

**UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA ELECTRÓNICA**



PROYECTO DE GRADO

**“DISEÑO DE SISTEMA DE ALERTA TEMPRANA COMPLEMENTARIA PARA
PREVENCIÓN DE DESASTRES”.**

**Aplicación: sistema nacional de alerta temprana para desastres SINAGER-SNATD y canal
de televisión, utilizando la norma ISDB-Tb**

POSTULANTE: UNIV. BISMARCK PONCE LIMACHI

TUTOR: ING. JOSÉ CAMPERO BUSTILLOS

DOCENTE: ING. GONZALO CABA MORALES

LA PAZ - BOLIVIA

2020



**UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE INGENIERIA**



LA FACULTAD DE INGENIERIA DE LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

- a) Visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
- b) Copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
- c) Copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la cita o referencia correspondiente en apego a las normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

TODOS LOS DERECHOS RESERVADOS. EL USO NO AUTORIZADO DE LOS CONTENIDOS PUBLICADOS EN ESTE SITIO DERIVARA EN EL INICIO DE ACCIONES LEGALES CONTEMPLADAS EN LA LEY DE DERECHOS DE AUTOR.

DEDICATORIA

Este proyecto simboliza el final de una gran etapa en mi vida llena de sacrificios, esfuerzos y logros, por este motivo quiero dedicarlo a mis padres, Félix Ponce Condori y Betzabe Limachi Rodríguez; quienes, con su apoyo constante, comprensión y paciencia me han inculcado los valores y las bases para llegar a ser una buena persona.

A mis hermanos, Ronald y Neyda, por ser una inspiración y un ejemplo a seguir como personas y profesionales.

A los amigos que conocí en esta linda etapa y que se convertirán en grandes profesionales, este proyecto se los dedico como reflejo del compañerismo, de la amistad y del trabajo en equipo que se realizó durante la carrera universitaria.

AGRADECIMIENTO

De todo corazón agradezco a todos los integrantes de mi familia, cuyo apoyo incondicional me han llevado hasta donde estoy ahora, son ejemplo de trabajo y esfuerzo, durante toda mi formación académica han sido un pilar fundamental.

También quiero agradecer a la Universidad Mayor de San Andrés y a la carrera de Ingeniería Electrónica, por permitirme realizar mis estudios y a todos los ingenieros que fueron mis profesores en las diferentes materias porque de ellos aprendí todas las cosas que hoy aplicare en el ámbito profesional; especialmente al Ing. Gonzalo Caba Morales y al Ing. José Campero quienes, muy acertadamente, dirigieron mi proyecto de grado.

Finalmente, a mis amigos con los cuales compartí y seguiré compartiendo muchas experiencias que nunca olvidare.

RESUMEN

Ahora en Bolivia con la migración de televisión analógica a la televisión digital y gracias a las características que posee el estándar adoptado por Bolivia (ISDB-Tb) en cuanto a interactividad para el envío de señales de alerta, se podrá mejorar los planes de contingencia en momentos de eventos climatológicos, los cuales van dejando secuelas continuamente en el territorio nacional.

El presente trabajo muestra el diseño de un sistema de alerta temprana para desastres, el cual complementaria al actual sistema que posee el Vice-ministerio de Defensa Civil (SNATD), mejorando la forma que tiene de difundir sus boletines, utilizando una nueva tecnología como es la televisión Digital, con el que se pretende disminuir las pérdidas materiales y pérdidas humanas, dicho diseño presentara dos etapas:

La parte del diseño de un enlace VPN (Virtual Private Network) mediante la creación y configuración de archivos del software OpenVPN, el cual permitiría que SNATD (sistema nacional de alerta temprana para desastres) pueda enviar información acerca del evento a la estación del canal de televisión en cuestión.

La segunda parte pertenece a la creación del código de programa NCL (Nested Context Lenguaje), dicho programa NCL permite que la información enviada por SNATD pueda ser visualizada en televisores y dispositivos móviles, pudiendo así difundir el mensaje de alerta temprana a los lugares que se verían afectados.

ABSTRACT

Now in Bolivia with the migration of analog television to digital television and thanks to the characteristics of the standard adopted by Bolivia (ISDB-Tb) in terms of interactivity for sending warning signals, the contingency plans can be improved at times of events climatological, which are leaving sequels continuously in the national territory.

The present work shows the design of an emergency warning system for disasters, which will complement the current system of the Vice Ministry of Civil Defense (SNATD), this will improve your way of spreading alert messages, using a new technology such as television Digital, which aims to reduce material losses and human losses, this design will present two stages:

The first part is the design of a VPN link (Virtual Private Network) through creating and configuring OpenVPN files, so that SNATD (national early warning system for disasters) can send information about the event to the television station.

The second part is the creation of the program code NCL (Nested Context Language), this NCL program allows the information sent by SNATD can be displayed on televisions and mobile devices, spreading the early warning message to affected places.

ÍNDICE DEL CONTENIDO

CAPITULO I: INTRODUCCIÓN	1
1 Antecedentes	1
2 Planteamiento del problema	4
3 Objetivos	5
3.1 Objetivo general.....	5
3.2 Objetivos específicos	5
4 Justificación.....	5
4.1 Justificación Social	5
4.2 Justificación Tecnológica.....	5
5 Alcances y Limites	6
5.1 Alcances.....	6
5.2 Limites	6
CAPITULO II: FUNDAMENTOS TEÓRICOS.....	7
1 Televisión Digital	7
1.1 Beneficios de la Televisión Digital Terrestre	7
1.2 Estándares de Televisión Digital Terrestre	10
2 Estándar ISDB-Tb	11
2.1 Características y especificaciones técnicas del estándar ISDB-Tb.....	11
2.1.1 Capa de Transmisión	12
2.1.2 Capa de Multiplexación	13
2.1.3 Capa de Compresión	14
2.1.4 Capa Middleware	15
2.1.5 Capa de Aplicación	15
2.2 Middleware GINGA	15

2.2.1	Arquitectura Ginga.....	15
2.2.1.1	Ginga - NCL.....	16
2.2.1.2	Ginga - J	16
2.2.1.3	Ginga Common Core	17
2.2.2	Funcionamiento	18
2.3	NCL, Nested Context Languaje.....	19
2.3.1	Estructura de un documento NCL.....	19
2.3.2	Módulos de NCL.....	21
2.3.2.1	Área funcional Estructure.....	22
2.3.2.2	Área funcional Layout.....	23
2.3.2.3	Área funcional Components.....	23
2.3.2.4	Área funcional Interfaces	24
2.3.2.5	Área funcional de Especificación de Presentación.....	25
2.3.2.6	Área funcional Linking.	26
2.3.2.7	Área funcional Connectors.....	27
2.3.2.8	Área funcional de Control de Presentación.....	28
2.3.2.9	Área funcional Timing	30
2.3.2.10	Área funcional Reuse	30
2.3.2.11	Área funcional Navigational Key.....	31
2.3.2.12	Área funcional Animation.....	32
2.3.2.13	Área funcional SMIL Transition Efects	32
2.3.2.14	Área funcional Metainformation.....	32
3	EWBS, Emergency Warning Broadcasting System.....	33
3.1	Transmisión de la alerta temprana	35
3.2	Esquema de transmisión de las señales de TDT	36

3.3 Recepción de la señal EWBS.....	38
3.4 EWBS por One-seg en terminales de mano.....	39
4 SAT en Bolivia.....	41
4.1 SNATD, Sistema Nacional de Alerta Temprana de Desastres	41
4.1.1 Aplicación del SAT en Bolivia	41
4.1.2 Innovaciones tecnológicas del SAT nacional.....	41
4.1.3 Comunicación para controlar el riesgo.....	42
4.2 Componentes del sistema de alerta temprana de desastres (SNATD).....	43
4.2.1 Sistema de observación	43
4.2.2 Sistema de comunicaciones.....	43
4.2.3 Plataforma DEWETRA para la gestión y prevención de desastres naturales	44
4.2.3.1 ¿Qué es la plataforma DEWETRA?.....	45
4.2.3.2 Arquitectura y acceso a la plataforma DEWETRA.....	46
4.2.3.3 Evaluación en tiempo real del riesgo	47
4.3 Colores de avisos de alerta implementados en el Sistema de Alerta Temprana (SAT)....	48
4.3.1 Precipitaciones Pluviales.....	49
4.3.2 Temperaturas	50
4.3.3 Vientos	50
4.3.4 Sequias	51
4.4 Boletines de riesgo.....	53
5. Red Privada Virtual.....	53
5.1 Definición de una VPN.....	53
5.2 Propiedades de una VPN	55
5.2.1 Encapsulación.....	55
5.2.2 Autenticación	55

5.2.3 Cifrado de datos	55
5.3 Beneficios de una VPN.....	56
5.3.1 Seguridad.....	56
5.3.2 Transparencia	56
5.3.3 Cobertura.....	56
5.3.4 Ahorro en los costos.....	56
5.4 Elementos de una VPN	57
5.5 Arquitecturas de una VPN	58
5.5.1 Arquitectura VPN basados en hardware	59
5.5.2 Arquitectura VPN basados en cortafuegos.....	59
5.5.3 Arquitectura VPN basados en software	59
5.6 VPN basadas en SSL/TLS	59
6. OpenVPN	62
6.1 ¿Qué es OpenVPN? Características Principales	62
6.2 OpenVPN y Paquetes Necesarios	64
6.2.1 Los archivos de configuración de OpenVPN.....	65
6.2.2 Seguridad con OpenSSL	65
6.2.2.1 Algoritmo Diffie-Hellman.....	66
6.2.2.2 Algoritmo de cifrado AES-256	66
6.3 Modo de funcionamiento de OpenVPN.....	67
6.3.1 Capa de Cifrado de OpenVPN	67
6.3.2 Asignación de Direcciones.....	68
CAPITULO III: ANÁLISIS PRELIMINAR	69
1 Modelo Preliminar.....	69
2 Herramientas a utilizar	70

2.1 Herramientas para TDT	70
2.1.1 Eclipse NCL.....	70
2.1.2 Ginga4Windows.....	71
2.2 Herramientas para el enlace VPN	72
2.2.1 OpenVPN	72
CAPITULO IV: DESARROLLO DEL PROYECTO	73
1 Diseño de una VPN mediante la creación y configuración de archivos del software OpenVPN para una comunicación entre dos puntos remotos	73
1.1 Introducción	73
1.2 Análisis de la situación actual de SNATD.....	73
1.3 Configuración del escenario	75
1.4 Requerimientos para la instalación de OpenVPN.....	76
1.5 Realización de la instalación y configuración	77
1.5.1 Instalación de OpenVPN en Windows.....	77
1.5.2 Implementación de la Seguridad	79
1.5.2.1 Creación de Clave (Certificado de la CA).....	80
1.5.2.2 Creación de clave y certificado del servidor	82
1.5.2.3 Creación de la clave y certificado del cliente.....	84
1.5.2.4 Creación de los parámetros Diffie Hellman	85
1.5.3 Configuración de archivos	86
1.5.3.1 Servidor	86
1.5.3.2 Cliente	88
1.5.4 Inicialización de la conexión.....	89
1.6 Calculo del tráfico de datos.....	90
2 Diseño del prototipo funcional con el software Eclipse para la emisión de alertas del SNATD (Sistema Nacional de Alerta Temprana de Desastres)	91

2.1	Introducción	91
2.2	Recolección de parámetros para la elaboración del prototipo	91
2.2.1	Tipos de amenazas	92
2.2.2	Nivel de alerta	92
2.2.3	Municipios.....	93
2.2.4	Fechas y características	93
2.2.5	Mapa de vulnerabilidad.....	93
2.2.6	Recomendaciones.....	94
2.3	Construcción del modelo	94
2.3.1	¿Qué vamos a mostrar? - Objetos Media	95
2.3.2	¿Dónde los vamos a mostrar? - Regiones	96
2.3.3	¿Cómo los vamos a mostrar? – Descriptores	97
2.3.4	¿Cuándo los vamos a mostrar? - Links y Conectores	97
2.4	Modelo de infraestructura de la TDT.....	101
2.4.1	Parámetros de entrada	102
2.4.2	El servidor de aplicación y contenidos.....	102
2.4.3	El servidor de playout	103
2.4.4	Parámetros de salida.....	104
2.4.5	Set-top box	104
2.5	Realización del prototipo	104
2.5.1	Creación de un nuevo proyecto NCL en Eclipse	104
2.5.2	Desarrollo del código del programa	108
2.6	Resultados	115
CAPITULO VI: COSTOS		118
1	Análisis de costos para el diseño de la red VPN	118

1.1 Costos de software	118
1.2 Costos del recurso humano	118
2 Análisis de costos para la elaboración del prototipo de sistema de alerta temprana.....	119
3 Costos totales para la hipotética implementación del proyecto	119
CAPITULO VII: CONCLUSIONES Y RECOMENDACIONES	120
1 Conclusiones	120
2 Recomendaciones.....	121
GLOSARIO.....	122
REFERENCIAS	126
ANEXO A: CÓDIGO DEL PROGRAMA.....	129
ANEXO B: BASE DE CONECTORES	131
ANEXO C: ARCHIVOS DE CONFIGURACION OPENVPN.....	155

ÍNDICE DE TABLAS

Tabla 1. Descripción de los Estándares Mundiales de TDT	10
Tabla 2. Áreas Funcionales.	21
Tabla 3. Módulo de Estructura Extendida.....	23
Tabla 4. Módulo Layout Extendida.....	23
Tabla 5. Módulo de Media Extendida.	24
Tabla 6. Módulo de Contexto Extendido.	24
Tabla 7. Módulo de MediaContentAnchor.	24
Tabla 8. Módulo de CompositeNodeInterface Extendido.....	25
Tabla 9. Módulo de PropertyAnchor Extendido	25
Tabla 10. Módulo de SwitchInterface Extendido.....	25
Tabla 11. Módulo del Descriptor Extendido.	26
Tabla 12. Módulo Linking Extendido.	27
Tabla 13. Módulo del CausalConnectorFunctionality Extendido.....	28
Tabla 14. Módulo del TestRule Extendido.	29
Tabla 15. Módulo TestRuleUse Extendido.	29
Tabla 16. Módulo ContentControl Estendido.	30
Tabla 17. Módulo DescriptorControl Extendido.....	30
Tabla 18. Módulo Import Extendido.....	31
Tabla 19. Módulo Meta-Information Extendido.	33
Tabla 20. Especificaciones de las Funciones que debe tener un Receptor ISDB-Tb.....	39
Tabla 21. Tabla de Costos de Software.....	118
Tabla 22. Tabla de Costos Recurso Humano	118
Tabla 23. Tabla de Costos para elaboración del programa NCL	119
Tabla 24. Tabla de Costo Total	119

ÍNDICE DE FIGURAS

Figura 1. Componentes del SINAGER-SAT	2
Figura 2. Esquema conceptual de utilización del ancho de banda disponible	9
Figura 3. Estructura del estándar ISDB-Tb.....	12
Figura 4. Segmentación del ancho de banda disponible en un canal de 6MHz.	12
Figura 5. Generación del flujo de transporte de difusión a transmitirse	13
Figura 6. Multiplexación de audio, video y datos	14
Figura 7. Arquitectura del Middleware Ginga	16
Figura 8. Estructura de Ginga CC	17
Figura 9. Funcionamiento de Ginga.....	19
Figura 10. Estructura Básica de un Documento NCL.....	20
Figura 11. Funciones Básicas del Sistema EWBS.....	34
Figura 12. Estructura de los Paquetes TS, Flujo BTS y cuadro Multiplex.	34
Figura 13. Diagrama de Transmisión Incluyendo una Señal EWBS.....	36
Figura 14. Conformación de los Paquetes de Flujo de Transporte.	37
Figura 15. Estructura del Descriptor EWBS en una Tabla PMT	38
Figura 16. Activación Automática de Terminales de mano One-Seg.....	40
Figura 17. Señales de Alerta de Emergencia en ISDB-Tb.....	40
Figura 18. Estaciones Meteorológicas en Bolivia (ubicación).....	41
Figura 19. Sistema Nacional de Alerta Temprana de Desastres (SNATD)	42
Figura 20. Sistema de Comunicación de la Red WAN.....	44
Figura 21. Plataforma Dewetra.	45
Figura 22. Interface de trabajo de la plataforma informática DEWETRA.	46
Figura 23. Lectura de Alerta de las Precipitaciones Pluviales en el Municipio de Magdalena, Beni-Bolivia.	49

Figura 24. Lectura de la Alerta de Temperatura en el Municipio de Magdalena, Beni-Bolivia....	50
Figura 25. Diagrama Lógico de una VPN.....	54
Figura 26. Situación de SSL/TLS en la pila de protocolos OSI.....	60
Figura 27. Arquitectura SSL/TLS.	61
Figura 28. Formato del Paquete Encapsulado por OpenVPN.....	63
Figura 29. Espacio de Direcciones IP para las Redes Privadas según la IANA.	68
Figura 30. Esquema del funcionamiento propuesto	69
Figura 31. IDE Eclipse	71
Figura 32. Reproductor NCL-NCLua	72
Figura 33. Logo del Software OpenVPN	72
Figura 34. Diagrama de bloques para la elaboración del diseño VPN.....	73
Figura 35. Diagrama de Red Local SINAGER-SNATD	74
Figura 36. Esquema del Escenario	75
Figura 37. Pasos 1 y 2 en la instalación de OpenVPN.....	78
Figura 38. Paso 3 Componentes a Instalar en Windows	78
Figura 39. Directorios de instalación y finalización de Instalación	79
Figura 40. Icono del Proceso de la GUI Instalada.....	79
Figura 41. Ejecución del Comando "init.config"	81
Figura 42. Configuración de Parámetros en VARS	81
Figura 43. Creación del Certificado CA.....	82
Figura 44. Llenado de datos para la Creación de Certificado y llave del Servidor.....	83
Figura 45. Creación del Par Certificado y Clave del Servidor.....	83
Figura 46. Llenado de Datos para la Creación de Certificado y llave Cliente	84
Figura 47. Creación del Par Certificado y Clave del Cliente	85
Figura 48. Creación del Archivo dh1024.pem	86

Figura 49. Configuración del Archivo servidor-vpn.ovpn.....	87
Figura 50. Configuración del Archivo cliente.ovpn.....	88
Figura 51. Inicialización del Servidor y Cliente	90
Figura 52. Diagrama de bloques para la generación del código NCL	91
Figura 53. Clasificación de alertas	92
Figura 54. Mapa de Inundaciones, deslizamientos, desbordes y/o riadas.....	93
Figura 55. Estructura de un Documento NCL.....	94
Figura 56. Imagen Principal del prototipo	95
Figura 57. Imagen Secundaria del prototipo	95
Figura 58. Distribución de Regiones.....	96
Figura 59. Puerto del nodo de composición.....	98
Figura 60. Diagrama de tiempo del prototipo	98
Figura 61. Diagrama de Flujo del código NCL.....	101
Figura 62. Infraestructura del sistema de alerta temprana	102
Figura 63. Funcionalidad de un servidor Playout	103
Figura 64. Set-Top box Comercial.....	104
Figura 65. Creación de un nuevo proyecto	105
Figura 66. Selección de NCL Proyect.....	105
Figura 67. Ingreso del Nombre del proyecto.....	106
Figura 68. Creación del Proyecto NCL.....	106
Figura 69. Creación de un Nuevo Documento NCL.....	107
Figura 70. Nombre del documento NCL.....	107
Figura 71. Esqueleto Básico de un Programa NCL	108
Figura 72. Atributos de posicionamiento de la región	109
Figura 73. Adhiriendo Ginga4Windows a Eclipse.....	115

Figura 74. Ingresando la dirección de Ginga4Windows en Eclipse	116
Figura 75. Ejecución de la Aplicación con Ginga-Play	116
Figura 76. Visualización de la imagen principal del prototipo	117
Figura 77. Visualización de la imagen secundaria del prototipo	117

CAPITULO I: INTRODUCCIÓN

1 Antecedentes

En los últimos años, la vigilancia y la disponibilidad de pronósticos del tiempo y el clima, se han convertido en una actividad científica sumamente sofisticada, dedicada en particular a la protección de las vidas y los bienes. La mejora continua de los pronósticos del tiempo ha permitido salvar numerosas vidas. Hoy en día la población en general, desde los agricultores y los urbanistas o los encargados de gestionar emergencias, pasando por los gestores de recursos hídricos o los excursionistas de fin de semana, hasta los funcionarios públicos, se benefician de los servicios meteorológicos y climáticos.

En Bolivia, la vigilancia Meteorológica está a cargo del SENAMHI (Servicio Nacional de Meteorología e Hidrología) que cuenta con sistemas de observación, servicios de telecomunicaciones, centro de procesamiento de datos y unidad de pronósticos que generan y disponen de información para su aplicación por los diversos sectores, pero principalmente en la gestión del riesgo para la operatividad del Sistema de Alerta temprana (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

El SENAMHI elabora avisos de alerta temprana meteorológicos e hidrológicos cuando las condiciones atmosféricas y de comportamiento de caudales y niveles de ríos lo ameriten y corresponda avisar a las autoridades de Defensa Civil, en la probabilidad de ocurrencia de evento adverso.

Defensa Civil en función al aviso de alerta emitido por el SENAMHI, elabora un boletín de criticidad para la toma de decisiones por parte de Municipios y Gobernaciones para hacer frente a la posible formación de evento adverso (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

El Vice-ministerio de Defensa Civil (VIDECI) a través del Sistema Integrado de Información y Alerta para la Gestión del Riesgo de Desastres (SINAGER-SAT), ofrece una base de información sobre amenazas, vulnerabilidades, niveles o escenarios de riesgo, de alerta, de capacidad de respuesta y parámetros de riesgo.

El SINAGER-SAT está integrado por el Sistema Nacional de Alerta Temprana para Desastres (SNATD), el Observatorio Nacional de Desastres (OND), la Infraestructura de Datos Espaciales (GEOSINAGER) y la Biblioteca Virtual de Atención y Prevención de Desastres (BIVAPAD) (Viceministerio de Defensa Civil, 2016).

Componentes del SINAGER-SAT



Figura 1. Componentes del SINAGER-SAT

Fuente: (Viceministerio de Defensa Civil, 2016)

El sistema Nacional de alerta temprana de desastres (SNATD), es el sistema de vigilancia y monitoreo de amenazas probables frente a las condiciones de vulnerabilidad existentes y tiene la finalidad de proporcionar información sobre el nivel o escenario de riesgos, para activar protocolos de prevención y preparación de transmisión rápida, para esto SNATD manda boletines a las Gobernaciones a través de Internet, fax y celulares (Viceministerio de Defensa Civil, 2016).

Por otra parte, dada la necesidad de difundir el mensaje de alerta con más efectividad, es que surgen nuevas tecnologías de sistemas de alertas tempranas.

En el mundo los primeros sistemas de alertas tempranas tienen origen en el continente asiático debido a la posición geográfica en la que se encuentra han sido víctimas de varios eventos naturales muchos de estos desastrosos, por este motivo ha existido la necesidad de crear sistemas de alerta mucho más sofisticados empleados en las nuevas tecnologías existentes como televisión digital terrestre (TDT).

Desde su invención la televisión se convirtió en un medio de información de gran importancia e influencia en la sociedad. A medida que los sistemas electrónicos evolucionaban, las operadoras de televisión empezaban a digitalizar sus equipos. Surgieron varios estándares de televisión digital

terrestre, para nuestro caso se estudiará el estándar japonés con modificación brasileña ISDB-Tb (Terrestrial Integrated Services Digital Broadcasting), ya que es la que se adoptó en Bolivia con el Decreto Supremo “D.S. 819 de 16/03/2011”.

El estándar adoptado por Bolivia otorgará la posibilidad de visualización del contenido en dispositivos móviles como celulares, tabletas y computadoras portátiles. Adicionalmente, mediante la utilización de aplicaciones interactivas dentro de la TDT se podrá recibir alertas de emergencia y de seguridad nacional.

En Bolivia hoy en día ya se han decretado normas y resoluciones administrativas para la implementación de dicho estándar.

- Decreto Supremo 3152 de 19 de abril de 2017, el cual aprueba el Plan de Implementación de Televisión Digital Terrestre en el Estado Plurinacional de Bolivia, establece los lineamientos, procedimientos y plazos para la transición de la televisión con tecnología analógica a la televisión digital terrestre (Autoridad de Regulación y Fiscalización de Telecomunicaciones y transporte, s.f.).

En la actualidad una cantidad limitada de emisoras nacionales se encuentran transmitiendo en señal analógica y digital a la vez, con el objetivo de realizar pruebas de la señal digital para llegar a cumplir con el apagón analógico, el cual consiste en apagar las emisiones analógicas de televisión y transmitir solo señales digitales. Bolivia espera cubrir la mayor parte de la población con emisiones digitales y realizar el apagón analógico en noviembre del 2021 en ciudades troncales y noviembre del 2025 para el territorio nacional restante (Bolivia. Concejo de Ministros , 2019).

- Instructivo técnico para el desarrollo de aplicaciones interactivas con la plataforma GINGA “ATT-DJ_RAR_TL LP 585/2017 de 19/07/2017”, el cual establece las normas técnicas para el desarrollo de aplicaciones interactivas, para implementarla en televisión digital terrestre, con la plataforma GINGA.

2 Planteamiento del problema

Bolivia es el segundo país más vulnerable de Sudamérica y el quinto menos preparado para mitigar los daños del cambio climático, según una investigación del programa ND-GAIM Country Index, del proyecto Iniciativa de Adaptación Global de la Universidad de Notre Dame (Estados Unidos). Este dato corrobora lo que hace un par de años advirtió la Organización de las Naciones Unidas (ONU): que Bolivia era uno de los países más expuestos al fenómeno del calentamiento global.

Para determinar qué país está más preparado y cuál corre mayor peligro frente a los efectos del calentamiento global, el proyecto recoge datos que muestran la vulnerabilidad al cambio climático y otros desafíos globales en combinación con su prontitud para reaccionar ante una catástrofe y su capacidad de adaptación (Los Tiempos , 2017).

Mencionando algunos de los desastres relevantes que ocurrieron en Bolivia como inundaciones y granizadas de 1983, 1993, 2007 y 2008, o efectos del fenómeno de El Niño en 2015 y 2016 que acarrearón “incalculables pérdidas económicas y han desestabilizado sensiblemente las opciones de desarrollo de muchas áreas empobrecidas de la subregión andina”, es que se muestra que aún queda mucho por hacer, por ejemplo, códigos de edificación obligatorios, sistemas de drenajes eficientes y el mejoramiento de alertas tempranas.

En Bolivia, el sistema nacional de alerta temprana para desastres (SNATD) envía boletines de criticidad a través de internet fax celular, y como se sabe, en casos de posibles desastres, es de valiosa necesidad el difundir la información a la mayor gente posible, ya que mientras más gente esté informada menos serán los daños causados.

Por tanto, se plantea la necesidad de un sistema que optimice el sistema de alerta temprana actual del Vice-ministerio de Defensa Civil.

Para lo cual se propone un sistema de alerta temprana complementaria, utilizando la señal de televisión digital como medio de difusión.

3 Objetivos

3.1 Objetivo general

Optimizar el sistema nacional de alerta temprana de desastres mediante el diseño de un prototipo que utilice el sistema middleware Ginga - NCL del sistema de televisión digital ISDB-Tb, para incorporarlo a las redes existentes de televisión, informando de los posibles eventos de riesgo de desastres en el país.

3.2 Objetivos específicos

- Recolectar datos relevantes de los boletines que envía SNATD para la realización del prototipo.
- Crear y configurar los archivos del software OpenVPN para una comunicación remota entre dos puntos.
- Generar un código de programación sencillo y flexible, que permita visualizar los parámetros característicos de cada evento de desastre de SNATD.
- Resaltar la importancia de la televisión digital terrestre (TDT) y del estándar que adoptamos (ISDB-Tb) en el país.

4 Justificación

4.1 Justificación Social

El presente diseño serviría para alertar a pobladores de posibles desastres climatológicos que ocurren y ocurrirán inevitablemente en el país, y que tienen como objetivo la disminución de los daños materiales y pérdidas humanas, a través de medios visuales. Teniendo en cuenta que el medio de comunicación de mayor penetración es la televisión, es este ideal para alertar en caso de emergencia.

4.2 Justificación Tecnológica

Este proyecto será de gran importancia en el ámbito de las Telecomunicaciones y la Gestión de Riesgos, ya que permitirá la inserción de un nuevo recurso televisivo como lo es el Middleware Ginga para la transmisión de alertas de emergencia de desastre, combinando datos (programa NCL) con la señal de televisión, para emitir alertas oportunas a la población y poder enfrentar

eventualidades, que tienen gran posibilidad de ocurrencia en Bolivia. Con su desarrollo se beneficiará a todos los actores involucrados en el proceso de comunicación; los televidentes, gracias a la implementación de un sistema moderno que permitirá estar al tanto de estas situaciones; y las estaciones televisivas, ya que podrán emitir información confiable y con mayor rapidez. Este diseño se presenta como una alternativa a implementar en caso de no poseer receptores EWBS, sin embargo, se mencionará también esta característica que posee el estándar ISDB-Tb.

5 Alcances y Limites

5.1 Alcances

- Se utilizará datos recopilados de antiguos acontecimientos ocurridos en Bolivia para aplicarlo en el desarrollo del prototipo.
- Este diseño de prototipo informara:
El tipo de riesgo de desastre que se acerca, el rango de fechas que posiblemente ocurra y la lista de municipios que serán afectados.
- Se utilizará algunos atributos de interactividad que posee el lenguaje NCL, como acceder a los botones del control remoto para realizar algunas tareas específicas.
- Se creará y configurará los archivos de OpenVPN, para que en un futuro las oficinas de SNATD pueda comunicarse con la estación de televisión.

5.2 Limites

- Este diseño se centrará únicamente a localidades donde llegue la señal del canal de televisión.
- Debido a los costos de equipamiento y la falta de auspicios, el proyecto solo se centrará en el diseño del enlace y la plantilla generada con el lenguaje Ginga-NCL, la cual es la que se observara en pantalla de los televisores y celulares.
- El Diseño de la VPN se centrará en la creación y configuración de archivos del software OpenVPN, tomando en cuenta que las oficinas que se desea comunicar ya cuentan con servicio a Internet y su propia red interna.

CAPITULO II: FUNDAMENTOS TEÓRICOS

1 Televisión Digital

A la televisión se la puede considerar como un sistema de transmisión de contenidos audiovisuales en tiempo real; específicamente, la televisión digital realiza dicha transmisión codificando de forma binaria las señales de audio y video. Este nuevo sistema de televisión puede presentarse de distintas maneras, dependiendo del modo y el medio de transmisión, como puede ser: Televisión Digital por cable, Televisión Digital por satélite y Televisión Digital terrestre.

La Televisión Digital Terrestre (TDT) es un sistema de difusión de información de acceso prácticamente universal, ya que sus transmisiones son de tipo punto a multipunto y pueden ser de acceso libre y gratuito, o por suscripción. Los flujos de información se los transmite a través del aire utilizando técnicas de modulación digital que ocupan una porción del espectro radioeléctrico, esta porción puede ser de 6, 7 u 8 MHz de ancho de banda, dependiendo del estándar que se adopte.

1.1 Beneficios de la Televisión Digital Terrestre

- **Aprovechamiento del espectro radioeléctrico**

Al concentrar más señales en las mismas bandas de frecuencia e incluso utilizando canales adyacentes libres en la televisión analógica, ya que las señales digitales son menos vulnerables a ruidos e interferencias (Frenzel, 2003).

- **Transmisión de contenidos en alta definición (HD), con una notable mejora en la calidad de audio y video**

La TDT permite que la calidad de la señal permanezca constante en toda el área de cobertura; y solo cuando existe un deterioro significativo de la misma, tanto que los sistemas de corrección de errores no puedan recuperarla, se produce un corte abrupto de la transmisión. De acuerdo a la resolución y calidad de la señal digital transmitida, ésta se clasifica en:

- ✓ LDTV (Low Definition Television), señal de baja resolución (320 x 240 pixeles) utilizada en transmisiones para dispositivos móviles.

- ✓ SDTV (Estándar Definition Television), señal de resolución estándar (720 x 576 pixeles) empleada comúnmente para multiprogramación.
- ✓ EDTV (Enhanced Definition Television), señal de resolución mejorada o intermedia (1280 x 720 pixeles) también utilizada en multiprogramación.
- ✓ HDTV (High Definition Television) señal de alta resolución (1920 x 1080 pixeles) utilizada para transmitir imágenes de gran calidad (Pisciotta, Liendo, & Lauro, 2013).

- **Integración de contenidos con Internet**

Que puede ser información relacionada o totalmente independiente de la programación, ya que los receptores pueden tener conexión a Internet a través de una interfaz Ethernet generando un canal de retorno para una comunicación bidireccional entre la estación transmisora y el usuario. Alguno de los servicios adicionales que pueden ofrecerse con esta integración son: Aplicaciones que se ejecuten en cualquier momento, Posibilidad de grabar la programación Alarmas y Alertas tempranas, Compras y Transacciones en tiempo real, así como información de Eventos culturales, musicales, deportivos, entre otros (Cubero, 2009).

- **Servicios asociados con la interactividad**

Esto se logra al emitir información adicional desde la estación transmisora, los datos son cargados en el receptor del usuario y pueden ser visualizados en simultáneo con la programación transmitida. Existen dos tipos de Interactividad, la local y la completa; la interactividad local consiste en aplicaciones que se instalan en el receptor con la señal transmitida, sin necesidad de conexiones adicionales, comúnmente están enfocadas a brindar al usuario información relevante de momento, como estadísticas deportivas o avisos de emergencia. Mientras que la interactividad completa, si necesita una conexión a Internet para establecer el canal de retorno entre el receptor la estación, dicha interactividad puede ocuparse para votaciones en programas en vivo o compra de productos. Además, a través de las mismas se puede visualizar módulos de subtítulo (CC), así como guías de programación electrónica (EPG).

- **Multiprogramación**

Al permitir la transmisión de varias señales ocupando el mismo ancho de banda o canal que la televisión analógica; aquí se debe considerar que la TDT transmite tres tipos de flujos binarios

como son: el video y audio correspondientes a la programación, los datos que corresponden a información adicional, la codificación y sincronización destinados a proteger los flujos útiles de las interferencias y detectar el esquema de transmisión utilizado. Algunas de las alternativas para la utilización del ancho de banda de un canal de televisión de 6 MHz es la que se muestra en la Figura 2.

