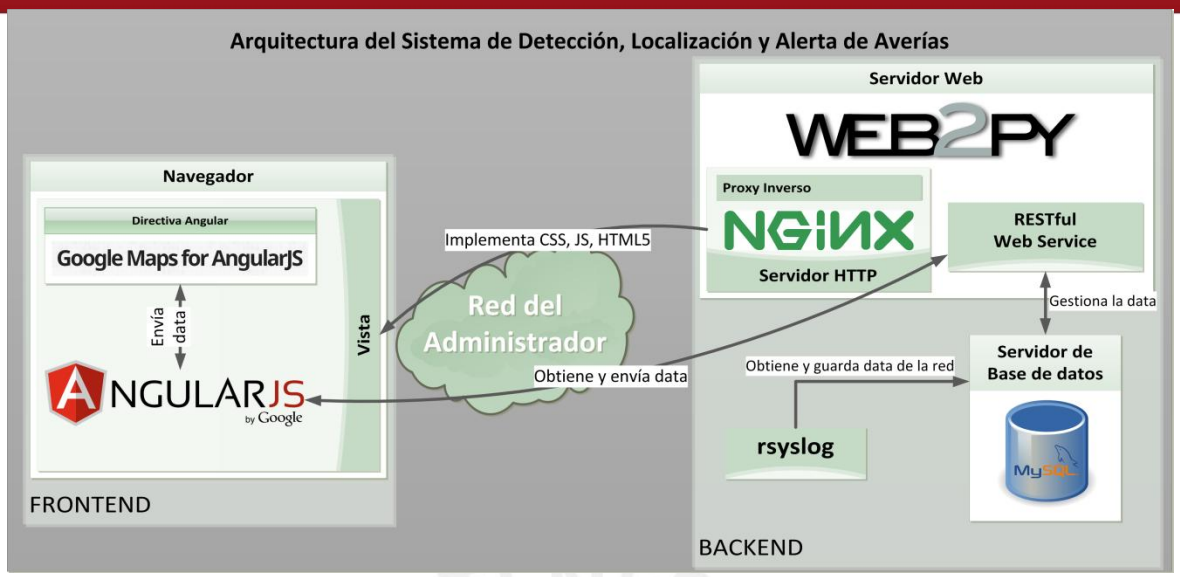


Figura 3.8. Diagrama de bloques de la arquitectura del sistema [Elaboracion propia]

A continuación se detalla cada uno ellos y como es que rol desempeñan para lograr que la aplicación web funcione.

3.2.1 Web Framework web2py

Este framework enfocado el modelo MVC que separa la representación de datos (el modelo), de la presentación de datos (la vista) y de la lógica y funcionamiento de la aplicación (el controlador) [25] y como su nombre indica funciona sobre lenguaje Python. Es un *framework Full-Stack,* es decir, que viene equipado con una serie de componentes esenciales para poder implementar por sí solo una aplicación web completamente funcional [25], por ejemplo, cuenta con soporte de múltiples servidores web (como Apache), también cuenta con una interfaz gráfica de administración propia que permite crear y gestionar las aplicaciones que uno desea (ver Figura 3.9), posee una Capa de Abstracción de Base de Datos (DAL) [25] que escribe código SQL dinámicamente de modo que puede generar de manera transparente bases de datos en MySQL, Oracle o MSSQL, por ejemplo. De igual forma provee el manejo de controladores para la implementación de las vistas en diferentes lenguajes de codificación (HTML, Javascript, etc.) (Ver Figura 3.10).

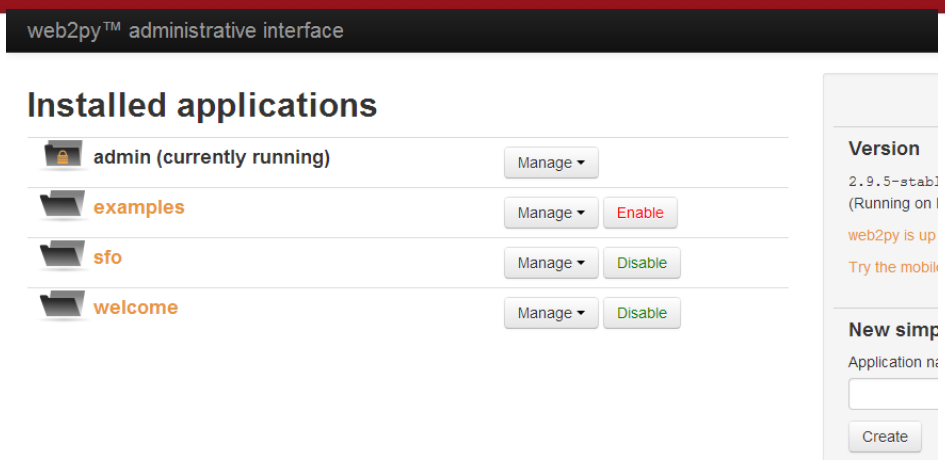


Figura 3.9. Interfaz gráfica de administración web2py
[Elaboracion propia]

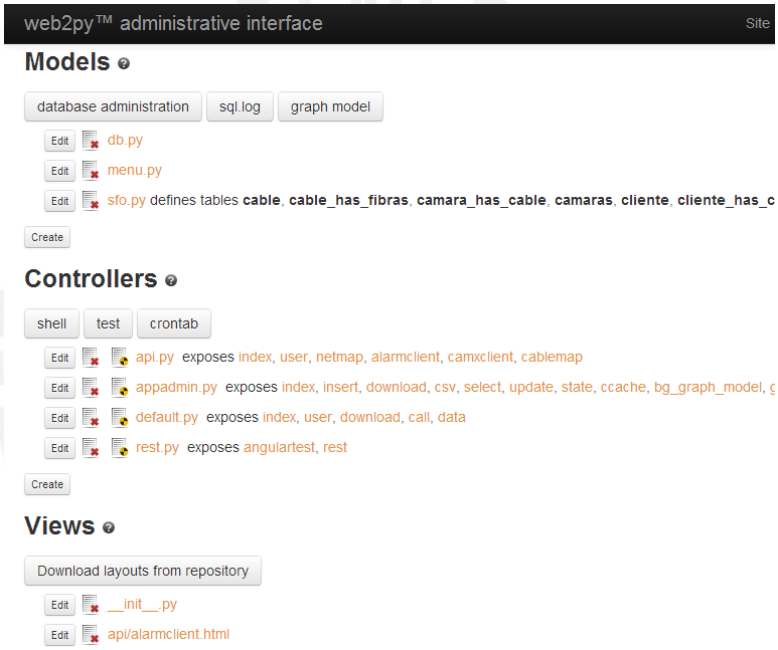


Figura 3.10. Distribución de elementos en la interfaz gráfica de administración basada en el modelo MVC
[Elaboración propia]

3.2.1.1 NGINX

Es un servidor HTTP de alto desempeño y *proxy* inverso que no utiliza *threads* como la mayoría de servidores tradicionales, sino pequeños bloques de memoria predecibles para atender peticiones (*requests)* siguiendo una arquitectura en base a eventos [33]. Es un *proxy* inverso, porque gracias a su arquitectura puede realizar *caching* (*proxy*) de los recursos que se enviarán hacia el *frontend* (inverso), de modo que si se accede múltiples veces al mismo recurso, este no tendrá que ser obtenido directamente desde el servidor sino que será almacenado en un contenedor temporal y será accedido desde ahí; como consecuencia se

percibe una mayor velocidad en la que la información se muestra al usuario y aligera la cantidad de procesamiento que realiza el servidor web. Por lo expuesto anteriormente, se eligió como servidor web para la implementación de web2py.

El objetivo de usar web2py en este proyecto es que sirva como el núcleo del funcionamiento de la aplicación, ya que además de funcionar como servidor tiene a su cargo las siguientes tareas:

- Facilitar el mantenimiento de la aplicación, sus funcionalidades y la base de datos, gracias a las herramientas gráficas que viene embebidas en el *framework*, además de permitir el registro de cualquier falla que se presente a nivel del servidor, la aplicación y/o base de datos.
- La implementación de cada una de las vistas y sobre las cuales se implementará el *framework* AngularJS y Google Maps ®.
- Gestionar el acceso, creación y mantenimiento de usuarios, para las funcionalidades del programa a través del módulo de autenticación básica que viene integrado con el *framework*.
- Y la implementación del *RESTful Web Service*, que no es otro que el servicio que permite obtener, guardar y actualizar datos en la base de datos; este servicio es crucial para el funcionamiento de Angular JS, el cual será explicado más adelante.

3.2.2 Javascript Framework AngularJS

AngularJS es un *framework* estructural para aplicaciones web dinámicas. Te permite utilizar HTML como lenguaje plantilla y te permite extender la sintaxis HTML para exponer los componentes de la aplicación desarrollada [26]. Tiene como finalidad apuntar el desarrollo web principalmente al entorno del cliente, es decir, todo sucede en el navegador del cliente, lo que lo hace compatible con cualquier tecnología de servidor [26]. AngularJS se caracteriza, principalmente por ser *Stand-Alone* respecto al uso de dependencias, ya que no requiere de ninguna librería para poder funcionar y por su flexibilidad, ya que cualquier de sus componentes puede ser modificado para adaptarse a según la aplicación como a cualquier *framework* que estemos utilizando (en este caso web2py), cuenta con un lenguaje sencillo, ya que los módulos de AngularJS son simples objetos de

Javascript (JS), lo que permite que sea fácil de evaluar, reutilizar, mantener y además es libre de *boilerplate code* [26], es decir, evita código innecesario, incluyéndolo implícitamente.

El uso de AngularJS en este trabajo sigue las prácticas recomendadas por los sus desarrolladores, estas prácticas consisten en crear y separar los componentes de AngularJS a utilizar de acuerdo a su función para luego ser "inyectados" unos dentro de otros según como se necesite utilizarlos. Pues bien de estos componentes resaltan los más relevantes:

3.2.2.1 Módulo:

Es un contenedor sobre el cual se pueden agregar las diferentes partes de la aplicación, ya sean controladores, servicios, etc. vía *"Dependency Injection"* (ver Figura 3.11) [27], los cuales permitirán el uso de los elementos, servicios, directivas o funciones definidas a nivel de AngularJS en nuestra vista.

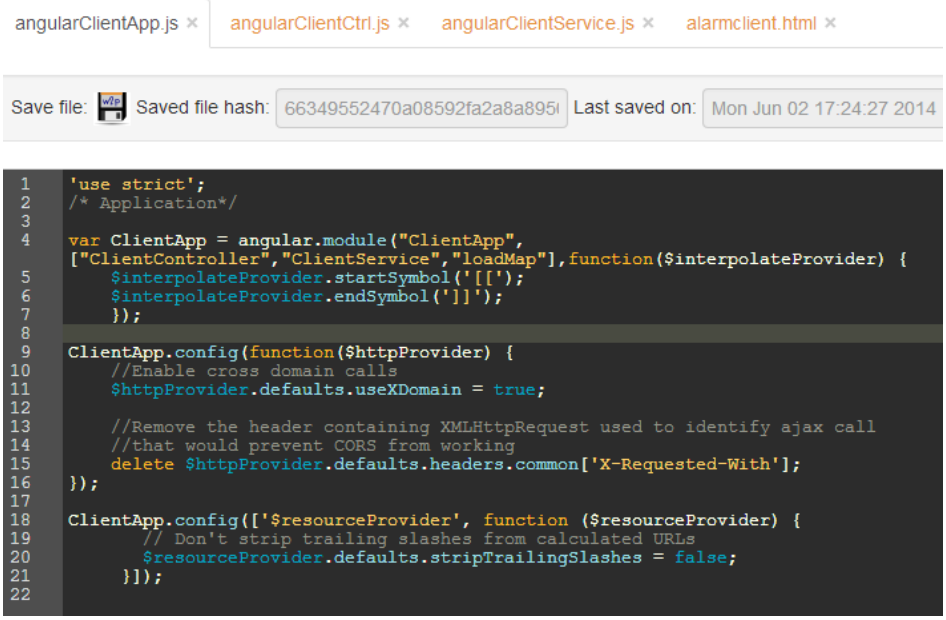


Figura 3.11. Módulo de AngularJS "ClientApp", que tiene "inyectados" los componentes "ClientController", "ClientService" y "loadMap" como dependencias para su funcionamiento.
[Elaboracion propia]

3.2.2.2 Controladores:

Estos módulos son una funciones constructoras de Javascript que permiten aumentar el *scope* de Angular*;* cuando un controlador es adjuntado a un elemento DOM a través de la directiva *ngController* (ver Figura 3.12), se instancia un nuevo objeto "Controlador" y a su vez un nuevo *child scope* estará disponible para ser "inyectado" bajo el nombre *$scope* a la función constructora del controlador [27]. Como cualquier módulo de AngularJS, es necesario "inyectar" cualquier directiva y/o elemento proveniente de otro módulo que se vaya a utilizar en nuestro controlador (ver Figura 3.13). El uso de un controlador tiene 2 objetivos:

- Asignar el valor inicial del objeto *$scope. [27]*
- Agregar el comportamiento al objeto *$scope. [27]*

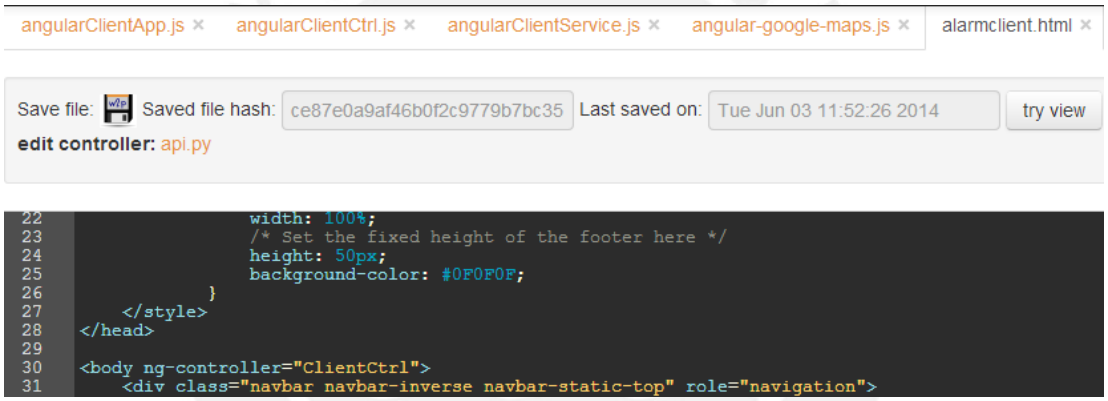


Figura 3.12. Uso de la directiva "ngController" para asignar un controlador de AngularJS al elemento DOM "<body>" en la vista "alarmclient.html" [Elaboracion propia]

Figura 3.13. Módulo controlador que tiene "inyectados" los módulos $scope, $rootScope y los elementos setClient y Cientes para ser procesados y utilizados por las variables dentro del ámbito de la vista
[Elaboracion propia]