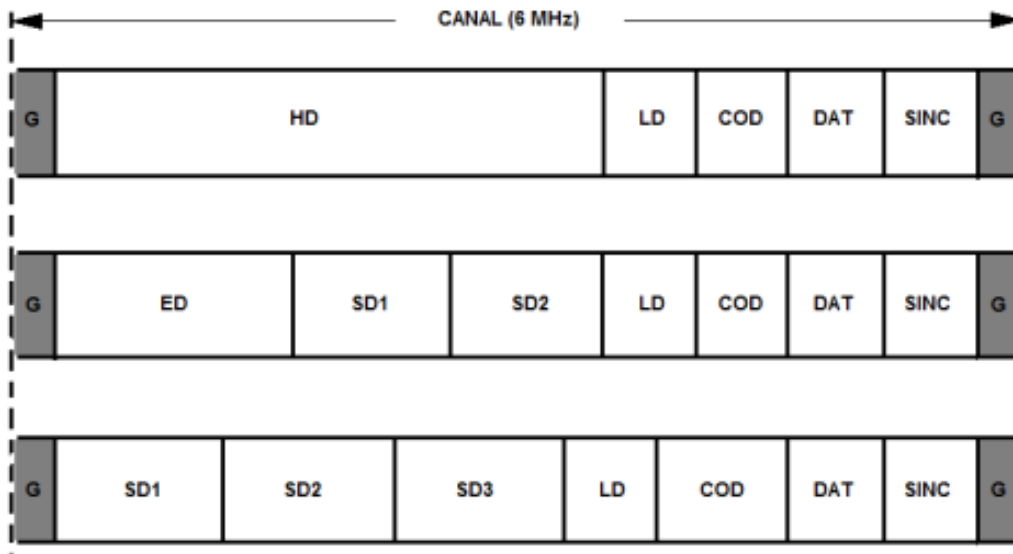


Figura 2. Esquema conceptual de utilización del ancho de banda disponible

Fuente: (Pisciotta, Liendo, & Lauro, 2013)

Como se observa, cada flujo de información ocupa una parte del espectro disponible en el canal; en la parte inferior y superior del mismo se encuentran los intervalos de guarda (G) como espacios de protección, el video y audio dependiendo de la calidad requerida (HD/ED/SD/LD), datos adicionales (DAT), codificación (COD) y sincronización (SINC) (Pisciotta, Liendo, & Lauro, 2013).

- **Portabilidad en varios tipos de dispositivos**

La TDT ha sido incorporada a televisores con sintonizador incluido, decodificadores fijos o portables e incluso teléfonos móviles compatibles con el estándar, ya que la tecnología digital y el servicio de telefonía móvil brindan esta posibilidad.

- **Implementación de Redes de Frecuencia Única (SFN)**

Que permiten ampliar el área de cobertura y mantener la uniformidad de los niveles de intensidad de campo dentro de la misma. Todas las señales deben ser transmitidas con la misma frecuencia que la estación principal en toda el área de interés, efectivizando el uso del espectro radioeléctrico. Esta configuración de frecuencias no afecta la recepción en zonas de solapamiento, ya que los transmisores son sincronizados para que la o las señales lleguen al receptor dentro del tiempo asignado para el intervalo de guarda.

Además; permite una disminución de costos de los equipos de transmisión y del consumo eléctrico, ya que los niveles de potencia requeridos para lograr un área de cobertura igual a la de los sistemas analógicos son más bajos (Pisciotta, Liendo, & Lauro, 2013).

1.2 Estándares de Televisión Digital Terrestre

Los estándares de TDT se diferencian entre sí por varios aspectos tecnológicos como, la transmisión y codificación de las señales, o la plataforma sobre la que se desarrolla, cada uno de estos estándares responden a distintos modelos de migración de lo analógico a lo digital, teniendo como principal objetivo fortalecer y mejorar la calidad del servicio de televisión tradicional. Hasta el momento se han desarrollado cuatro normas oficiales y algunas versiones de las mismas, que se detallan en la Tabla 1.

Tabla 1. Descripción de los Estándares Mundiales de TDT

Fuente: (pisciotta, Liendo, & Lauro,2013)

Estándar	Descripción	Origen
ATSC	Advanced Television Systems Committee	Estado Unidos
DVB-T	Digital Video Broadcasting – Terrestrial	Europa
ISDB-T	Integrated Services of Digital Broadcasting - Terrestrial	Japón
ISDB-Tb	ISDB-T con modificaciones propuestas por Brasil	Brasil
DMB-T	Digital Multimedia Broadcasting - Terrestrial	China

2 Estándar ISDB-Tb

El estándar japonés ISDB-T, desarrollado por la Asociación de Empresas e Industrias de Radio de ese país (ARIB), está pensado esencialmente para transmisiones terrestres de televisión ya que ofrece el establecimiento de jerarquías en el transporte de señales, brindando varios parámetros de calidad en las mismas. Esta norma permite la transmisión de gráficos, textos, programas informáticos, guías de programación, además del audio y video propios de la señal; esto se logra al emplear una Multiplexación por División de Frecuencia Ortogonal con transmisión en banda segmentada, denominada BST-OFDM, misma que también es aprovechada en la difusión de señales compatibles con dispositivos móviles y portátiles, utilizando solo uno de los segmentos del canal.

En el año 2006, el Gobierno brasileño adopta el estándar japonés ISDB-T como norma para la emisión de la TDT, posteriormente se realizaron algunas modificaciones con la finalidad de adecuar el estándar a las necesidades tecnológicas, políticas y económicas del país; es así que en el año 2007 se implementó el Sistema Brasileño de Televisión Digital Terrestre o ISDB-T Internacional. Junto con ésta implementación se firmaron acuerdos de colaboración técnica entre los gobiernos involucrados, uno de los cuales consiste en difundir el estándar en toda Sudamérica con el objetivo de promover investigaciones relacionadas con la televisión; bajo esta consigna, casi todos los países de la región han adoptado el estándar Nipo- Brasileño (Pisciotta, Liendo, & Lauro, 2013).

2.1 Características y especificaciones técnicas del estándar ISDB-Tb

La organización por capas del sistema ISDB-Tb permite que las acciones realizadas en las capas inferiores sirvan de base para cada una de las superiores; es así que la estructura del estándar comienza desde la capa de transmisión, continua con la de multiplexación, la de compresión, luego la middleware y finalmente la de aplicación, esta distribución se indica más detallada en la Figura 3 (DiBEG, 2008).

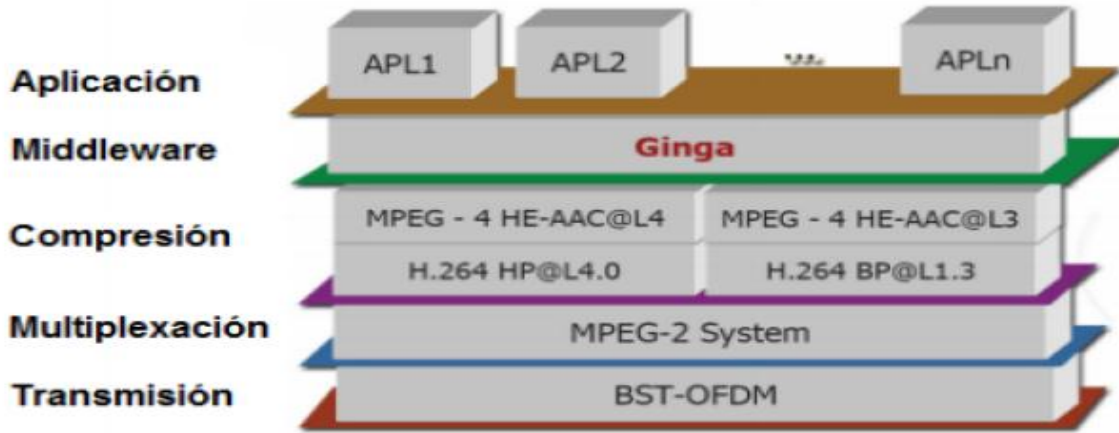


Figura 3. Estructura del estándar ISDB-Tb

Fuente: (DiBEG, 2007)

2.1.1 Capa de Transmisión

Permite la transmisión de información digitalizada a través del espectro disponible; el ancho de banda asignado para un canal de televisión en Bolivia es de 6 MHz, mismos que para la aplicación del estándar deben ser divididos en 14 segmentos, cada uno de los cuales tienen un ancho de banda de 428.57 KHz, dando origen a la denominada BST-OFDM u OFDM con Transmisión de Banda Segmentada.

La distribución de los 14 segmentos en los 6 MHz de ancho de banda del canal se muestra en la Figura 4, en la que se puede apreciar que solo 13 de los 14 segmentos son utilizables ya que uno fue repartido entre los intervalos de guarda superior e inferior del canal, el segmento central está destinado para la difusión de señales hacia receptores móviles, mientras que los 12 segmentos restantes se destinan para la transmisión de señales hacia dispositivos fijos.

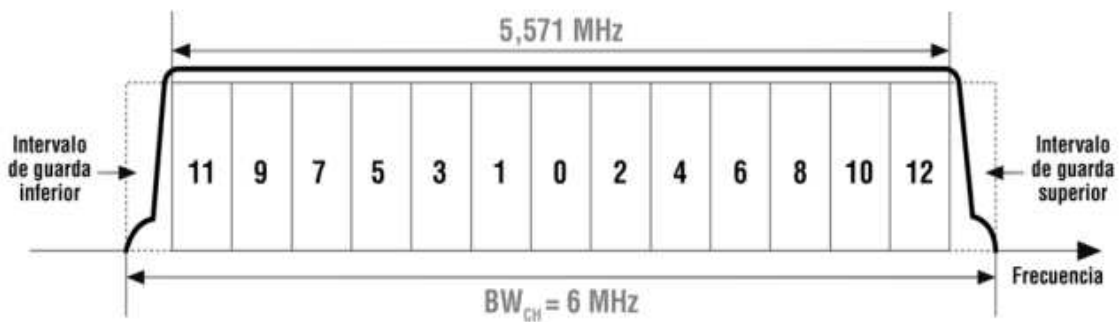


Figura 4. Segmentación del ancho de banda disponible en un canal de 6MHz.

Fuente: (Pisciotta, Liendo, & Lauro, 2013)

2.1.2 Capa de Multiplexación

Es también llamada capa de transporte y es la encargada de generar un único flujo de transporte de información de señales de audio, video y datos denominado Transport Stream (TS), este flujo contiene una secuencia digitalizada de dichas señales, que en un principio se manejan con tasas de bits bastante elevadas y que después del proceso de codificación se reducen y se forma el primer TS. En el caso de que se transmitan varias programaciones se generan varios TS adicionales, mismos que son combinados en el multiplexor, obteniendo a su salida un flujo único llamado Transport Stream MPEG-2.

Finalmente, y para poder enviar la información hacia la planta transmisora, se re-multiplexa el flujo de transporte de entrada con paquetes de transporte nulos, con la finalidad de mantener una tasa de transmisión constante, obteniéndose a la salida un Broadcast Transport Stream. Este proceso del que resulta el BTS se muestra gráficamente en la Figura 5 (Asociación Brasileña de Normas Técnicas, 2007).

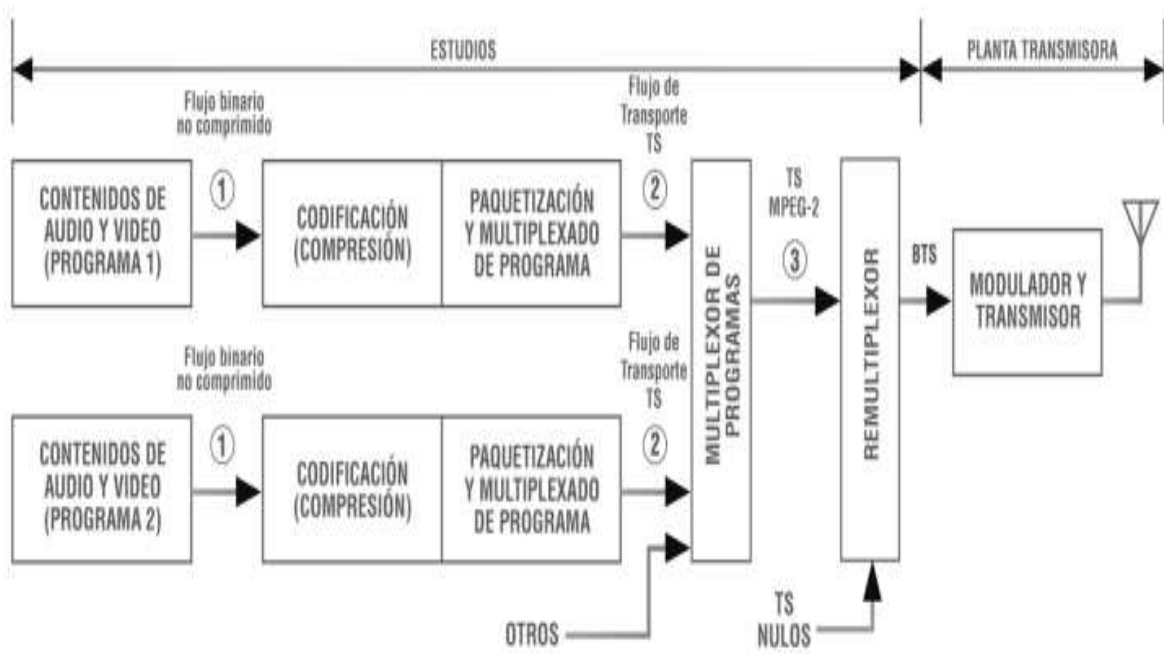


Figura 5. Generación del flujo de transporte de difusión a transmitirse

Fuente: (Asociación Brasileña de Normas Técnicas, 2007)

Al realizar la multiplexación de los datos no se puede usar este mismo tipo de multiplexación ya que el comportamiento del usuario al solicitar los datos es aleatorio y no determinístico, es decir,

no se puede saber en qué momento un televidente va a presionar un botón de control remoto para iniciar una aplicación de interactividad. Este problema lo soluciona el middleware Ginga poniendo los datos en forma de carrusel. Este nombre se refiere a un conjunto de medios que son transmitidos en forma secuencial y repetidamente ya que es probable que un usuario no este sintonizando un canal cuando este lanza una aplicación. Al transmitirse continuamente los datos se garantiza que en algún instante el receptor se encontrara con el código.

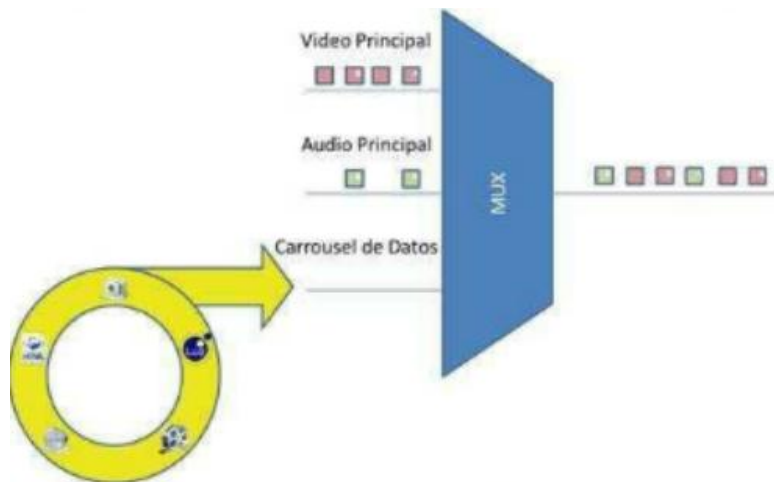


Figura 6. Multiplexación de audio, video y datos

Fuente: (DiBEG, 2008)

2.1.3 Capa de Compresión

Se encarga de eliminar redundancias temporales y espaciales de las señales audiovisuales, disminuyendo en gran medida la cantidad de datos necesarios para su almacenamiento y transmisión, sin que esto signifique una pérdida en la calidad de las señales, para esto se utiliza la técnica MPEG-4 de compresión, misma que organiza los objetos según su frecuencia de repetición, como son: imágenes fijas, música ambiental, personas hablando, etc.

Específicamente, para la compresión de video se utiliza el formato H.264/MPEG-4, ya que ofrece una calidad de imagen nítida durante la visualización en tiempo real; por su parte, en la compresión de audio se utiliza la variante MPEG-4:HE-ACC, que mejora la codificación de audio en receptores fijos y móviles (Ochoa Domínguez, Mireles García, & Cota Ruíz, 2007).

2.1.4 Capa Middleware

Corresponde al software intermedio entre el código de las aplicaciones y la infraestructura de ejecución (hardware y el sistema operativo). Permite la integración de todas las capas inferiores de la estructura, así como la ejecución de un sin número de utilidades para el desarrollo de contenidos y aplicaciones interactivas, independientemente del tipo de receptor y plataforma que se utilice. El middleware oficial del estándar ISDB-Tb es el denominado Ginga (Barba Cherrez, 2018).

2.1.5 Capa de Aplicación

Es la capa que permite la ejecución y presentación de las señales de audio, video y datos, en televisores que cuenten con sintonizador ISDB-T integrado, Set-Top Box, o dispositivos móviles compatibles con el estándar (Barba Cherrez, 2018).

2.2 Middleware GINGA

El Middleware Ginga para la norma ISDB-Tb surge con el desarrollo de proyectos de investigación coordinados por los laboratorios Telemídia en la PUC-Rio y LAViD en la UFPB. Ginga está formado por un conjunto de tecnologías estandarizadas e innovaciones brasileñas que lo convierten en la especificación middleware más avanzada (Sitio Oficial de Middleware Ginga, 2007).

Ginga nació como software libre por la necesidad de realizar una inclusión social/digital en la totalidad de hogares, permitiendo que todos tengan acceso al aprendizaje, servicios sociales, información, entre otros por medio de su televisor (Rodríguez Ortega, 2017).

2.2.1 Arquitectura Ginga

La arquitectura de Ginga se divide en tres subsistemas Ginga-NCL, Ginga-JAVA y Ginga-CC. Estas capas permiten que los sistemas, que pueden ser del tipo declarativo o imperativo, se conecten con los diferentes televisores o decodificadores existentes en el mercado para brindar el servicio de interactividad al usuario.

Da soporte al desarrollo de aplicaciones tanto empleando un paradigma declarativo, imperativo o ambos. Los dos ambientes de ejecución son exigidos en los receptores fijos y portátiles, mientras que solo el ambiente declarativo es exigido en los receptores portátiles (Sitio Oficial de Middleware Ginga, 2007).

Las librerías incorporadas le permiten ser multiplataforma, por tal razón no existen inconvenientes al momento de desarrollar un sistema en cualquier ambiente de desarrollo porque Ginga se encarga de las conexiones.



Figura 7. Arquitectura del Middleware Ginga

Fuente (Asociación Brasileña de Normas Técnicas, 2007)

2.2.1.1 Ginga - NCL

Es una categoría del middleware Ginga que realiza el procesamiento de documentos escritos en el lenguaje de programación NCL. Por lo que permite la ejecución de aplicaciones interactivas basadas en dicha plataforma de programación. Este subsistema tiene un motor que decodifica los ficheros declarativos denominado Formateador.

Ginga-NCL debe ofrecer soporte a dos lenguajes procedurales, como son LUA y JAVA. Lua es el lenguaje script de NCL, y Java debe seguir las especificaciones de Ginga-J.

El ambiente declarativo es en sí muy limitado. Las aplicaciones que utilicen éste deben tener su enfoque sobre el sincronismo, siendo el foco del lenguaje NCL exactamente eso, no la interactividad, ya que la interacción es tratada como resultado de la sincronización.

2.2.1.2 Ginga - J

La parte imperativa del sistema Ginga es manejada por JAVA, este es un lenguaje que se programa de acuerdo a algoritmos y lo más importante es que también está equipado de librerías que complementan a Ginga para el desarrollo de sistemas de televisión digital.

2.2.1.3 Ginga Common Core

Se le denomina al núcleo común del middleware Ginga. Su función es interpretar las API's para la ejecución de aplicaciones interactivas escritas tanto en el lenguaje NCL como en Java. Dentro de esta capa se encuentra el sintonizador de canales, terminal gráfico, sistema de almacenamiento, entre otros. Este núcleo contiene los decodificadores de los formatos PNG, JPEG, MPEG, entre otros (Rodríguez Ortega, 2017)

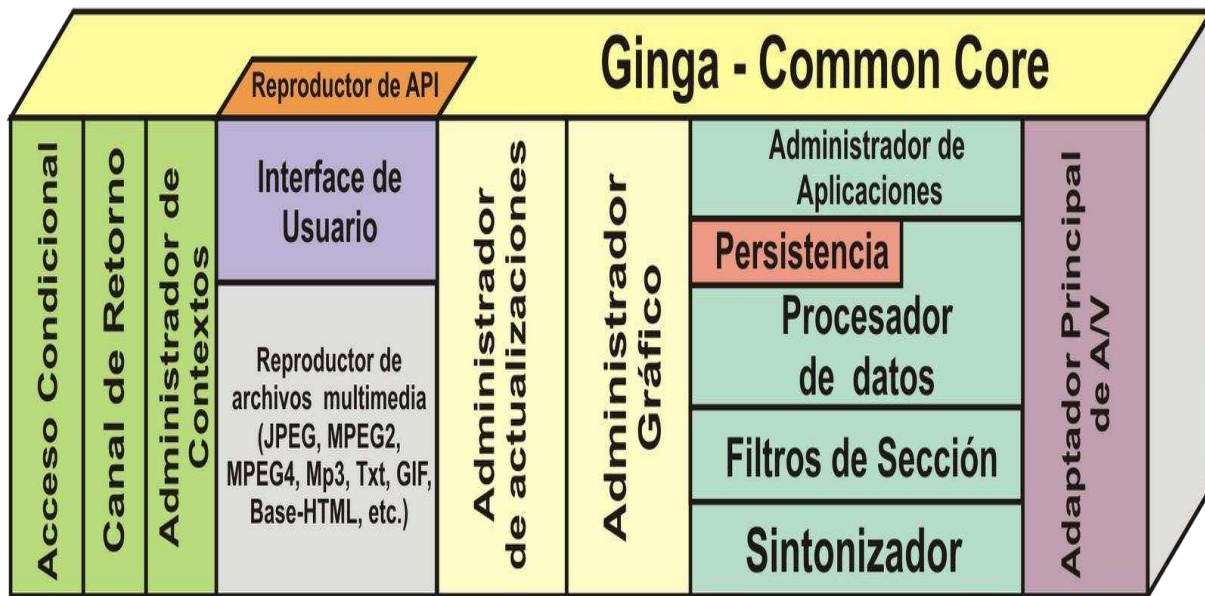


Figura 8. Estructura de Ginga CC

Fuente: (Asociación Brasileña de Normas Técnicas, 2007)

- **Sintonizador:**

Es el responsable de sintonizar un canal, seleccionando un canal físicamente dentro del Transport Stream enviado.

- **Filtro de secciones:**

Una vez sintonizado, el middleware debe acceder a partes específicas del Transport Stream; para esto existe el Filtro de Secciones, capaz de buscar en el flujo lo necesario para su ejecución.

- **Procesador de datos:**

Es el elemento responsable de acceder, procesar y reenviar los datos recibidos por la capa física.

- **Persistencia:**

Una vez finalizado el proceso, se almacenan y guardan los archivos para poder reutilizarlos.

- **Iniciador de aplicaciones:**

Gestiona las aplicaciones, carga, configura e inicializa cualquier aplicación, tanto declarativa como imperativa.

- **Gestor de gráficos:**

Los estándares del middleware definen como se deben mostrar las imágenes, videos y los datos a los usuarios, esto gestionando las presentaciones.

- **Gestor de actualizaciones:**

Gestiona las actualizaciones del sistema, verificando, y descargando las actualizaciones del middleware.

- **Visualizador de medios:**

Proporciona herramientas necesarias para visualizar los archivos multimedia recibidos (Barba Cherez, 2018).

2.2.2 Funcionamiento

El funcionamiento de Ginga viene determinado por los subsistemas en los que se divide. Trabaja con diferentes lenguajes de programación que le ayudan a soportar aplicaciones distribuidas y multiplataforma.

La señal digital es una de las causantes de la interactividad en aplicaciones desarrolladas con el middleware Ginga, porque permite el envío de datos, audio y video. Pero el inconveniente es que el envío se produce en un solo sentido, es en este punto donde entra el middleware transformando los datos recibidos y generando una respuesta que será enviada por un canal de retorno.

El canal de retorno funciona gracias al middleware, el cual permite que las aplicaciones trabajen con los datos que se reciben.

Estos datos pueden llegar con el audio y video de la señal digital o pueden ser incrustados desde un servidor externo, de esta forma se logra tener aplicaciones que no necesariamente deben ser aplicadas a la señal digital, más bien son independientes de esta y funcionan bajo sus propios medios, todo se logra gracias al poderoso middleware Ginga.

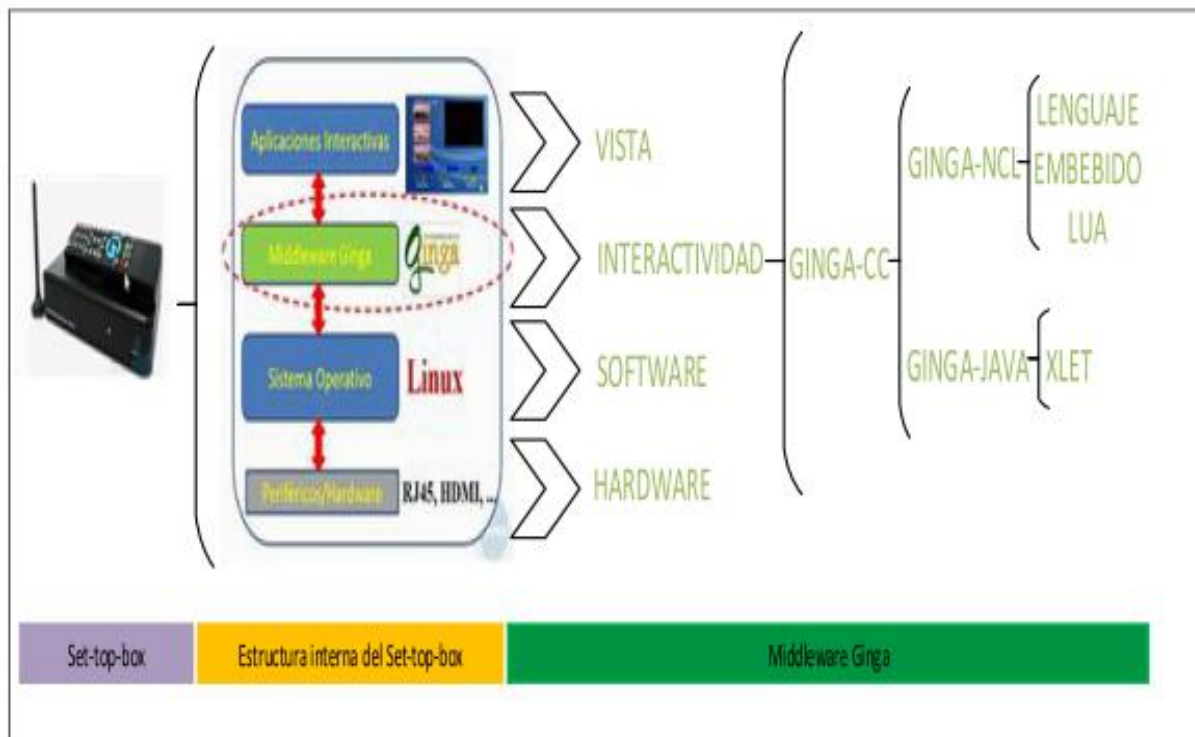


Figura 9. Funcionamiento de GINGA.

Fuente: (Rodríguez Ortega, 2017)

2.3 NCL, Nested Context Language

NCL es un lenguaje declarativo XML (lenguaje de marcado extensible) basado en el modelo conceptual NCM (modelo de contexto anidado). Además, NCL tiene la facilidad para especificar los aspectos de la interactividad, en tiempo-espacio entre los objetos de las medias, posee adaptabilidad, soporte a múltiples dispositivos. Tiene la ventaja adicional en el uso del lenguaje de script Lua, para la manipulación de sus variables, mediante la adopción del paradigma imperativo, ser eficiente, rápido y ligero y diseñados para extender las aplicaciones.

2.3.1 Estructura de un documento NCL

Todo contenido de un documento NCL está definido dentro del elemento <ncl> siendo su estructura dividida en dos grandes partes, la cabecera <head> y el cuerpo del texto <body>.

Encabezado de archivo NCL	<pre> 1. <?xml version="1.0" encoding="ISO-8859-1"?> 2. <ncl id="ejemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd"></pre>	1
Encabezado de programa	3. <head>	
Base de regiones	<pre> 4. <regionBase> 5. <!-- regiões da tela onde as mídias são apresentadas --> 6. </regionBase></pre>	2
Base de descriptores	<pre> 7. <descriptorBase> 8. <!-- descritores que definem como as mídias são apresentadas --> 9. </descriptorBase></pre>	3
Base de conectores	<pre> 10. <connectorBase> 11. <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12. </connectorBase></pre>	8
Cuerpo de programa	13. </head>	
Punto de entrada al programa	14. <body>	
Contenido de programa	<pre> 15. <port id="pInicio" component="ncPrincipal" interface="iInicio"/> 16. <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos --></pre>	5, 6, 7, 4
	17. </body>	
Termino	18. </ncl>	

Figura 10. Estructura Básica de un Documento NCL.

Fuente: (Galabay Teolongo & Vivar Espinoza, 2012)

- Archivo de encabezado NCL (líneas 1 y 2).
- Una sección del encabezado (sección head, las líneas 3 a 13), que define las regiones, los descriptores, los conectores y las reglas utilizadas por el programa.
- El cuerpo del programa (sección body, líneas 14 a 17), donde se definen los contextos, en los nodos de media, enlaces y otros elementos que describen el contenido y la estructura del programa.
- Al menos una de las puertas que indica dónde el programa comienza a ser exhibido (puerto pInicio, línea 15).
- La terminación del documento (línea 18).

En un documento NCL se debe incluir obligatoriamente las instrucciones de procesamiento, es decir el encabezado básico del programa. Éstas identifican documentos NCL que contengan sólo los elementos definidos en esta Norma, y la versión NCL con la cual el documento está de acuerdo.

```
<?xml versión="1.0" encoding="ISO-8859-1 "?>
<ncl id="qualquer string" xmlns="http://www.ncl.org.br/NCL3.0/proleName">
```

El atributo id del elemento <ncl> puede recibir cualquier cadena de caracteres como valor. El número de versión de una especificación NCL consiste en un número principal y otro secundario separados por un punto. Los números son representados como una cadena de caracteres formada por números decimales, en la cual los ceros a la izquierda se suprimen. El número de versión inicial del estándar es 3.0.

2.3.2 Módulos de NCL

Un módulo es un conjunto de elementos relacionados que representan una unidad utilizable. A continuación, se muestra los módulos del NCL dependiendo del área de funcionalidad. Existen 14 áreas funcionales, que se dividen nuevamente en módulos, y son los siguientes:

Tabla 2. Áreas Funcionales.

Fuente: (Asociación Brasileña de Normas Técnicas, 2007)

Área Funcional	Módulos
Structure	Módulo Structure
Layout	Módulo Layout
Components	Módulo Media Módulo Context
Interfaces	Módulo MediaContentAnchor Módulo CompositeNodeInterface Módulo PropertyAnchor Módulo SwitchInterface
Presentation Specification	Módulo Descriptor
Linking	Módulo Linking

Connectors	Módulo ConnectorCommonPart Módulo ConnectorAssessmentExpression Módulo ConnectorCausalExpression Módulo CausalConnector Módulo CausalConnectorFunctionality Módulo ConnectorBase
Presentation Control	Módulo TestRule Módulo TestRuleUse Módulo ContentControl Módulo DescriptorControl
Timing	Módulo Timing
Reuse	Módulo Import Módulo EntityReuse Módulo ExtendedEntityReuse
Navigational Key	Módulo KeyNavigation
Animation	Módulo Animation
Trasition Effects	Módulo TransitionBase Módulo Trasition
Meta	Information

En las tablas se emplean los siguientes símbolos: (?) opcional (zero o una ocurrencia), (|) o, (*) zero o más ocurrencias, (+) una o más ocurrencias, lo que indica el número de ocurrencias que debe tener cada atributo en un programa NCL.

2.3.2.1 Área funcional Estructure.

Esta área funcional consta de un solo modulo (structure) que presenta el formato básico que debe tener un documento NCL para poder ser correctamente exhibido.

El ncl es el encabezado del documento NCL, head es el encabezado del documento NCL y body es el contenido del documento NCL.

Tabla 3. Módulo de Estructura Extendida.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
ncl	<i>id, title, xmlns</i>	(head?, body?)
head		(importedDocumentBase?, ruleBase?, transitionBase?, regionBase*, descriptorBase?, connectorBase?, meta*, metadata*)
body	<i>id</i>	(port property media context switch link meta metadata)*

2.3.2.2 Área funcional Layout.

Esta área funcional está formada por el módulo layout, en el cual se especifica cómo se va a presentar los elementos y atributos en las diferentes regiones de los dispositivos que se van a utilizar para presentar el documento NCL.

Tabla 4. Módulo Layout Extendida.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
regionBase	<i>id, device, region</i>	(importBase region)+
region	<i>id, title, left, right, top, bottom, height, width, zIndex</i>	(region)*

2.3.2.3 Área funcional Components.

Esta área funcional está compuesta por dos módulos, denominados Media y Context. En general este módulo define componentes de un documento NCL, llamando componente a todo elemento que pueda ser iniciado para mostrar algún tipo de contenido ya sea éste audio-visual (en el caso del módulo de Media), o lógico (en el caso del módulo de Context).

- **Media:** el módulo se define utilizando la etiqueta <media> y sus atributos para representar de alguna manera algún contenido físico de media.

Tabla 5. Módulo de Media Extendida.

Fuente: (ABNT NBR 15606-2, 2009)

Elements	Attributes	Content
media	<i>id, src, refer, instance, type, descriptor</i>	(area property)*

- **Context:** el módulo especifica la etiqueta <context> que es el responsable por la definición de contextos internos en documentos, una etiqueta <context> especifica un conjunto de links y medias que no serán accesibles en el cuerpo global del documento, permitiendo el acceso solamente cuando el contexto fuera inicializado.

Tabla 6. Módulo de Contexto Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elements	Attributes	Content
context	<i>id, refer</i>	(port property media context link switch meta metadata)*

2.3.2.4 Área funcional Interfaces

Esta área funcional trata acerca de las interfaces de los objetos <media>, <switch>, <context>, etc., por medio de estas se puede acceder a estos para realizar sincronizaciones tiempo-espacio. Está compuesto por 4 módulos.

- **MediaContentAnchor:** Permite definir interfaces en los elementos <medias>.

Tabla 7. Módulo de MediaContentAnchor.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
area	<i>id, coords, begin, end, text, position, first, last, label</i>	vacío

- **CompositeNodeInterface:** Permite definir interfaces en los elementos <context> y <switch>.

Tabla 8. Módulo de CompositeNodeInterface Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
port	<i>id, component, interface</i>	vacío

- **PropertyAnchor:** En este módulo se definen las propiedades de estas interfaces.

Tabla 9. Módulo de PropertyAnchor Extendido

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
property	<i>name, value</i>	vacío

- **SwitchInterface:** Permite definir algunas interfaces especiales en los <switch>

Tabla 10. Módulo de SwitchInterface Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
switchPort	<i>id</i>	mapping+
mapping	<i>component, interface</i>	vacío

2.3.2.5 Área funcional de Especificación de Presentación.

Esta funcionalidad específica el módulo Descriptor. Este módulo es importante para la visualización de contenido en los documentos NCL, ya que especifica la etiqueta <descriptor> que contiene toda la información necesaria para que las medias puedan ser correctamente exhibidas. Para que la definición de la etiqueta <descriptor> sea hecha, es necesaria la definición de la etiqueta

<descriptorBase> que contiene un conjunto de <descriptor> y está definido dentro de la etiqueta <head> de un documento NCL. La etiqueta tiene un atributo "id" que ha de ser único en un documento, por medio de este atributo y la etiqueta <media> se consigue referenciar la etiqueta <descriptor> a través de su atributo "descriptor".

Tabla 11. Módulo del Descriptor Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
descriptor	<i>id, player, explicitDur, region, freeze, moveLeft, moveRight, move Up, moveDown, focusIndex, focusBorderColor, focusBorderWidth, focusBorderTransparency, focusSrc, focusSelSrc, selBorderColor, transIn, transOut</i>	(descriptorParam)*
descriptorParam	<i>name, value</i>	vacio
descriptorBase	<i>id</i>	(importBase descriptor descriptorSwitch)+

2.3.2.6 Área funcional Linking.

El área funcional Linking define el módulo Linking, responsable de la definición de los eslabones, que utilizan conectores. Un elemento <link> puede tener un atributo id, que es identificado dentro del documento y debe tener obligatoriamente un atributo xconnector, que se refiere al URI de un conector hipermedia. La referencia debe tener obligatoriamente el formato alias#connector_id, o documentURI_value#connector_id, para conectores definidos en un documento externo, o simplemente connector_id, para conectores definidos en el propio documento.

El elemento <link> contiene elementos hijos denominados <bind>, que permiten asociar nodos a papeles (roles) del conector. Para hacer esta asociación, un elemento <bind> tiene cuatro atributos básicos.

- El primero se denomina role, que se utiliza para hacer referencia a un papel del conector.
- El segundo se denomina component, que se utiliza para identificar el nodo.
- El tercero es un atributo opcional denominado interfaz, usado para hacer referencia a una interfaz del nodo.

- El cuarto es un atributo opcional denominado descriptor, usado para hacer referencia a un descriptor a ser asociado con el nudo, como se definió en el módulo Descriptor.

Si el elemento conector define parámetros, conviene a los elementos <bind> o <link> definir valores para esos parámetros, a través de sus elementos-hijos denominados <bindParam> y <enlaceParam>, respectivamente, ambos con atributos name y value. En ese caso, el atributo name debe obligatoriamente hacer referencia al nombre de un parámetro del conector, mientras que el atributo value debe definir obligatoriamente un valor a ser atribuido al respectivo parámetro

Los elementos del módulo Linking, sus atributos y sus elementos hijos deben estar de acuerdo obligatoriamente con la Tabla 12.

Tabla 12. Módulo Linking Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
bind	<u>role</u> , <u>component</u> , <u>interface</u> , <u>descriptor</u>	(bindParam)*
bindParam	<u>name</u> , <u>value</u>	vacío
linkParam	<u>name</u> , <u>value</u>	vacío
link	<u>id</u> , <u>xconnector</u>	(linkParam*, bind+)

2.3.2.7 Área funcional Connectors.

Esta es el área funcional del núcleo del lenguaje NCL. Donde se define en si las relaciones espacio temporales de los componentes. Está compuesto por 7 módulos básicos, de los cuales 4 son los más esenciales (Connector Common Part, Connector Assessment Expresion, Connector Causal Expresion, Causal Connector) que se encuentran comprendidos en el módulo CausalConnectorFunctionality.

Los elementos del módulo CausalConnectorFunctionality, sus elementos-hijos y sus atributos deben estar de acuerdo obligatoriamente con la Tabla 13.

Tabla 13. Módulo del CausalConnectorFunctionality Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
causalConnector	<u>id</u>	(connectorParam*, (simpleCondition compoundCondition), (simpleAction compoundAction))
connectorParam	<u>name</u> , <u>type</u>	vacío
simpleCondition	<u>role</u> , <u>delay</u> , <u>eventType</u> , <u>key</u> , <u>transition</u> , <u>min</u> , <u>max</u> , <u>qualifier</u>	vacío
compoundCondition	<u>operator</u> , <u>delay</u>	((simpleCondition compoundCondition)+, (assessmentStatement compoundStatement)*)
simpleAction	<u>role</u> , <u>delay</u> , <u>event Type</u> , <u>action Type</u> , <u>value</u> , <u>min</u> , <u>max</u> , <u>qualifier</u> , <u>repeat</u> , <u>repeatDelay</u> , <u>duration</u> , <u>by</u>	vacío
compoundAction	<u>operator</u> , <u>delay</u>	(simpleAction compoundAction)+
assessmentStatement	<u>comparator</u>	(attributeAssessment, (attributeAssessment valueAssessment))
attributeAssessment	<u>role</u> , <u>eventType</u> , <u>key</u> , <u>attributeType</u> , <u>offset</u>	vacío
valueAssessment	<u>value</u>	vacío
compoundStatement	<u>operator</u> , <u>isNegated</u>	(assessmentStatement compoundStatement)+

Un elemento <causalConnector> representa una relación causal que puede ser utilizada por elementos <link> en documentos. En una relación causal, una condición debe ser obligatoriamente satisfecha para disparar una acción.

2.3.2.8 Área funcional de Control de Presentación.

Esta área es la encargada de detallar opciones y contenido de los documentos NCL, consta de 4 módulos:

- **TestRule:** describe un conjunto de reglas en la etiqueta <ruleBase>, esta etiqueta debe ser definida como hija de la etiqueta <head> en un documento, Esas reglas pueden ser simples definidas por el elemento <rule>, o compuestas definidas por el elemento

<compositeRule>. Las reglas simples definen un identificador (atributo id), una variable (atributo var), un valor (atributo value) y un comparador (atributo comparator) relacionando la variable a un valor.

Tabla 14. Módulo del TestRule Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
ruleBase	<i>id</i>	(importBase rule compositeRule)+
Rule	<i>id, var, comparator, value</i>	vacío
compositeRule	<i>id, operator</i>	(rule compositeRule)+

- **TestRuleUse:** define el elemento <bindRule>, En este módulo se asocian las reglas a los elementos del documento NCL.

Tabla 15. Módulo TestRuleUse Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
bindRule	<i>constituent, rule</i>	vacío

- **ContentControl:** explica la etiqueta <switch> que permite la definición de nodos alternativos a ser elegidos en tiempo de presentación del documento. Las reglas de prueba utilizadas para escoger el componente del switch a ser presentado se definen por el módulo TestRule, o son reglas de prueba específicamente definidas e incorporadas en una implementación del formateador NCL. El módulo ContentControl también puntualiza el elemento <defaultComponent>, cuyo atributo component identifica el elemento (default) que debe ser obligatoriamente seleccionado si ninguna de las reglas bindRule es evaluada como verdadera.

Tabla 16. Módulo ContentControl Estendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
switch	<i>id, refer</i>	defaultComponent?, (switchPort bindRule media context switch)*
defaultComponent	<i>component</i>	vacío

- DescriptorControl:** define el elemento <descriptorSwitch>, que contiene un conjunto de descriptores alternativos a ser asociado a un objeto. Análogamente al elemento <switch>, la elección <descriptorSwitch> se realiza en tiempo de presentación, utilizando reglas de prueba descritas por el módulo TestRule, o reglas de prueba específicamente detalladas e incorporadas en una implementación del formateador NCL. El módulo DescriptorControl también precisa el elemento <defaultDescriptor>, cuyo atributo descriptor identifica el elemento (default) que debe ser obligatoriamente seleccionado cuando ninguna de las reglas bindRule es evaluada como verdadera.

Tabla 17. Módulo DescriptorControl Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
descriptorSwitch	<i>id</i>	(defaultDescriptor?, (bindRule descriptor)*)
defaultDescriptor	<i>descriptor</i>	vacío

2.3.2.9 Área funcional Timing

Define el módulo Timing que permite la descripción de atributos temporales para componentes de un documento, este módulo detalla atributos para especificar qué pasa con un objeto al final de su presentación (freeze) y la duración ideal de un objeto (explicitDur). Estos atributos pueden ser incorporados por los elementos <descriptor>.

2.3.2.10 Área funcional Reuse

NCL permite una gran reutilización de sus elementos mediante el área funcional Reuse que se divide en tres módulos:

- **Import:** para permitir que una base de entidades sea incorporada a otra ya existente, el módulo Import define el elemento <importBase>, que tiene dos atributos: DocumentURI y alias. El atributo documentURI se refiere a un URI correspondiente al documento NCL conteniendo la base a ser importada. El atributo alias especifica un nombre a ser utilizado como código cuando sea necesario referirse a elementos de esa base importada. EL nombre del atributo alias debe ser obligatoriamente único en un documento y su alcance está supeditado al documento que lo definió.

Los elementos del módulo Import, sus elementos-hijos y sus atributos deben estar de acuerdo obligatoriamente con la Tabla 18.

Tabla 18. Módulo Import Extendido

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
importBase	<i>alias, documentURI, región</i>	vacío
imported DocumentBase	<i>id</i>	(importNCL)+
importNCL	<i>alias, documentURI</i>	vacío

- **EntityReuse:** Este módulo permite que un documento NCL sea reutilizado.
- **ExtendedEntityReuse:** En este módulo se definen otras características para poder importar documentos NCL.

2.3.2.11 Área funcional Navigational Key

Esta área se encuentra constituida por el módulo KeyNavigation el cual permite modificar características del movimiento del foco (esto se lo define en el descriptor con el comando focusIndex). Entre los comandos más utilizados están tres: FocusBorderColor (define el color del borde del foco), focusBorderWidth (define el ancho del borde del foco) y focusBorderTransparency (define el porcentaje de transparencia del borde del foco). Si estos valores no son definidos, toman valores automáticamente en el documento NCL.

2.3.2.12 Área funcional Animation

Una animación es una combinación de dos factores: soporte al dibujo y al movimiento del objeto, o más propiamente, soporte para la alteración del objeto en función del tiempo. En NCL no se pueden crear objetos de media, pero se puede utilizar como un formato de escalonamiento y orquestación. Eso significa que NCL no se puede utilizar para hacer dibujos animados, pero se puede utilizar para exhibir objetos de dibujo animado en el contexto de una presentación general y para alterar las propiedades de sincronización y exhibición de un objeto de un dibujo animado (o cualquier otro) globalmente, mientras el objeto está siendo exhibido.

El área funcional Animation define el módulo Animation que suministra las extensiones necesarias para describir qué pasa cuando el valor de una propiedad de un nodo es alterado. Básicamente, el módulo describe atributos que pueden ser incorporados por los elementos <simpleAction> de un conector, si su valor eventType es attribution. Dos nuevos atributos son definidos: duration y by. Al atribuir un nuevo valor para una propiedad, la alteración es instantánea por default (duration=0), pero la alteración también se puede realizar durante un período explícitamente declarado, especificado por el atributo duration.

2.3.2.13 Área funcional SMIL Transition Effects

El área funcional Transition Effects se divide en dos módulos:

- **TransitionBase:** es definido por la NCL 3.0 y consiste en un elemento <transitionBase> que especifica un conjunto de efectos de transición y se debe definir obligatoriamente como elemento-hijo del elemento <head>.
- **Transition:** Solamente tiene un elemento llamado <transition>. Es especificado en el elemento <transitionBase> y permite que sea definido un estándar (template) de transición. Cada elemento <transition> define un estándar único de transición y debe tener obligatoriamente un atributo id para que pueda ser referido dentro de un elemento <descriptor>.

2.3.2.14 Área funcional Metainformation

Una meta información contiene informaciones sobre el contenido utilizado o exhibido. Esta área está compuesta por el módulo metainformation, derivado del módulo Metainformation SMIL. El

elemento <meta> especifica un único par de propiedad/valor en los atributos name y content, respectivamente.

El elemento <metadata> contiene informaciones que también se relacionan con la metainformación del documento. Actúa como el elemento raíz del árbol RDF. El elemento <metadata> puede tener como elementos-hijos: Elementos RDF y sus subelementos.

Tabla 19. Módulo Meta-Information Extendido.

Fuente: (ABNT NBR 15606-2, 2009)

Elementos	Atributos	Contenido
meta	<i>name, content</i>	vacío
metadata	<i>empty</i>	RDF tree

3 EWBS, Emergency Warning Broadcasting System

Uno de los principales beneficios ofrecidos por el estándar ISDB-Tb es el sistema de transmisión de alerta de emergencias (EWBS). El cual es otra opción para la difusión de alertas tempranas, permitiendo la emisión de señales de emergencias durante la ocurrencia de un fenómeno natural, desde la estación televisiva hasta los dispositivos receptores, independientemente si estos están encendidos o en modo de espera. Este Sistema tiene como objetivo prevenir y disminuir las pérdidas, humanas y materiales, causadas por fenómenos naturales, entregando información oportuna a la población afectada, a través de un medio de comunicación masivo como es la televisión (Barba Cherrez, 2018).

La información emitida por la estación puede presentarse a través de textos independientes o propiamente con el programa, cabe destacar que la señal EWBS está constituida por una bandera de activación (flag activation) y un descriptor de la emergencia en la tabla del mapa de la programación transmitida (PMT). Finalmente, la alerta de emergencia se muestra por superposición de datos, solamente en los receptores que se encuentren en el área de cobertura correspondiente al código indicado. Todas las funciones descritas del sistema EWBS se esquematizan en la Figura 11 (Foro Internacional ISDB-T, 2013).



Figura 11. Funciones Básicas del Sistema EWBS.

Fuente: (Foro Internacional ISDB-T, 2013)

El estándar ISDB-Tb define el uso de la señal de Control de Multiplexación y Transmisión, también llamada TMCC, en la que se especifica información como: el modo de transmisión, intervalo de guarda, esquemas de modulación y codificación, cantidad de segmentos, activación de alertas, etc., misma que es necesaria para garantizar la comunicación entre el transmisor y los receptores. Esta señal TMCC junto con otra información relacionada con la sincronización forman un único Transport Stream denominado ISDB-Tb Information Packet (IIP), que luego será re-multiplexado con los demás flujos de transporte, formando el Broadcast Transport Stream (BTS) definitivo; la estructura de los paquetes en la etapa final del BTS es la que se muestra en la Figura 12 (Pisciotta, Liendo, & Lauro, 2013).

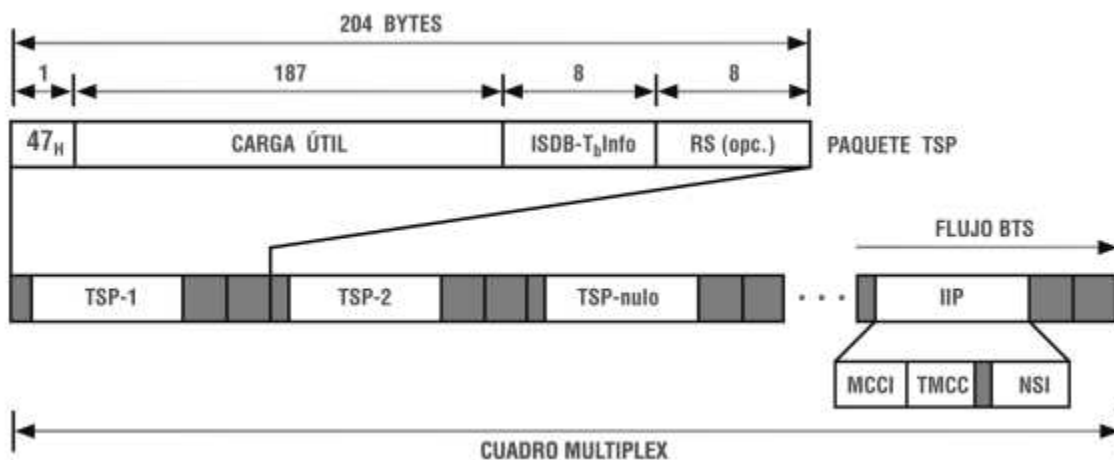


Figura 12. Estructura de los Paquetes TS, Flujo BTS y cuadro Multiplex.

Fuente: (Pisciotta, Liendo, & Lauro, 2013)

- **Bandera de activación (flag activation)**

Representa una parte de la Señal de Control de Multiplexación y Transmisión, ocupa solamente un bit de los 204 que conforman el paquete IIP, específicamente el bit 26 correspondiente al sistema de alertas de radiodifusión; los valores que pueden ser asignados son “1” y “0”, cuando está disponible el control de inicio de alerta de emergencias y cuando no se cuenta con el control, respectivamente.

- **Descriptor con información de la emergencia**

El descriptor que contiene la información relacionada con la emergencia tiene una longitud de 64 bits, de los cuales se detallara posteriormente.

3.1 Transmisión de la alerta temprana

La señal EWBS tiene su origen en la capa de transmisión del estándar ISDB-Tb, misma que debe poseer una mayor robustez que garantice una transmisión y recepción constantes. En la generación de esta señal se deben tener en cuenta las siguientes especificaciones:

- El descriptor en el que se incluye la información sobre la emergencia debe ser transferido a la PMT para todos los servicios; en este descriptor se incluirá, las condiciones de inicio y final del EWBS, así como el código de identificación de área y de servicio. Además, se debe establecer el flag activation en “1” para el Sistema de Alerta dentro de la Señal TMCC que será emitida por la estación.
- Los mensajes de texto que contienen información relacionada con la emergencia, pueden ser incluidos de manera sobrepuesta a la programación. Por su parte, los programas con información adicional deben ser transmitidos sobre el propio servicio de transporte de la señal EWBS.
- Cuando la señal de alerta se detiene, el flag activation debe cambiar su valor lógico a “0” y el descriptor con la información de la emergencia debe eliminarse de la tabla PMT.

La Figura 13 esquematiza el funcionamiento de la etapa de transmisión del sistema EWBS, mismo que ha sido incluido en la señal emitida por una estación que ofrece servicios de multiprogramación (Foro Internacional ISDB-T, 2013).

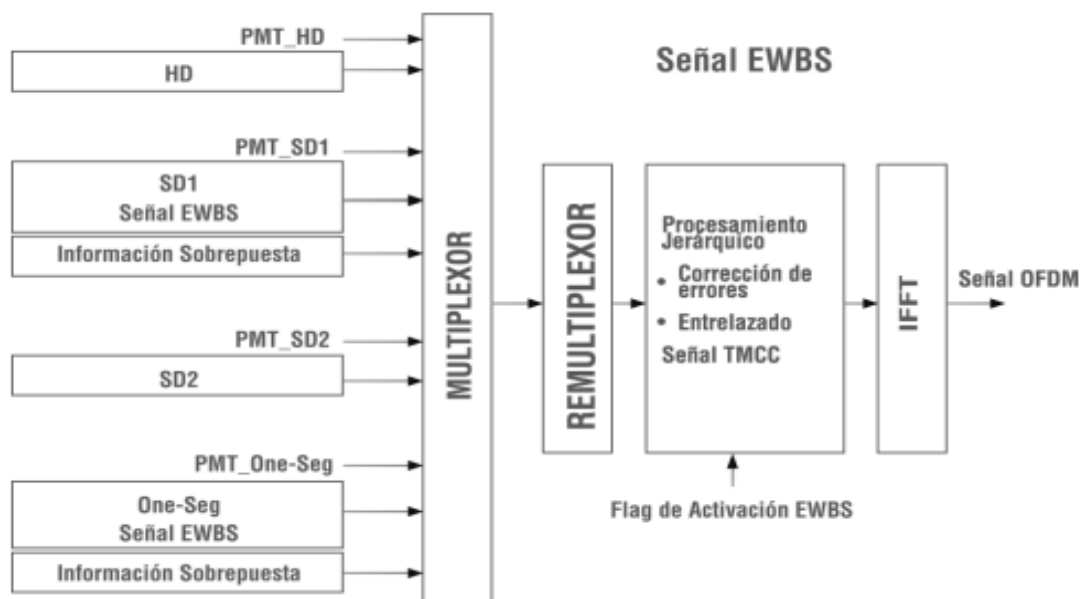


Figura 13. Diagrama de Transmisión Incluyendo una Señal EWBS.

Fuente: (Foro Internacional ISDB-T, 2013)

Para el caso de transmisión de alerta utilizando el middleware Ginga, los datos del programa deben ser especificado en la tabla PMT de la misma manera.

3.2 Esquema de transmisión de las señales de TDT

Las señales de alerta junto con los datos de las aplicaciones interactivas, deben emitirse en simultáneo con la programación habitual transmitida por las televisoras.

La estación televisiva, encargada de difundir el Sistema de Alerta Temprana propuesto, procesa el audio y video de la programación de forma independiente, sin importar que estas sean de alta, estándar o baja definición; y luego las combina para formar un único flujo binario de datos.

Al contar con tasas demasiado elevadas como para poder ser transmitidas, se aplican modelos de codificación que reducen sustancialmente el tamaño del flujo de datos, luego se los agrupa en paquetes denominados PES.

A la carga útil de cada paquete se le adhiere una cabecera con información relevante sobre las señales de audio, video y datos, luego los paquetes son seccionados en flujos de 188 bytes, generando los denominados Transport Streams (TS).

Cada flujo TS tiene 184 bytes de información propiamente dicha y 4 bytes de cabecera. La Figura 14 muestra gráficamente el proceso de generación de los paquetes TS.

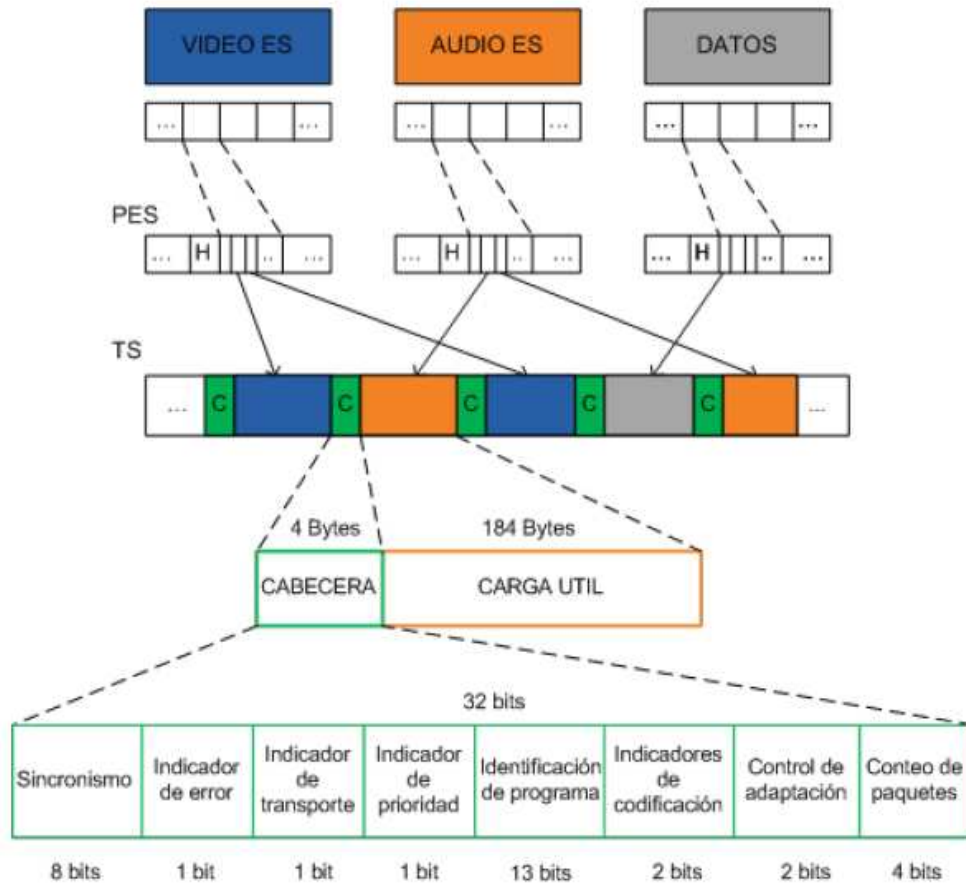


Figura 14. Conformación de los Paquetes de Flujo de Transporte.

Fuente: (Barba Cherez, 2018)

Para la ejecución del sistema EWBS se utilizan secciones que también se transmiten en paquetes normalizados de 188 bytes que, a diferencia de los de audio, video y datos, contienen información adicional de la emergencia y que está organizada en tablas Program Map Table (PMT).

En las tablas PMT se detalla información relacionada con el programa que se está transmitiendo, además contienen varios descriptores como: tipos de codificación de las señales, datos de carrusel, información de emergencia, etc. que ayudan al receptor a determinar la estructura del flujo de transporte. El receptor realiza la lectura de la tabla PMT en la carga útil del TS cuando en la cabecera del mismo, la identificación de programa toma el valor correspondiente.

La estructura de la Tabla PMT, así como la del descriptor destinado a informar sobre la situación de emergencia es la que se indica en la Figura 15; El descriptor 1 de dicha Tabla es el reservado para el Sistema EWBS, a continuación, se detalla los bloques de bits que lo conforman:

- **Descriptor_tag:** contiene el valor de la etiqueta que describe la emergencia.
- **Descriptor_length:** contiene el valor de la cantidad de bits que continúan a este bloque.
- **Service_id:** contiene la identificación del programa o evento transmitido.
- **Start_end_flag:** corresponde a la señal de inicio y final de la transmisión EWBS.
- **Signal_level:** contiene la información del tipo de emergencia, es determinado por el ente encargado de la gestión de riesgos.
- **Area_code_length:** contiene el valor de la cantidad de bits que se utilizarán en el bloque area_code.
- **Area_code:** contiene la información correspondiente al código de área en el que se difundirá la señal EWBS.

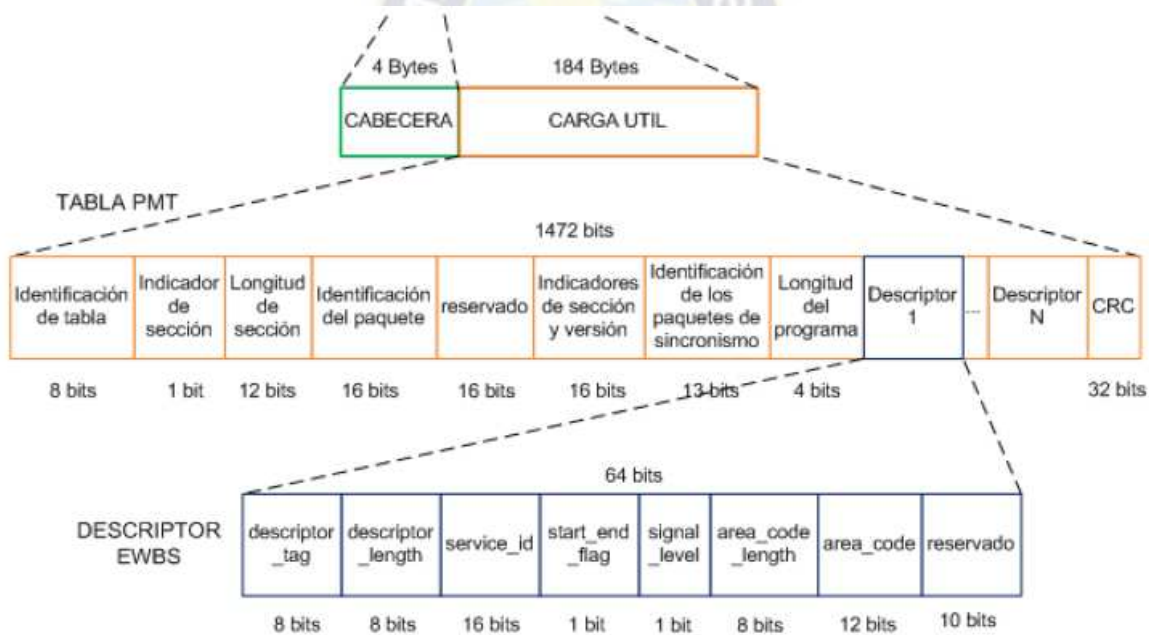


Figura 15. Estructura del Descriptor EWBS en una Tabla PMT

Fuente: (Barba Cherez, 2018)

3.3 Recepción de la señal EWBS

Las señales del sistema EWBS que llegan a los receptores, fijos y móviles, deben mostrarse en la pantalla de los mismos, ya sea con textos superpuestos a la señal propia de televisión o como información adicional contenida en ella; además los dispositivos deben contar con la capacidad de ejecutar ciertas funciones que se describen en la Tabla 20.

Tabla 20. Especificaciones de las Funciones que debe tener un Receptor ISDB-Tb

Fuente: (Foro Internacional ISDB-T, 2013)

Función	Receptores Fijos		Receptores móviles	
	Televisión con sintonizador integrado	Set-Top Box	One-seg	Exclusivo para EWBS
Inicio automático	Recomendado	Recomendado	Recomendado	Recomendado
Preset del código de área	Obligatorio	Obligatorio	Obligatorio	Obligatorio
Superposición	Obligatorio	Obligatorio	Obligatorio	Obligatorio
Decodificación del programa	Obligatorio	Obligatorio	Obligatorio	--

Ya en la etapa de recepción propiamente dicha, los receptores deben monitorear permanentemente el bit correspondiente al flag de activación del paquete IIP, aun cuando el dispositivo este en modo de espera; cuando al estado del flag cambia su valor lógico a “1” se inicia el control del descriptor de la información de emergencia en la tabla PMT, luego se verifica que el código de área sea el mismo que se programó en el receptor, de ser así, el dispositivo se enciende y la alerta de emergencia se visualiza. El estado de alerta mostrado en la pantalla finaliza cuando el flag vuelve a ser “0” y el descriptor de información se elimina de la tabla PMT (Foro Internacional ISDB-T, 2013).

3.4 EWBS por One-seg en terminales de mano.

El sistema de transmisión de alertas de emergencias cuenta con terminales de mano con la función One-seg que se activa automáticamente cuando se produce un desastre y proporciona información esencial para los usuarios. A diferencia de las redes de comunicación que tienden a cogestionarse, la información puede ser transmitida al instante a un gran número de terminales de mano One-seg. Entonces como podemos apreciar en la siguiente figura, que los terminales de mano se activen automáticamente con la emisión de una señal de alerta de emergencia, es un tema muy importante para la preparación contra desastres naturales de gobiernos naciones y locales (Espin Lucas, 2014).

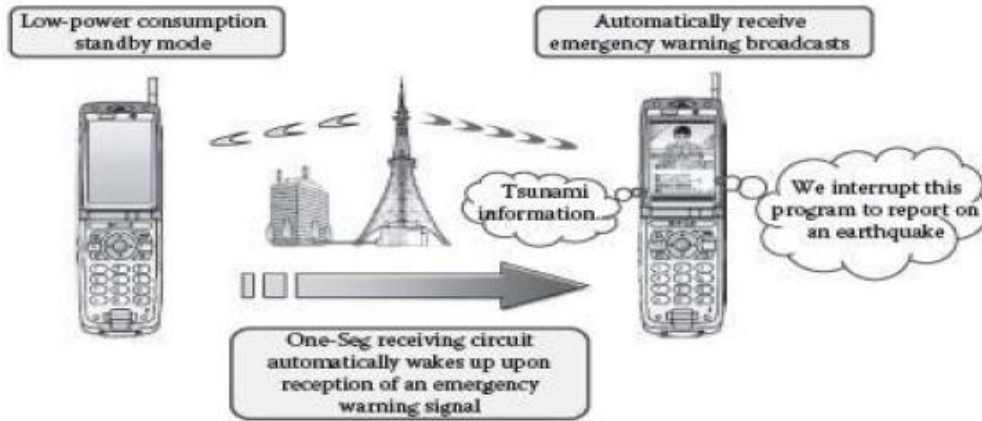


Figura 16. Activación Automática de Terminales de mano One-Seg

Fuente: (Espin Lucas, 2014)

En el estándar ISDB-Tb como se muestra en la siguiente figura, una señal de información de una alerta de emergencia es colocada en el bit 26 de TMCC. Durante la emisión de la alerta de emergencia, ese bit se convierte en 1.

Para recibir la transmisión de la alerta de emergencia, el terminal de mano debe siempre estar atento a esta señal, y cuando el bit llega a ser 1, este presenta el contenido del programa transmitido mientras que la difusión de una alarma indica que esto es la transmisión de una alerta de emergencia.