3.2.2.3 Servicios:

Los servicios de AngularJS son objetos sustituibles que están interconectados entre sí vía "*Dependency Injection*", de modo que se puede organizar y utilizar código a través de cualquier parte de la aplicación [28], también se usa para obtener datos de servicios externos (ejm. *RESTful Web Service*) y utilizarlos en la aplicación. Para usar los servicios en nuestra aplicación, basta con "inyectarlos" en el componente deseado, ya sea controlador, directiva, etc. que dependa del servicio en cuestión (ver Figura 3.14). Como es de esperarse de un componente propio de AngularJS, el nombre de un servicio antepone "$" antes de su para fácil identificación, por ejemplo "*$resource"*. [28]

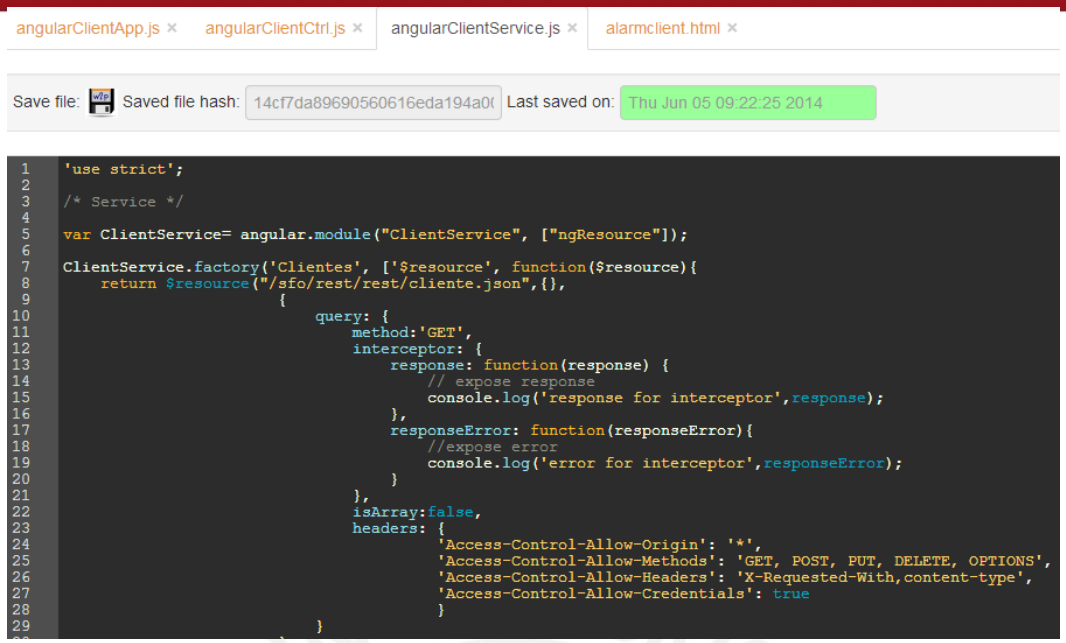


Figura 3.14. Servicio "$resource" inyectado en el módulo "ClientService" para obtener los datos de la URI : "/sfo/rest/rest/cliente.json" [Elaboracion propia]

3.2.2.4 Directivas:

Las directivas son marcadores en un elemento DOM que le indican al “compilador HTML” de AngularJS que debe asignar un comportamiento o funcionalidad específica a dicho elemento o bien transformarlo incluyendo a sus subelementos [29]. Estos marcadores se colocan dentro del elemento DOM como el nombre de dicho elemento, un atributo, un comentario o una clase CSS, de modo que cuando AngularJS se inicializa en la vista, el compilador cruza las directivas DOM con los elementos DOM que las tienen asignadas [29] (ver Figura 3.15).

Directiva

```
angular.module('docsSimpleDirective', [])
  .controller('Controller', ['$scope', function($scope) {
    $scope.customer = {
      name: 'Naomi',
      address: '1600 Amphitheatre'
    };
  }])
  .directive('myCustomer', function() {
    return {
      template: 'Name: {{customer.name}} Address: {{customer.address}}'
    };
  });
```

Vista

```
<div ng-controller="Controller">
  <div my-customer></div>
</div>
```

Name: Naomi Address: 1600 Amphitheatre

Figura 3.15. Ejemplo de uso de una directiva simple "myCostumer" para mostrar una plantilla de datos [29].

Entonces, tomando en cuenta los componentes descritos anteriormente, en el presente trabajo se utilizaron todos ellos, cada uno con un propósito diferente:

- Cada vista implementada por el *framework* web2py cuenta con su propia aplicación de AngularJS, es decir, se creó un módulo con sus respectivas dependencias "inyectadas" (controlador, servicio y directiva), para cada funcionalidad y según sus necesidades.
- Se creó un módulo que actúa como controlador para cada funcionalidad a implementar para gestionar y procesar la información de la red que se habilitará en el *scope* de la vista asociada.
- Se creó un módulo que actúa como de servicio para cada funcionalidad para obtener la información que sería procesada por el controlador. Se utilizó específicamente el servicio *$resource* para conectarse al *RESTful Web Service* implementado en *framework* web2py.
- Se utilizó la directiva Angular-Google-Maps perteneciente a los desarrolladores [30], para implementar todas las funcionalidades que utilizan Google Maps® para mostrar información de la red en el navegador en tiempo real, de modo que se aprovechan las ventajas de AngularJS frente a usar código estático como del API de Google Maps®.

- Debido a que todos estos módulos necesitan ser agregados como archivos *script* en la vista (ver Figura 3.16), se encuentran almacenados dentro de la aplicación en una dirección estática y son administrados a través de la herramienta gráfica de web2py (ver Figura 3.17)

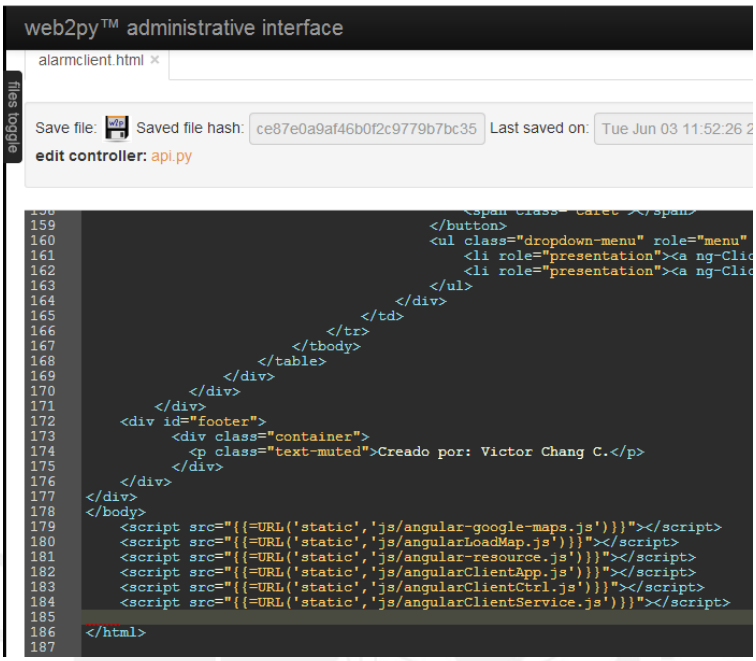


Figura 3.16. Ejemplo de uso de componentes de AngularJS en la vista "alarmclient.html".
[Elaboracion propia]

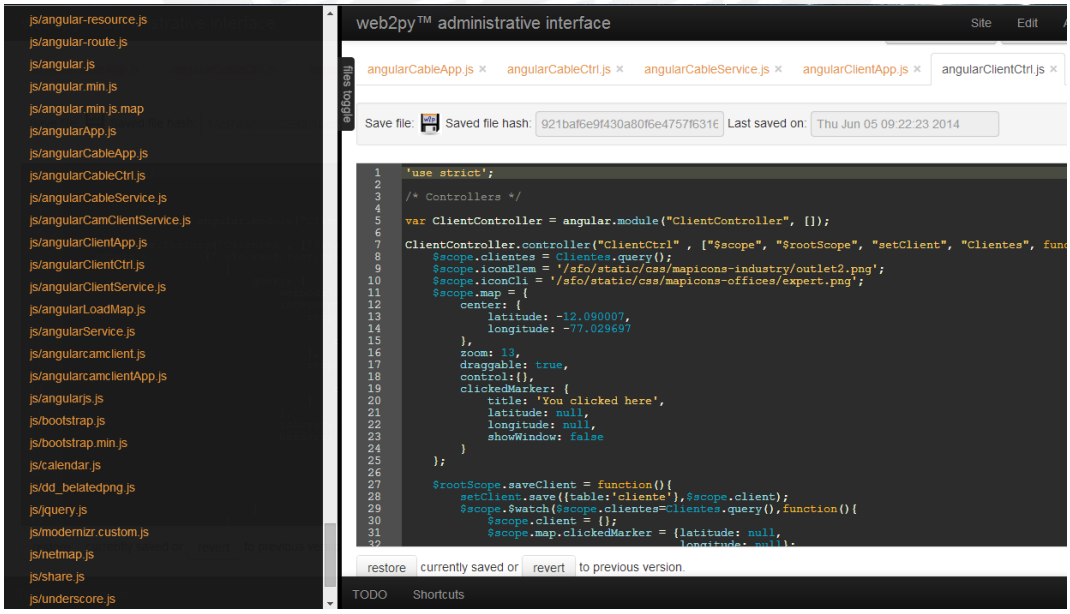


Figura 3.17. Interfaz de edición de archivos de web2py junto con la lista de los archivos de AngularJS según su ruta de almacenamiento.
[Elaboracion propia]

3.2.3 MySQL® Server

MySQL® es un sistema de administración de bases de datos relacionales SQL, que tiene como servidor a MySQL® Server para la administración y mantenimiento de bases de datos de gran capacidad y en entornos de todo tipo [31]. Es un *software Open Source* ya que su código fuente puede ser modificado según las necesidades del administrador, adicionalmente provee de herramientas gráficas como MySQL® Workbench (ver Figura 3.18) para hacer más sencilla la labor de gestión del servidor y las bases de datos. También está diseñado para trabajar en un entorno cliente/servidor, ya que puede trabajar con diferentes *backends*, programas, API's, entre otros [31].

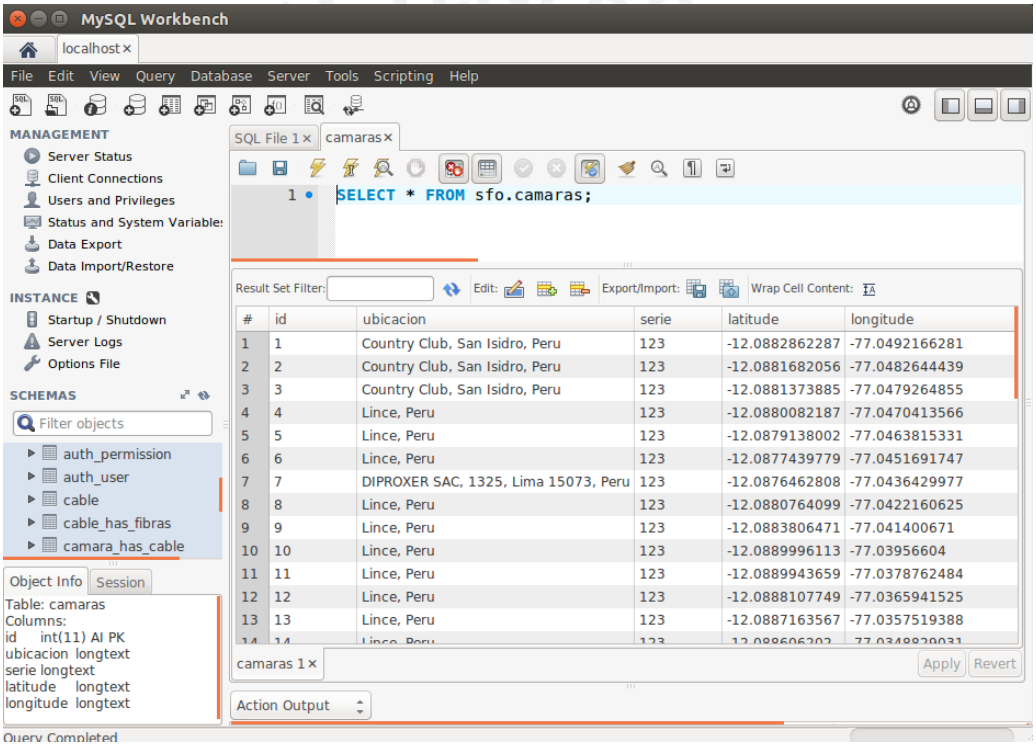


Figura 3.18. Interfaz gráfica de administración MySQL® Workbench [Elaboracion propia]

En el presente trabajo, MySQL® Server cumple el rol principalmente de almacenamiento y gestión de la base de datos utilizada por el programa, si bien el framework web2py posee su propio gestor para estos recursos, a modo de seguir las buenas prácticas de diseño de un sistema, se optó por separar esta característica para hacerla sea independiente del framework, es decir, que se pueda dar mantenimiento y gestión sin la necesidad de web2py; además nos da la ventaja de que esta información pueda ser reutilizada en el futuro por otros

sistemas y no tenga que ser exportada manualmente. Por otra parte, el gestor de base de datos de web2py posee una interfaz poco intuitiva para la administración y cuenta con funciones básicas para realizar operaciones de CRUD, mientras que MySQL® Workbench nos da una gama de herramientas amigables para el usuario que sirven para el mismo propósito y facilitan el proceso.

Es importante resaltar que la capacidad de creación y eliminación de tablas de las base de datos se deja estrictamente en manos de web2py, debido a que crea archivos que se asocian a las tablas de las bases de datos de manera local a la hora de cargar la aplicación. De modo que si uno crease o eliminase una tabla desde MySQL®, se crearía un conflicto en el framework al momento de ejecutar la aplicación, ya que el archivo local asociado a dicha tabla o bien no fue creado o no fue eliminado y se traducirá en un error.

3.2.4 Rocket-Fast System for Log Processing (RSYSLOG)

RSYSLOG es un conjunto de herramientas para el registro y procesamiento de eventos que puede ser altamente personalizado de acuerdo al flujo de la información y mensajes [32], es compatible con múltiples tecnologías y con sistemas operativos como Windows, esto se puede observar en detalle en la Figura 3.19 . Está basado en el estándar *Syslog* [33] para el envío de mensajes en una red IP de modo que pueda identificarse los eventos de manera inequívoca y se envíe información relevante por cada elemento de la red.

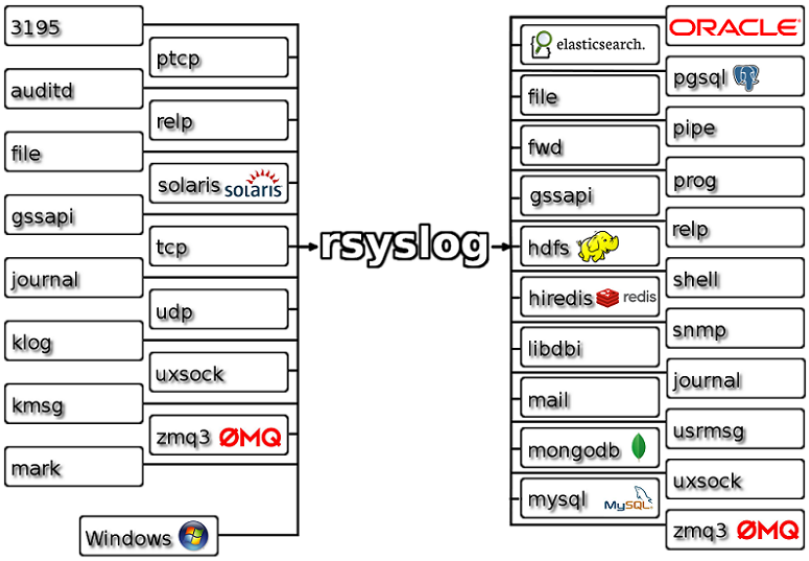


Figura 3.19. Árbol de compatibilidad y características de RSYSLOG [32].

Su rol en el sistema es el de encargarse del registro del estado de los enlaces de la red y almacenarlos en la base de datos para ser utilizados en el proceso de detección de averías. Para este objetivo, es necesario implementar una base de datos adicional a la usada por la aplicación web como podemos ver en la Figura 3.20 , de modo que aprovechamos no sobrecargar la base de datos que contiene la información de la red y evitamos afectar su desempeño al tener que buscar trabajar con grandes cantidades de datos.

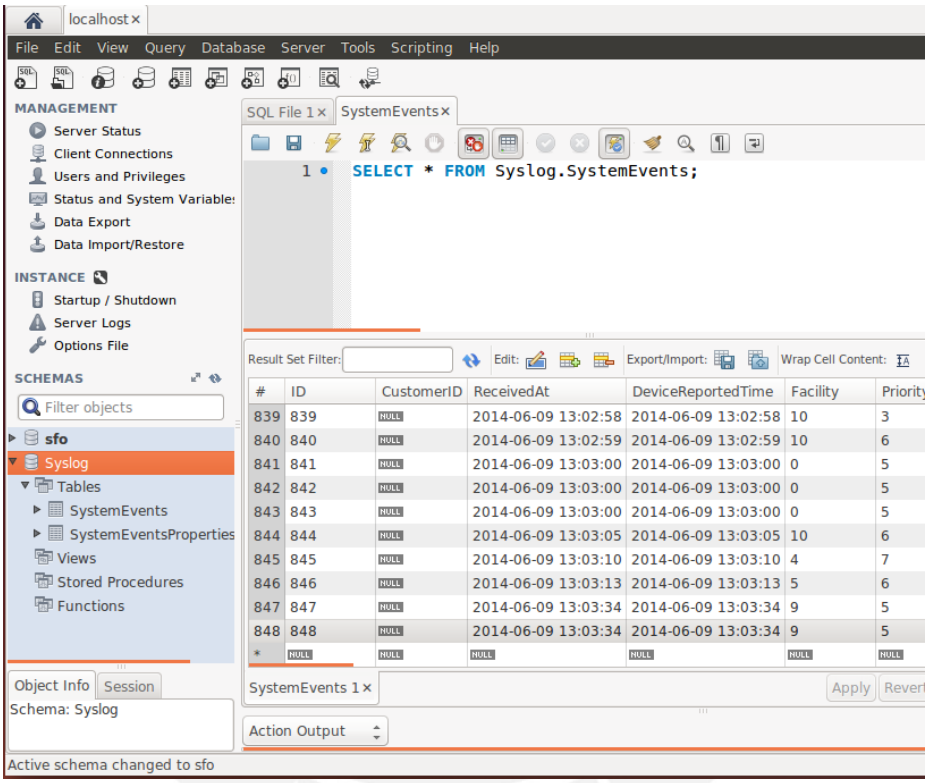


Figura 3.20. Vista del esquema de base de datos "Syslog" utilizado por RSYSLOG para almacenar los mensajes del estado de los enlaces en la red [Elaboracion propia].

3.3 Implementación del sistema

Para la implementación se utilizó un servidor con el sistema operativo Ubuntu Server 14.04 LTS, sobre el cual se comenzó por desplegar todos los componentes del *backend*: instalar web2py, sus dependencias, implementar la base de datos MySQL e implementar el *RESTful Web Service*; realizando las pruebas de interacción necesarias para comprobar el correcto funcionamiento de cada uno, ya que se trata del núcleo del sistema y sin este no se podría desplegar el *frontend.* Una vez terminadas las pruebas, se procedió a continuar con el *frontend* que se conforma principalmente de las vistas HTML, *AngularJS* y

Angular-Google-Maps para la visualización de los mapas y elementos de la red. A continuación, se detalla cómo es que interactúan el *backend* y *frontend* tanto a nivel interno como a nivel externo.

3.3.1 Backend

Tal y como observamos en la Figura 3.21, para implementar el servidor web es necesario iniciar web2py, por ello se utilizó un *script* de configuración inicial brindado por el mismo *framework* que despliega todas las dependencias necesarias para web2py, así como la configuración del servidor HTTP NGINX para acceder a la aplicación través del puerto 80. Una vez completado el despliegue, el framework se encarga establecer conexión contra MySQL® Server en consecuencia a la base de datos SFO ya través del puerto 3306, luego el *RESTful Web Service* será implementado a través de un controlador a nivel de web2py para realizar las peticiones asíncronas de datos que se utilizarán para desplegar las vistas o guardar información a pedido del usuario. Así mismo, rsyslog se despliega unilateralmente en el sistema operativo de modo que obtenga la información del estado de los enlaces de la red vía mensajes *syslog*, de modo que habilita el puerto 514 para la captura y luego los escriba directamente en la base de datos designada en el MySQL® Server.

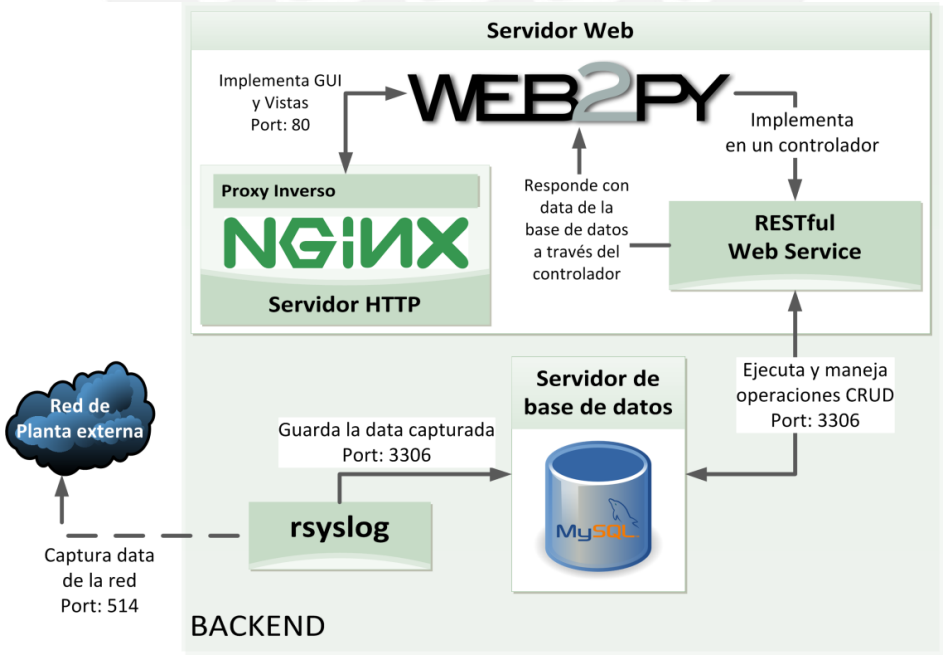


Figura 3.21.Comunicación interna entre el servidor HTTP, RESTful Web Service, rsyslog y el servidor de base de datos [Elaboracion propia].

3.3.2 Frontend

Para el escenario que vemos en la Figura 3.22 se desplegó *AngularJS* a nivel de web2py, es decir, dentro su repositorio de archivos estáticos de modo que puedan ser utilizados de manera local en las vistas, cada vista obtiene desde el servidor web, un módulo que instancia todos los elementos creados en *AngularJS*, un controlador para habilitar las funciones agregadas, un servicio que obtiene la información usada por la vista a través del *RESTful Web Service* y la directiva *Angular-Google-Maps* de ser necesaria para mostrar la información de red en un mapa de calles de *Google Maps*®. Por otro lado, cuando se trata de las vistas que permiten ingresar información nueva de la red como las funcionalidades "Agregar Clientes" y "Agregar Elementos de Red", el mismo *AngularJS* mediante el uso del de una función en el controlador llama al servicio que obtiene la información, pero para realizar el guardado de los datos también a través del *RESTful Web Service.* Una vez concluida la operación, el controlador actualizará automáticamente la información en pantalla.

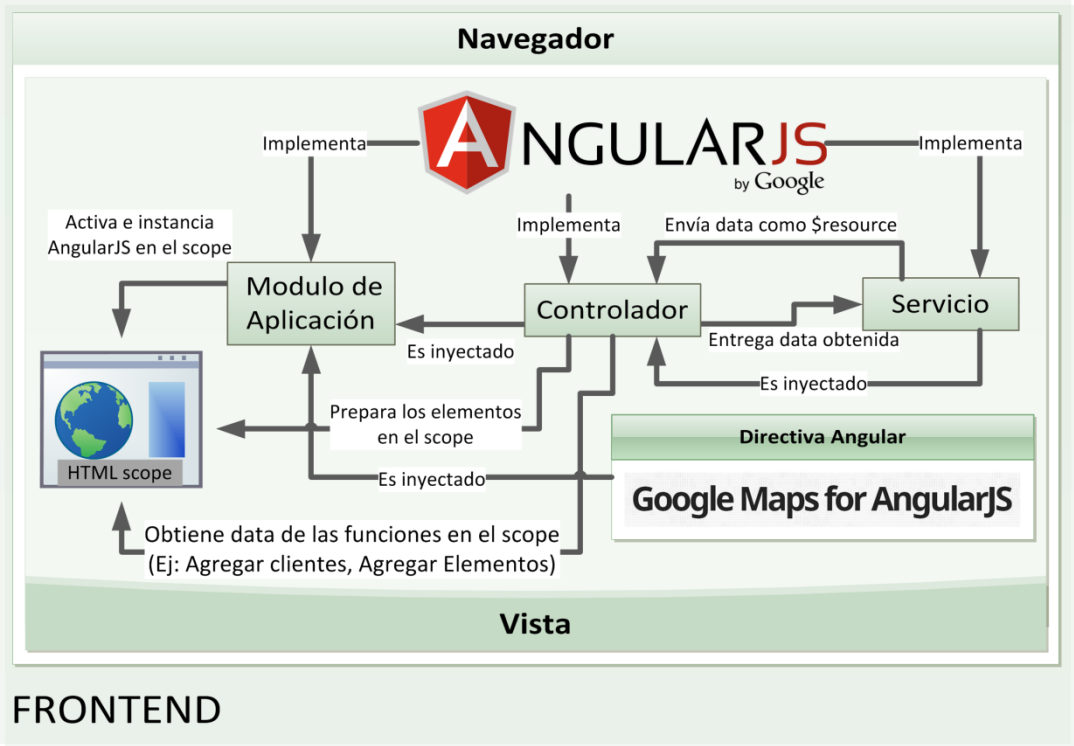


Figura 3.22. Comunicación interna entre los componentes de AngularJS, la directiva Angular-Google-Maps y el scope de la vista HTML [Elaboración propia]

Finalmente ambos extremos en conjunto se conectan según lo observado en la Figura 3.23:

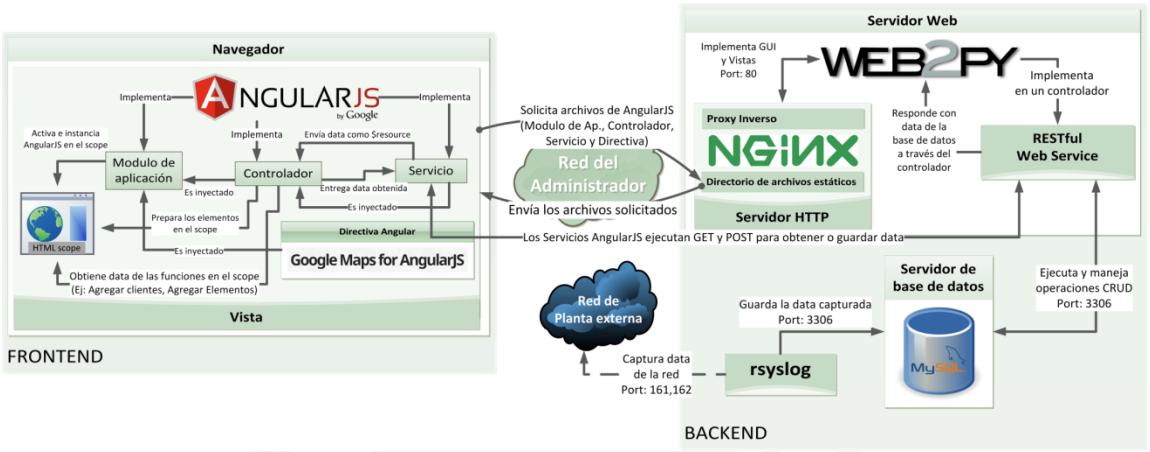


Figura 3.23. Comunicación entre el frontend y el backend a través de la red del administrador del sistema
[Elaboracion propia]

3.3.3 Funcionalidades del sistema

3.3.3.1 Registrar usuario

Esta funcionalidad se implementó utilizando la plantilla de inicio de sesión de *web2py* que utiliza el formulario "*Auth()*", autogenerado por el mismo framework, que valida los campos de la tabla "*auth_user*" de la base de datos al habilitar el "decorador" de autenticación en el controlador de las vistas (ver Figura 3.24). Se tiene un botón llamado "*Register*" en la interfaz de inicio de sesión, el cuál llama a la siguiente vista que muestra el formulario "*Auth()"* donde se aprecian los campos de usuario, contraseña, datos personales, grupo o rol (Operador o Especialista), etc. pertenecientes a la tabla antes mencionada, de modo que una vez llenados, se presiona el botón "*Register"* y este envía todos los datos ingresados a la base de datos para ser almacenados en la tabla "*auth_user*" (ver Figura 3.25). Una vez guardados, el usuario puede ser autenticado a través de la interfaz de inicio de sesión.

```
@auth.requires_login()
def index():
    return locals()

def user():
    return dict(form=auth())

@auth.requires_login()
def netmap():
    return locals()

@auth.requires_login()
def alarmclient():
    return locals()

@auth.requires_login()
def camxclient():
    return locals()

@auth.requires_login()
def cablemap():
    return locals()
```

Figura 3.24. Contenido de controlador de web2py "api.py" que contiene las vistas a implementar junto con el "decorador" que solicita la autenticación de usuario para acceder a las mismas
[Elaboración propia]

Figura 3.25. Diagrama de funcionamiento de la funcionalidad "Registrar usuario"
[Elaboración propia]

3.3.3.2 Elementos de la red

La presente funcionalidad utiliza el controlador "api.py" para ser implementada por *web2py*, a su vez los componentes utilizados como *Google Maps*®, y las funciones de *AngularJS* para agregar un elemento de red y visualizar los datos de los elementos, son invocados desde la vista para habilitar su funcionamiento y permitir observar la información respectiva en tiempo real. Luego, mediante el módulo de aplicación "angularElemApp.js" de *AngularJS* es posible habilitar las funciones definidas a nivel del controlador "angularElemCtrl.js", las cuales junto con la utilización del servicio "angularElemService.js" para obtener la información

de la tabla “camaras”, que contiene la información de los elementos de la red, a través del *RESTful Web Service,* que establece una sesión contra la base de datos “Sfo” la cual contiene la tabla antes mencionada, permiten llenar la tabla con datos de los elementos de la red, cargar los parámetros básicos del mapa (centro, tipo de mapa, zoom, etc.), asignar la posición de los marcadores que representan los elementos de la red en el mapa y agregar nuevos elementos de la red a través de un formulario que se llena con datos obtenidos del mapa al hacer clic en él (ver Figura 3.26).