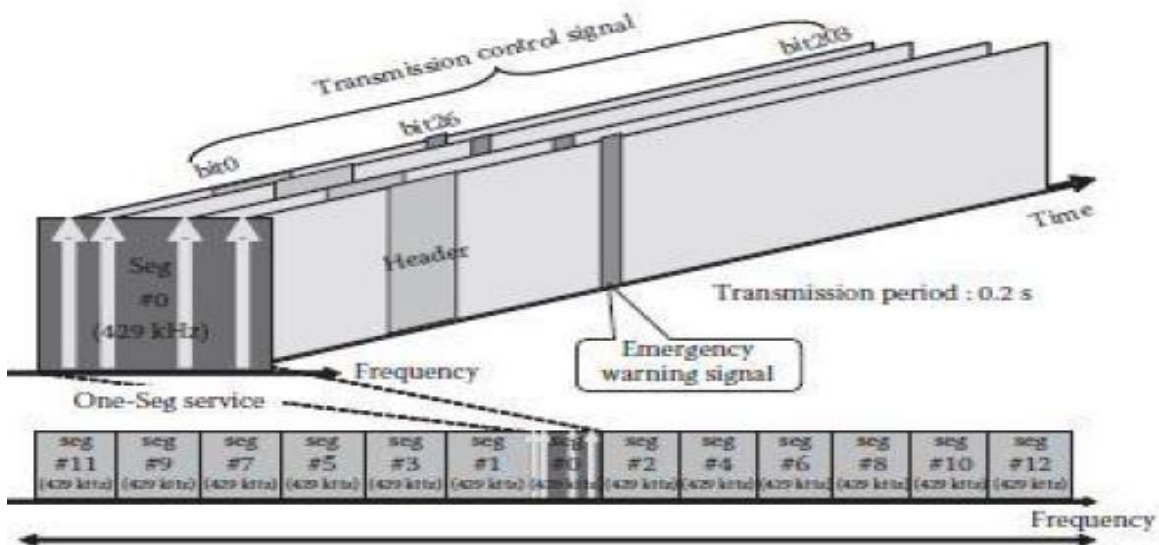


Figura 17. Señales de Alerta de Emergencia en ISDB-Tb.

Fuente: (Espin Lucas, 2014)

4 SAT en Bolivia

4.1 SNATD, Sistema Nacional de Alerta Temprana de Desastres

4.1.1 Aplicación del SAT en Bolivia

SAT (Sistema de alerta temprana), En Bolivia el SENAMHI (Servicio Nacional de Meteorología e Hidrología) entidad creada con la finalidad de realizar el monitoreo hidro-meteorológico en el país, en coordinación con la Cooperación Italiana han implementado a partir del año 2008, un sistema de comunicación y difusión de información de pronósticos y alertas hidro-meteorológicas en tiempo real.

La red de monitoreo hidro-meteorológico cuenta en la actualidad con 119 estaciones meteorológicas e hidrológicas en distintos puntos estratégicos del país (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

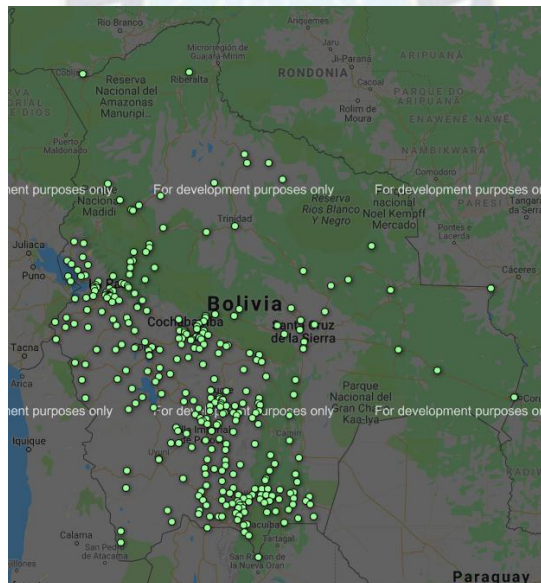


Figura 18. Estaciones Meteorológicas en Bolivia (ubicación).

Fuente: (Senamhi, s.f.)

4.1.2 Innovaciones tecnológicas del SAT nacional

En un principio, el sistema estuvo a cargo del SENAMHI quien enviaba información de alertas de amenazas y pronósticos meteorológicos a los departamentos y municipios integrados a la red. En la actualidad, se ha implementado la plataforma DEWETRA en el SENAMHI a fin de facilitar y mejorar los pronósticos y alertas. Las cuáles serán enviadas al Vice-ministerio de Defensa Civil

(VIDECI) a fin que esta instancia envíe alertas de riesgos con base a información hidrometeorológica del SENAMHI a las Gobernaciones y Municipios.

4.1.3 Comunicación para controlar el riesgo

El presente esquema explica la forma de comunicación del Sistema Nacional de Alerta Temprana (SNATD).

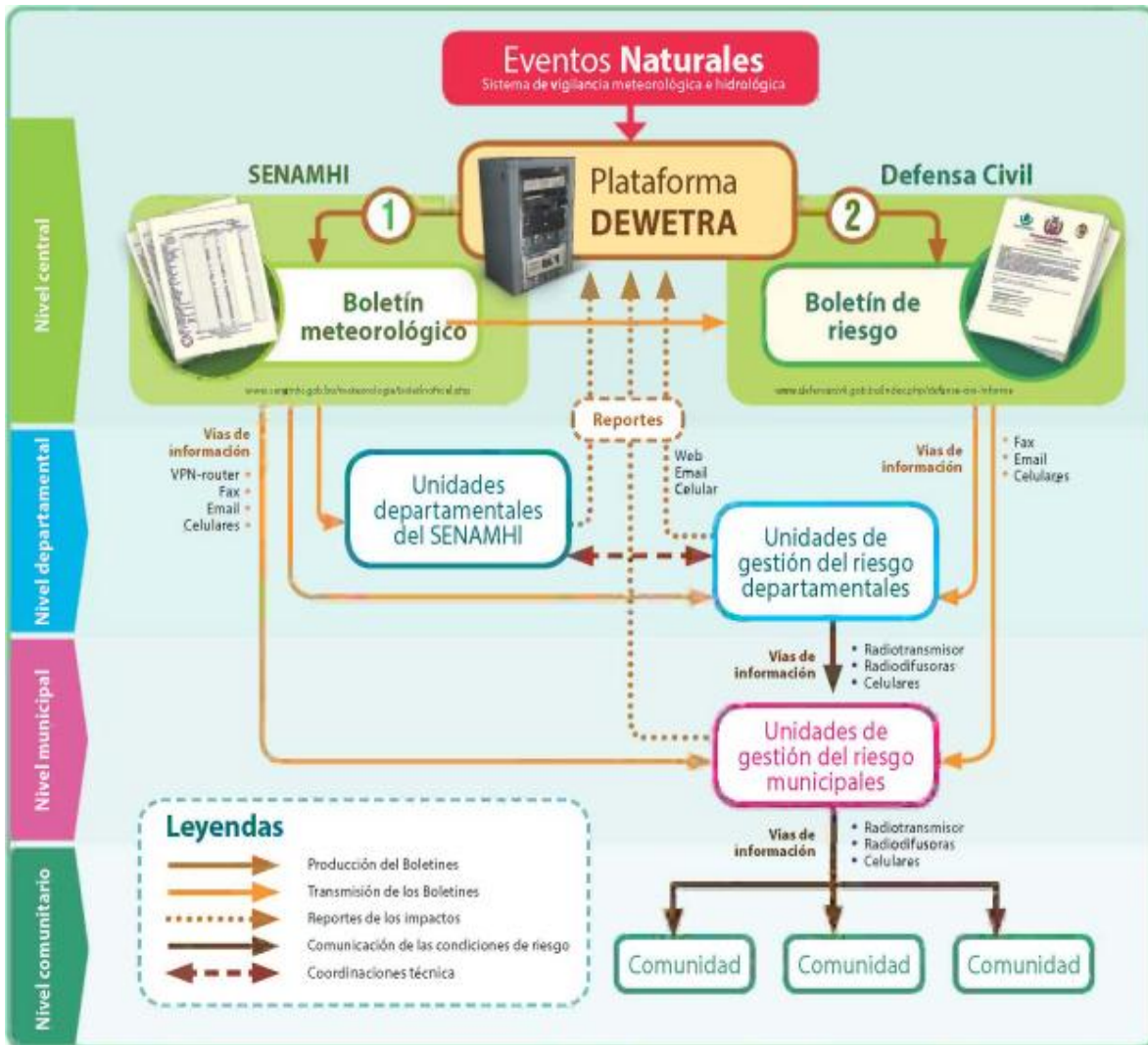


Figura 19. Sistema Nacional de Alerta Temprana de Desastres (SNATD)

Fuente: (Viceministerio Defensa Civil, s.f.)

En el flujo superior, se puede observar que el SENAMHI con el uso del DEWETRA elaborará pronósticos y alertas meteorológicas las cuales son enviadas en boletines y mensajes al VIDECCI, a

las Gobernaciones y a los Municipios a través de la red VPN, internet y celulares, asimismo a través de las oficinas regionales del SENAMHI en los nueve departamentos.

El VIDECI con el uso del DEWETRA elaborará un boletín y mensajes de criticidad (riesgo) que tome en cuenta de los posibles efectos al suelo de los fenómenos meteorológicos, el cual es enviado a las Gobernaciones y Municipios a través de Internet, fax y celulares.

4.2 Componentes del sistema de alerta temprana de desastres (SNATD)

Los componentes del sistema de alerta temprana son:

- Sistema de vigilancia meteorológica e hidrológica.
- Sistema de comunicaciones.
- Planes de prevención y contingencia.

4.2.1 Sistema de observación

El país cuenta con red de observación meteorológica e hidrológica, constituye la base o la espina dorsal de todas las actividades meteorológicas, climatológicas, agro meteorológicas e hidrológicas, proporciona la materia prima básica sin la cual no sería posible desarrollar ninguna otra actividad. La complejidad de la atmósfera, de sus propiedades físicas y químicas, hace que sean necesarias diferentes técnicas de observación que se concretan mediante el despliegue de diferentes redes, cada una de las cuales nos proporcionan una visión "parcial" de la misma y que, una vez integradas, contribuyen a proporcionarnos una visión de conjunto.

4.2.2 Sistema de comunicaciones

Con la finalidad de disponer el sistema de comunicación entre el Servicio Nacional de Meteorología e Hidrología, Defensa Civil, las UGR's (unidad de gestión de riesgo) de Gobernaciones y Municipios, se implementó el sistema permanente de comunicación WAN que permite el intercambio de información meteorológica, climatológica, agro meteorológica e hidrológica de tal forma que logró construir un sistema de alerta temprana a través de la visualización de la información de los diversos fenómenos adversos.

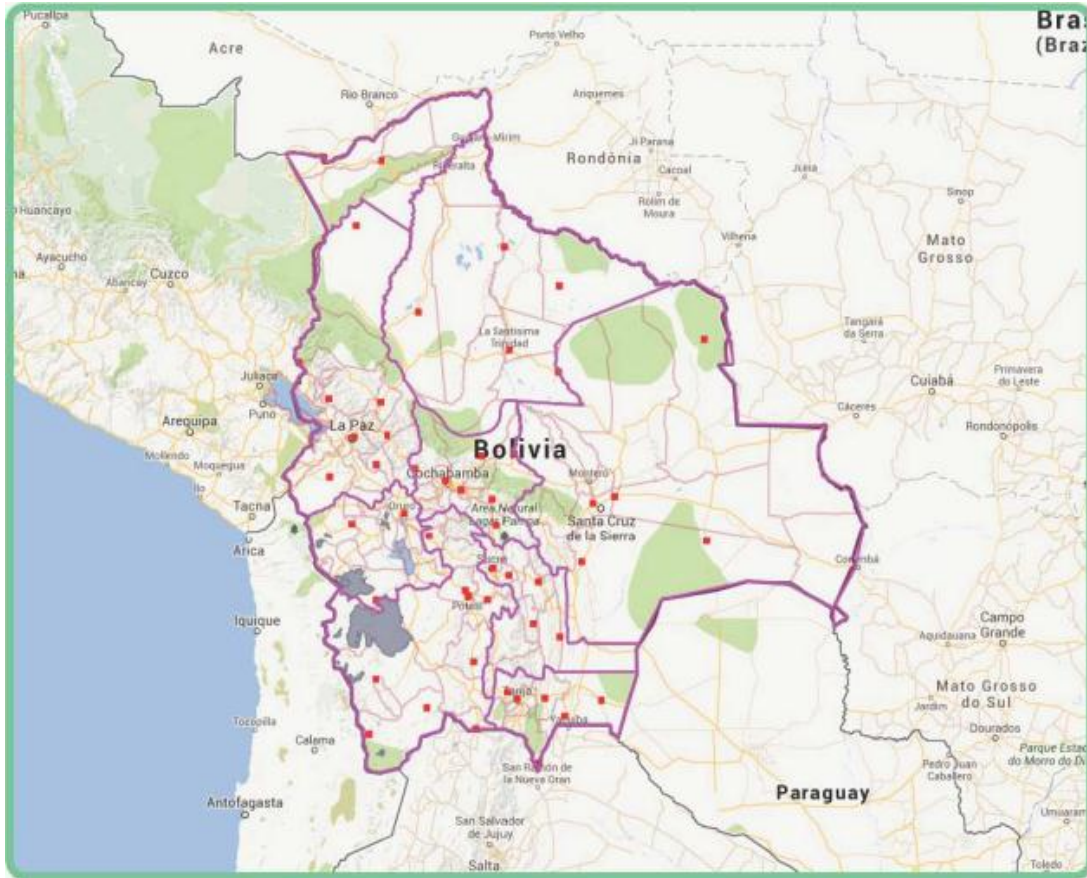


Figura 20. Sistema de Comunicación de la Red WAN.

Fuente: (Mariaca, Trujillo, Rossi, & Mendoza, 2013)

4.2.3 Plataforma DEWETRA para la gestión y prevención de desastres naturales

El establecimiento de un pronóstico fiable y de un sistema de alerta para las comunidades en situación de riesgo, requiere de la combinación de: datos, herramientas de monitoreo, pronóstico y de personal técnico preparado.

Estos componentes deben estar disponibles en cada Centro de Gestión de Emergencia, quienes deben ser capaces de emitir alertas automáticamente, estos centros además deberán estar conectados a una red de intercambio de datos, procedimientos, modelos y conocimientos técnicos. La eficacia del esquema de trabajo se basa en el rápido acceso y la disponibilidad de datos, para que el sistema de pronóstico pueda producir en tiempo real escenarios fiables. Por otra parte, la coordinación y el intercambio de datos puede aumentar significativamente la cantidad de información disponible. En este contexto, es muy importante obtener toda la información observada por las diferentes fuentes disponibles.

Los datos hidro-meteorológicos exactos, no tienen ningún valor si no llegan oportunamente a los usuarios y si no se toman las decisiones necesarias para actuar adecuadamente en consecuencia. Al igual, los resultados de los modelos de pronósticos "multi-riesgo" específicos (por ejemplo, inundaciones, incendios forestales, deslizamiento) deben estar disponibles y ser difundidos de manera oportuna para que se puedan tomar las decisiones y adoptar las medidas para reducir el impacto del evento en curso. El apoyo a la toma de decisión abarca a todos los actores del ciclo de emergencia, desde el experto meteorólogo en fase de pronóstico hasta la toma de decisiones adoptada por el alcalde de una comunidad propensa a un determinado riesgo.

4.2.3.1 ¿Qué es la plataforma DEWETRA?

DEWETRA es una plataforma de pronóstico y monitoreo que se encarga de recolectar y sistematizar todos los datos registrados de una forma automática o manual y producir reportes con valor agregado: las observaciones terrestres y modelos de previsión son integrados con datos de vulnerabilidad y exposición para producir un escenario de riesgo en tiempo real. La plataforma es actualmente utilizada a nivel nacional.

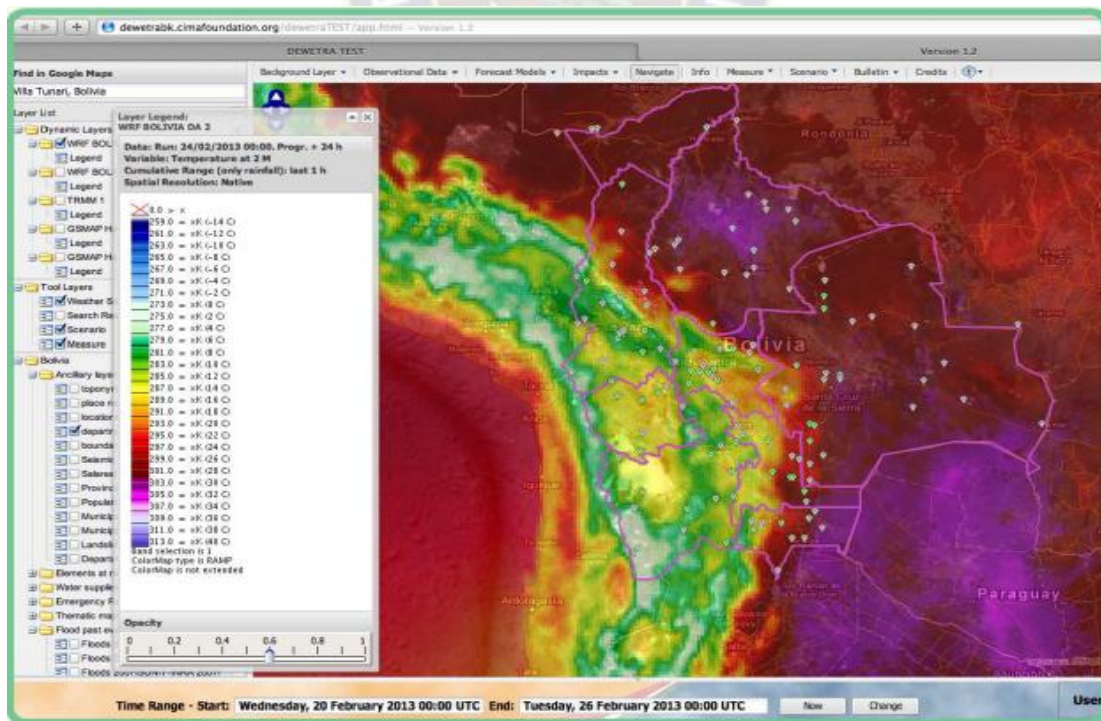


Figura 21. Plataforma Dewetra.

Fuente: (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

4.2.3.2 Arquitectura y acceso a la plataforma DEWETRA

La arquitectura de la plataforma informática DEWETRA está basada en un software de tres capas ("three-tiers architecture") distribuido en varios servidores. Esta arquitectura multicapa asegura flexibilidad y a la vez, confiabilidad al sistema a través de un robusto "middleware" que optimiza los flujos de informaciones. Este tipo de arquitectura es ampliamente utilizada para aplicaciones cliente/servidor dentro de un sistema "net-centric", donde el fin último es facilitar informaciones desde proveedores heterogéneos hacia un gran número de usuarios, eventualmente de diferentes tipologías.

Los usuarios tienen acceso a través de una aplicación publicada en internet (protocolo estándar http). El uso de la aplicación en una web 2.0 garantiza la accesibilidad al sistema desde cualquier lugar y la rapidez de proporcionar actualizaciones que son directamente publicadas en el server e inmediatamente alcanzan a todos los clientes conectados.

A través de una interfaz gráfica de usuario (GUI) multicapa se puede proporcionar informaciones de alta resolución sobre el riesgo observado y previsto. La interfaz de DEWETRA cuenta con seis áreas principales de control (vea A,B,C,D,E en la siguiente Imagen).

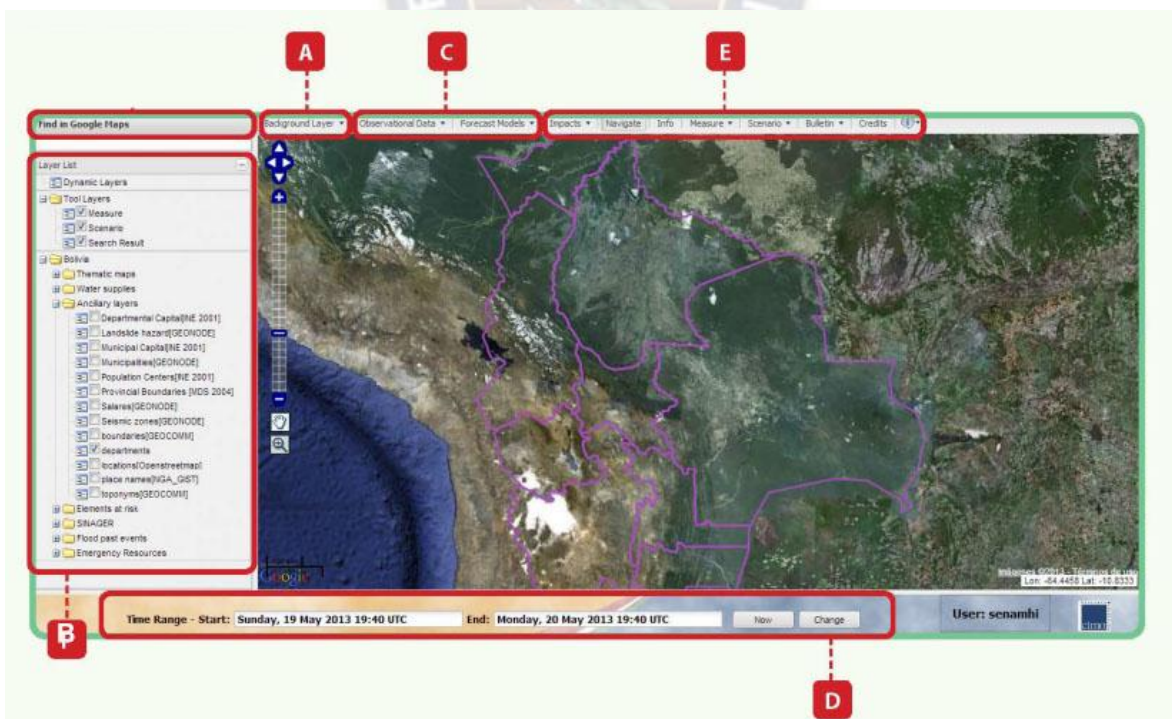


Figura 22. Interface de trabajo de la plataforma informática DEWETRA.

Fuente: (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

A) Capas informativas de fondo: mapas satelitales actualizados por Google.

B) Informaciones estáticas publicadas en formato Web Map Service y navegables a través del árbol de navegación que aparece a lado izquierdo de la ventana de DEWETRA. La información está agrupada en diferentes temas: datos (límites administrativos, capitales, ciudades, otros); elementos en riesgo (poblaciones e infraestructuras que pueden verse afectadas por un evento adverso, otros); recursos hídricos; recursos que pueden ser activados durante la emergencia; mapas temáticos (uso del suelo, cultivos, otros.); eventos históricos (mapas de desastres ocurridos en el pasado, otros).

C) Informaciones dinámicas: observaciones de sensores (red de estaciones meteorológicas, observaciones de radar hidro-meteorológicos, satélites estacionarios y polares, sondeos); elaboración de las observaciones (algoritmos de interpolaciones, fusión, algoritmos para la estimación de la humedad del terreno); informaciones de pronóstico (modelos meteorológicos de circulación global o de área limitada, modelos de inundaciones extensas, inundaciones locales, deslizamientos, incendios, modelos fenológicos).

D) El intervalo de tiempo de los datos visualizados por DEWETRA permite seleccionar y navegar a través de una escala de tiempo utilizada por la representación de los datos. El usuario puede seleccionar una ventana temporal desde una fecha específica en el pasado hasta el momento actual, hasta 72 horas desde la última corrida del sistema en un tiempo futuro.

E) Serie de herramientas de navegación y consulta que permiten al usuario obtener información adicional sobre las capas visualizadas. En particular, sobre el panel de control para la navegación a diferentes niveles de acercamiento (zoom), herramientas de medición de distancias y áreas, herramientas para la construcción de escenarios de impacto, herramientas para la compilación semiautomática de reportes y boletines.

4.2.3.3 Evaluación en tiempo real del riesgo

La plataforma DEWETRA fue diseñada con un enfoque integral de gestión de desastres, para permitir realizar análisis multi-riesgo, combinando la información de amenazas y vulnerabilidades. Esos datos pueden ser utilizados convenientemente para rastrear eventos meteorológicos significativos, construir escenarios de eventos detallados y evaluar los impactos potenciales de eventos esperados sobre las comunidades.

DEWETRA puede visualizar al mismo tiempo observaciones en tiempo real de diferentes sensores (in situ o remotos) como las estaciones hidro-meteorológicas automáticas, radio-sondeos, radares

meteorológicos, satélites polares y satélites geoestacionarios. En DEWETRA, el conjunto de datos de diferentes fuentes es combinados para sacar el máximo contenido de informaciones.

En la fase de respuesta, DEWETRA puede proporcionar rápidamente información sobre el impacto del evento producido a través de la elaboración de imágenes satelitales, sensores in situ, reportes recompilados con la misma plataforma.

El sistema puede ayudar a los tomadores de decisiones involucrados en la gestión y planificación de recursos, proporcionando información para apoyar el despliegue táctico de los equipos de emergencia y la creación de medidas restrictivas en algunas zonas afectadas del territorio.

4.3 Colores de avisos de alerta implementados en el Sistema de Alerta Temprana (SAT)

Los colores verde, amarillo, naranja y rojo identifican el nivel de amenaza en las situaciones de observación y/o pronósticos de fenómenos meteorológicos, climatológicos e hidrológicos adversos en el territorio boliviano. Estos colores forman parte de un protocolo utilizados cuando se pronostican fenómenos adversos y que requieran la intervención por parte de los responsables de: Municipios, Gobernaciones, Defensa Civil y otras entidades como las de ayuda humanitaria.

Estos fenómenos adversos que normalmente tienen intensidades fuertes y extremas son: lluvia, tormentas eléctricas, viento, nieve, heladas, niebla, llovizna, olas de frío, olas de calor (ambas relacionadas a temperaturas extremas); o fenómenos climatológicos como la sequía y fenómenos hidrológicos como ser las riadas e inundaciones.

La amenaza meteorológica, climatológica e hidrológica que genera riesgo para la población, se relaciona con umbrales o presencia del fenómeno con intensidad moderada, fuerte o muy fuerte, ya que, cuando mayor sea esta, menos preparada esta la población para enfrentarse a sus efectos.

Para determinar los umbrales, el SENAMHI ha desarrollado estudios para cada serie de datos correspondientes a cada una de las estaciones meteorológicas y que representan a un área determinada y correspondiente a un municipio en particular: a partir de ello, ha establecido para cada fenómeno y de aplicabilidad y entendimiento sencillo, los umbrales:

“verde”, “amarillo”, “naranja”, “rojo”.

4.3.1 Precipitaciones Pluviales

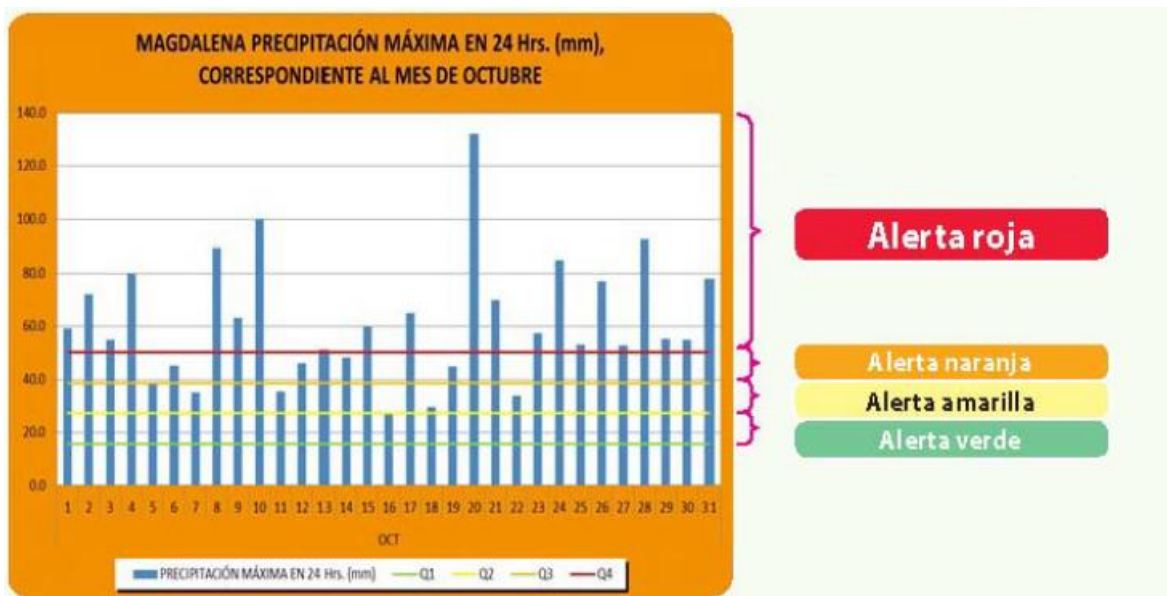


Figura 23. Lectura de Alerta de las Precipitaciones Pluviales en el Municipio de Magdalena, Beni-Bolivia.

Fuente: (Mariaca, Trujillo, Rossi, & Mendoza, 2013)

Como muestra la gráfica de precipitación diaria en 24 horas, se ha identificado para cada mes y para cada estación meteorológica, las intensidades y cantidades acumuladas que están por encima del cuartil cuarto (Q4, línea roja), respecto a la determinación del promedio, el que muestra por encima de este valor las cantidades acumuladas que representan potencial peligro y que pueden causar desastres en el municipio (fenómeno de intensidad fuerte o excepcional y con riesgo alto para cualquier actividad productiva, por ejemplo ocasionan rápidas inundaciones barriales).

En el cuartil tercero (Q3, línea naranja) se pueden observar las cantidades correspondientes entre el 75 al 100% de precipitación (generan riesgo con cierto grado de peligro, generando grandes charcos o riachuelos).

En el cuartil dos (Q2, línea amarilla), se encuentra las cantidades de precipitación entre 50% al 75% (precipitaciones que pueden ocasionar conflictos en algunas actividades, por ejemplo, turismo); todo el resto de precipitaciones por debajo de la línea amarilla representa precipitaciones dentro de la normalidad.

4.3.2 Temperaturas

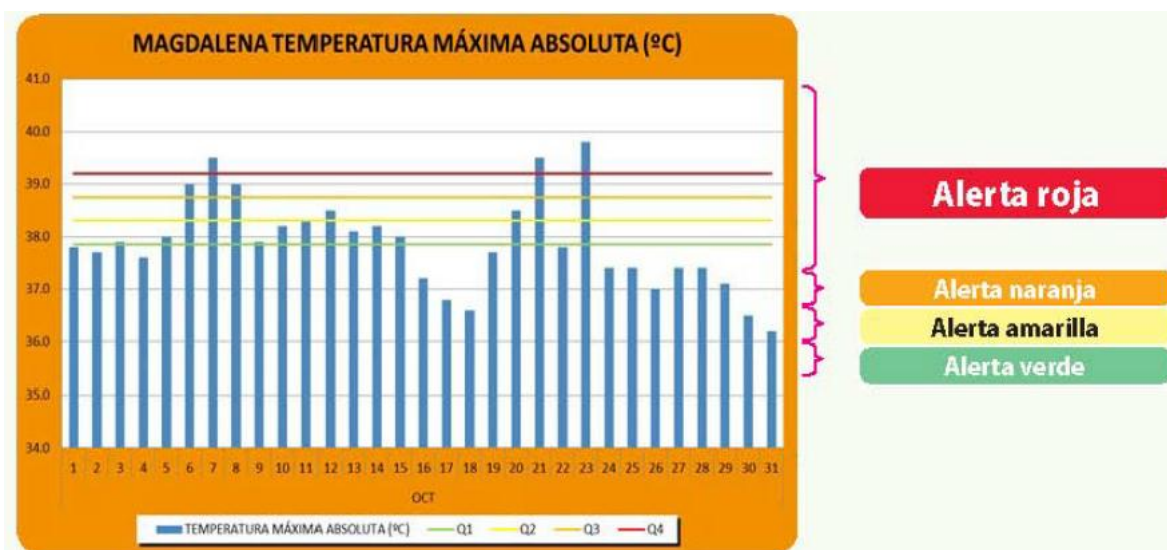


Figura 24. Lectura de la Alerta de Temperatura en el Municipio de Magdalena, Beni-Bolivia.

Fuente: (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

Respecto a las temperaturas, se han realizado análisis del comportamiento de las mismas por meses y épocas, determinándose umbrales para cada lugar donde se cuenta con observación meteorológica, concluyendo que cuando hay persistencia de cinco o más días continuos con temperaturas superiores a un umbral determinado (línea roja), nos referimos a ola de calor. Ambas, olas de calor o frio tienen repercusiones importantes en la salud de las personas, de los animales y las plantas. También se considera para emitir avisos de alerta, cambios bruscos de temperatura, por ejemplo, cuando ingresa un frente frio.

Existen descensos que se producen entre dos a tres horas con valores mayores a 10°C, y que se prolongan durante dos o tres días; estos descensos de temperatura causan también impactos en la salud.

4.3.3 Vientos

Con respecto a los vientos, se tienen umbrales de intensidad, de tal forma que cuando existen vientos de intensidad menores a 40Km/h se consideran normales; cuando existen vientos con intensidades entre 40 y 50 Km/hora se considera alerta amarilla el cual puede causar conflictos a los techos y letreros mal contruidos, puede desgajar algunas ramas de árboles pequeños. Se

considera alerta naranja cuando las intensidades están entre 50y 60 Km/hora, representa peligro porque es dificultoso caminar para las personas en contra el viento, se mueven árboles grandes. Intensidades mayores a 60Km/h se considera alerta roja porque las personas no pueden caminar en contra el viento, se hace muy difícil la respiración, puede producir caída de árboles grandes, fácilmente puede destechar casas, al arrastrar materiales desde el suelo pueden impactar a las personas afectando su salud y causando daños a estructuras de edificios.

4.3.4 Sequias

Respecto a las sequías es necesario considerar inicialmente lo siguiente: es un fenómeno perjudicial y subrepticio que se produce a raíz de niveles de precipitación inferiores a lo esperado o a lo normal y que, cuando se prolonga durante una estación o durante períodos más largos, hace que las precipitaciones sean insuficientes para responder a las demandas de la sociedad y del medio ambiente.

La sequía es una imperfección transitoria climática y en ello se diferencia de la aridez, que es una característica permanente del clima. La aridez estacional, es decir, una estación seca claramente demarcada, es también distinta de la sequía, aunque frecuentemente existe una confusión entre ambas palabras o se utilizan indistintamente. Hay que entender las diferencias entre ambas, a fin de reflejarlas de manera adecuada en los sistemas de vigilancia y alerta temprana de la sequía y en los planes de preparación frente a ella. La sequía debe considerarse como un estado relativo y no absoluto. Se manifiesta tanto en regiones muy lluviosas como poco lluviosas y prácticamente en todos los regímenes climáticos. La sequía es un componente del clima, aunque su extensión geográfica y su gravedad variarán a escala estacional o anual. La sequía no es en sí misma un desastre. Puede llegara serlo en función de sus efectos sobre la población local, sobre la economía y sobre el medio ambiente y en función de la capacidad de estos últimos para hacer frente al fenómeno y recuperarse de tales efectos. Por consiguiente, la clave para comprender la sequía está en calibrar sus dimensiones tanto naturales como sociales. Las sequías presentan tres rasgos distintivos: intensidad, duración y extensión.