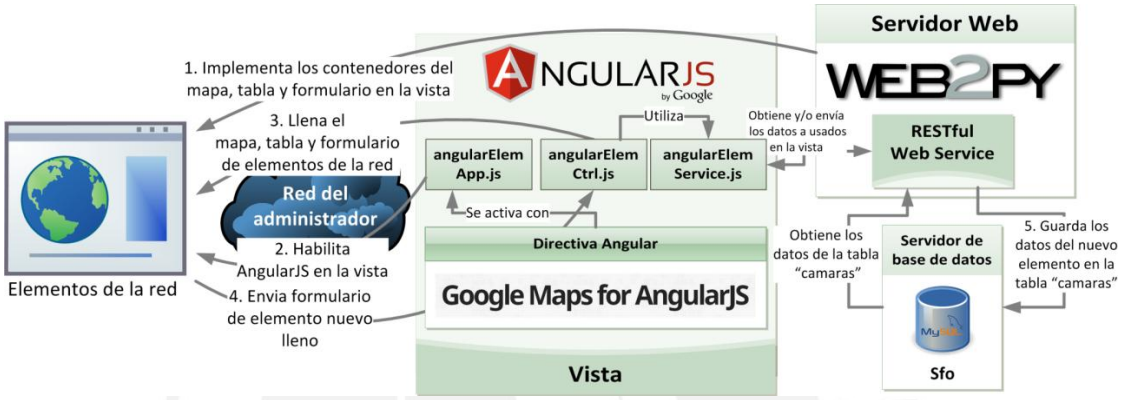


Figura 3.26. Diagrama de funcionamiento de la funcionalidad "Elementos de la red"
[Elaboración propia]

3.3.3.3 Clientes

Esta funcionalidad también es implementada a través del controlador de *web2py* "api.py", utiliza el *RESTful Web Service* también conectado a la base de datos "Sfo" y, de manera similar a la funcionalidad anterior, los componentes "angularClientApp.js", "angularClientCtrl.js" y "angularClientService.js" (módulo de aplicación, controlador y servicio de *AngularJS*, respectivamente) son utilizados para habilitar el mapa, los marcadores que representan a los clientes, el formulario para agregar nuevos clientes y la tabla con información de los clientes en la vista, todo lo anterior con los datos obtenidos de la tabla "cliente" (ver Figura 3.27).

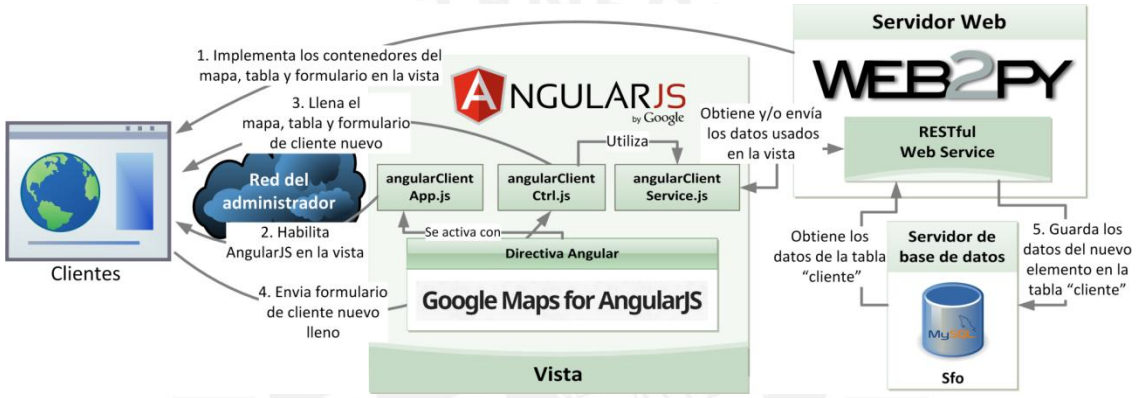


Figura 3.27. Diagrama de funcionamiento de la funcionalidad "Clientes" [Elaboración propia]

3.3.3.4 Enlaces en la red

Al igual que las demás funcionalidades, el controlador de *web2py* "api.py" implementa esta funcionalidad, a su vez utiliza los componentes de *AngularJS* "angularCamClientApp.js", "angularCamClientCtrl.js" y "angularCamClientService.js" al igual que el *RESTful Web Service* para habilitar el mapa, los marcadores que representan los clientes y elementos de la red y la tabla con información de los clientes con sus respectivos elementos de red, similar a las funcionalidades anteriores (ver Figura 3.28). En adición a las tareas anteriores, se creó un botón desplegable dentro de la tabla con información de clientes, para cada cliente, llamado "Opciones" que contiene los botones "Ver ruta" y "Ocultar ruta"; el primero se encarga de mostrar la ruta que sigue el enlace de un cliente a través de los elementos asociados al mismo, para ello llama a la función de controlador de *AngularJS* que obtiene los elementos involucrados en el enlace del cliente mediante la asociación del campo "*id"* del cliente contra el campo "*id"* de los elementos a través de la tabla "cliente_has_camaras", de modo que una vez obtenidos los "*id*" de los elementos relacionados, otra función se encarga de comparar los mismos con los "*id's*" de los elementos de la red obtenidos de la tabla "camaras". Los que concuerden serán los que conforman la ruta del enlace, de los cuales se utilizará su posición (latitud y longitud) para realizar el trazado de una polilínea que tendrá como puntos de unión a dichos elementos seleccionados. Estos puntos serán almacenados en el atributo "*path*" de la variable "*line*", la cual será utilizada por la directiva *Angular Google Maps* para mostrar el trazado en el mapa. Y el segundo, se encarga de vaciar el atributo "*path*" de modo que no posea valores de latitud y longitud, lo que por ende ocasiona que no exista la línea.

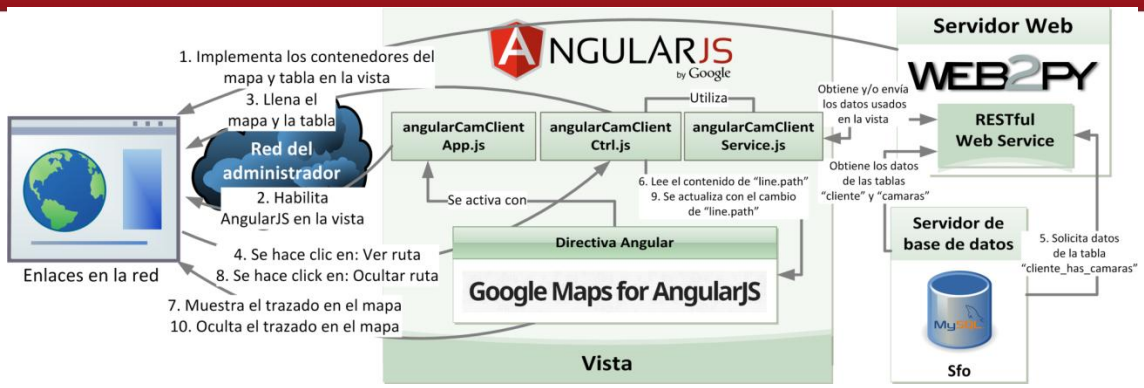


Figura 3.28. Diagrama de funcionamiento de la funcionalidad "Enlaces en la red" [Elaboración propia]

3.3.3.5 Cables de fibra óptica

En esta funcionalidad se utilizó tanto el controlador “api.py”, como el *RESTful Web Service* en conexión con la base de datos “Sfo” y los componentes de *AngularJS* “angularCableApp.js”, “angularCableCtrl.js” y “angularCableService.js”, para mostrar los elementos de la red involucrados en la extensión de los cables, estos elementos fueron obtenidos de la tabla “camaras”. Así mismo, la lista de cables disponibles que aparece en pantalla fue obtenida de la tabla “cable”, donde el código de cable que aparece en la lista, es un botón desplegable que posee dos opciones: “Ver ruta” y “Ocultar ruta”, estos dos últimos funcionan bajo la misma lógica que en la funcionalidad anterior, el primero llama a la función que obtiene la los elementos de la red relacionados al cable a través de la asociación del campo “*id*” del cable contra el campo “*id*” de los elementos de la red según la tabla “camara_has_cable”, que luego de hacer la comparación de los datos obtenidos contra los elementos de la red, guarda los valores de latitud y longitud en el atributo “*path*” de la variable “*line*” para dibujar y mostrar el trazado del cable de fibra óptica en el mapa a través de la directiva *Angular Google Maps*. Y de igual forma el segundo, se encarga de vaciar la variable que permite visualizar el trazado sobre el mapa (ver Figura 3.29).

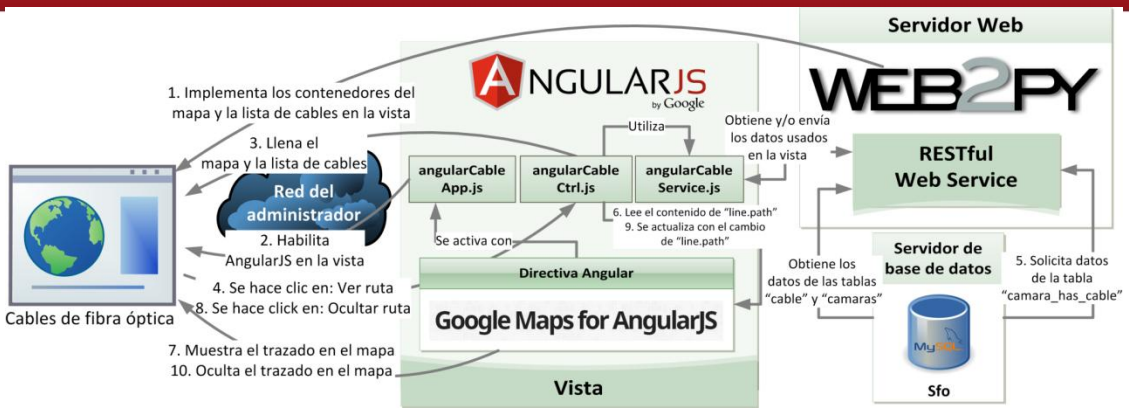


Figura 3.29. Diagrama de funcionamiento de la funcionalidad "Cables de fibra óptica"
[Elaboración propia]

3.3.3.6 Alertas

En esta funcionalidad se utilizó de igual forma el controlador “api.py”, el primer cambio es la conexión con la base de datos “*Syslog*”, que es la que contiene los registros del estado de los enlaces de la red que fueron almacenados mediante *rsyslog* en la tabla “*SystemEvents*”, luego de capturar y procesar los mensajes transmitidos al puerto 514 desde los *routers* u otros equipos de la red. Por otro lado, para realizar el proceso de detección de averías fue necesario adicionar un complemento de *web2py* conocido como “scheduler.py”, con el cual es posible realizar un *polling* constante cada 5 segundos a la base de datos “Syslog” en búsqueda de nuevos registros con estado de enlace “*down*”(siguiendo es estándar de *syslog)* dentro del campo “*Message*”. De modo que, de encontrar que se cumpla la condición, se pasa a ejecutar la lógica de localización de la avería, que comienza con ejecutar la función dentro de “*scheduler.py”* que obtiene el cliente con la avería y la ruta de su enlace, los clientes que pertenecen al mismo cable de fibra óptica también con sus rutas de enlace y las alertas relacionadas a estos últimos. Si hubiese alguna alerta de los demás clientes dentro del mismo intervalo de tiempo que el que presentó la avería inicialmente, la función compara la longitud de cada ruta de enlace (cantidad de elementos por enlace de cliente) de mayor a menor donde se intersectan sus rutas y determina que el tramo (par de elementos de red) anterior a la intersección es donde se encuentra la avería. Una vez determinado el tramo de la falla, este es almacenado en la tabla “schedule_run” desde donde los componentes de *AngularJS*: “angularAlertApp.js”, “angularAlertCtrl.js” y “angularAlertService.js” se encargaran de la obtención e impresión de la información de la averías en pantalla, es decir, de la visualización del mapa con elementos de la red junto con el cliente involucrado y el cable cuyo enlace se encontraba averiado, así como la posición de la avería marcada en color rojo entre el par de elementos de red determinado (ver Figura 3.30).

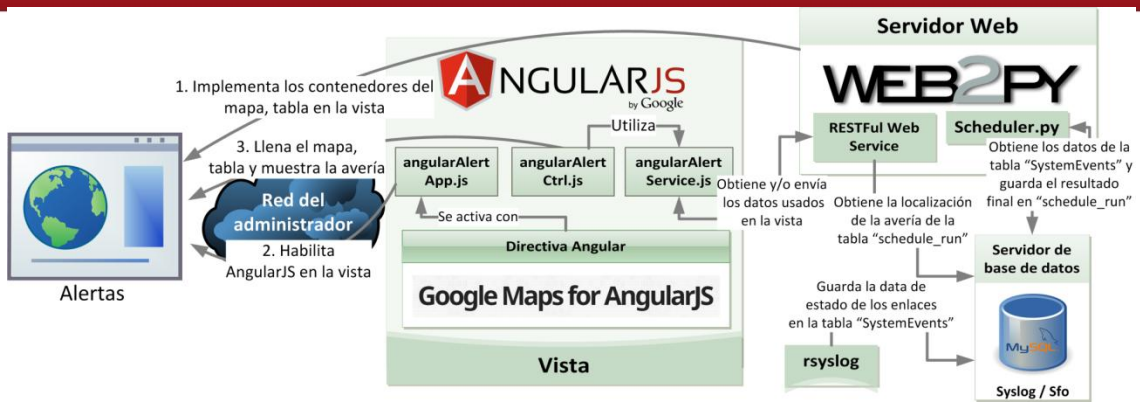


Figura 3.30. Diagrama de funcionamiento de la funcionalidad "Alertas" [Elaboración propia]

CAPITULO IV
PRUEBAS Y RESULTADOS

En el presente capítulo se presenta el prototipo del sistema basado en el diseño elaborado en el capítulo anterior, así como las pruebas de concepto realizadas para corroborar el correcto funcionamiento del prototipo implementado.

4.1 Interfaces del prototipo

4.1.1 Inicio de sesión

Para la interfaz de inicio de sesión se optó por un diseño sencillo y de fácil uso, por ello se utilizó el modelo predefinido de *web2py* que consiste en un formulario de usuario y contraseña para realizar *login*, así como las opciones de recuperar contraseña y registrar un usuario.

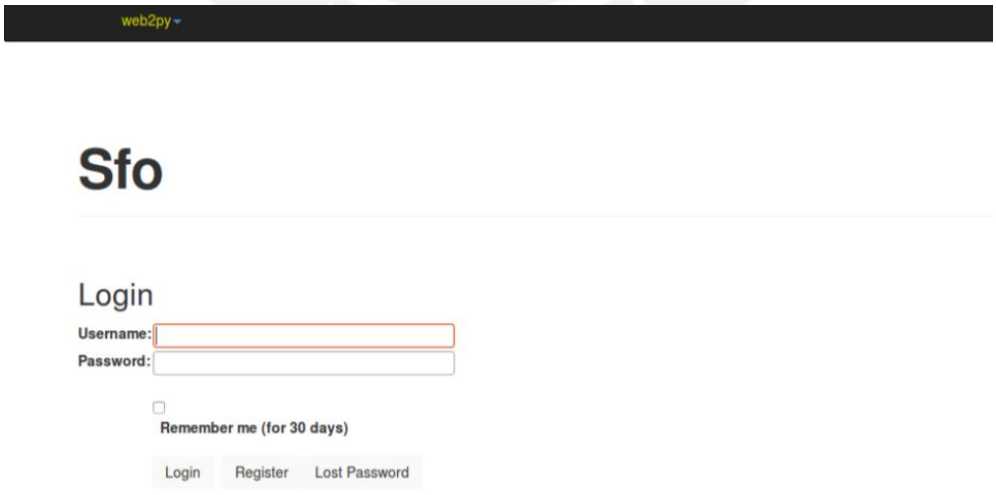


Figura 4.1. Interfaz de inicio de sesión
[Elaboración propia]

4.1.2 Registrar usuario

Para la interfaz de registro de usuario también buscó simplificar su uso y se utilizó el modelo de formulario predefinido por *web2py*. Este consiste en un campo de nombre, uno de apellido, de *e-mail*, de nombre de usuario, de contraseña y uno para verificar la contraseña (ver Figura 4.2).

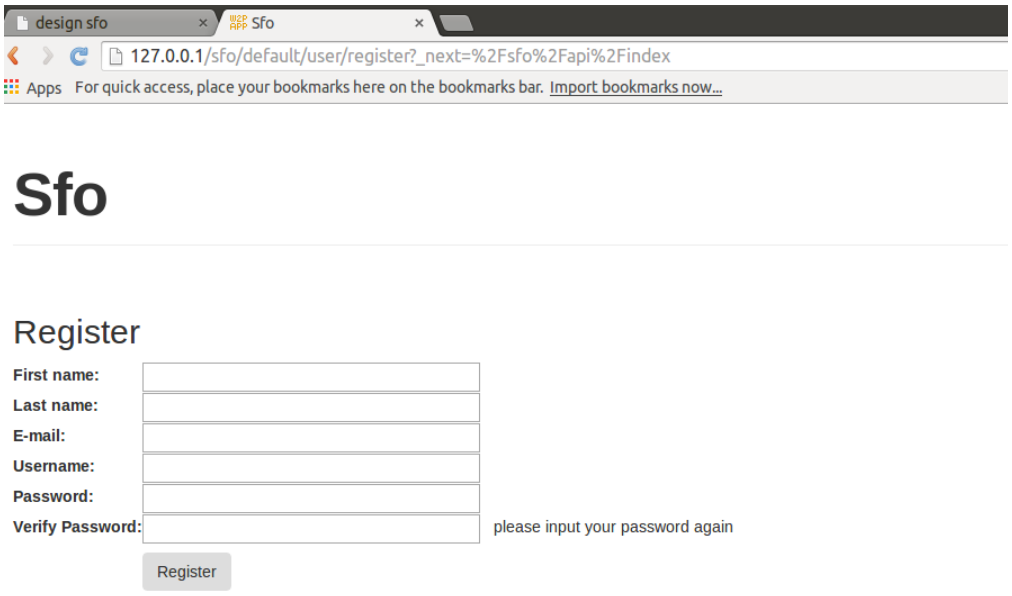


Figura 4.2. Interfaz de registro de usuario
[Elaboración propia]

4.1.3 Menú principal

Para la interfaz del menú principal (ver Figura 4.3) se buscó simplificar el acceso a las funcionalidades del sistema, al colocar en cada una ellas una descripción breve de lo que permiten visualizar junto con un botón con un color específico para una fácil identificación.

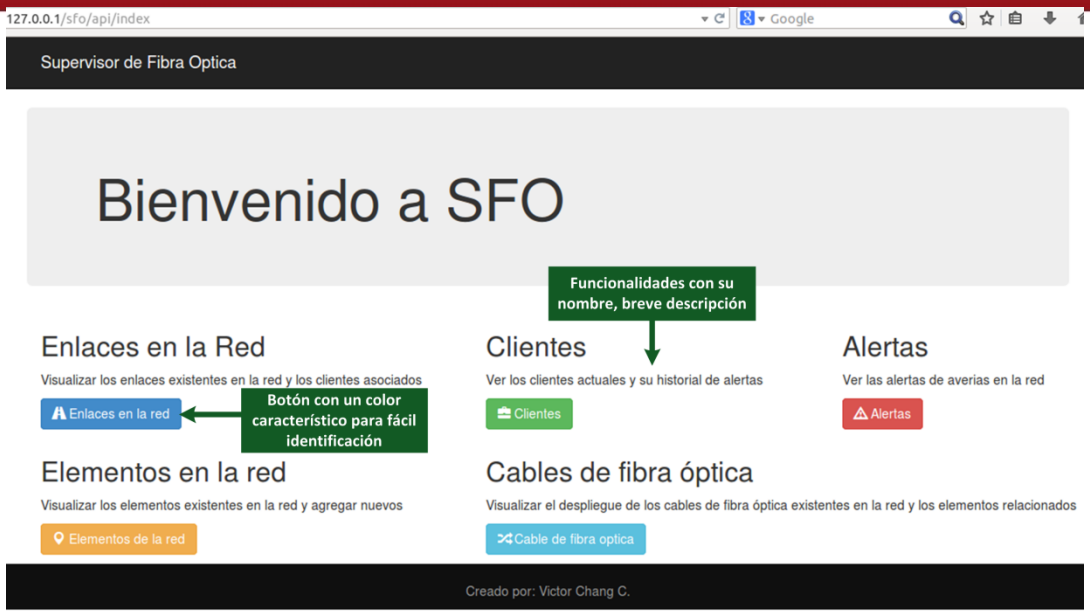


Figura 4.3. Interfaz del menú principal, con las cinco funcionalidades disponibles [Elaboración propia]

4.1.4 Elementos de la red

En la interfaz de la funcionalidad “Elementos de la red” (ver Figura 4.4), se utilizó un menú de navegación basado en el menú principal del sistema, el cual está presente en todas las vistas y cuenta con los mismos botones y títulos agrupados. Así mismo, cuenta con un *banner* que indica en que funcionalidad se encuentra el usuario y un botón para volver al menú principal. Respecto a la funcionalidad, se muestra la pestaña para “Agregar elemento de red” (solo para usuario Especialista), el mapa de calles donde aparecen los elementos de red actuales y el panel inferior donde aparecen los datos de todos los elementos, este panel cuenta con filtros de búsqueda y de ordenamiento de datos según lo indicado por el usuario.

Figura 4.4. Interfaz de la funcionalidad "Elementos de red": Vista inicial [Elaboración propia]

4.1.5 Clientes

Para la interfaz de la funcionalidad "Clientes" (ver Figura 4.5), se utilizó también el menú de navegación en la parte superior de la página, el banner que indica en que funcionalidad se está visualizando, el mapa con los clientes presentes en la red y el panel inferior que cuenta con la tabla con información relevante de los clientes, de igual forma que en la interfaz anterior, el panel cuenta con un filtro de búsqueda y de ordenamiento de los datos según lo elegido por el usuario. Además, también se aprecia el botón "Agregar cliente de la red" que despliega el formulario para llenar los datos del nuevo cliente (solo usuario Especialista).

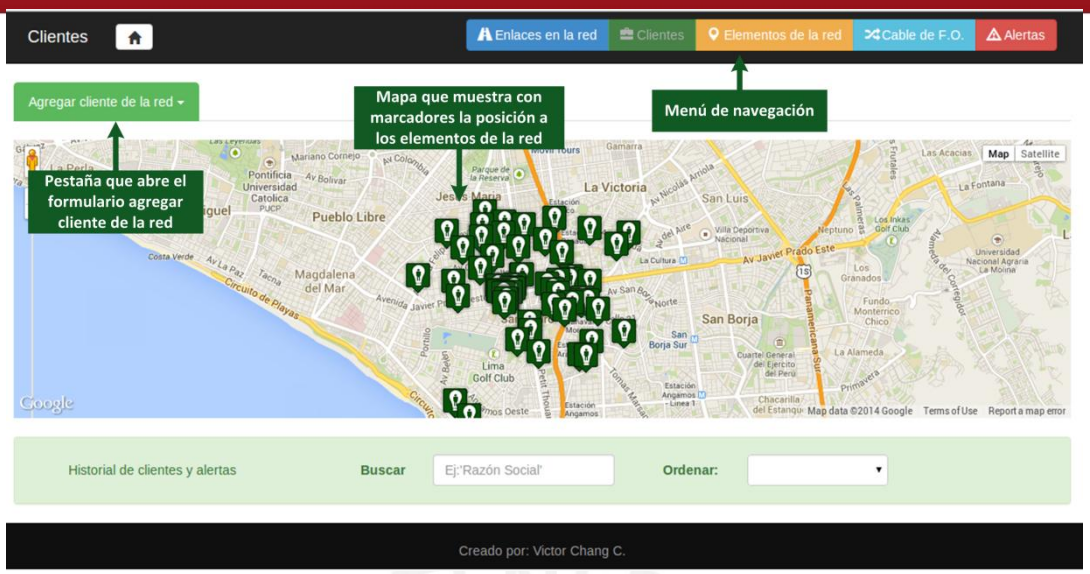


Figura 4.5. Interfaz de la funcionalidad "Clientes": Vista inicial [Elaboración propia]

4.1.6 Enlaces en la red

La interfaz de la funcionalidad “Enlaces en la red” (ver Figura 4.6), se muestra el menú de navegación en la barra superior, el *banner* con el nombre de la funcionalidad en la que nos encontramos, el mapa de calles con los elementos y los clientes presentes en la red y el panel que cuenta con la tabla con la información relevante de los clientes, el panel también cuenta con filtro de búsqueda y de ordenamiento de datos según lo indicado por el usuario.

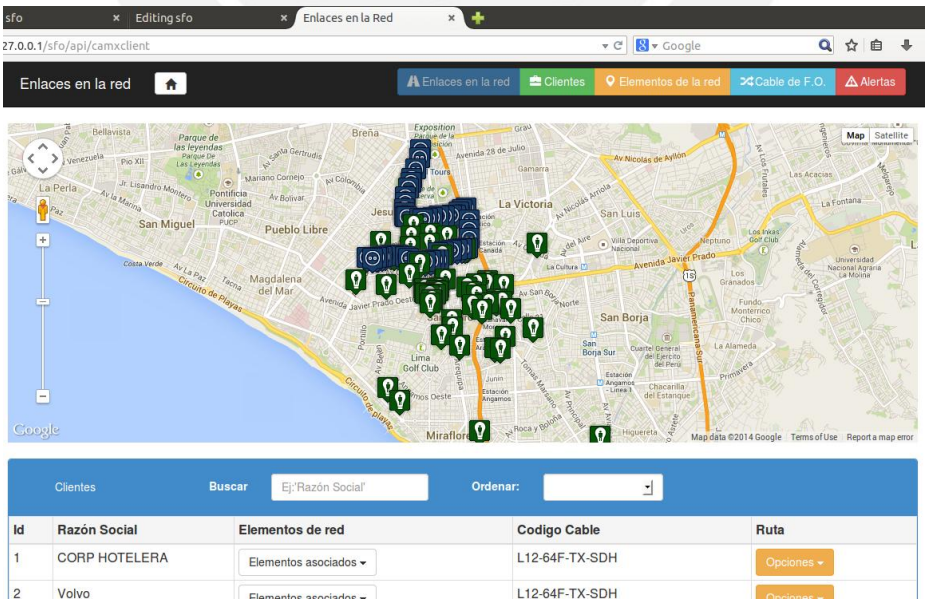


Figura 4.6. Interfaz de la funcionalidad "Enlaces en la red": Vista inicial [Elaboración propia]

4.1.7 Cables de fibra óptica

En la interfaz de esta funcionalidad (ver Figura 4.7) se cuenta también con el menú de navegación en la parte superior, el *banner* respectivo, el mapa junto con los elementos de la red que conforman la ruta que siguen los cables, el cuadro de texto que muestra los elementos asociados a cada cable y el panel inferior que contiene la lista de cables de fibra óptica.