La intensidad refleja el déficit de precipitación y la gravedad de los efectos asociados a ese déficit. Dado que no existe una definición única de la sequía para todas las situaciones, los planificadores agrícolas e hídricos entre otros, tienen que hacer uso de diversos tipos de datos o de índices expresados en forma de mapa o en forma gráfica. El SENAMHI, por la disponibilidad de

parámetros meteorológicos y en forma coyuntural está utilizando el índice normalizado de precipitación porque nos muestra un diagnóstico de severidad de sequía y se basa en la probabilidad de lluvia en cualquier periodo de tiempo, también cuantifica el déficit de precipitación durante múltiples periodos de tiempo (dos, tres meses; uno, dos, tres años, etc.), calculado con la siguiente fórmula:

$$\left[\frac{P_i - P_m}{P_i} \right] * P_m$$

PPN = Porcentaje de la precipitación normal.
Pi = Precipitación acumulada durante 30 días.
Pm = Precipitación promedio en 30 días.

Del análisis anterior se considera alerta roja cuando se presente cinco o más meses continuos con déficit de precipitación hasta un 75% respecto de la normal. Se emitirá alerta naranja cuando se prevea el déficit de precipitación entre dos a cuatro meses con el 50% respecto a la normal. Se emitirá alerta amarilla cuando se prevea déficit mensual del 50% o más respecto al promedio. Como se explicó anteriormente existen muchos eventos adversos y en general se constituyen como amenazas que exponen a riesgo a las personas y de acuerdo con ello, para una mejor comprensión, se ha establecido la emisión de boletines de avisos de alerta distinguidos por colores que se explican a continuación.

- **Alerta Verde.** Cuando aún no ha ocurrido el evento adverso y se considera una situación de normalidad. Ante alertas de esta clase los distintos ministerios y las instancias encargadas de la atención ante desastres y/o emergencias, así como los gobiernos autónomos departamentales y municipales, efectuarán, entre otras: actividades de mantenimiento, reparación de infraestructura y equipos; capacitarán permanentemente al personal para fines de respuesta.
- **Alerta Amarilla.** Cuando la proximidad de la ocurrencia de un evento adverso se encuentra en fase inicial de desarrollo o evolución.

- **Alerta Naranja.** Cuando se prevé que el evento adverso ocurra y su desarrollo pueda afectar a la población, medios de vida, sistemas productivos, accesibilidad a servicios básicos y otros.
- **Alerta Roja.** Cuando se ha confirmado la presencia del evento adverso y por su magnitud o intensidad puede afectar y causar daños a la población, medios de vida, sistemas productivos, accesibilidad, servicios básicos y otros.



4.4 Boletines de riesgo

el boletín de riesgo representa una síntesis del riesgo debido a fenómenos meteorológicos de interés y que podrán interesar a corto plazo a la población de Bolivia.

El boletín elaborado por el VIDECI tiene la finalidad de alertar a las autoridades competentes en el territorio nacional para que puedan tomar medidas de mitigación y prevención de los efectos al suelo y de eventos climatológicos adversos (Mariaca, Trujillo, Rossi, & Mendoza, 2013).

5. Red Privada Virtual

5.1 Definición de una VPN

Una Red Privada Virtual (VPN – del inglés Virtual Private Network) es una red, construida sobre la infraestructura de una red pública, que permite a usuarios remotos comunicarse de forma transparente y segura. Tal y como indica su nombre, es privada dado que garantiza la privacidad en sus comunicaciones, y virtual porque se implementa sobre la infraestructura de redes públicas (por ejemplo, Internet).

Las conexiones VPN permiten a los usuarios acceder, desde casa o desde un punto remoto, al servidor de su organización a través de la infraestructura de encaminamiento que proveen las redes públicas. Desde el punto de vista del usuario, la conexión VPN es una conexión punto a punto entre

su ordenador y el servidor de la organización. La naturaleza de las redes intermedias es irrelevante para el usuario, ya que, para él, los datos son enviados como si hubiera un enlace dedicado hasta el servidor. La tecnología VPN también permite a las empresas conectar con sus filiales, o con otras empresas, a través de la infraestructura de redes públicas, ofreciendo comunicaciones seguras. También en este caso, la conexión aparece al usuario como si se estableciera en una red privada. Por tanto, las redes privadas virtuales intentan dar soluciones principalmente a problemas tales como:

- El acceso remoto a la red del servidor.
- La interconexión de múltiples sitios remotos.
- Establecimiento de conexiones seguras.

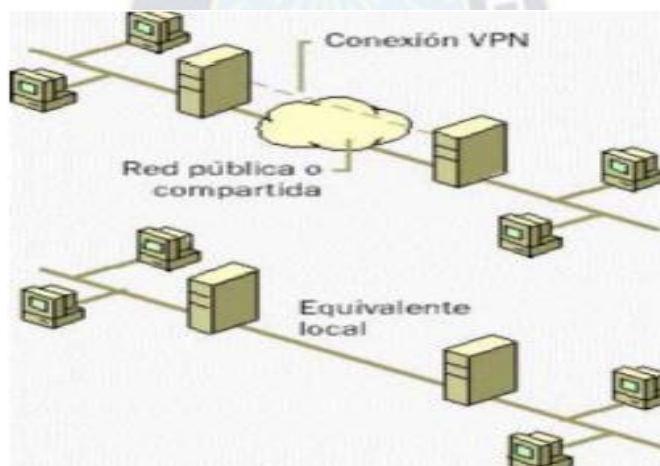


Figura 25. Diagrama Lógico de una VPN.

Fuente: (Tomás Cánovas, 2008)

Para hacer esto posible de manera segura es necesario proveer los medios para garantizar la autenticación, integridad y confidencialidad de toda la comunicación:

- **Autenticación y autorización:** ¿Quién está del otro lado? Usuario/equipo y qué nivel de acceso debe tener. Emisor y receptor son capaces de determinar de forma inequívoca sus identidades, de tal manera que no exista duda sobre las mismas. Esto puede conseguirse mediante firmas digitales.

- **Integridad:** La garantía de que los datos enviados no han sido alterados, es decir, los datos que le llegan al receptor sean exactamente los que el emisor transmitió por el canal. Para esto se pueden utilizar firmas digitales.
- **Confidencialidad:** Dado que los datos viajan a través de un medio potencialmente hostil como Internet, los mismos son susceptibles de interceptación, por lo que es fundamental el cifrado de los mismos. De este modo, la información no debe poder ser interpretada por nadie más que los destinatarios de la misma. La manera de conseguir esto es mediante técnicas de encriptación (Ñacato Gualotuña, 2007).

5.2 Propiedades de una VPN

Las Redes Privadas Virtuales presentan las siguientes propiedades:

5.2.1 Encapsulación

Las VPN's encapsulan datos privados con un encabezado que les permite atravesar la red pública.

5.2.2 Autenticación

Con esta propiedad el servidor VPN autentifica al cliente VPN que desea conectarse y al mismo tiempo verifica que tenga los permisos adecuados debido a que la autenticación entre ellos es mutua.

La autenticación en una Red Privada Virtual consiste en la autenticación de datos y usuarios. Es importante ofrecer cada una de éstas debido a que, la autenticación de datos reafirma que el mensaje ha sido enviado completamente y que no ha sido alterado en forma alguna, en cambio que la autenticación de usuarios es un proceso que permite al usuario tener acceso a la red.

5.2.3 Cifrado de datos

Esta propiedad permite convertir algún texto legible en un texto ilegible, logrando de esta manera que sólo la persona a la que se le envía lo convierta en un texto legible.

Desde la perspectiva de la seguridad se tiene que saber qué tipo de tecnología de cifrado utiliza la organización. La fortaleza del cifrado afecta directamente a la seguridad de la Red Privada Virtual (Ñacato Gualotuña, 2007).

5.3 Beneficios de una VPN

Los beneficios de una Red Privada Virtual (VPN) es sólo un término general, que se utiliza para describir todas las utilidades potenciales cuando se implementa la tecnología VPN, entre estos podemos señalar:

5.3.1 Seguridad

Una VPN tiene una alta seguridad. Específicamente, una VPN requiere el establecimiento de un túnel en la red corporativa y la encriptación de los datos entre la PC del usuario y los servidores corporativos. Utilizan protocolos estándar de autenticación y encriptación que permiten ocultar los datos en entornos inseguros de Internet, pero que son accesibles a los usuarios corporativos mediante una VPN.

5.3.2 Transparencia

En lo que se refiere a la transparencia, el usuario de la Red Privada Virtual (VPN) tiene total control del proceso de transferencia de la información, una vez que éste ha sido autenticado por el servidor VPN.

5.3.3 Cobertura

Las redes basadas en Internet ya están extendidas en la mayoría de los centros de negocio del mundo, por lo que la cobertura es quizás la ventaja concreta más importante que se obtiene al trasladar la Red Privada a una infraestructura basada en IP. A diferencia de las líneas permanentes de las compañías telefónicas, una conexión punto a punto con Internet es como una tupida malla que enlaza con una gran cantidad de redes, todas ellas interconectadas, que llegan hasta el último rincón del planeta.

5.3.4 Ahorro en los costos

Una Red Privada Virtual ofrece ahorros directos sobre otros métodos de comunicación tales como líneas privadas y llamadas de larga distancia, también ofrece beneficios indirectos como resultado de la reducción de costos de entrenamiento y equipamiento.

Las líneas privadas convencionales ofrecen beneficios para una corporación con gran demanda de ancho de banda (E1 a 2.048 Mbps y superior). Sin embargo, gran parte de ese ancho de banda no

es utilizado en todo momento por una corporación de medianas necesidades. En todo caso, es necesario pagar costos por interconexión entre nodos remotos debido, principalmente, a que se trata de enlaces fijos de recursos reservados.

Una Red Privada Virtual no necesita de enlaces dedicados, puesto que utiliza los recursos ya existentes en la red mundial, Internet, para establecer canales virtuales y temporales conocidos como túneles.

Una conexión por Internet es mucho más barata que una conexión permanente a través de una línea alquilada. Por lo general, las líneas alquiladas tienen un costo fijo de conexión y otro costo que depende de la distancia; sin embargo, con este tipo de líneas, a medida que aumenta el tamaño de la red, los costes se disparan. Por el contrario, el coste de las conexiones por Internet depende sólo de la capacidad de la línea, y no de la distancia; además, al tratarse de un servicio totalmente gestionado, los costes de explotación de cada nodo de VPN se reducen considerablemente (Ñacato Gualotuña, 2007).

5.4 Elementos de una VPN

Para realizar el diseño de una Red Privada Virtual (VPN) es necesario la utilización de ciertos elementos, los mismos que se detallaran a continuación:

- **Servidor VPN**

Es un computador que recibe conexiones VPN de clientes VPN mediante acceso remoto.

- **Cliente VPN**

Es un computador que inicia un enlace o conexión a un servidor VPN.

- **Conexión VPN**

Es la parte del enlace en la cual los datos son encriptados para obtener una conexión más segura, los datos deben ser encriptados a lo largo de toda la conexión VPN.

- **Túnel**

Constituye la parte de la conexión en la cual los datos están encapsulados. Protocolos de Túnel son usados para administrar los túneles y la encapsulación de los datos privados.

- **Red Publica**

Es aquella red que permite transferir datos de un lugar a otro, de una manera poco segura, ya que la información que viaja por ésta puede ser interceptada por terceros. Una red pública permite la conexión VPN a través de la cual los datos viajan encapsulados.

- **Switch**

Es un dispositivo de interconexión de redes de ordenadores/computadoras que opera en la capa 2(nivel de enlace de datos) del modelo OSI (Open Systems Interconnection). Este interconecta dos o más segmentos de red, funcionando de manera similar a los puentes (bridges), pasando datos de una red a otra, de acuerdo con la dirección MAC de destino de los datagramas en la red.

- **Enrutador o Router**

Es un dispositivo que toma decisiones sobre cuál sería el camino menos costoso para transferir la información; realiza el encaminamiento entre dos o más redes.

Al encaminador se lo usa en situaciones donde las funciones de un puente no son lo suficientemente robustas para las operaciones requeridas por la red.

El objetivo del encaminamiento es trasladar a través de la red información desde una fuente hasta un destino. La diferencia principal entre un router y un puente es que el puente trabaja a nivel de la capa 2 del modelo OSI (capa de enlace), en cambio que el router trabaja a nivel de la capa 3 de modelo OSI (capa de red).

- **Firewall**

Un firewall es un sistema de defensa que se basa en la instalación de una "barrera" entre una PC y los datos que viajan por la red a través de software o hardware, permitiendo autorizar o denegar estos, bajo determinadas instrucciones que se hayan configurado en el mismo. En otras palabras, un firewall es un dispositivo conectado en el perímetro de la red de una organización para impedir el acceso de usuarios no autorizados (Ñacato Gualotuña, 2007).

5.5 Arquitecturas de una VPN

La arquitectura de una Red Privada Virtual se determina de acuerdo a las necesidades que presentan cada una de las organizaciones o empresas que la van a implementar, en cuanto a lo relacionado con el tipo de red que manejan, la capacidad técnica y adecuada para mantener e instalar este tipo de red, la seguridad, la infraestructura de hardware y el número de usuarios que la vayan a utilizar. En la actualidad las arquitecturas de Redes Privadas Virtuales más utilizadas son las siguientes:

5.5.1 Arquitectura VPN basados en hardware

Los sistemas basados en hardware, son routers que encriptan. Son seguros y fáciles de usar, simplemente que conectarlos y ya está. Ofrecen un gran rendimiento, porque no malgastan ciclos de procesador haciendo funcionar un Sistema Operativo. Es hardware dedicado, muy rápido, y de fácil instalación.

5.5.2 Arquitectura VPN basados en cortafuegos

Estos se implementan con software de cortafuegos (firewall). Tienen las ventajas de los mecanismos de seguridad que utilizan los cortafuegos, incluyendo el acceso restringido a la red interna. También realizan la traducción de direcciones (NAT). Estos satisfacen los requerimientos de autenticación fuerte.

Muchos de los cortafuegos comerciales, aumentan la protección, quitando al núcleo del Sistema Operativo algunos servicios peligrosos que llevan éstos de serie, y les provee de medidas de seguridad adicionales, que son mucho más útiles para los servicios de VPN.

El rendimiento en este tipo decrece, ya que no se tiene hardware especializado de encriptación.

5.5.3 Arquitectura VPN basados en software

Éstos sistemas son ideales para las situaciones donde los dos puntos de conexión de la VPN no están controlados por la misma organización, o cuando los diferentes cortafuegos o routers no son implementados por la misma organización. Este tipo de VPN's ofrecen el método más flexible en cuanto al manejo de tráfico. Con este tipo, el tráfico puede ser enviado a través de un túnel, en función de las direcciones o protocolos, en cambio en las VPN por hardware, todo el tráfico es enrutado por el túnel. Podemos hacer un enrutamiento inteligente de una manera mucho más fácil (Ñacato Gualotuña, 2007).

5.6 VPN basadas en SSL/TLS

Los protocolos SSL (Secure Socket Layer) y TLS (Transport Layer Security) son protocolos de la capa de transporte que proporcionan comunicaciones seguras en Internet. Los protocolos SSL versión 3.0 y TLS versión 1.0 son idénticos, una de las diferencias, son sus diseñadores.

SSL/TLS permite la autenticación tanto de cliente como servidor, usando claves públicas y certificados digitales y proporciona comunicación segura mediante el cifrado de la información

entre emisor y receptor. Este protocolo está muy extendido para realizar actividades de comercio electrónico de tal manera que Visa, MasterCard, American Express y muchas de las principales instituciones financieras han aprobado SSL para el comercio sobre Internet.

Hay que destacar que SSL/TLS se compone de cuatro protocolos:

- **Record Protocol:** es el protocolo de transporte que proporciona a cada conexión:
 Confidencialidad: utilizando una clave compartida generada durante el protocolo Handshake para el cifrado convencional de los datos.
 Integridad del mensaje: el protocolo de Handshake también genera una clave secreta común que se usa para formar el MAC o código de autenticación del mensaje.
- **Handshake Protocol:** Mediante este protocolo se generan los parámetros criptográficos que van a definir el estado de una sesión (una sesión SSL/TLS siempre empieza con el Handshake). Este protocolo permite la autenticación entre el cliente y el servidor y la negociación de los algoritmos de cifrado y las claves.
- **Change Cipher Spec Protocol:** Consiste en un único mensaje de un byte de contenido 1 que tiene como objetivo pasar del modo pendiente, para negociar los parámetros de una conexión, al modo operativo, en el que los parámetros ya se han establecido y la comunicación es totalmente segura.
- **Alert Protocol:** señala alertas y errores en la sesión establecida.

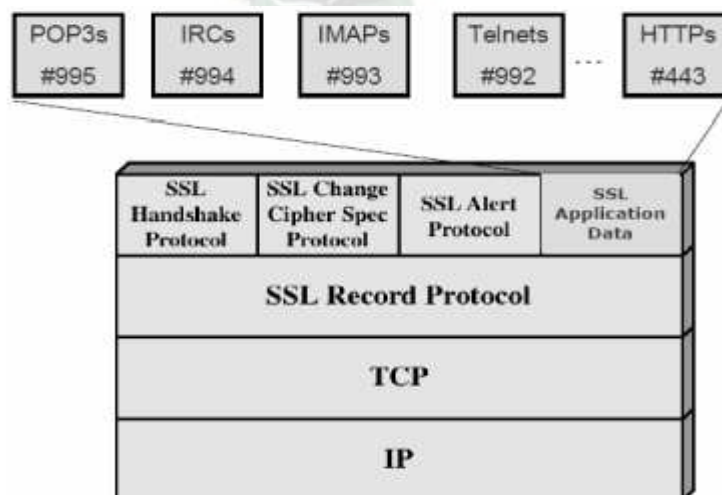


Figura 26. Situación de SSL/TLS en la pila de protocolos OSI.

Fuente: (Tomás Cánovas, 2008)

SSL es capaz de trabajar con la mayoría de protocolos que trabajan sobre TCP de tal manera que el IANA les tiene asignado un número de puerto por defecto, por ejemplo, el protocolo HTTP sobre SSL ha sido denominado HTTPS y tiene como puerto el 443.

SSL se basa en un esquema de clave pública para el intercambio de claves de sesión. En primer lugar, cliente y servidor intercambian una clave de longitud suficiente mediante un algoritmo de cifrado asimétrico como RSA o Diffie-Hellman utilizando certificados. Mediante esa clave se establece un canal seguro, utilizando para ello un algoritmo simétrico previamente negociado. Los mensajes a ser transmitidos, se fragmentan en bloques, se comprimen y se les aplica un algoritmo Hash para obtener un resumen (MAC del mensaje) para asegurar la integridad.

En SSL/TLS una sesión es una asociación entre un cliente y un servidor. Las sesiones se crean mediante el protocolo Handshake y coordina los estados del cliente y del servidor. El estado de una sesión incluye la siguiente información (Tomás Cánovas, 2008):

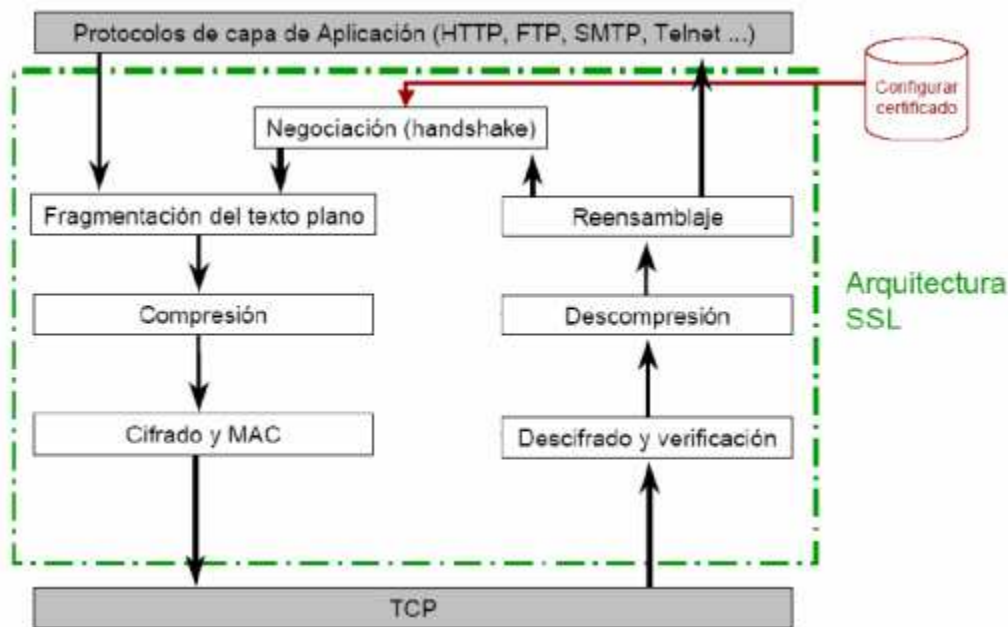


Figura 27. Arquitectura SSL/TLS.

Fuente: (Tomás Cánovas, 2008)

6. OpenVPN

6.1 ¿Qué es OpenVPN? Características Principales

OpenVPN es una solución de conectividad basada en software, una utilidad de código abierto (está publicado bajo la licencia GPL, de software libre) para soluciones SSL/TLS VPN. Fue creado por James Yonan en el año 2001 y ha estado siendo mejorado desde entonces. OpenVPN tiene asignado y reservado el puerto 1194 de manera oficial por la IANA. El modelo de seguridad de OpenVPN está basado en la arquitectura SSL/TLS, que es el estándar escogido actualmente por la industria para establecer comunicaciones seguras a través de Internet.

Un aspecto que hay que tener en cuenta durante todo el estudio con OpenVPN es que esta herramienta implementa redes seguras en la capa 2 o 3 (según el modo que utilice OpenVPN: Tunnel o Bridge) de la pila de protocolos OSI utilizando como extensión el protocolo SSL/TLS, soportando métodos de autenticación del cliente de manera flexible. Las implementaciones de SSL/TLS más conocidas hoy en día operan a través de un navegador Web (lo que se conoce como HTTPS), ya que se han implementado aproximaciones de SSL/TLS para aplicaciones en la capa de aplicación de la pila OSI (capa 7). Pero hay que tener muy en cuenta que este tipo de implementación Web con SSL/TLS no es una VPN y que OpenVPN no es una aplicación Web, ni opera a través de un navegador Web, ya que opera sobre la capa 2 o 3 de la pila OSI. Por tanto, un cliente de OpenVPN no podrá utilizar un navegador Web para conectarse al servidor de OpenVPN y mantener una comunicación segura a través de la VPN.

Además, OpenVPN es una herramienta multiplataforma, soportadas en sistemas operativos como Linux, Windows, OpenBSD, FreeBSD, NetBSD, Mac OS X y Solaris.

OpenVPN no opera en el kernel, sino que opera en el espacio de usuario incrementando de esta manera la seguridad y la escalabilidad. Según el autor, James Yonan, uno de los mayores frenos de IPSec es que añade una gran complejidad en kernel. El problema de añadir dicha complejidad de seguridad software en el kernel es que se ignora un importante principio de los sistemas de seguridad: nunca se ha de diseñar un sistema en el que si uno de los componentes cae pueda poner en peligro todo el sistema. Un simple desbordamiento de un buffer en el espacio del kernel provocaría un compromiso total en la seguridad del sistema. Es por esta razón que OpenVPN ubica su complejidad y ejecuta su código dentro del espacio de usuario pudiendo contener los fallos en este espacio más rápidamente sin comprometer la seguridad del sistema. En ocasiones, para proveer

de encriptación al enlace, las aplicaciones necesitan intervenir con el kernel del sistema operativo para ganar acceso a bajo nivel en la interfaz de red del enlace. Para estos casos, OpenVPN emplea “interfaces virtuales” del espacio de usuario para controlar y acceder sin la necesidad de depender del kernel. Estas interfaces virtuales ofrecen, a las VPN que usan el espacio de usuario, un punto extra de seguridad respecto a otras tecnologías VPN como IPSec, además de ofrecer más flexibilidad a la hora de exportar dicha herramienta a otros sistemas operativos y facilidad de instalación y uso en dichos sistemas operativos (Tomás Cánovas, 2008).

Por otra parte, OpenVPN permite encapsular el tráfico en paquetes que utilicen como protocolo de transporte TCP o UDP.

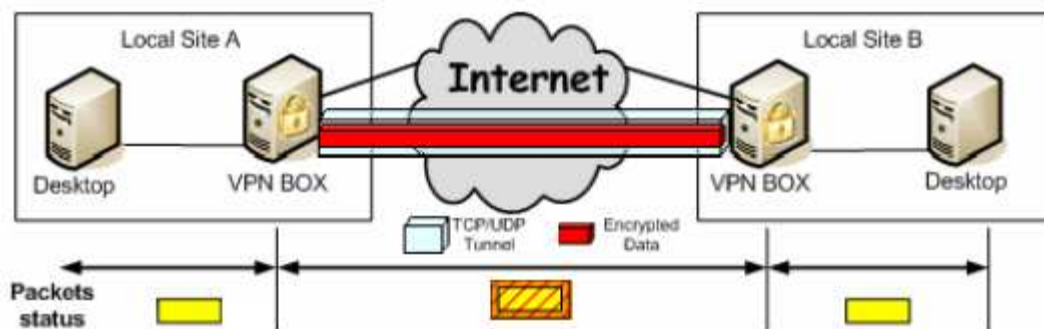


Figura 28. Formato del Paquete Encapsulado por OpenVPN.

Fuente: (Tomás Cánovas, 2008)

De entre todas las aplicaciones y características que OpenVPN ofrece se pueden citar las siguientes:

- Posibilidad de implementar dos modos básicos, “bridge” o “túnel”, en la capa 2 o capa 3 respectivamente, con lo que se logran túneles capaces de enviar información en otros protocolos no IP como IPX o broadcast (Netbios).
- Uso de todos los protocolos de cifrado, autenticación y certificación que ofrece la librería OpenSSL para proteger el tráfico de la VPN durante el tránsito por Internet. Esta librería nos ofrecerá muchos algoritmos de cifrado, tamaños de clave, o resúmenes HMAC (para la integridad de los mensajes).
- Uso de compresión en tiempo real.
- Ningún problema con NAT. Tanto los clientes como el servidor pueden estar en la red usando solamente direccionamiento IP privado.

- Controlar y monitorizar conexiones OpenVPN mediante interfaces graficas de usuario (GUI) para diferentes plataformas (Windows, Linux, MAC OS X).
- Permite su instalación en casi cualquier plataforma. Tanto la instalación como su uso son increíblemente simples.
- Solo se ha de abrir un puerto para permitir conexiones, dado que desde OpenVPN 2.0 se permiten múltiples conexiones en el mismo puerto TCP o UDP.
- OpenVPN se ha construido con un diseño firmemente modular. Todo lo relacionado con la encriptación está soportado por la librería OpenSSL y todo lo relacionado con la funcionalidad de los túneles IP está proporcionado por el adaptador virtual TUN/TAP. Los beneficios de esta modularidad pueden ser vistos, por ejemplo, en el caso de que OpenVPN puede enlazarse dinámicamente con una nueva versión de la librería OpenSSL e, inmediatamente, tener acceso a cualquier función ofrecida por esta nueva versión. Por ejemplo, cuando OpenVPN es implementada junto con la última versión de la librería OpenSSL, se puede acceder automáticamente a nuevos algoritmos de cifrado como AES-256 (Advanced Encryption Standard con 256 bits de clave) y, también, a las nuevas capacidades de OpenSSL que permiten la utilización de nuevos aceleradores hardware para optimizar el cifrado, descifrado y los métodos de autenticación. La portabilidad de OpenVPN hacia otros entornos también se ve facilitada gracias a esta modularidad.
- OpenVPN es rápido, ejecutándose en una máquina Pentium II a 266 MHz, OpenVPN puede lograr una velocidad de transferencia de 1,455 megabytes por segundo de tiempo de CPU (Tomás Cánovas, 2008).

6.2 OpenVPN y Paquetes Necesarios

La instalación, configuración y puesta en marcha de OpenVPN es bastante fácil. La herramienta OpenVPN debe ser instalada tanto en las máquinas que hagan de servidor como en las que hagan de clientes. No existe una versión para clientes y otra para servidores, desde que OpenVPN se instala en el PC, proporciona tanto las funciones del cliente como las del servidor con sus respectivos comandos y directivas.

OpenVPN corre casi completamente en el espacio de usuario no exigiendo ninguna modificación ni componente en el kernel del sistema operativo que no sean los controladores virtuales “TUN/TAP drivers” disponibles para muchas plataformas tales como Windows, Linux y MAC.

OpenVPN también puede ser vinculado con la librería de compresión de datos en tiempo real LZO (Lempel Ziv Oberhumer). De esta manera OpenVPN soporta compresión con adaptación en tiempo real, que significa que permitirá realizar la compresión sólo cuando flujo de datos del túnel pueda ser comprimido.

6.2.1 Los archivos de configuración de OpenVPN

Según la arquitectura de la VPN que se quiera implementar con OpenVPN se tendrá que hacer un fichero de configuración en modo cliente, en modo servidor de túneles o prescindiendo de ninguno de estos modos, como un simple extremo de la comunicación. En cualquiera de estos casos, cada máquina ha de ejecutar un fichero de configuración independientemente de los scripts o plugins que también deba ejecutar.

Una vez creados los archivos de configuración hay que ejecutarlos mediante el programa OpenVPN (OpenVPN, s.f.).

6.2.2 Seguridad con OpenSSL

OpenSSL es un proyecto de software de libre distribución desarrollado por los miembros de la comunidad Open Source para libre descarga. Consiste en un robusto paquete de herramientas de administración y librerías relacionadas con la criptografía, que suministran funciones criptográficas a otros paquetes como OpenVPN, OpenSSH y navegadores web (para acceso seguro a sitios HTTPS). Estas herramientas ayudan al sistema a implementar el protocolo Secure Socket Layer (SSL), así como otros protocolos relacionados con la seguridad, como el protocolo Transport Layer Security (TLS).

OpenSSL es compatible con un gran número de sistemas operativos, entre los que se encuentran sistemas operativos tipo Windows, UNIX/Linux o también MAC OS X. Según el sistema operativo en el que se instale OpenVPN será necesario o no una instalación a parte de la herramienta OpenSSL. Por ejemplo, en sistemas operativos Windows, el ejecutable que se utiliza para instalar OpenVPN también instala las librerías y programas de OpenSSL necesarios para OpenVPN (Tomás Cánovas, 2008).

OpenSSL soporta un gran número de algoritmos criptográficos diferentes según la finalidad:

- Algoritmos de cifrado: Blowfish, AES, DES, RC2, RC4, RC5, IDEA, Camellia.

- Algoritmos para funciones hash: MD5, SHA, MD2, MDC-2.
- Algoritmos de intercambio de clave pública: RSA, Diffie-Hellman, DSA.

Además, OpenSSL proporciona las herramientas y funciones para:

- Generar distintas claves para distintos algoritmos de cifrado.
- Generar claves asimétricas para los algoritmos habituales (RSA, Diffie-Hellman, DSA).
- Generar números aleatorios y pseudos aleatorios.
- Utilizar los algoritmos para firmar, certificar y revocar claves.

6.2.2.1 Algoritmo Diffie-Hellman

El algoritmo criptográfico Diffie-Hellman (en honor a sus creadores, Whitfield Diffie y Martin Hellman), es un algoritmo de establecimiento de claves entre partes que no han tenido contacto previo, utilizando un canal inseguro y de manera anónima (no autenticada).

Se emplea generalmente como medio para acordar claves simétricas que serán empleadas para el cifrado de una sesión.

El sistema se basa en la idea de que dos interlocutores pueden generar conjuntamente una clave compartida sin que un intruso, que esté escuchando las comunicaciones, pueda llegar a obtenerla. Para ello se eligen dos números públicos y, cada interlocutor, un número secreto. Usando una fórmula matemática, que incluye la exponenciación, cada interlocutor hace una serie de operaciones con los dos números públicos y su número secreto. A continuación, los interlocutores se intercambian los resultados de forma pública. En teoría revertir esta función es tan difícil como calcular un logaritmo discreto (un sextillón de veces más costosa que la exponenciación usada para transformar los números). Por eso se dice que este número es el resultado de aplicar una función unidireccional al número secreto.

6.2.2.2 Algoritmo de cifrado AES-256

AES significa Advanced Encryption Standard. Aunque sus raíces se remontan a 1997, actualmente sigue siendo el único algoritmo en la lista del National Institute of Standards and Technology (NIST) para proteger datos clasificados.

AES es lo que se conoce como un cifrado simétrico por bloques, lo que significa que cifra y descifra los datos en bloques de 128 bits cada uno. Para ello, utiliza una clave criptográfica específica, que

es efectivamente un conjunto de protocolos para manipular información. Esta clave puede ser de 128, 192 o 256 bits de tamaño.

AES-256 – la versión clave de 256 bits de AES – Es la forma más avanzada del cifrado y consiste en 14 rondas de sustitución, transposición y mezcla para un nivel de seguridad excepcionalmente alto.

AES-256 también tiene la ventaja de ser extremadamente rápido. Cuando navegas por la web con una VPN que utiliza el cifrado AES-256 en sus servidores, no experimentarás ninguna disminución en el rendimiento en comparación con otro protocolo de seguridad.

Por razones de seguridad y conveniencia, exige una VPN con cifrado AES-256.

6.3 Modo de funcionamiento de OpenVPN

OpenVPN puede trabajar en dos modos: modo “tun” (o modo Tunnel) o modo “tap” (o modo Bridge). En concreto, utilizando el adaptador TUN para transmitir tráfico IP a lo largo del túnel o, por el contrario, utilizando el adaptador TAP para transmitir tráfico Ethernet por el túnel. Como sabemos, la configuración de estos adaptadores es muy fácil y solo requiere de un conjunto de comandos o de líneas introducidas en un fichero de configuración. Una vez la interfaz TUN/TAP se ha establecido ya se puede hacer una comunicación mediante OpenVPN.

6.3.1 Capa de Cifrado de OpenVPN

OpenVPN tiene dos posibles modos de autenticación:

- Utilizar claves estáticas, es decir, utilizar claves pre-compartidas.
- Utilizar TLS, es decir, utilizar los protocolos de SSL/TLS y certificados para la autenticación y el intercambio de claves.