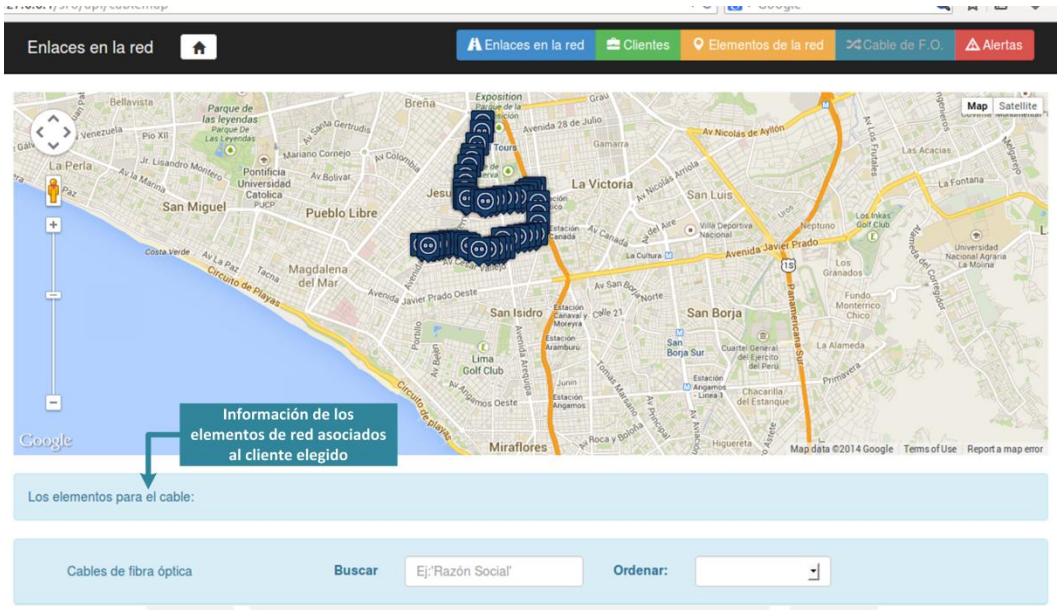


Figura 4.7. Interfaz de la funcionalidad "Cables de fibra óptica": Vista inicial [Elaboración propia]

4.1.8 Alertas de la red

De igual forma, la interfaz de esta funcionalidad (ver Figura 4.8), contiene los mismos elementos que las anteriores: menú de navegación, mapa para mostrar las averías, panel con tabla con información de las alertas junto con sus opciones de filtrado y ordenamiento. Inicialmente se observa que el mapa está vacío puesto que no hay ninguna avería detectada.

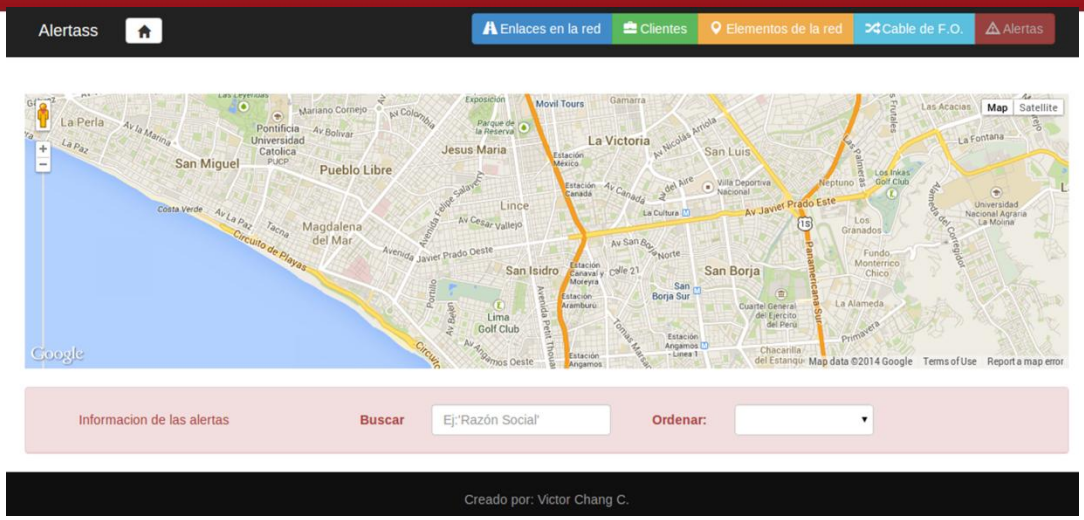


Figura 4.8. Interfaz de la funcionalidad "Alertas": Vista inicial sin alertas [Elaboración propia]

4.2 Pruebas de funcionamiento

4.2.1 Registrar usuario e Inicio de sesión

El registro de usuario puede ser realizado directamente por el mismo personal en cuestión que desea utilizar la aplicación, sin embargo, la asignación de permisos es realizada por el administrador del sistema. Este último se encarga de asociar a nivel de la base de datos el rol del usuario que acaba de registrarse, de modo que le habilita el acceso a las funcionalidades según su rol. En la Figura 4.9 podemos observar que se ha registrado un usuario "*operatortest*" al cual se le asignará el rol de "Operador" tal y como observamos en la Figura 4.10. Una vez asignado el rol, el usuario ya puede probar ingresar a la funcionalidad cables de fibra óptica (ver Figura 4.11), pero si quisiera entrar a la funcionalidad de elementos de red que cuenta con el formulario para agregar nuevos elementos, se le denegará el acceso (ver Figura 4.12).

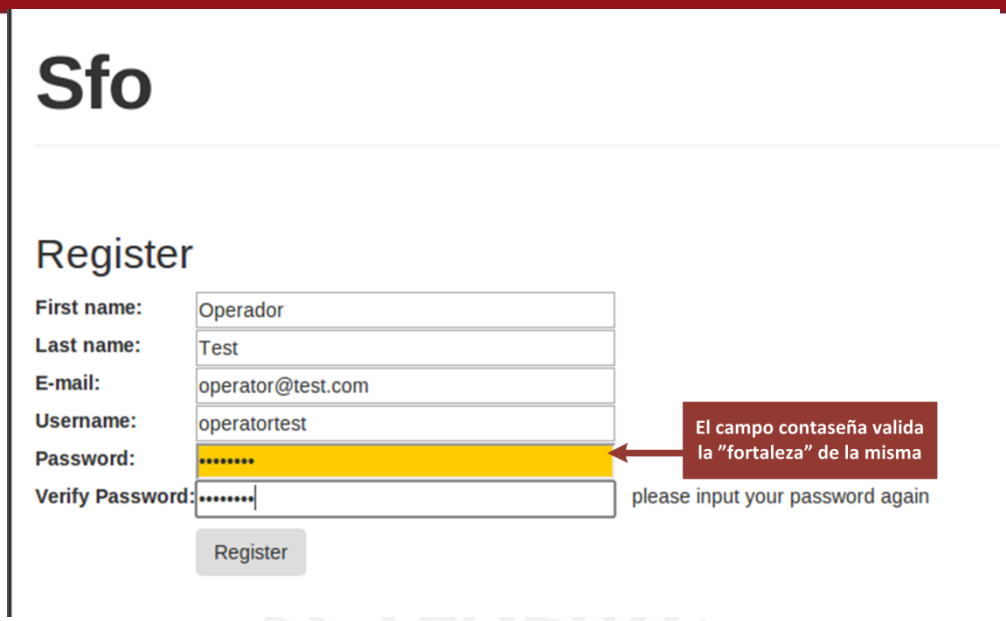


Figura 4.9. Formulario de registro del usuario Operador en el sistema [Elaboración propia]

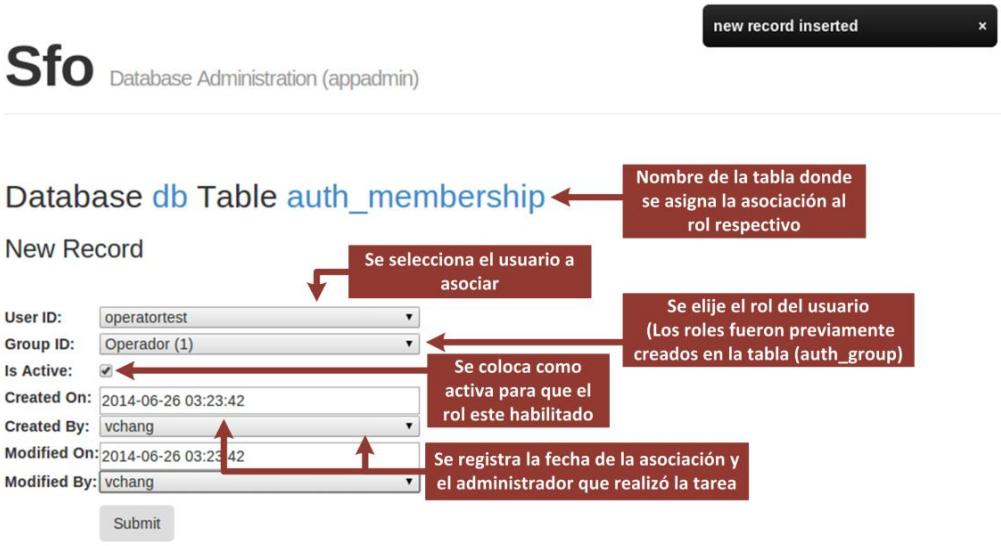


Figura 4.10. Formulario para ingresar un nuevo registro en la tabla "auth_membership" para asignar el rol respectivo a un usuario [Elaboración propia]

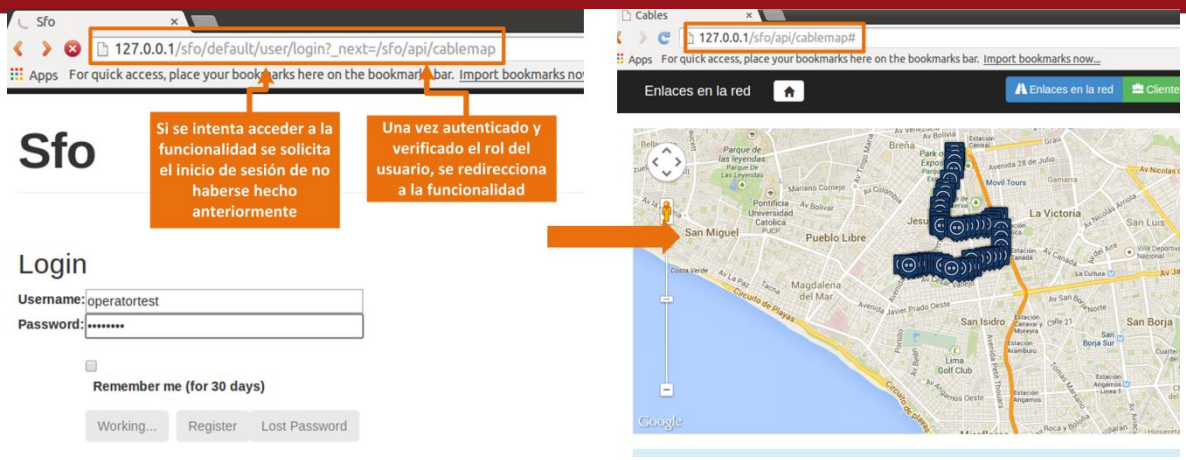


Figura 4.11. Proceso de inicio de sesión y verificación de rol para acceder a una funcionalidad
[Elaboración propia]

Figura 4.12. Proceso de denegación de acceso a una funcionalidad según el rol del usuario
[Elaboración propia]

4.2.2 Visualizar y agregar elementos de la red

Como podemos ver en la Figura 4.13, la interfaz nos muestra los elementos de la red disponibles como marcadores azules en el mapa; en la parte inferior, dentro del panel, podemos ver la información de la tabla de elementos de la red cuyos datos han sido filtrados para ver los que contienen la letra “d” y están ordenados por el campo más reciente primero (el número mayor de ID en la base de datos implica que el campo fue el último o el más reciente en ser ingresado).

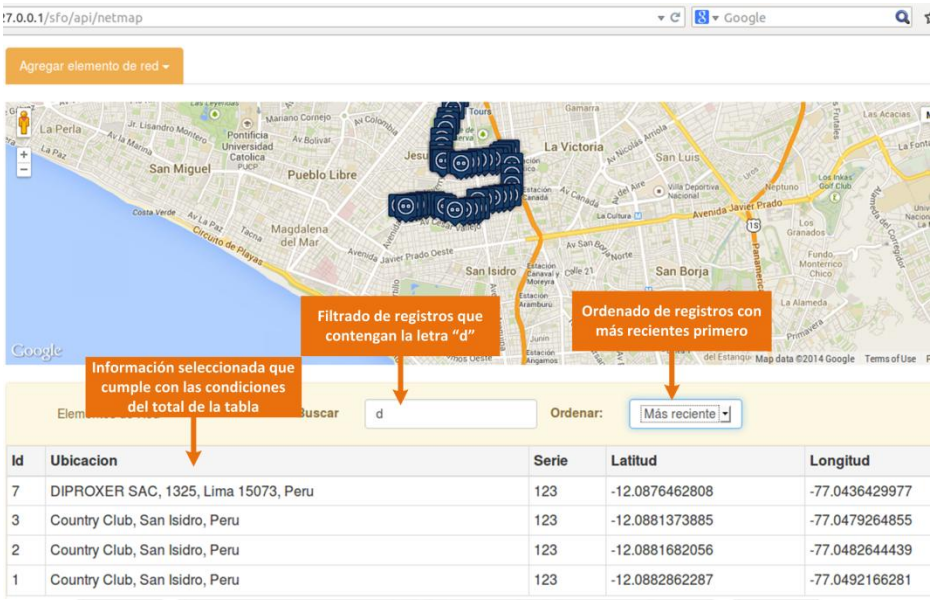


Figura 4.13. Funcionalidad "Elementos de la red" que muestra los datos de los elementos seleccionados según los filtros aplicados y la posición de todos los elementos de la red en el mapa.
[Elaboración propia]

Luego si hacemos clic en la pestaña “Agregar elemento de red” se desplegará el formulario que vemos en la Figura 4.14, este formulario se llena automáticamente con datos obtenidos del marcador creado al hacer clic en un punto en el mapa donde ubicaremos al nuevo elemento; una vez confirmados los datos se hace clic en el botón envíar para que sea almacenado en la base de datos e inmediatamente se actualizará en el mapa el marcador del nuevo elemento junto con su información en la tabla del panel inferior.

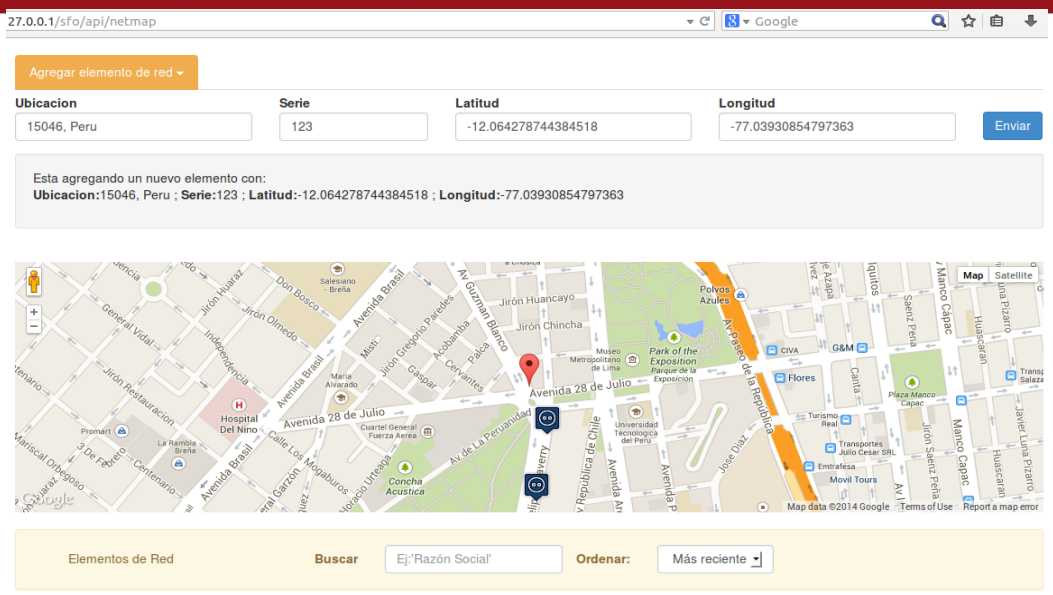


Figura 4.14. Vista de la función "Agregar elemento de red" dentro de la funcionalidad "Elementos de la red"
[Elaboración propia]

4.2.3 Visualizar y agregar clientes de la red

Tal y como observamos en la Figura 4.15, la interfaz nos muestra todos los clientes actuales de la red en forma de marcadores verdes sobre el mapa, así como los datos de los clientes que cumplen la condiciones de los filtros en el panel inferior.