En el modo de claves estáticas, una clave pre-compartida es generada y compartida entre los dos participantes de OpenVPN antes de iniciar el túnel. Por defecto, en el modo de clave pre-compartida, la clave estática consta de 4 claves independientes: una HMAC enviada, una HMAC recibida, una clave para cifrar y una clave para descifrar (aunque también se puede establecer una clave con 2 claves: una HMAC para ambos extremos y una clave para cifrar/descifrar). Con el

comando --secret, indicamos que se quiere utilizar el modo de claves estáticas y según el parámetro que le pasemos a dicho comando se utilizarán 2 claves o 4 claves para dicha clave pre-compartida. La ventaja de utilizar el modo de seguridad mediante claves pre-compartidas, frente al modo SSL/TLS, es su facilidad de configuración, ya que los participantes no necesitan manejar ningún certificado y no hace falta la intervención de una autoridad certificadora (CA). Por el contrario, dichas claves han de estar a buen recaudo, por lo que este modo se considera menos seguro que al utilizar SSL/TLS (Tomás Cánovas, 2008).

6.3.2 Asignación de Direcciones

En este apartado se comenzará por aclarar el direccionamiento IP utilizado típicamente para las redes privadas, como puede ser cualquier VPN particular implementada con OpenVPN. Para ello hay que decir que la IANA ha reservado los siguientes espacios de direcciones IP para las redes privadas:

Grupo A	10.0.0.0	10.255.255.255	10/8
Grupo B	172.16.0.0	172.31.255.255	172.16/12
Grupo C	192.168.0.0	192.168.255.255	192.168/16

Figura 29. Espacio de Direcciones IP para las Redes Privadas según la IANA.

Fuente: (Tomás Cánovas, 2008)

Estos bloques de direcciones IP son normalmente utilizados en las configuraciones de las VPN, pero es importante seleccionar las direcciones que minimizan la probabilidad de que haya un conflicto con las direcciones IP de red o de subred. Los tipos de conflictos que se han de prevenir son:

- Conflictos provocados por tener diferentes sitios de la VPN con LANs que utilizan la misma numeración de red.
- Conexiones de acceso remoto desde sitios que están utilizando redes privadas que provocan conflictos con las redes o subredes de la propia VPN.

CAPITULO III: ANÁLISIS PRELIMINAR

Con este proyecto se propone diseñar un prototipo funcional con el lenguaje Ginga-NCL, que dé a conocer las alertas emitidas por SNATD, transmitiendo dichas alertas mediante la señal de televisión digital terrestre, para que llegue a las terminales receptoras fijas y móviles, y así de esta manera el usuario pueda informarse de la situación y pueda también tomar sus precauciones.

1 Modelo Preliminar

Dicho prototipo constara de la realización de una plantilla en la cual se podrá introducir toda la información que se quiera difundir. Esta información será extraída de los boletines que envía SNATD, tomando en cuenta los parámetros más relevantes del boletín.



Figura 30. Esquema del funcionamiento propuesto

Fuente: Elaboración propia

En este esquema se sugiere el funcionamiento del sistema optimizado de alerta temprana:

- Llegan boletines del SENAMHI a oficinas de SINAGER-SNATD.
- SINAGER-SNATD realiza boletines de riesgo de desastres analizando la criticidad de la situación.
- SINAGER-SNATD enviaría notas de texto con la información del evento a ocurrir a la estación de radiodifusión (servidor de aplicaciones de Bolivia Tv) mediante un enlace VPN.

- Estas notas de texto junto con la plantilla-NCL serían enviadas al carrusel de datos, donde serían multiplexadas al flujo de datos y se activaría el mensaje de alerta, para luego enviar el flujo completo a la planta transmisora.
- La planta transmisora modularía y enviaría la señal, la cual llegaría a los receptores móviles y fijos de los municipios afectados.

2 Herramientas a utilizar

Todas las herramientas exploradas en esta sección son gratuitas y de código abierto que evolucionan constantemente.

2.1 Herramientas para TDT

El objetivo de utilizar estas herramientas para el desarrollo de aplicaciones interactivas Ginga-NCL en una PC es poder disponer de un entorno similar al middleware Ginga, que se encontrará habitualmente ya instalado en los receptores, pudiendo ser estos adaptadores SetTopBoxes o televisores integrados, obteniendo así una plataforma de simulación del funcionamiento real de dichas aplicaciones.

Para el desarrollo de aplicaciones interactivas Ginga se utilizarán dos tipos de herramientas:

- De desarrollo (Eclipse NCL)
- De presentación. (Ginga4Windows)

2.1.1 Eclipse NCL.

Eclipse es un IDE (Entorno de Desarrollo Integrado) de programación para desarrollar aplicaciones en varios lenguajes (C, java, php. . .). Esta plataforma es extremadamente modular ya que por medio de la adición de plugins se puede extender la funcionalidad, es decir provee la infraestructura para edición de productos a partir de estos. La característica fundamental que fomenta la adaptación de ésta a nuevos entornos de desarrollo es que posee una licencia de tipo EPL de código abierto que permite, usar, modificar, copiar y distribuir nuevas versiones del producto.

Para la creación del lenguaje NCL la Universidad Federal do Maranhao desarrollo NCL-Eclipse que es un editor XML/NCL creado como un plugin para la plataforma Eclipse, el cual facilita la

integración con otras herramientas de desenvolvimiento para la TV Digital, como, por ejemplo, Lua Eclipse.

Estos complementos auxilian y agilizan la creación de aplicaciones, permitiendo que facilidades extras de Eclipse sean reutilizadas e integradas con otras herramientas de desenvolvimiento.



Figura 31. IDE Eclipse

Fuente: <http://www.eclipse.org>

Debido a la gran cantidad de desarrollo de plugins para Eclipse esta herramienta usualmente es utilizada como base para las demás soluciones que facilitar el desenvolvimiento de aplicaciones declarativas, ya que permiten la integración de lenguaje procedural al lenguaje declarativo NCL, convirtiéndose así en la herramienta con más potencia para el desarrollo de contenido interactivo en la actualidad.

2.1.2 Ginga4Windows

Para la visualización de las aplicaciones en entorno Ginga-NCL, existen varios aplicativos, para este proyecto se utilizará Ginga4windows.

Es un reproductor NCL-NCLua desarrollado por la PUC de Río de Janeiro, Brasil. Este software es compatible con el sistema operativo de Windows. Tiene una gran utilidad, al funcionar como un simulador de aplicaciones interactivas escritas bajo el lenguaje de programación NCL.

Las teclas asociadas a los botones del control son:

- Tecla F1-boton rojo
- Tecla F2-boton verde
- Tecla F3-boton amarillo

- Tecla F4-boton azul
- Tecla Enter- botón OK o SELECT



Figura 32. Reproductor NCL-NCLua

Fuente: <http://www.ginga.org.br/es>

2.2 Herramientas para el enlace VPN

2.2.1 OpenVPN

OpenVPN es una herramienta de conectividad basada en software libre: SSL (Secure Sockets Layer). OpenVPN ofrece conectividad punto-a-punto con validación jerárquica de usuarios y host conectados remotamente. Está publicado bajo la licencia GPL, de software libre.

OpenVPN es una excelente nueva solución para VPN que implementa conexiones de capa 2 o 3, usa los estándares de la industria SSL/TLS para encriptar.



Figura 33. Logo del Software OpenVPN

Fuente: <https://openvpn.net>

CAPITULO IV: DESARROLLO DEL PROYECTO

1 Diseño de una VPN mediante la creación y configuración de archivos del software OpenVPN para una comunicación entre dos puntos remotos

1.1 Introducción

La presente parte del proyecto comprende la creación y configuración de archivos del software OpenVPN para una comunicación entre dos puntos remotos, donde SINAGER-SNATD será el servidor, esto permitirá que SNATD puedan enviar la información de sus boletines.



Figura 34. Diagrama de bloques para la elaboración del diseño VPN

Fuente: Elaboración Propia

1.2 Análisis de la situación actual de SNATD

Actualmente la oficina de SNATD está ubicada en el edificio Cofadena en la ciudad de La Paz, dicho edificio consta de 12 pisos, en la cual la oficina de SINAGER-SNATD está ubicada en el piso 4.

La red local existente de SINAGER-SNATD utiliza la tecnología Ethernet 100BASE-TX, teniendo equipos a su disposición para cada área de SINAGER, las cuales utilizan Sistema Operativo Windows 10.

La salida a Internet se la obtiene a través de un ISP (Internet Service Provider) con un ancho de banda de 20/20 Mbps que les llega por fibra óptica. Cabe recalcar que dichos ambientes tienen todo el equipamiento necesario para la implementación de la red VPN.

Para las pruebas del enlace con OpenVPN se lo realizo con:

El modem ZXHN F660, con las siguientes especificaciones:

- **LAN:** Admite cuatro Ethernet 10/100/1000 Base-T Interfaces con conector RJ-45, admite Half / Full Duplex y control de flujo, automático negociación o configuración manual, admite detección automática MDI / MDIX.
- **Power:** Interfaz de entrada de alimentación de +12 V DC (a través de un adaptador externo AC/DC: 90-240V, entrada de CA de 50/60 Hz, salida de CC de 12V).
- **PON:** Interfaz SC/APC para conexión de fibra. Totalmente compatible con los estándares ITU-T G.984.x GPONI.

Los switch´s Cloud Router Switch Microtik, que tienen las siguientes especificaciones:

- **Arquitectura:** MIPSBE
- **Sistema Operativo:** RouterOS
- **Tamaño de RAM:** 128 MB
- **Tamaño de almacenamiento:** 128 MB
- **Estándares inalámbricos de 2.4 GHz:** 802.11 b/g/n
- **Puertos Ethernet 10/100/1000:** 16

La red asignada para las oficinas de SINAGER: 192.168.5.0/24, los mismos que pertenecen a redes privadas de clase C.

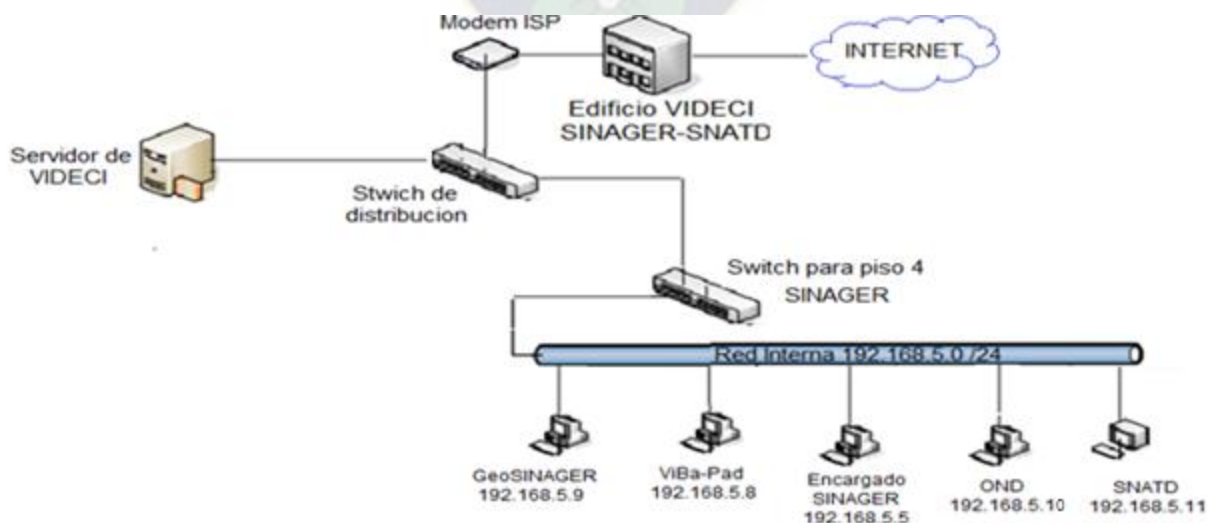


Figura 35. Diagrama de Red Local SINAGER-SNATD

Fuente: Red Sinager

1.3 Configuración del escenario

Una configuración muy típica dentro de las tecnologías VPN consiste en conectar dos sucursales, delegaciones u oficinas, que están separadas por la red de redes (Internet), a través de un túnel seguro, permitiendo a las redes internas de cada oficina transmitir datos seguros, al resto de oficinas, a través de dicho túnel.

En la siguiente figura se muestra el diagrama de red que se pretende diseñar para la conexión remota. Simulando una red privada local para el cliente ya que no se necesitaría de datos específicos, bastaría con que tenga conexión a Internet; sin embargo, el servidor si requiere de información específica, como la dirección IP de la máquina que hará de servidor, para este diseño el cliente se conectara a la IP 52.168.0.1. e ira por el puerto 1194 a la maquina 192.168.5.11.

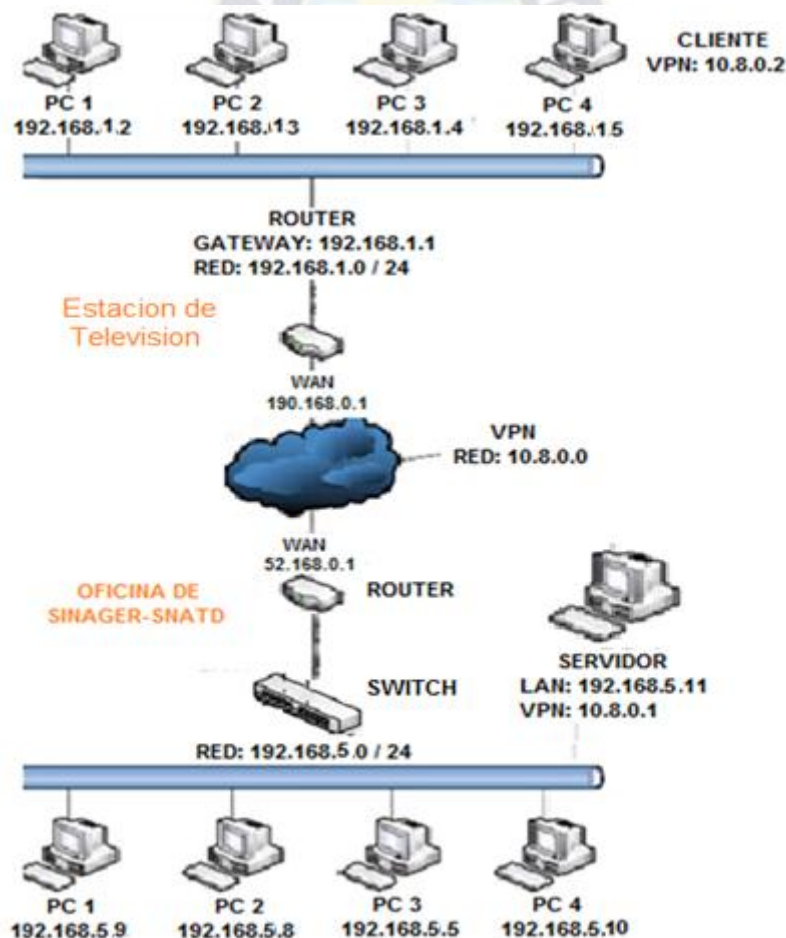


Figura 36. Esquema del Escenario

Fuente: Elaboración Propia

El enlace VPN tendrá las siguientes características:

- **Protocolo:** OpenVPN utiliza el protocolo SSL/TLS (Secure Sockets Layer / Transport Layer Security), los cuales proporciona autenticación y privacidad de la información entre extremos sobre Internet mediante el uso de criptografía. Habitualmente la autenticación mutua requiere un despliegue de infraestructura de claves públicas (o PKI). Esto se lo realizará con la librería OpenSSL que posee OpenVPN.
- **Direcciones IP:** para este diseño se utilizará la red 10.8.0.0 /24 que corresponde a red privada clase A. OpenVPN asignará la dirección 10.8.0.1 al servidor y el 10.8.0.2 al cliente.
- **Puertos:** para este diseño se utilizará el puerto 1194, el cual es el puerto designado por la IANA para OpenVPN.
- **Controlador Virtual:** OpenVPN depende de los controladores virtuales TUN/TAP (modo túnel y modo puente) para poder establecer túneles punto a punto entre los dos extremos de la comunicación. Estos controladores son instalados automáticamente junto con el paquete de instalación de OpenVPN; en este caso lo configuraremos para que sea modo túnel, que vendría a ser el modo punto a punto.
- **Protocolo de transporte:** OpenVPN utiliza los protocolos TCP o UDP como protocolos de transporte, en el caso de este diseño se lo configurara con UDP. Ya que no necesitaríamos de la verificación de si el paquete llega de TCP, dado que OpenVPN tiene su propia verificación.
- **Encriptación:** OpenSSL soporta un gran número de algoritmos criptográficos diferentes. Para el caso de intercambio de clave pública se utilizará el algoritmo Diffie-Hellman con un tamaño de 1024 bits, y para el cifrado se configurará con el algoritmo AES (Advanced Encryption Standard).

En este escenario las máquinas que hacen de extremos del túnel (cliente y servidor OpenVPN) tendrán sistemas operativos Windows.

1.4 Requerimientos para la instalación de OpenVPN

A continuación, se muestra algunos prerequisites si se quiere instalar OpenVPN en una máquina. Se recomienda tener las siguientes características de sistema para que OpenVPN pueda operar:

- Memoria RAM: 64 MB (mínimo) – 512 MB (recomendado).
- Espacio en disco duro: 512 MB (mínimo) – 2 GB (recomendado).
- Procesador (ver arquitecturas): Intel x86-compatible (32 bit) (Intel Pentium I/II/III/IV/Celeron/Xeon, AMD K6/II/III, AMD Duron, Athlon/XP/MP), Intel Itanium (64 bit), Advanced Micro Devices AMD64(Athlon 64, etc) e Intel EM64T (64 bit),

De acuerdo al estudio realizado en las oficinas de SINAGER-SNATD, en lo que se refiere a hardware y software, se ha determinado que los equipos que actualmente están disponibles tienen las características mínimas necesarias para la instalación de OpenVPN, además de poseer ya con conexión a Internet que es el requisito indispensable.

En las instalaciones de Windows (archivos .exe ejecutable) se instalan a la vez todos los paquetes que hacen falta para poder utilizar OpenVPN, es decir, no es necesario descargar ningún archivo de instalación excepto el que proporciona OpenVPN.

1.5 Realización de la instalación y configuración

En este apartado se comentará de manera breve el proceso de instalación de OpenVPN en sistema operativo Windows. Para este caso se va a instalar la versión OpenVPN 2.4.7, ya que es la más actual, realizada el 21 de febrero de 2019. La descarga se ha realizado desde la Web: <http://openvpn.net>.

1.5.1 Instalación de OpenVPN en Windows

La instalación es muy sencilla, ya que todas las aplicaciones y paquetes necesarios (OpenVPN, librerías de OpenSSL, librerías de compresión LZ0 y los drivers TAP) se instalan a partir de un mismo archivo ejecutable, los cuales serán mostrados en las siguientes figuras:



Figura 37. Pasos 1 y 2 en la instalación de OpenVPN

Fuente: Elaboración Propia

Para el paso 3 es importante hacer click en los espacios que no estén marcados, ya que todos estos archivos que se descargaran son muy importantes para el correcto funcionamiento de OpenVPN.



Figura 38. Paso 3 Componentes a Instalar en Windows

Fuente: Elaboración Propia

Por ultimo damos click en instalar y posteriormente en finalizar para terminar con la instalación del Software.

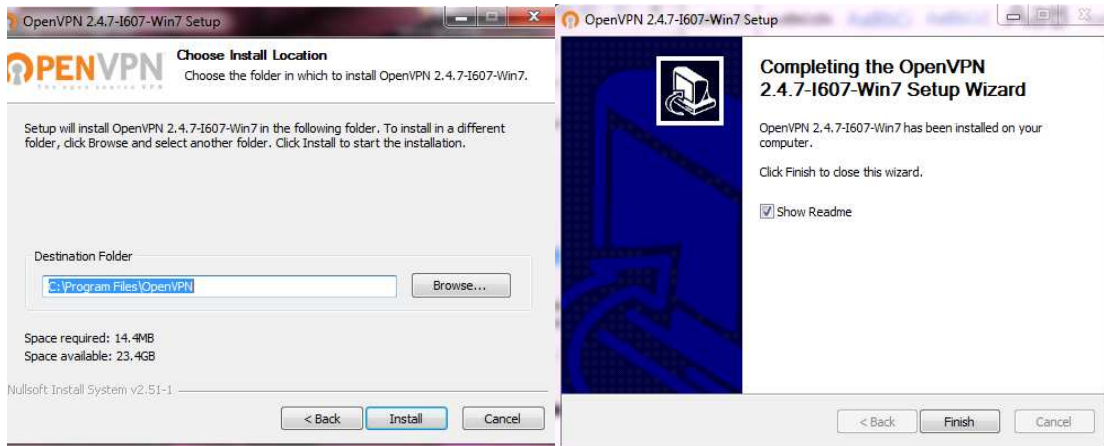


Figura 39. Directorios de instalación y finalización de Instalación

Fuente: Elaboración Propia

Una vez terminado el proceso de instalación, podemos comprobar la obtención de un icono en la barra de tareas que presenta la interfaz TUN/TAP.

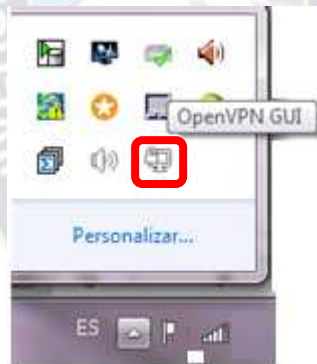


Figura 40. Icono del Proceso de la GUI Instalada

Fuente: Elaboración Propia

Aunque se haya instalado una GUI de OpenVPN en nuestro Windows, su uso solo se limita a conectar/desconectar el túnel, con lo cual se procederá a crear la seguridad en dicho túnel posteriormente.

1.5.2 Implementación de la Seguridad

Para la implementación se tendrá que hacer uso de algunos scripts para añadir algunos métodos de seguridad a nuestra VPN.

Con estos scripts podremos crear claves, certificados, etc. Los cuáles serán necesarios cuando vayamos a utilizar algún método de seguridad.

Para poder utilizar seguridad mediante TLS tendremos que establecer un PKI infraestructura de clave pública, que consiste en:

- Un certificado público y una clave privada para el servidor, y un certificado Público y una clave privada para el cliente.
- Un certificado público y clave privada de la Autoridad Certificadora (CA) que tendrá la función de firmar cada uno de los certificados del cliente y del servidor.

El servidor y el cliente se autenticarán verificando que el certificado que le ha presentado el otro participante fue firmado por la Autoridad Certificadora (CA) para luego comprobar algunos campos del certificado como el “common name” del certificado o el tipo de certificado (cliente o servidor). A continuación, se mostrarán características de este modelo de seguridad:


- No es necesario que el servidor conozca los certificados de cada uno de los clientes que puedan conectarse.
- El servidor solo aceptara clientes cuyos certificados fueron firmados por el certificado de la Autoridad Certificadora (CA), ya que dicho certificado lo tiene tanto el servidor como los clientes.

1.5.2.1 Creación de Clave (Certificado de la CA)

Para la generación del certificado público y la clave privada de la CA, se va a hacer uso de los scripts proporcionados por OpenVPN que se hallan en la carpeta “easy-rsa” en la ruta donde se haya instalado OpenVPN. El procedimiento para ejecutar dichos scripts se hace mediante consola.

Ya dentro la consola, nos ubicaremos en el directorio de: `c:\program files\Openvpn\easy-rsa`

El primer paso será ejecutar el archivo “**init-config.bat**” desde la consola, para copiar algunos archivos necesarios al mismo directorio (esto además, sobrescribirá los archivos “vars.bat” y ”openssl.cnf” ya existentes).

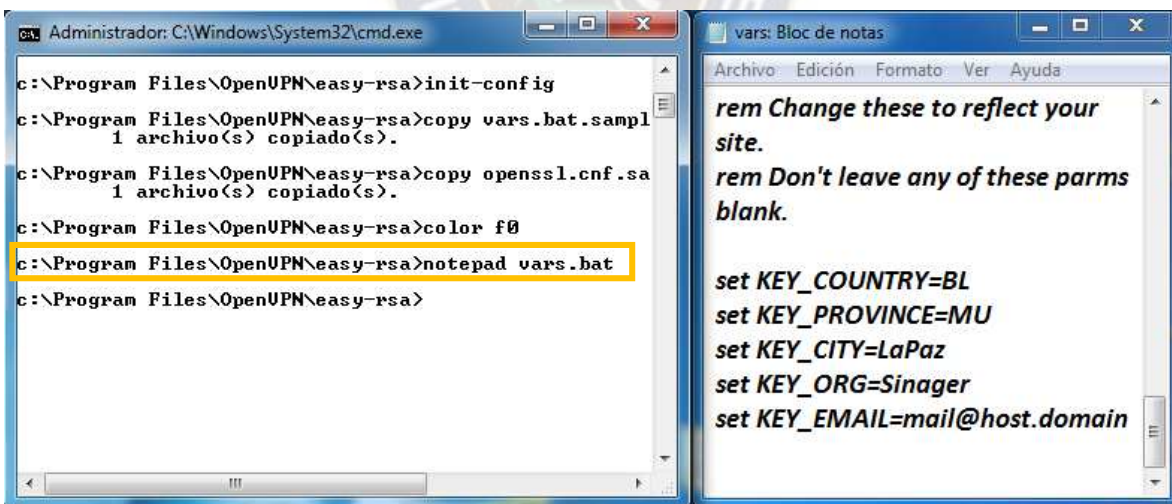


```
Administrador: C:\Windows\System32\cmd.exe
c:\Program Files\OpenUPN\easy-rsa>init-config
c:\Program Files\OpenUPN\easy-rsa>copy vars.bat.sample vars.bat
1 archivo(s) copiado(s).
c:\Program Files\OpenUPN\easy-rsa>copy openssl.cnf.sample openssl.cnf
1 archivo(s) copiado(s).
c:\Program Files\OpenUPN\easy-rsa>color f0
c:\Program Files\OpenUPN\easy-rsa>
```

Figura 41. Ejecución del Comando "init.config"

Fuente: Elaboración Propia

Una vez tengamos el archivo vars.bat, lo editamos en los parámetros: KEY_COUNTRY, KEY_PROVINCE, KEY_CITY, KEY_ORG, KEY_EMAIL introduciendo nuestros datos.



```
Administrador: C:\Windows\System32\cmd.exe
vars: Bloc de notas
Archivo Edición Formato Ver Ayuda
rem Change these to reflect your
site.
rem Don't leave any of these parms
blank.

set KEY_COUNTRY=BL
set KEY_PROVINCE=MU
set KEY_CITY=LaPaz
set KEY_ORG=Sinager
set KEY_EMAIL=mail@host.domain

c:\Program Files\OpenUPN\easy-rsa>init-config
c:\Program Files\OpenUPN\easy-rsa>copy vars.bat.sample vars.bat
1 archivo(s) copiado(s).
c:\Program Files\OpenUPN\easy-rsa>copy openssl.cnf.sample openssl.cnf
1 archivo(s) copiado(s).
c:\Program Files\OpenUPN\easy-rsa>color f0
c:\Program Files\OpenUPN\easy-rsa>notepad vars.bat
c:\Program Files\OpenUPN\easy-rsa>
```

Figura 42. Configuración de Parámetros en VARS

Fuente: Elaboración Propia

Después de editarlo, ejecutamos el script “vars”.

Por último, solo queda ejecutar el script “**build-ca**” para generar la clave privada y el certificado de la Autoridad Certificadora.

Ejecutado el comando “**build-ca**” se tendrá que ingresar el nombre para el certificado, para este diseño se llamara:

Common name: llave-servidor

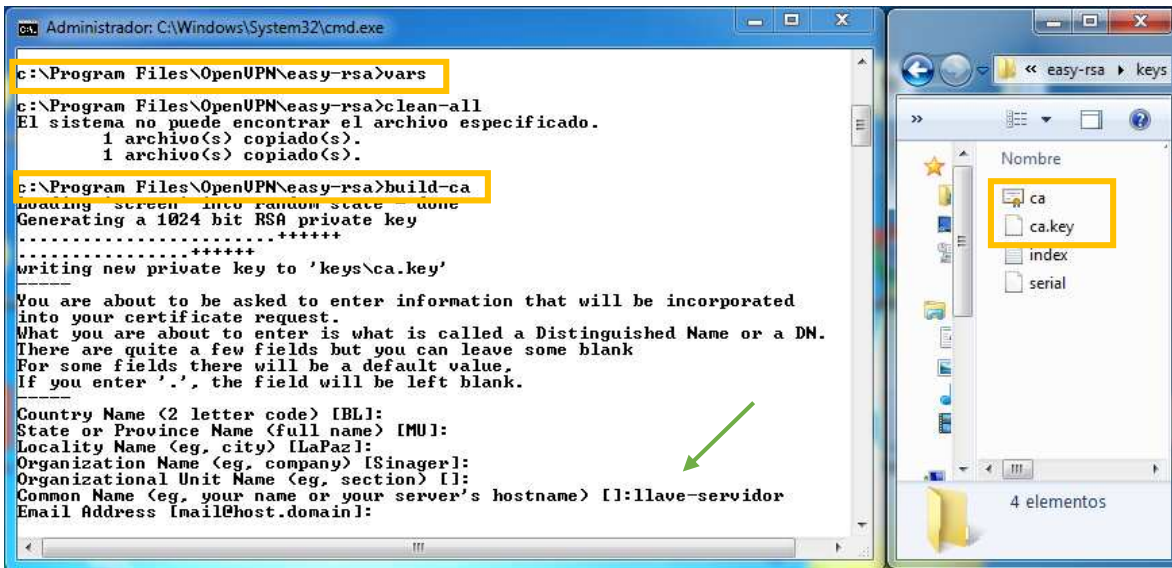


Figura 43. Creación del Certificado CA

Fuente: Elaboración Propia

Con este procedimiento se generará dos nuevos ficheros.

- **ca.crt:** fichero correspondiente al certificado público de la CA.
- **ca.key:** fichero correspondiente a la clave privada de la CA, la cual debe mantenerse protegida ya que es la clave más importante de toda la PKI.

1.5.2.2 Creación de clave y certificado del servidor

Los pasos son los mismos que fueron utilizados para la creación del certificado de la CA, para ello ejecutamos nuevamente el archivo “**vars**”, seguido de esto ejecutamos el script “**build-key-server**” para la creación de clave/certificado del servidor.

Para lo cual debemos introducir algunos parámetros como:

Common name: servidor-vpn.

Password: sinager-snatd.

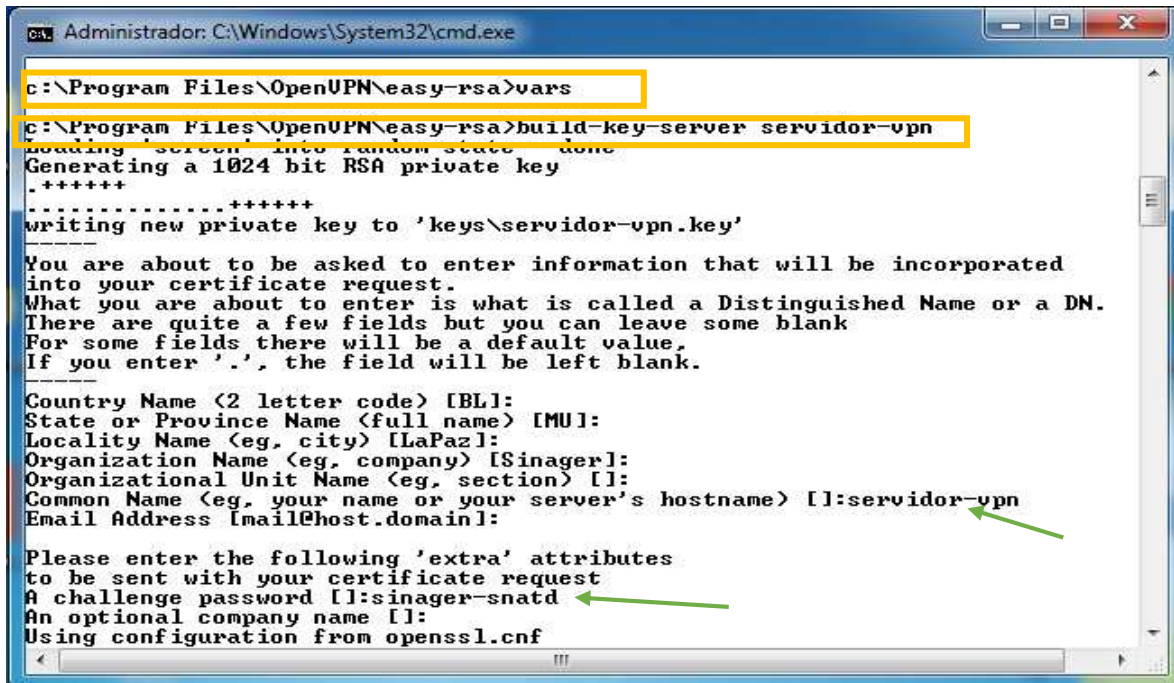


Figura 44. Llenado de datos para la Creación de Certificado y llave del Servidor

Fuente: Elaboración Propia

Como podemos observar en la siguiente imagen, tras completar todos los campos del certificado del servidor, nos pregunta si queremos firmar el certificado, utilizando para ello el certificado de la CA, y tendremos que responder que sí.