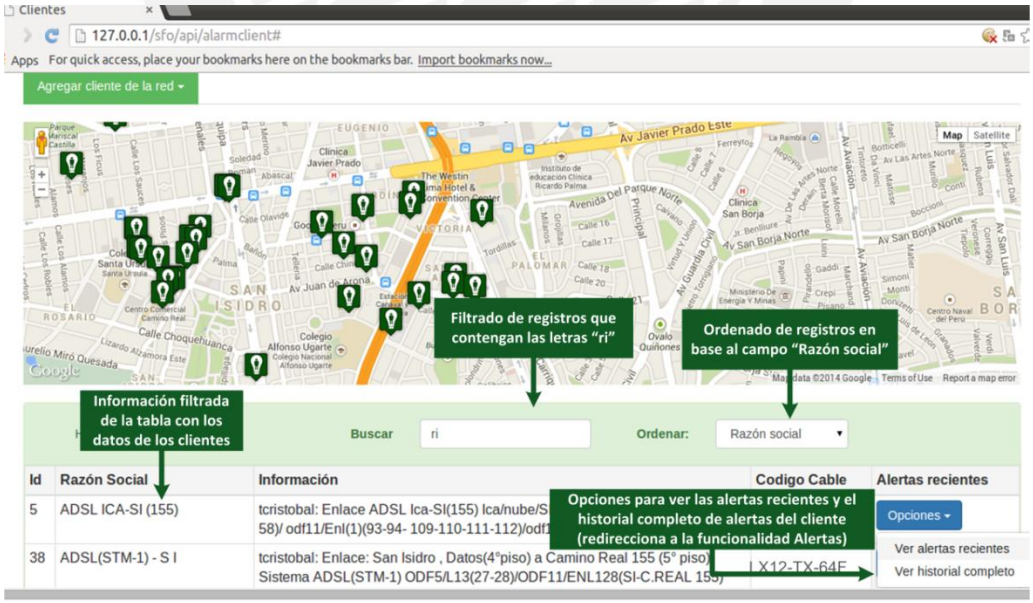


Figura 4.15. Funcionalidad "Clientes" que muestra los datos de los clientes seleccionados según los filtros aplicados y la posición de todos los clientes de la red en el mapa.
[Elaboración propia]

Luego en la Figura 4.16, podemos observar que luego de hacer clic en la pestaña “Agregar cliente de la red” se desglosa el formulario que también se llena de forma automática al hacer clic en el punto en el mapa donde queremos agregar al cliente. Los datos agregados son su posicionamiento (obtenido del marcador) y los campos Razón social, Información y código de cable, donde los datos que aparecen en estos últimos no son más que un autollenado que el usuario debe cambiar con la información del cliente respectivo. Una vez listo el formulario, se presiona el botón “Enviar” y automáticamente se actualiza el mapa con el marcador del nuevo cliente y su información en la tabla de contenido inferior.

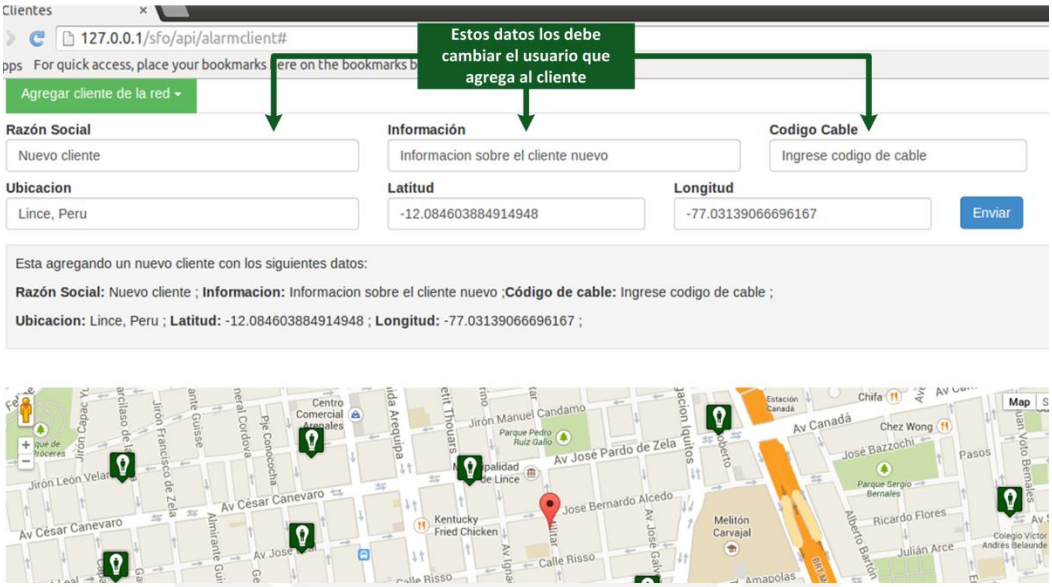


Figura 4.16. Vista de la función "Agregar cliente de la red" dentro de la funcionalidad "Clientes"
[Elaboración propia]

4.2.4 Visualizar enlaces en la red

En la Figura 4.17 podemos observar que se ha seleccionado el cliente Volvo, a través del filtro en el panel inferior y que se ha hecho clic en el botón “Elementos asociados” para mostrar los datos de los elementos de red por donde se despliega el enlace del cliente. Así mismo, de acuerdo con la vista inicial en el mapa se puede observar los clientes (color verde) y elementos de la red (color azul), basta con mantener el clic sobre el mapa y moverlo para ver a los demás marcadores.

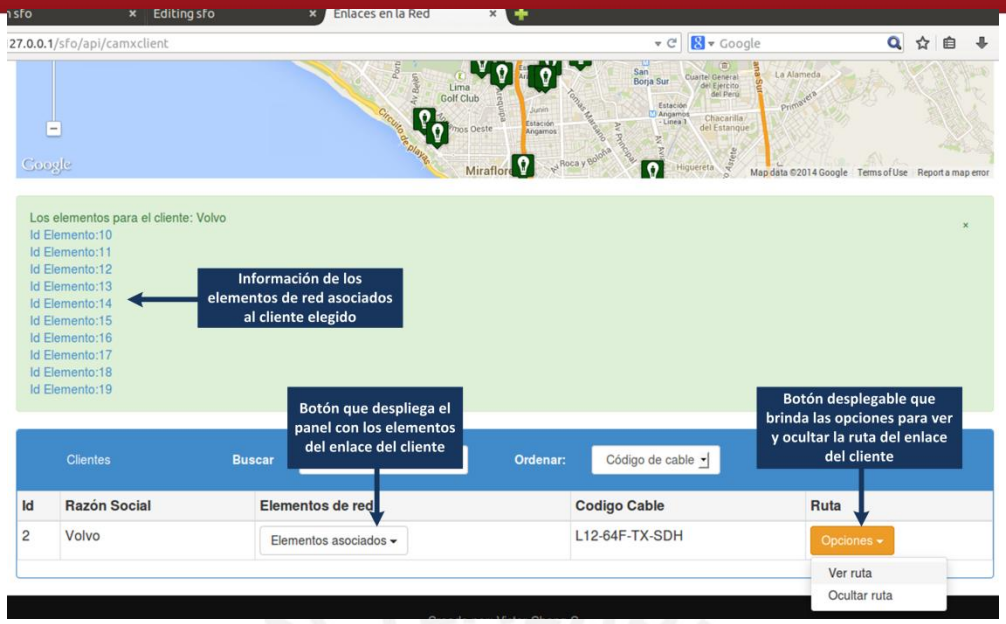


Figura 4.17. Funcionalidad "Enlaces en la red" que muestra los datos del cliente buscado, los datos de los elementos involucrados en su enlace. [Elaboración propia]

Si se hace clic en el botón “Ver ruta”, se mostrará lo que podemos ver en la Figura 4.18, que es el trazado de la ruta del enlace del cliente Volvo, donde se puede apreciar que el trazado pasa justamente por los elementos de red asociados al cliente que observamos en la Figura 4.17. Si se presiona la opción “Ocultar ruta” se obtendrá en pantalla lo que vimos previamente en la Figura 4.6.

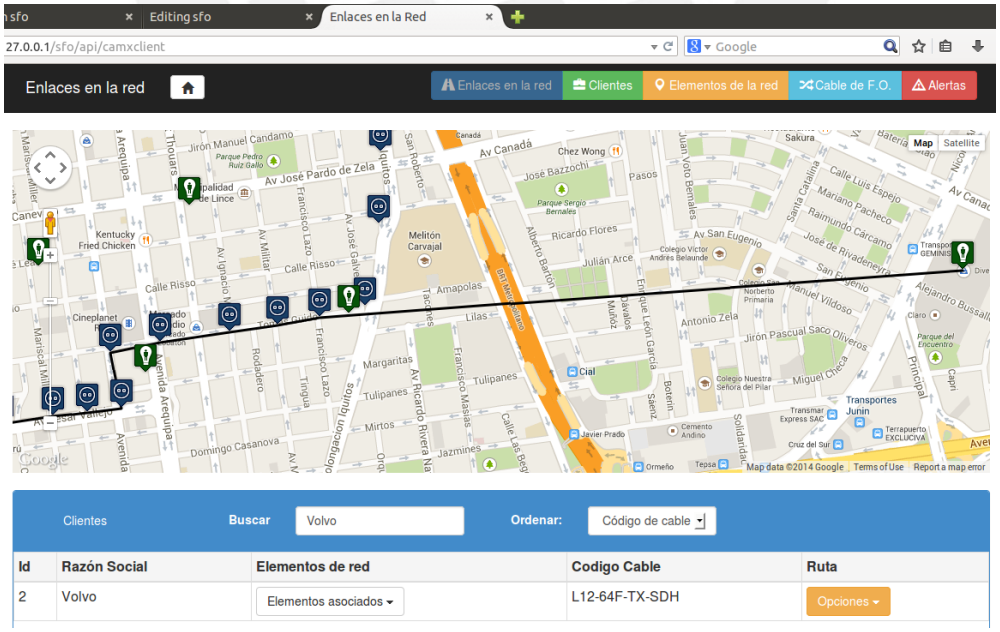


Figura 4.18. Visualización de la ruta del enlace del cliente Volvo mediante la función "Ver ruta" en las opciones del cliente. [Elaboración propia]

4.2.5 Visualizar cables de fibra óptica de la red

En la Figura 4.19 se puede observar la lista de cables, donde cada uno de los códigos de cable son un botón desplegable que permite elegir las opciones ver ruta y ocultar ruta del cable de fibra óptica, también es posible filtrar y/o ordenar la lista según un criterio dado, en este caso se filtra los cables que contenga los caracteres “L1” y se ordena por el más reciente.

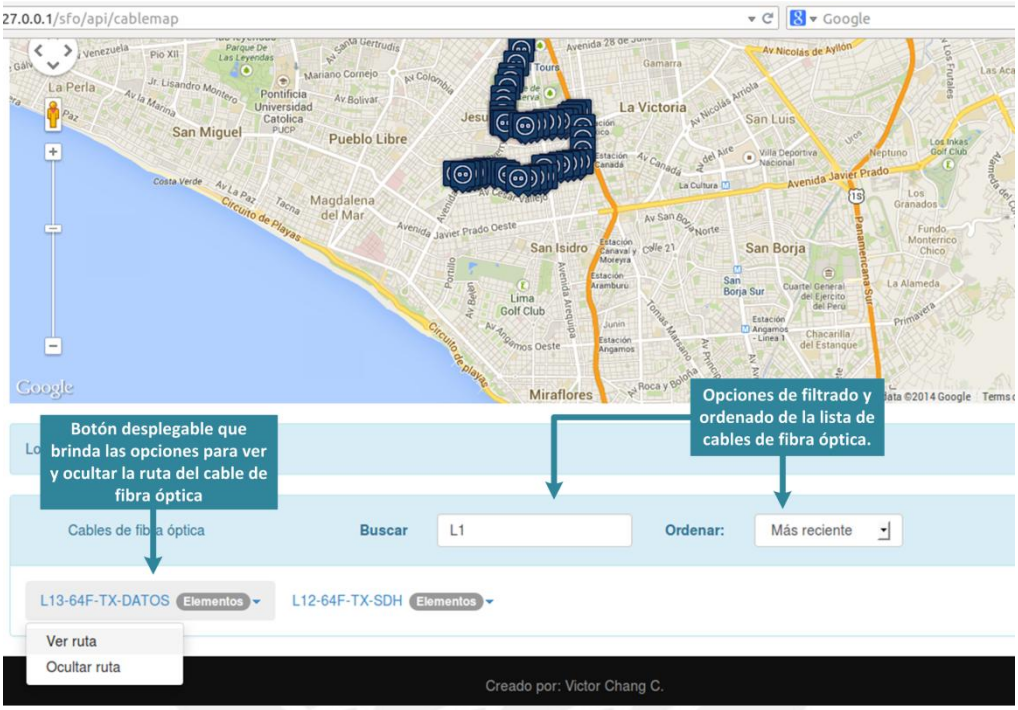


Figura 4.19. Visualización de la lista cables de fibra óptica según los criterios de filtrado y ordenado del panel inferior
[Elaboración propia]

Si se da clic a la opción “Ver ruta” se nos aparecerá lo que vemos en la Figura 4.20, el cuadro de texto encima del panel inferior nos mostrará los elementos de red asociados al cable de fibra óptica y a la vez se trazará la ruta del cable sobre los elementos de red asociados que visualizamos en el mapa. Y al igual que la funcionalidad enlaces de red, si presionamos la opción “Ocultar ruta” volvemos a la vista inicial de la interfaz que podemos observar en la Figura 4.7.

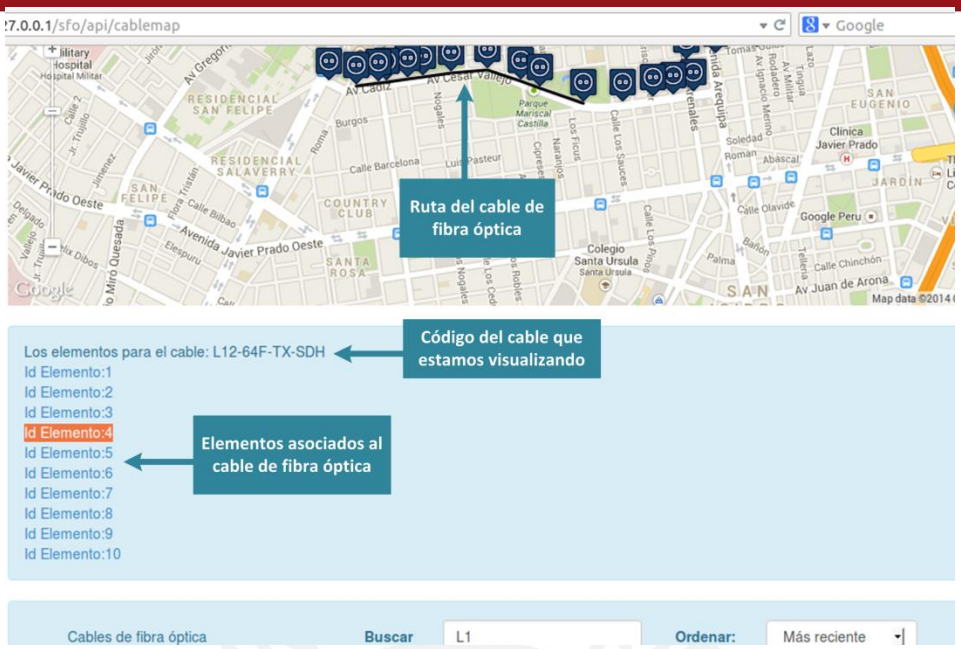


Figura 4.20. Vista del trazado de la ruta del cable de fibra óptica con código "L12-64F-TX-SDH" junto con sus elementos de red asociados.
[Elaboración propia]

4.2.6 Detectar y visualizar averías en la red

Cuando se detecte una avería ocurrirá primero el proceso de localización de la misma en segundo plano, una vez terminado se enviará la información a la vista para que esta pueda se mostrar en pantalla tal y como observamos en la Figura 4.21, primero se cargará en el mapa los marcadores de los elementos asociados (azules) y el cliente (verde) que presenta la avería, luego se mostrará la ruta del cable de fibra óptica asociado como un trazo en color negro y finalmente el tramo donde se encontró la avería se mostrará como una línea color rojo entre 2 elementos de la red, la cual estará superpuesta sobre la ruta del cable de fibra óptica.

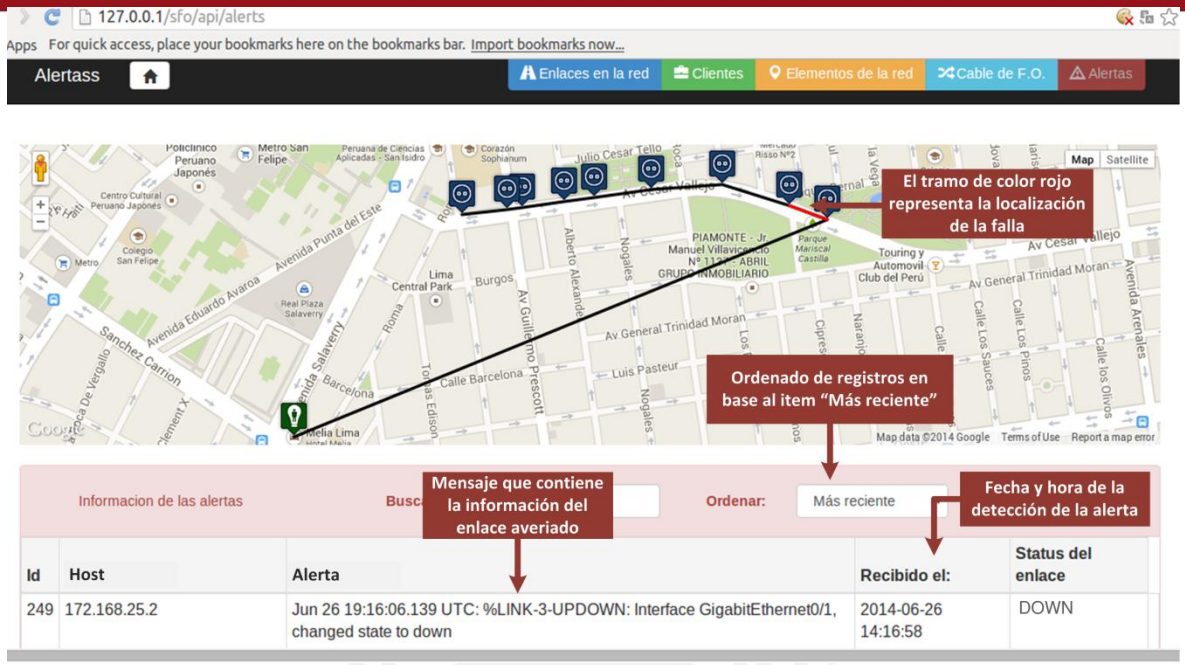


Figura 4.21. Visualización de la localización de una avería y su información asociada.
[Elaboración propia]

CONCLUSIONES

Finalizado este proyecto, se concluye lo expresado a continuación:

- Se logró implementar un sistema capaz de detectar y localizar averías sin la necesidad de utilizar equipamiento basado en la tecnología de reflectometría óptica y que a su vez brinda una alerta inmediata con la información relevante sobre la falla, como lo es la posición de la misma. Así mismo, el sistema lleva un historial de las alertas registradas, lo que permite hacer un seguimiento al estado de la red y su comportamiento. Esto hace posible que se pueda hacer un estudio de las averías y plantear estrategias para evitarlas a futuro.
- El uso de la información georeferenciada de la red, permite determinar la localización de la avería de forma rápida y precisa, lo que facilita la tarea de atender la falla en la planta externa, al no tener que usar tiempo para decidir por donde comenzar a buscar la avería con un OTDR. Esto contribuirá, en gran medida, a un ahorro de recursos y tiempo por parte de los encargados de supervisar y/o monitorear la red de planta externa metropolitana al momento de atender alguna falla.
- El sistema brinda herramientas de fácil uso que no requieren de conocimientos especializados para ser aprovechadas en su totalidad. Esto hace que los usuarios no tengan problemas a la hora visualizar la información que desean y que esta se presente de una forma bastante accesible.

RECOMENDACIONES

A continuación se presentan algunas recomendaciones derivadas de la presente tesis:

- Con la finalidad de facilitar el despliegue del sistema en un escenario real, se recomienda disponer de un servidor (de preferencia físico) con el sistema operativo Linux en sus distribuciones para servidor como Ubuntu Server o Red Hat, de modo que se puedan aprovechar los *scripts* de instalación automática de *web2py* junto con *Nginx*, así como poder utilizar la herramienta *rsyslog* de una forma más accesible.

- Respecto a *rsyslog*, es recomendable revisar previamente si no se tiene ningún servicio que capture o procese información del tipo "*syslog*" y desinstalarlo, ya que es altamente probable que también funcione en el mismo puerto que "*rsyslog*", lo cual ocasionará conflictos al intentar capturar los mensajes de estado de los enlaces.

BIBLIOGRAFÍA

(1) Ng Boon Chuan; Premadi, A.; Ab-Rahman, M.S.; Jumari, K., "Physical layer monitoring in 8-branched PON-based i-FTTH," Photonics (ICP), 2010 International Conference on , vol., no., pp.1,5, 5-7 July 2010
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5604408&isnumber=5604368

(2) Zhixin Liu; Ming Li; Chun-Kit Chan, "Fault localization in passive optical networks using OTDR trace correlation analysis," Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2012 and the National Fiber Optic Engineers Conference , vol., no., pp.1,3, 4-8 March 2012
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6192031&isnumber=6191948

(3) Maamoun, K.M.; Mouftah, H.T., "Novel techniques for deploying monitoring trails (m-trails) for fault localization in all-optical networks," High-Capacity Optical Networks and Enabling Technologies (HONET), 2010 , vol., no., pp.26,32, 19-21 Dec. 2010
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5715768&isnumber=5715739

(4) Hosszu, E.; Tapolcai, J.; Ronyai, L.; Soproni, P.; Babarczi, P.; Pin-Han Ho, "Fast failure localization in all-optical networks with length-constrained monitoring trails," Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on , vol., no., pp.677,683, 3-5 Oct. 2012
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6459752&isnumber=6459638

(5) Jingran Luo; Shanguo Huang; Jie Zhang; Xin Li; Wanyi Gu, "A novel multi-fault localization mechanism in PCE-based multi-domain large capacity optical transport networks," Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2012 and the National Fiber Optic Engineers Conference , vol., no., pp.1,3, 4-8 March 2012
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6476364&isnumber=6191948

(6) Ahuja, S.S.; Ramasubramanian, S.; Krunz, M.M., "Single-Link Failure Detection in All-Optical Networks Using Monitoring Cycles and Paths," Networking, IEEE/ACM Transactions, vol.17, no.4, pp.1080,1093, Aug. 2009
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4804678&isnumber=5208599

(7) Khair, M.; Kantarci, B.; Mouftah, H.T., "Connection provisioning constrained to fault localization in all-optical networks," Computer and Information Sciences, 2008. ISCIS '08. 23rd International Symposium on , vol., no., pp.1,6, 27-29 Oct. 2008
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4717966&isnumber=4717848

(8) Esmail, M.A.; Fathallah, H., "Current and next-generation passive optical networks monitoring solution," High Capacity Optical Networks and Enabling Technologies (HONET), 2011 , vol., no., pp.334,338, 19-21 Dec. 2011
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6149765&isnumber=6149763

(10) Ehrhardt, A.; Escher, F.; Schurer, L.; Foisel, H-M; Templin, A.; Adamy, M.; Gerlach, C., "PON measurements and monitoring solutions for FTTH networks during deployment and operation," Transparent Optical Networks (ICTON), 2011 13th International Conference on , vol., no., pp.1,6, 26-30 June 2011
Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5970861&isnumber=5970761

(11) Vivek Kumar, Deepak Rajouria, "Fault Detection Technique by using OTDR: Limitations and drawbacks on practical approach of measurement," International Journal of Emerging Technology and Advanced Engineering (IJETAE) ISSN 2250-2459 on vol. 2, Issue 6, June 2012
Available: http://www.ijetae.com/files/Volume2Issue6/IJETAE_0612_49.pdf

(12) ISO/IEC/IEEE 15289 Systems and software engineering — "Content of life-cycle information products (documentation) – Definitions"(2011)
Available: http://dictionary.ieee.org/definitions/15289-2011/

(13) IEEE Std 1682™-2011 IEEE Trial-Use "Standard for Qualifying Fiber Optic Cables, Connections, and Optical Fiber Splices for Use in Safety Systems in Nuclear Power Generating Stations - Definitions"(2011)
Available: http://dictionary.ieee.org/definitions/1682-2011/

(14) IEEE Std 1355-1995 "IEEE Standard for Heterogeneous InterConnect (HIC) (Low-Cost, Low-Latency Scalable Serial Interconnect for Parallel System Construction) – Definitions" (1995)
Available: http://dictionary.ieee.org/definitions/1355-1995/

(15) IEEE Std 1682™-2011 IEEE Trial-Use "Standard for Qualifying Fiber Optic Cables, Connections, and Optical Fiber Splices for Use in Safety Systems in Nuclear Power Generating Stations - Definitions"(1997)
Available: http://dictionary.ieee.org/definitions/211-1997/

(16) REAL ACADEMIA ESPAÑOLA "Diccionario de la Lengua Española – Vigésima segunda edición" (2001)
Available: http://lema.rae.es/drae/

(17) Gerd, Keiser, "Optical Fibers: Structures, Waveguides and Fabrication" in Optical Fiber Communications 2nd ed. Singapore, McGraw-Hill, 1991, ch.2, pp. 26-28.

(18) Tanenbaum, Andrew S., "Hardware de Redes" in Redes de Computadoras 4th ed. Amsterdam, Netherlands, McGraw-Hill, 2003, ch.1, sec.1.2, pp 14-23.
Available: http://www.youblisher.com/p/519571-Redes-de-Computadoras-Andrew-S-Tanenbaum-4ta-Edicion/

(19) Ministerio de Vivienda y Urbanismo, "Infraestructura de Telecomunicaciones" in Reglamento Nacional de Edificaciones – Separa Especial, pp. 75 - 76
Available: http://www.vivienda.gob.pe/Ley29090/documentos/RNE.pdf

(20) Salomé, Omar, Material del Curso de Comunicaciones Ópticas

(21) Elaboración propia

(22) 3M Communications Technologies, "3M™ Fiber Optic Splice Closure 2178-S/FR" in Fiber Closures and Terminals
Available:
http://solutions.3m.com/wps/portal/3M/en_US/NA_Communication_Technologies/Home/Products/~/3M-2178-Series-Closures?N=7569747+3294423422&rt=rud

(23) The Fiber Optics Association, "Basic splices" in Outside Plant, Termination, Splices
Available: http://www.thefoa.org/tech/ref/termination/Splices/Splices.htm

(24) Ministerio de Energía y Minas, "Sección2: Terminología Básica" in Código Nacional de Electricidad – Suministro, pp.17
Available: http://spij.minjus.gob.pe/Graficos/Peru/2011/Mayo/05/RM-214-2011-MEM-DM.pdf

(25) Di Pierro, Massimo, "Introduction" in Web2Py: Complete Reference Manual, 5th ed.
Available: https://dl.dropboxusercontent.com/u/18065445/web2py/web2py_manual_5th.pdf

(26) Google®, "What is Angular?" in AngularJS – HTML Enhanced for web apps
Available: https://docs.angularjs.org/guide/introduction

(27) Google®, "Understanding controllers" in AngularJS – HTML Enhanced for web apps
Available: https://docs.angularjs.org/guide/controller

(28) Google®, “Services” in AngularJS – HTML Enhanced for web apps
Available: https://docs.angularjs.org/guide/services

(29) Google®, “Directives” in AngularJS – HTML Enhanced for web apps
Available: https://docs.angularjs.org/guide/directives

(30) McCready, Nicholas and Laplante Nicolas, “Google Maps for AngularJS”
Available: http://angular-google-maps.org/

(31) Oracle Corporation, “What is MySQL?” in Overview of the MySQL Database Management System
Available: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html

(32) Adiscon GmbH, “The Rocket-Fast System for Log Processing”
Available: http://www.rsyslog.com/doc/v8-stable/configuration/basic_structure.html

(33) IETF Standards, “The Syslog Protocol” in RFC 5424, March 2009
Available: http://tools.ietf.org/html/rfc5424