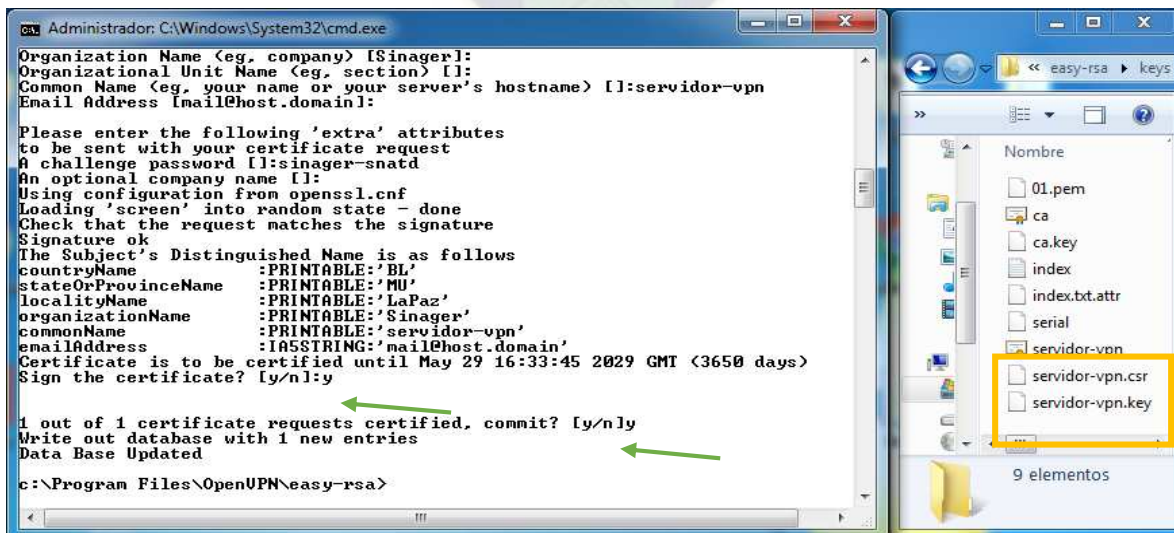


Figura 45. Creación del Par Certificado y Clave del Servidor

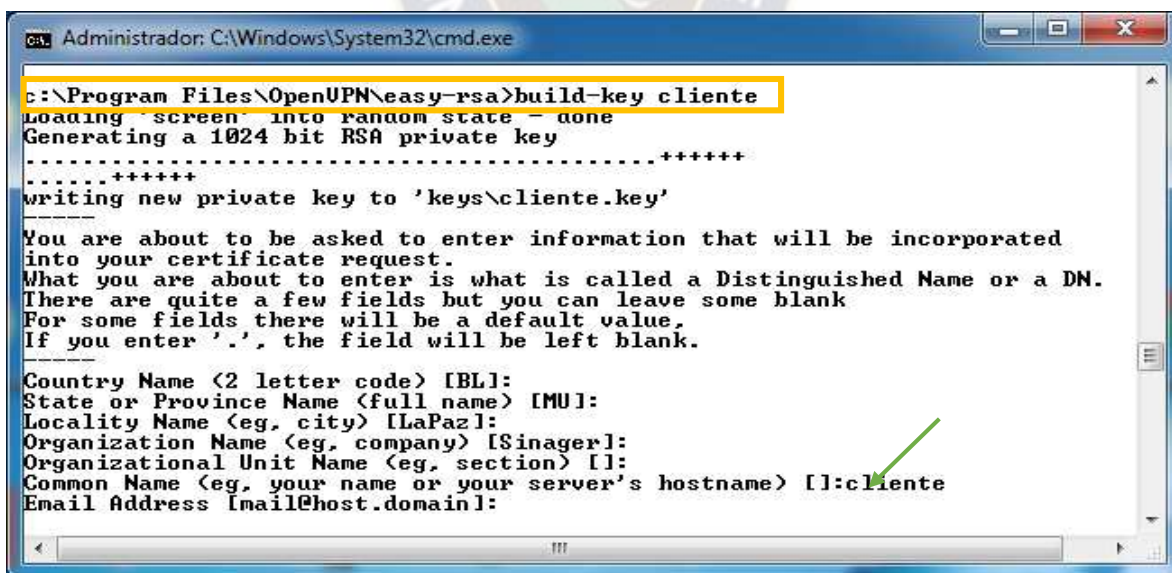
Fuente: Elaboración Propia

Tras realizar estos pasos hemos generado 4 ficheros nuevos:

- **Servidor-vpn.crt:** fichero correspondiente al certificado público del servidor.
- **Servidor-vpn.key:** fichero correspondiente a la clave privada del servidor, la cual debe permanecer protegida.
- **01.pem:** fichero correspondiente al certificado público del servidor en formato PEM. El nombre del fichero proviene de su número de serie del certificado, el cual es 01.
- **Servidor-vpn.csr:** este fichero sirve para poder crear el certificado del servidor en otra máquina que pueda crearlo y firmarlo, ya que este fichero tiene toda la información que le hace falta.

1.5.2.3 Creación de la clave y certificado del cliente

Para crear la clave privada y el certificado público de los clientes se hacen casi los mismos pasos que se han realizado para el servidor y para la Autoridad Certificadora (CA). De nuevo se ha de ejecutar el script “vars.bat”. En este caso, el script para crear la clave y el certificado del cliente se utiliza el script “build-key.bat”, y el nombre que tendrá el cliente, en este caso **cliente**.



```
Administrador: C:\Windows\System32\cmd.exe
c:\Program Files\OpenVPN\easy-rsa>build-key cliente
Loading screen into random state - done
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'keys\cliente.key'

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

-----
Country Name (2 letter code) [BL]:
State or Province Name (full name) [MU]:
Locality Name (eg, city) [LaPaz]:
Organization Name (eg, company) [Sinager]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []: cliente
Email Address [mail@host.domain]:
```

Figura 46. Llenado de Datos para la Creación de Certificado y llave Cliente

Fuente: Elaboración Propia

Al igual que en el caso del servidor, tras completar todos los campos del certificado de cada cliente nos preguntan si queremos firmar el certificado, utilizando para ello el certificado de la CA, y tenemos que responderle que sí.

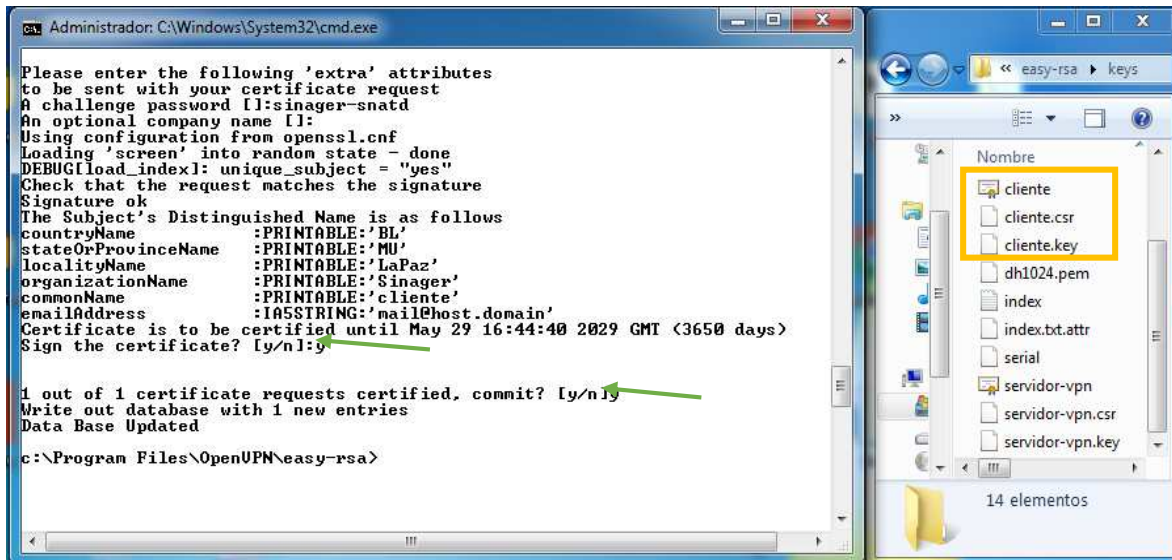


Figura 47. Creación del Par Certificado y Clave del Cliente

Fuente: Elaboración Propia

Tras realizar estos pasos hemos generado 3 ficheros nuevos:

- **cliente.crt:** fichero correspondiente al certificado público del cliente.
- **cliente.key:** fichero correspondiente a la clave privada del cliente, la cual debe permanecer protegida.
- **cliente.csr:** este fichero sirve para poder crear el certificado del cliente en otra máquina que pueda crearlo y firmarlo, ya que este fichero tiene toda la información que le hace falta.

1.5.2.4 Creación de los parámetros Diffie Hellman

Los parámetros Diffie Hellman deben ser generados para el servidor OpenVPN. Para ello se ha de ejecutar el script “**build-dh**”. El tamaño de estos parámetros dependerá del valor que le hayamos asignado a la variable **KEY_SIZE** en el script **vars** (lo normal es utilizar un tamaño de 1024 bits o 2048 bits). En este proyecto se ha utilizado un tamaño para los parámetros Diffie Hellman de 1024 bits.

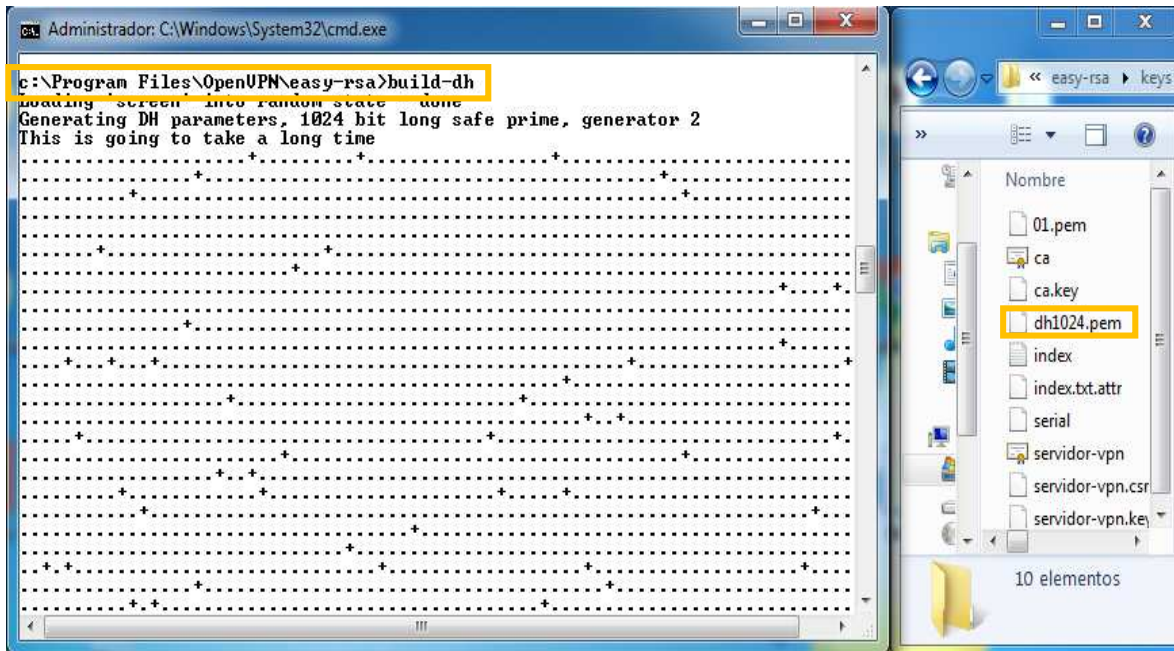


Figura 48. Creación del Archivo dh1024.pem

Fuente: Elaboración Propia

Tras ejecutar este script se genera un fichero en formato PEM llamado dh1024.pem. Los parámetros Diffie Hellman se utiliza para poder realizar el intercambio de una clave entre dos participantes de manera segura.

1.5.3 Configuración de archivos

En este apartado se realizará la configuración de los archivos:

“servidor.ovpn”, y “cliente.ovpn”, para que se puedan comunicar remotamente.

1.5.3.1 Servidor

El fichero de configuración del servidor OpenVPN (servidor.ovpn) es el siguiente :

A continuación vamos a configurar y describir brevemente lo que hace cada línea del fichero de configuración del servidor OpenVPN (servidor-vpn.ovpn).


```
#####  
# Sample OpenVPN 2.0 config file for #  
# multi-client server. #  
# This file is for the server side #  
# of a many-clients <-> one-server #  
# OpenVPN configuration. #  
# OpenVPN also supports #  
# single-machine <-> single-machine #  
# configurations (See the Examples page #  
# on the web site for more info). #  
#####  
tls-server  
port 1194  
dev tun  
proto udp  
dh dh1024.pem  
ca ca.crt  
cert servidor-vpn.crt  
key servidor-vpn.key  
server 10.8.0.0 255.255.255.0  
keepalive 10 120  
cipher AES-256-CBC  
persist-key  
persist-tun  
status openvpn-status.log
```

Figura 49. Configuración del Archivo *servidor-vpn.ovpn*

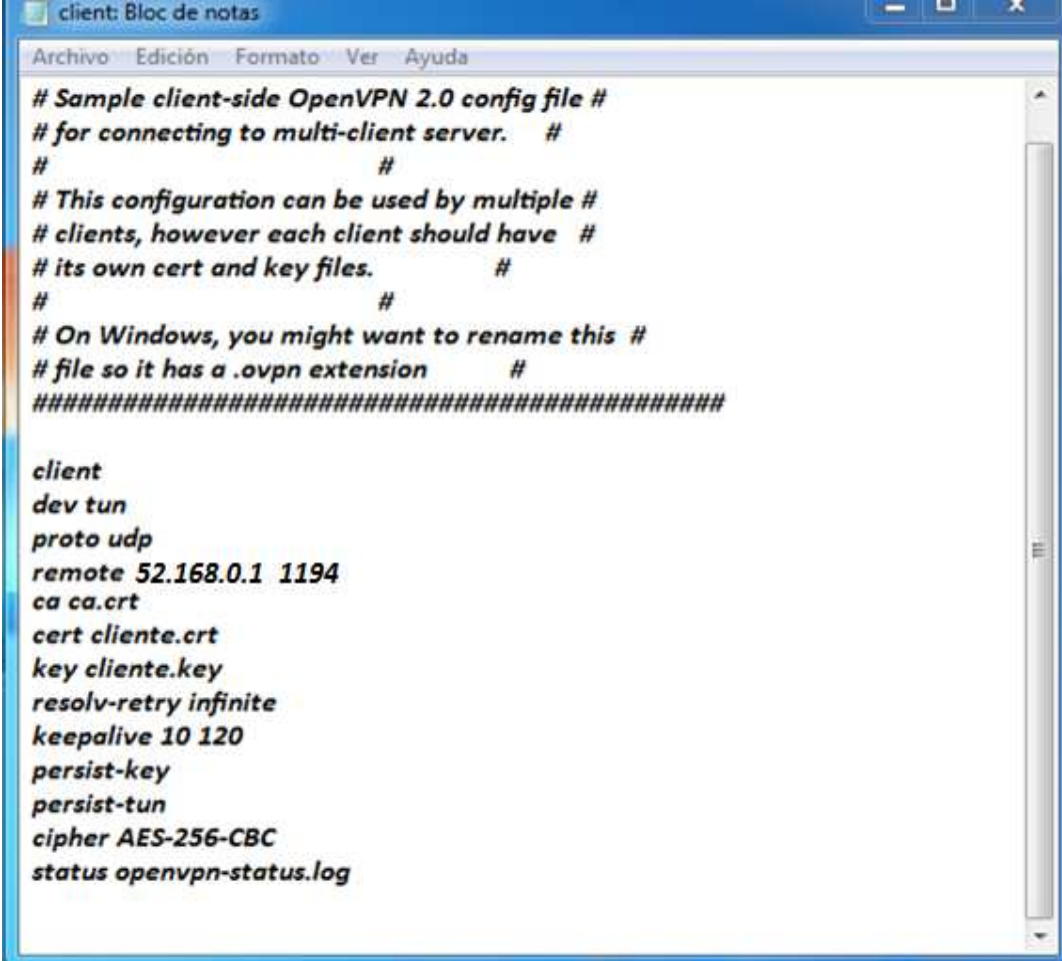
Fuente: Elaboración Propia

- **tls-server:** indica que esta máquina actuara como servidor durante el establecimiento del protocolo TLS.
- **port 1194:** utilizamos el puerto 1194.
- **dev tun:** indica que vamos a implementar un túnel mediante un dispositivo “tun”.
- **proto udp:** se utiliza el protocolo UDP como protocolos de transporte del túnel.
- **dh dh1024.pem:** cargamos los parámetros Diffie Hellman.
- **ca ca.crt:** cargamos el certificado público de la CA.
- **cert servidor-vpn.crt:** cargamos el certificado del Servidor OpenVPN.
- **key servidor-vpn.key:** cargamos la clave privada del Servidor OpenVPN.
- **keepalive 10 120:** se puede traducir en que el Servidor OpenVPN enviará un ping cada 10 segundos y esperará 120 segundos máximo para recibir contestación, sino deducirá que el otro extremo (el Cliente OpenVPN) ha caído.

- **persist-key:** permite que las claves no tengan que ser re-leídas cuando el Servidor OpenVPN es reiniciado.
- **persist-tun:** permite que el túnel no tenga que ser cerrado y re-abierto de nuevo tras reiniciar el Servidor OpenVPN.
- **status openvpn-status.log:** indica el fichero donde se volcará la información de estado del túnel.

1.5.3.2 Cliente

El fichero de configuración del Cliente OpenVPN (“cliente.ovpn”) es el siguiente:



```

# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server. #
# #
# This configuration can be used by multiple #
# clients, however each client should have #
# its own cert and key files. #
# #
# On Windows, you might want to rename this #
# file so it has a .ovpn extension #
#####

client
dev tun
proto udp
remote 52.168.0.1 1194
ca ca.crt
cert cliente.crt
key cliente.key
resolv-retry infinite
keepalive 10 120
persist-key
persist-tun
cipher AES-256-CBC
status openvpn-status.log

```

Figura 50. Configuración del Archivo cliente.ovpn

Fuente: Elaboración Propia

A continuación vamos a describir brevemente lo que hace cada línea del fichero de configuración del Cliente OpenVPN (client.ovpn).

- **client:** indica que esta máquina actuará como cliente durante el establecimiento del protocolo TLS.
- **dev tun:** indica que vamos a implementar un túnel mediante un dispositivo “tun” de los drivers TUN/TAP.
- **proto udp:** se utilizará el protocolo UDP como protocolo de transporte del túnel.
- **remote 52.168.0.1 1194:** indica a que dirección IP se tiene que conectar para establecer el túnel y utilizando el puerto 1194 asignado por la IANA. En este caso es la IP de la institución que hará de servidor (IP de SINAGER-SNATD).
- **ca ca.crt:** carga el certificado de la CA.
- **cert cliente.crt:** carga el certificado del Cliente OpenVPN.
- **key cliente.key:** carga la clave privada del Cliente OpenVPN.
- **resolv-retry infinite:** el cliente intentará de manera indefinida resolver la dirección o nombre de host dado por la directiva “remote”.
- **keepalive 10 120:** se puede traducir en que el Cliente OpenVPN enviará un ping cada 10 segundos y esperará 120 segundos máximo para recibir contestación, sino deducirá que el otro extremo (el Servidor OpenVPN) ha caído.
- **persist-key:** permite que las claves no tengan que ser re-leídas cuando el Cliente OpenVPN es reiniciado.
- **persist-tun:** permite que el túnel no tenga que ser cerrado y re-abierto de nuevo tras reiniciar el Cliente OpenVPN.
- **status openvpn-status.log:** indica el fichero donde se volcará la información de estado del túnel.

1.5.4 Inicialización de la conexión

Acabado los pasos de creación y configuración del servidor y cliente, se debe copiar los archivos generados a sus respectivas máquinas de servidor y cliente, y se procede a iniciar ambas maquinas

Al iniciar el servidor y el cliente se les asignara una IP del segmento de red según el rango que se designo en los archivos de configuración, en este caso 10.8.0.1 para el servidor, 10.8.0.2 para el cliente, y con esto se verificara la conexión exitosa.

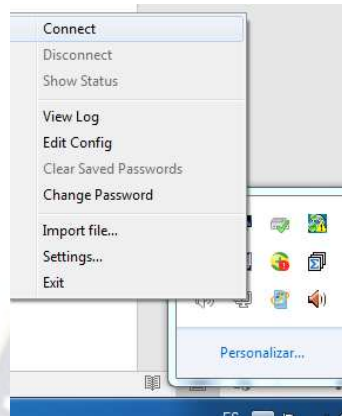


Figura 51. Inicialización del Servidor y Cliente

Fuente: Elaboración Propia

1.6 Calculo del tráfico de datos

Dado que los datos que enviaría SNATD son datos planos de texto (.txt), y que los boletines que envía son en promedio de 20 a 30 al año, y teniendo en cuenta que de esos boletines el 30% se envía en el mes de febrero por la subida de ríos, en el peor de los casos 10 boletines.

Para el cálculo del trafico tomando el caso extremo de un boletín por día y tomando en cuenta que las hojas de texto son de máximo 4 KB, se tendría que enviar 4KB en un día.

Por lo que se concluye que no es necesario un gran ancho de banda para el envío y recepción de la información, bastaría con un servicio de internet mínimo de 1Mbps.

2 Diseño del prototipo funcional con el software Eclipse para la emisión de alertas del SNATD (Sistema Nacional de Alerta Temprana de Desastres)

2.1 Introducción

La presente sección comprende el diseño de la plantilla, en la cual se pondrán los parámetros del boletín que envía SINAGER-SNATD, con la ayuda del software Eclipse, programando en el lenguaje NCL.



Figura 52. Diagrama de bloques para la generación del código NCL

Fuente: Elaboración Propia

Cabe recalcar que con el apagón analógico en Bolivia todas las emisoras de televisión están obligadas a renovar sus equipos. En este caso teniendo en cuenta que el canal en cuestión es el canal estatal y dicho canal ya está funcionado con la tecnología digital, por lo tanto, es importante hacer notar que todos los equipos necesarios para la implementación de este proyecto ya están disponibles.

2.2 Recolección de parámetros para la elaboración del prototipo

El sistema nacional de alerta temprana de desastres (SNATD), es el sistema de vigilancia y monitoreo de amenazas probables frente a las condiciones de vulnerabilidades existentes, anteriores a la ocurrencia de desastres y/o emergencias, con la finalidad de proporcionar información sobre el nivel o escenario de riesgos, para activar protocolos de prevención.

SNATD elabora un boletín de criticidad (riesgo) que es enviado a las gobernaciones y municipios, el cual contiene la siguiente información.

2.2.1 Tipos de amenazas

Se muestra el tipo de amenaza que se acerca y algunas características de la amenaza como ser: dirección e intensidad de los vientos, nombres de los ríos que subirán de caudal, nombres de cordilleras en donde nevara, etc. A continuación, mencionamos los tipos de amenazas que más ocurren en Bolivia:

- Riesgo de inundación, deslizamiento, y/o riadas a consecuencia de lluvias.
- Riesgo de incendios forestales por el incremento de las temperaturas máximas, humedad relativa media y vientos moderados, condiciones favorables para la propagación de los mismos.
- Pronostico de nevadas.
- Pronostico de frente frio.
- Pronostico de Sequias.
- Pronostico de vientos.

2.2.2 Nivel de alerta

En el boletín se envía el nivel de alerta según el nombre del municipio, solo se envía los niveles amarillo naranja y rojo, puesto que verde no tiene mucho grado de peligrosidad.

<p>Condiciones de normalidad y vigilancia de carácter preventiva frente a la probabilidad de amenaza recurrente</p> <p>Se ejecutan actividades de PREVENCIÓN y MITIGACIÓN.</p>	<p>Condiciones de vigilancia, se establece la probabilidad de la presencia de una amenaza, cuyo desarrollo y evolución aumenta el riesgo de afectación, para lo cual debe formular y/o ajustar el Plan de Contingencias.</p>	<p>mayor certeza de la presencia de la amenaza y se han detectado vulnerabilidades en las regiones de riesgo, requiere la activación del GOE y Plan de contingencias para acciones preventivas como evacuación a zonas seguras y de preparación: primera respuesta, grupos de búsqueda y rescate, asistencia médica, insumos humanitarios, asistencia alimentaria y otras acciones inherentes.</p>	<p>se ha confirmado la presencia del evento adverso con alto grado de riesgo de afectación por la magnitud del evento, debiendo aplicarse y activarse los mecanismos de atención y respuesta a la emergencia de acuerdo al Plan de Contingencia, poniendo a disposición de la MAE los recursos que se necesiten</p>
--	--	--	---

Figura 53. Clasificación de alertas

Fuente: (Asamblea Legislativa Plurinacional , 2014)

2.2.3 Municipios

El boletín también muestra una lista de los departamentos y sus municipios correspondientes que podrían ser afectados a causa del desastre natural. Esta lista de municipios con su respectivo nivel de alerta.

2.2.4 Fechas y características

Como son fenómenos naturales de largo rango de ocurrencia es necesario poner el rango de fechas en el que existe más probabilidad que sucedan dichos fenómenos naturales.

2.2.5 Mapa de vulnerabilidad

Muestra en el mapa de Bolivia los lugares que serán afectados por el desastre, pintando por colores el nivel de alerta.

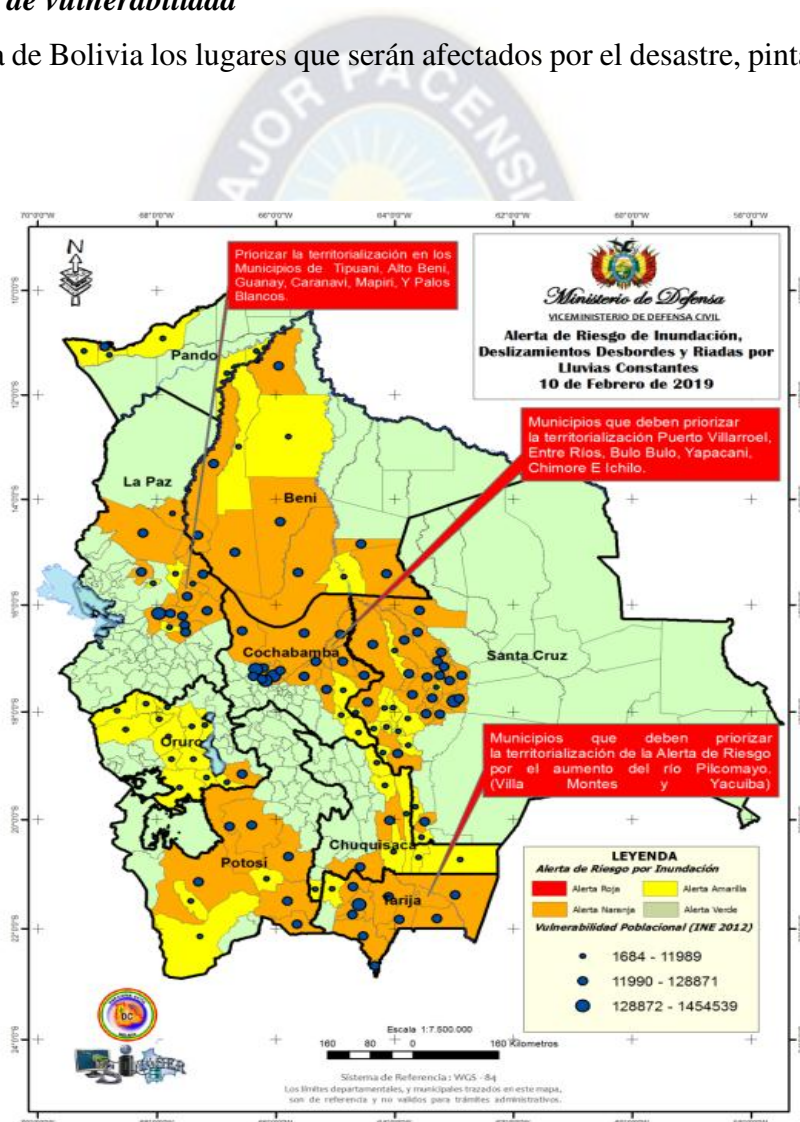


Figura 54. Mapa de Inundaciones, deslizamientos, desbordes y/o riadas

Fuente: Fuente especificada no válida.

2.2.6 Recomendaciones

Son recomendaciones que se da a la población afectada, indica de manera general si se debe evacuar o no, cantidad de cosas a llevar, etc.

2.3 Construcción del modelo

El objetivo principal del Sistema de Alerta, y en específico el de las aplicaciones interactivas diseñadas, es generar en el televidente un sentido de atención hacia el contenido informativo de la aplicación más que a la programación televisiva; basándose en este aspecto y teniendo en cuenta criterios de diseño gráfico, es recomendable que la aplicación tenga una interfaz amigable evitando estructuras complejas para la navegación a través de los menús, también se debe evitar cuadros con excesivos textos y pocos colores, y finalmente se debe permitir la finalización de la aplicación en cualquier punto de la misma.

La tendencia natural de los seres humanos de leer de izquierda a derecha y de arriba hacia abajo, hace que la parte izquierda de la pantalla tenga cierto tipo de prioridad sobre las demás, por lo que la mayor parte de la información de la aplicación será proyectada en esta región.

Para la construcción del modelo se recolectarán los parámetros más relevantes que muestra un boletín de SNATD. El prototipo a realizar se guía en la siguiente lógica:



Figura 55. Estructura de un Documento NCL

Fuente: (Galabay Teolongo & Vivar Espinoza, 2012)

2.3.1 ¿Qué vamos a mostrar? - Objetos Media

Al comenzar a diseñar un programa, es el contenido audiovisual interactivo lo primero que se escoge; que vídeos, imágenes, textos y otros medios será presentado por el programa, este contenido está representado por los nodos de media. Una media representa cada nodo de un documento que informa el descriptor cual está relacionado.

Para la realización del proyecto se tomarán 5 medias, de las cuales las últimas tres serán proporcionadas por SINAGER-SNATD y las primeras dos son las imágenes de fondo que se verán en pantalla:



Figura 56. Imagen Principal del prototipo

Fuente: Elaboración Propia



Figura 57. Imagen Secundaria del prototipo

Fuente: Elaboración Propia

- **Plantilla.png:** Es la imagen de fondo que se verá en pantalla
- **Button.png:** Es la imagen que se mostrara cuando termine la imagen principal “plantilla”.
- **Alerta.txt:** Esta hoja contendrá el nivel de alerta y el tipo de amenaza a difundir.
- **Fecha.txt:** Esta hoja contendrá el rango de fechas a ocurrir y especificaciones del desastre.
- **Municipios.txt:** Esta hoja contendrá los nombres de los municipios que se verán afectados.

Para las últimas tres medias se optó por escoger los parámetros más relevantes que envía SNATD, las cuales tendrán que estar en formato .txt.

2.3.2 ¿Dónde los vamos a mostrar? - Regiones

Después de definir el contenido multimedia del programa, se debe definir el Área en donde se va a mostrar cada elemento en la pantalla, por medio de elementos llamados regiones. Una región representa la posición y tamaño de la zona donde ciertos objetos media serán visualizados, es decir una región sirve para inicializar la posición de las medias en una ubicación específica

Aunque una región representa el lugar en donde se podría presentar un nodo media, ésta no indica que nodo medio será presentado en dicha región, esta asociación se realiza por medio de un descriptor.

Para la realización del prototipo se tendrá 5 regiones y estarán distribuidas como muestra la siguiente figura, de las cuales la plantilla.png que es la base de las hojas .txt, tendrá un 80% de la pantalla principal.

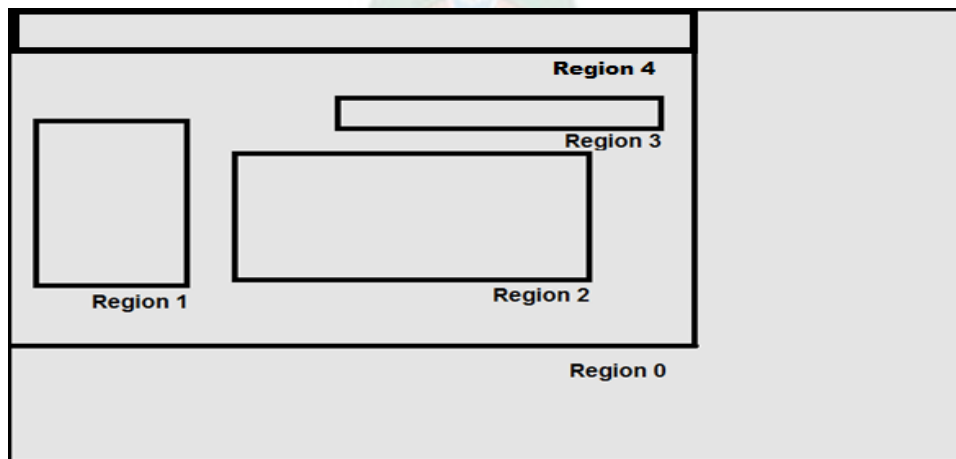


Figura 58. Distribución de Regiones

Fuente: Elaboración Propia

2.3.3 ¿Cómo los vamos a mostrar? – Descriptores

La definición de una región debe ser complementada con otra información que indique cómo cada nodo será presentado. Esta descripción de las características de cada nodo se realiza a través de los elementos llamados descriptores. Un descriptor puede detallar parámetros de representación de los nodos, incluyendo a la región donde tendrá lugar la presentación de su volumen, su transparencia, y el tiempo de duración, entre otros.

Cuando se define un descriptor, es necesario definir la región a la que se asocia. Todas las medias que utilizan éste son asociados a la región correspondiente.

A continuación, se muestra la relación entre regiones y medias y su tiempo de duración.

- Plantilla.png - Región cero (30 segundos).
- Municipios.txt - Región uno (30 segundos).
- Alerta.txt - Región tres (30 segundos).
- Fecha.txt - Región dos (30 segundos).
- Button.png – Región cuatro (28800 segundos = 8 horas).

2.3.4 ¿Cuándo los vamos a mostrar? - Links y Conectores

Una vez que se hayan seleccionado los nodos que formaran parte del programa, se hace necesario definir cuál será el primer nodo en ser presentado y el orden de ejecución de los demás. Esta definición se hace con el elemento llamado puerto, estos definen los nodos que serán presentados cuando un nodo de contexto iniciara.

Los links no definen todas las relaciones de sincronización entre los nodos y la interactividad del programa, para eso requiere el uso de conectores.

Como se observa en la siguiente figura, la región uno, dos y tres se adaptan a la región cero, que vendría a ser el puerto, y la región cuatro será visualizada al terminar el tiempo de vida de la región 0,1,2 y 3.

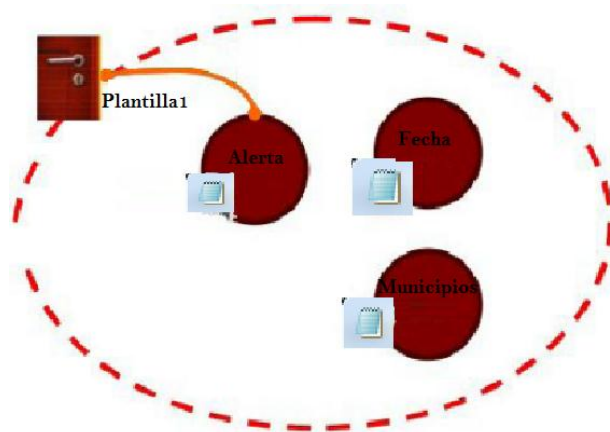


Figura 59. Puerto del nodo de composición

Fuente: Elaboración Propia

Para la sincronización del prototipo se pretende que siga el diagrama de tiempo que se muestra en la siguiente figura:

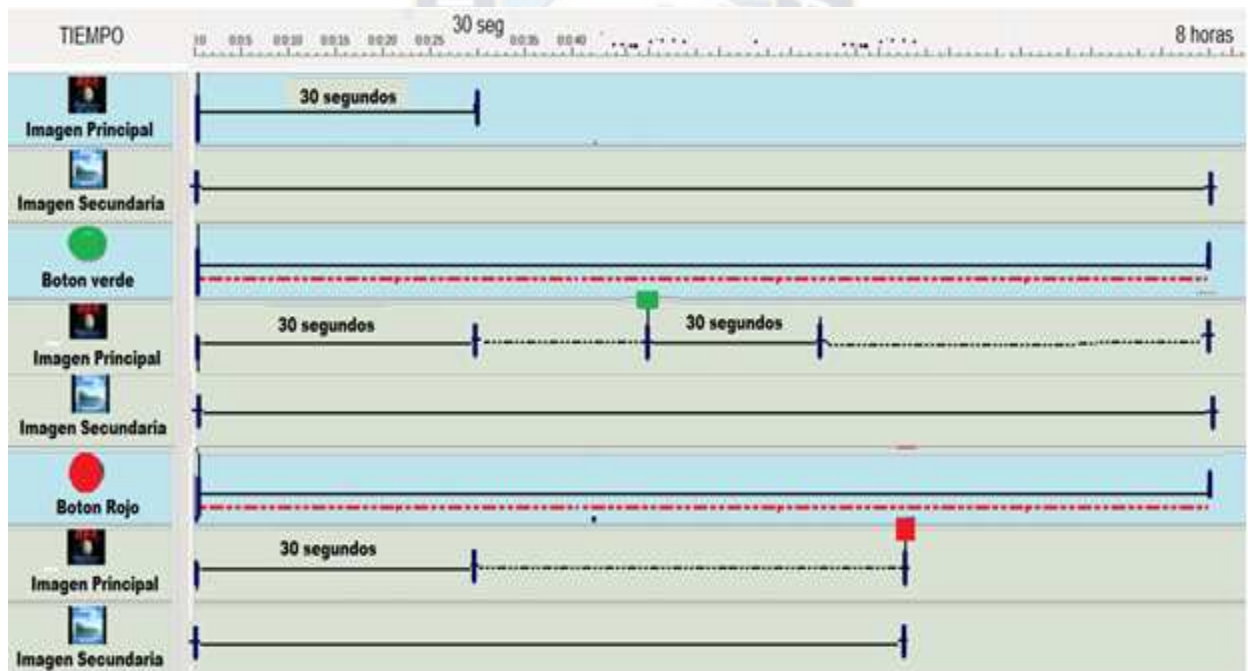


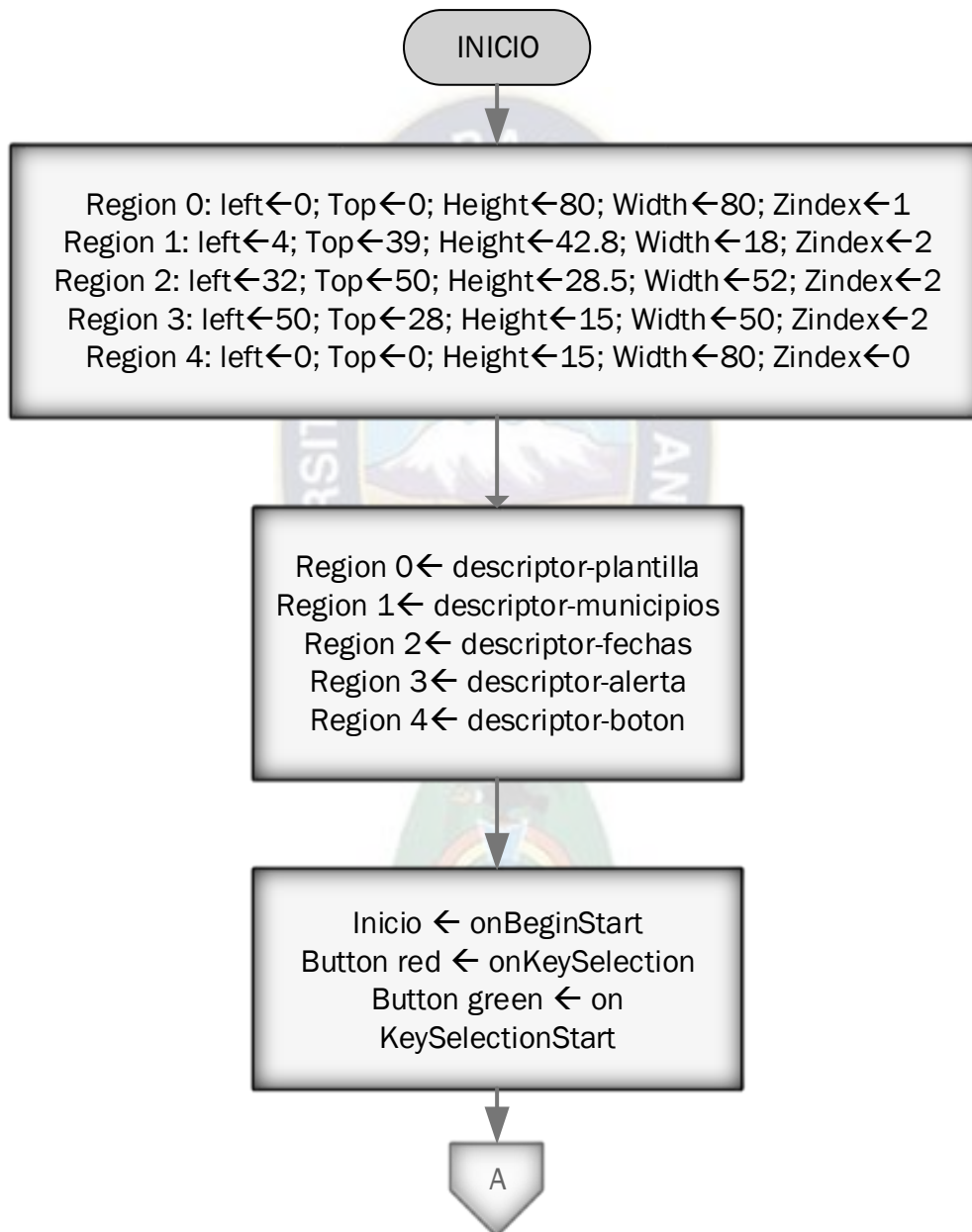
Figura 60. Diagrama de tiempo del prototipo

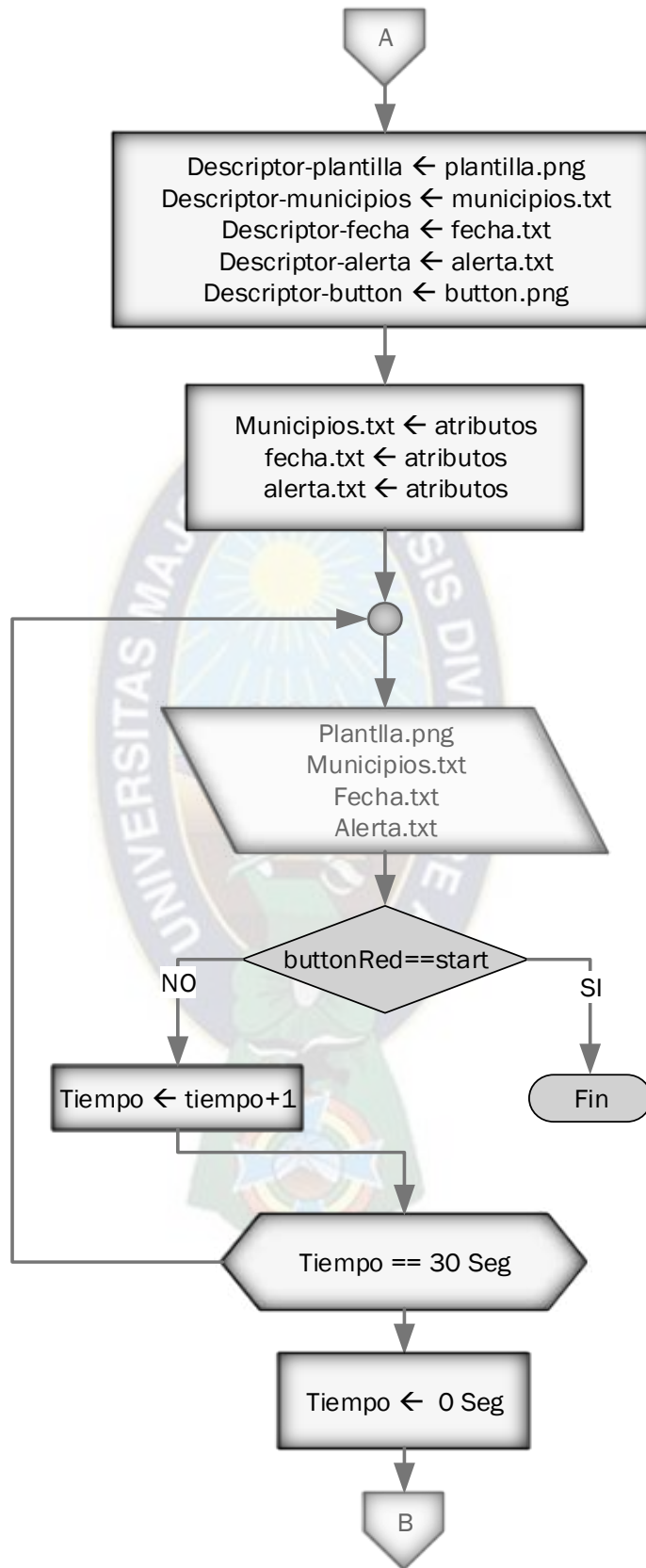
Fuente: Elaboración Propia

La alerta comenzara mostrando la imagen principal durante 30 segundos. Después de transcurrido este tiempo se mostrará la imagen secundaria durante 8 horas. Se utilizará los botones rojo y verde

del control remoto para que cumplan las siguientes tareas: rojo para terminar el programa y verde que inicia la imagen principal nuevamente.

A continuación, se muestra un diagrama de flujos del prototipo propuesto:





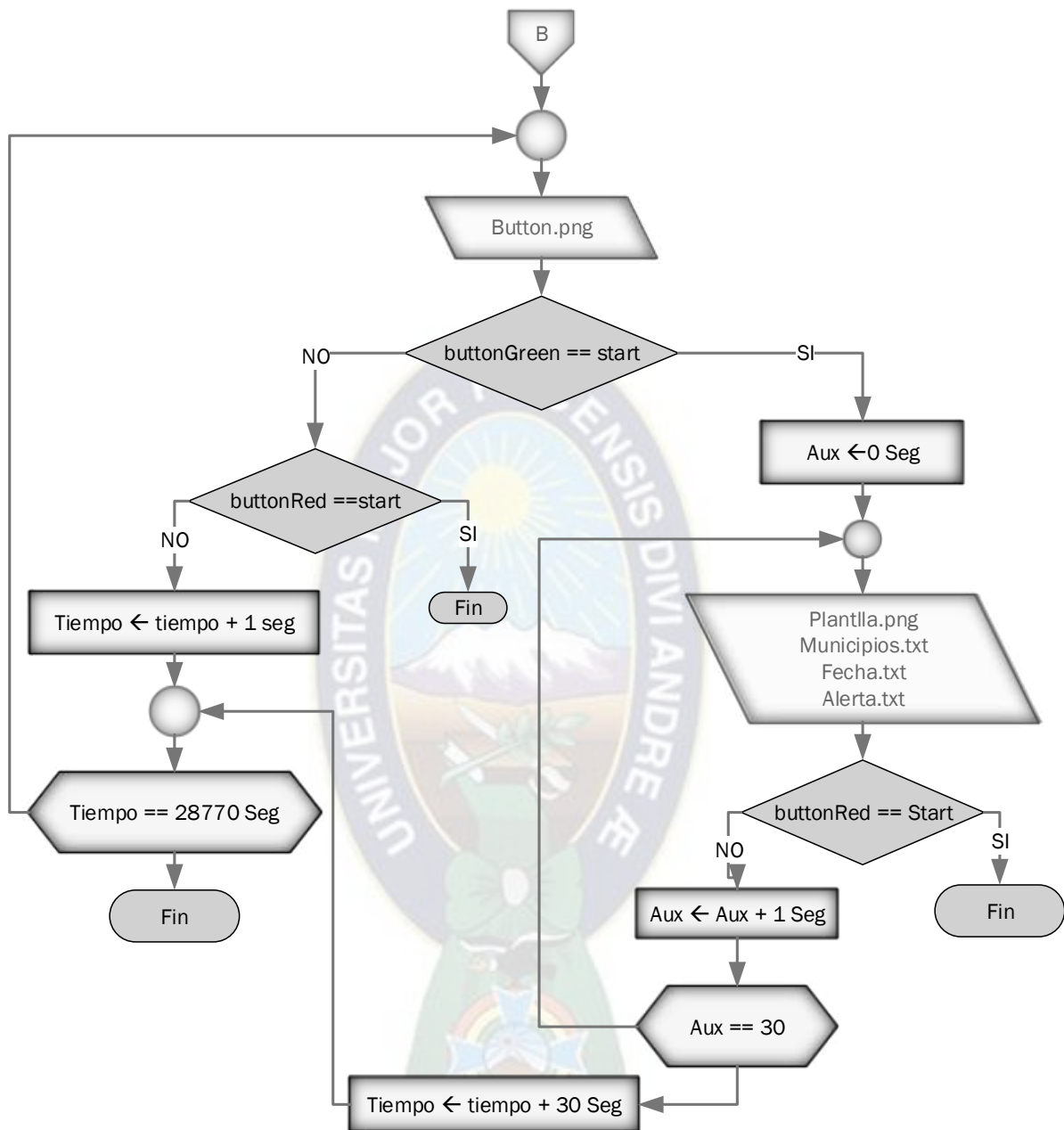


Figura 61. Diagrama de Flujo del código NCL

Fuente: Elaboración Propia

2.4 Modelo de infraestructura de la TDT

Un sistema de TDT tiene los siguientes elementos; en la infraestructura del proveedor: el servidor de aplicaciones y contenidos, y el servidor de playout; y en la infraestructura del usuario: el decodificador o set-top box. Por lo que, el tipo de aplicación interactiva será la que defina la

infraestructura necesaria para su implementación. El sistema de TDT se encuentra representado en la siguiente figura:

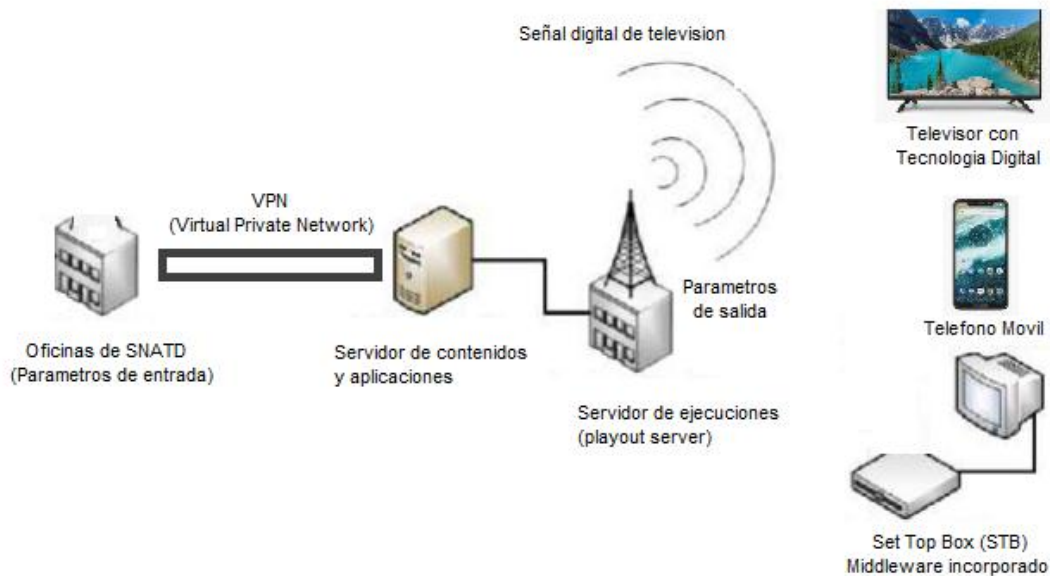


Figura 62. Infraestructura del sistema de alerta temprana

Fuente: Elaboración Propia

2.4.1 Parámetros de entrada

Los parámetros de entrada serán los datos de los boletines que proporciona SNATD, estos datos estarán en formato .txt para que el programa y el servidor pueda reconocerlo y así el prototipo del programa pueda correr a través del Middleware Ginga.

Se enviarán 3 documentos .txt, las que tendrán como nombre:

- Alerta.txt
- Fecha.txt
- Municipios.txt

Los cuales son los parámetros que se quieren mostrar.

2.4.2 El servidor de aplicación y contenidos

Se trata de un dispositivo de software que proporciona servicios de aplicación a las computadoras cliente.

Es un equipo en el cual se guardan las aplicaciones de NCL que se difundirán junto con el contenido audiovisual. Dentro de este dispositivo debe almacenarse adicionalmente, el software para el desarrollo de las aplicaciones interactivas (Dueñas Flores, 2017).

Los servidores de aplicaciones incluyen middleware (o software de conectividad) que les permite intercomunicarse con variados servicios para efectos de confiabilidad, seguridad, no repudio, etc. Los servicios de aplicación también brindan a los desarrolladores una interfaz para programación de aplicaciones (API), de tal manera que no tengan que preocuparse por el sistema operativo o por la gran cantidad de interfaces requeridas en una aplicación web moderna.

2.4.3 El servidor de playout

Con este equipo se realiza la modulación y transmisión de los contenidos de la televisión digital. Es el encargado de crear el flujo de datos de transporte, mediante la utilización del formato MPEG-2.

Mediante el esquema mostrado en la siguiente figura, se describe la funcionalidad del servidor de playout.

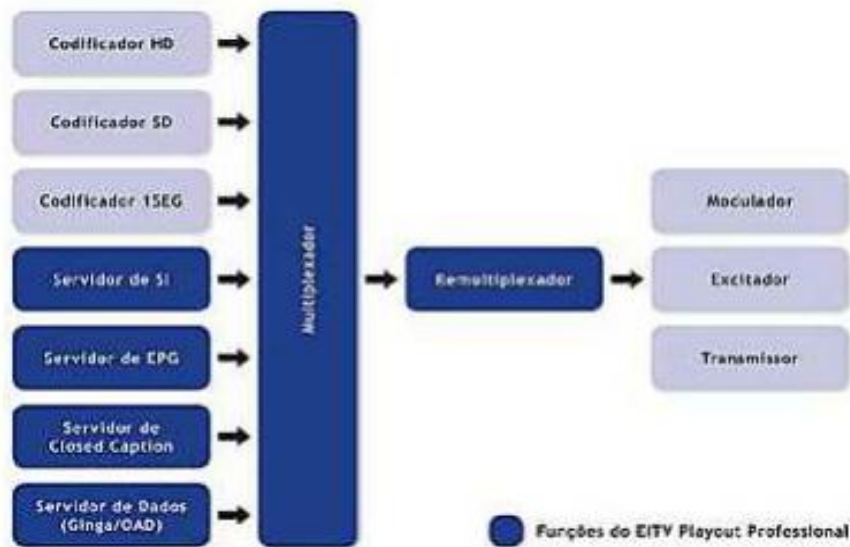


Figura 63. Funcionalidad de un servidor Playout

Fuente: (Dueñas Flores, 2017)

Este aparato tiene la capacidad de juntar la aplicación NCL con el flujo de datos (audio y video).

2.4.4 Parámetros de salida

El parámetro de salida es el código programado NCL, que tendrá el nombre de “ProyectoSinager.ncl”. en el cual estará el contenido de los documentos .txt enviados desde oficinas de SNATD.

2.4.5 Set-top box

Es el equipo encargado de recibir la señal digital mediante la antena del receptor, decodificarla y desplegarla en un televisor; dicho dispositivo se muestra en la siguiente figura. En el caso del estándar ISDB-Tb, tiene precargado el middleware Ginga para la ejecución de aplicaciones interactivas.



Figura 64. Set-Top box Comercial

Fuente: (Dueñas Flores, 2017)

2.5 Realización del prototipo

2.5.1 Creación de un nuevo proyecto NCL en Eclipse

Inicialmente clic en:

- File
- New
 - Project

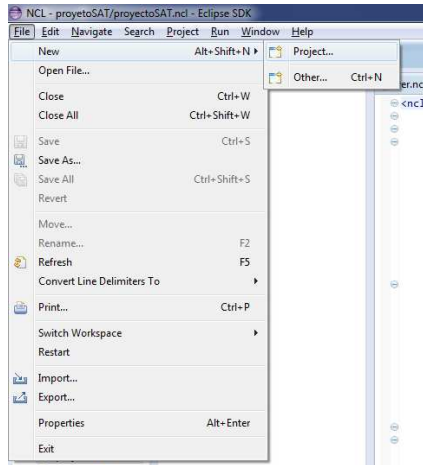


Figura 65. Creación de un nuevo proyecto

Fuente: Elaboración Propia

- NCL Project
- Next

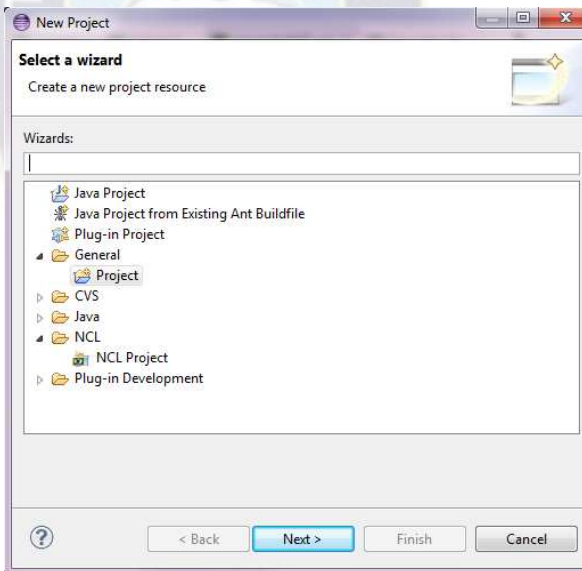


Figura 66. Selección de NCL Project

Fuente: Elaboración Propia

- Project name
- Next

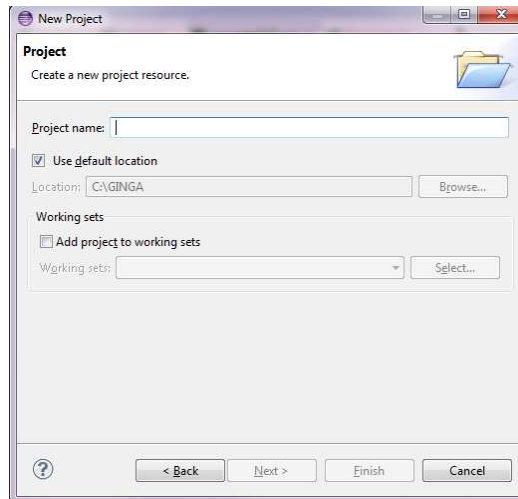


Figura 67. Ingreso del Nombre del proyecto

Fuente: Elaboración Propia

Después de crear el proyecto se procederá a crear una hoja de documento NCL, donde ira el código en el que se ingresaran las instrucciones detalladas anteriormente:

Clic derecho en:

- New NCL Project
- Other

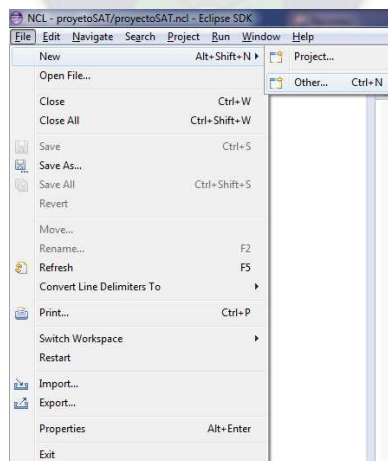


Figura 68. Creación del Proyecto NCL

Fuente: Elaboración Propia

- NCL Document
- Next

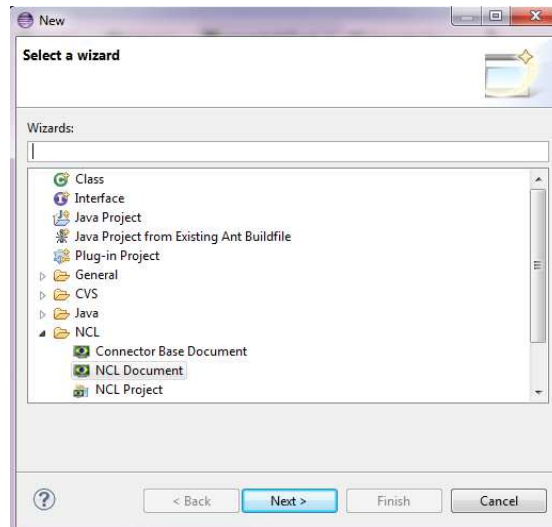


Figura 69. Creación de un Nuevo Documento NCL

Fuente: Elaboración Propia

Para finalizar pondremos el nombre del documento, la dirección de donde se encuentra el directorio proyecto y terminamos la creación del nuevo proyecto.

➤ Finish

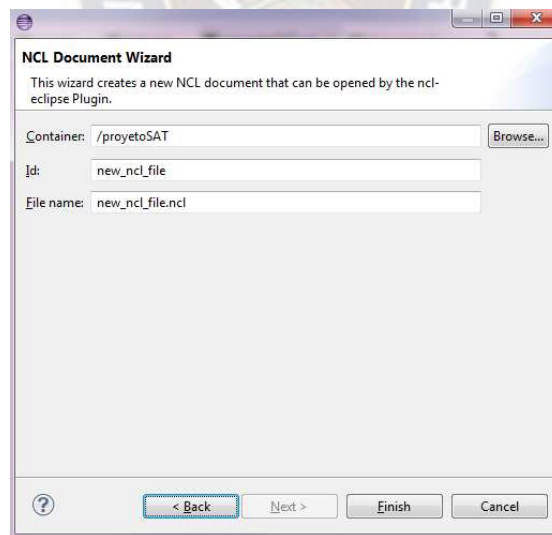


Figura 70. Nombre del documento NCL

Fuente: Elaboración Propia

Para que puedan ser accedidos por el código del programa los diferentes objetos media que presentara el programa NCL deben estar contenidos dentro del proyecto, esto se lo realiza de forma

sencilla, solamente se deben arrastrar de su ubicación de origen hacia la carpeta que lo contendrá para su muestra en el proyecto.

2.5.2 Desarrollo del código del programa

Todos los programas NCL parten de un esqueleto básico, este esqueleto se genera de forma automática mediante el plugin Ginga-NCL que está instalado en el software Eclipse, cuando se crea un documento NCL.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Generated by NCL Eclipse -->
<ncl id="proyectosinager" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>
  </head>
  <body>
  </body>
</ncl>
```

Figura 71. Esqueleto Básico de un Programa NCL

Fuente: Elaboración Propia

Paso 1: Definición de las regiones en la pantalla

Las regiones se definen por el elemento <region> en el encabezado del documento (entre las etiquetas <head> y </head>), en la sección de la base de la región <regionBase> (entre las etiquetas <regionBase> y </regionBase>), como se vio en el diseño, para este prototipo se utilizará 5 regiones ("cero, uno, dos, tres y cuatro").

La región 0 es la región base para las regiones 1,2 y 3, y la región 4 esta oculta bajo la región 0.

<regionBase>

```
<region id="cuatro" left="0%" top="0%" height="15%" width="80%" zIndex="0"/>
```

```
<region id="cero" left="0%" top="0%" height="80%" width="80%" zIndex="1">
```

```
<region id="uno" left="4%" top="39%" height="42.8%" width="18%" zIndex="2" />
```

```
<region id="dos" left="32%" top="50%" height="28.5%" width="52%" zIndex="2"/>
```

```
<region id="tres" left="50%" top="28%" height="15%" width="50%" zIndex="2"/>
```

```
</region>
```

</regionBase>

A continuación, se da detalle de cada una de los atributos utilizados:

id: identificador de la región, este valor que debe ser único, se utilizara cada vez que sea necesario referirse a la región.

left: define la coordenada horizontal izquierda de la región.

top: define la coordenada vertical superior de la región.

height: establece la dimensión vertical de la región.

width: establece la dimensión horizontal de la región.

zIndex: numero entre 0 y 255 que define la superposición de las capas de acuerdo con el valor en este atributo, una región se presentara “arriba” de otras regiones con zIndex menor y “debajo” de otras regiones con zIndex mayor.

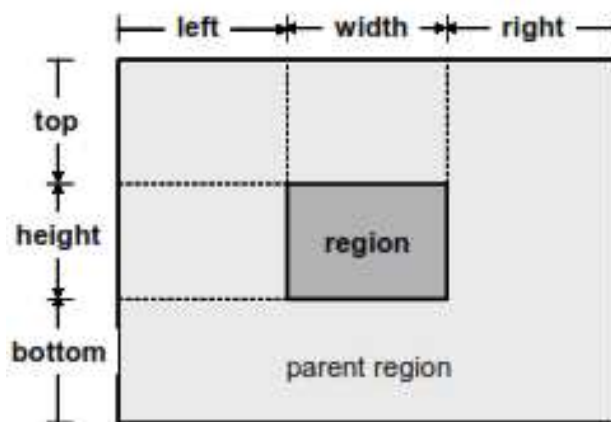


Figura 72. Atributos de posicionamiento de la región

Fuente: (ABNT NBR 15606-2, 2009)

Paso 2: Definición de descriptores que determinan como una media será exhibida

Después de especificar las regiones, se debe definir los descriptores. El descriptor se declara en el encabezado del documento (entre las etiquetas <head> y </head>), en la sección de la base de los descriptores <descriptorBase> (entre las etiquetas <descriptorBase> e </descriptorBase>).

Después de ingresar el descriptor con su respectivo nombre se debe poner el nombre de la región hacia quien va a apuntar dentro del documento, como también cuanto tiempo permanecerá en pantalla cada media.

```

<descriptorBase>
  <descriptor id="descriptor-plantilla.png" explicitDur="30s" region="cero"/>
  <descriptor id="descriptor-alerta.txt" explicitDur="30s" region="tres"/>
  <descriptor id="descriptor-fecha.txt" explicitDur="30s" region="dos"/>
  <descriptor id="descriptor-municipios.txt" explicitDur="30s" region="uno"/>
  <descriptor id="descriptor-button.png" explicitDur="28800s" region="cuatro"/>
</descriptorBase>

```

A continuación, se da detalle de cada una de los atributos utilizados:

id: identificador del descriptor.

región: determina la región a la que el descriptor está relacionado, presentando el nodo en la región correspondiente.

explicitDur: determina la duración de la presentación del elemento asociado con el descriptor, es decir el tiempo de presentación del objeto multimedia.

Paso 3: Definición de los conectores

Para que los enlaces inicien o terminen la presentación de un objeto media, es necesario crear conectores que describan ese comportamiento (iniciar, empezar, finalizar, parar, pausar, etc.). La definición de la base de conectores también se la realiza en el encabezado del documento NCL.

Para este prototipo se crearán tres conectores, una que será para dar comienzo a las medias, el segundo para dar fin a las medias si se pulsa el botón rojo del control remoto y un tercero para dar comienzo si se pulsa el botón verde.

El conector **onBeginStart**, que es el que se utilizara para dar comienzo al prototipo, y el conector **onKeySelection** que es el que se utilizara para acceder a los botones del control remoto.

```

<connectorBase>
  <causalConnector id="onBeginStart">
    <simpleCondition role="onBegin"/>
    <simpleAction role="start" max="unbounded" qualifier="par"/>
  </causalConnector>

```



```

<causalConnector id="onKeySelectionStop">
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
    <simpleAction role="stop" max="unbounded" qualifier="par"/>
</causalConnector>

<causalConnector id="onKeySelectionStart">
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
    <simpleAction role="start" max="unbounded" qualifier="par"/>
</causalConnector>
</connectorBase>

```

A continuación, se da detalle de cada una de los módulos y atributos utilizados:

<simpleCondition> define las condiciones simples de activación de un enlace que utiliza un conector y role es el que establece la función.

En este caso es la condición onBegin que se activa cuando la presentación de los elementos vinculados a esta función es iniciada, y la condición onSelection la cual nos indica que tenemos que ingresar un evento para que cumpla la condición, en este caso presionar los botones del control remoto.

<simpleAction> define las acciones simples que se realizaran cuando un enlace que utiliza un conector sea activado.

Role: es el que establece el nombre de la función de acción, para esta aplicación se utilizara “start” la cual inicia la presentación del elemento vinculado a esta función, y “stop” la cual finaliza la presentación del elemento vinculado a esta función.

Max: el número máximo de elementos para poder utilizar este documento, el valor “unbounded” especifica un número máximo ilimitado.

Qualifier: establece si la acción será ejecutada en paralelo o secuencialmente, en este caso se elegirá “par”

<connectorParam> descrito para definir parámetros y se utiliza para que el valor sea validado o establecido y puede estar indicado en el momento de uso del conector.

Paso 4: Definición del puerto

Para la sección <body> se comenzará insertando el puerto el cual es la puerta para el comienzo de las demás medias.

```
<port id="Port-plantilla.png" component="plantilla.png"/>
```

Donde:

id: identificador único que permite que esta sea referencia por otro elemento

Component: el componente que se asigna a través de la puerta, y que es el que será ejecutado por medio de la misma.

Paso 5: Definición de las medias

En esta parte se crea el nodo media que compone el documento, esta definición del nodo se la realiza en el body del documento NCL.

<media> en esta región principalmente se agrega la dirección de la carpeta raíz de los archivos multimedia además de la id del archivo que posteriormente se utilizará.

```
<media id="plantilla.png" src="plantilla.png" descriptor="descriptor-plantilla.png">
```

```
</media>
```

```
<media id="alerta.txt" src="alerta.txt" descriptor="descriptor-alerta.txt">
```

```
  <property name="fontSize" value="17" />
```

```
  <property name="fontColor" value="red"/>
```

```
</media>
```

```
<media id="fecha.txt" src="fecha.txt" descriptor="descriptor-fecha.txt">
```

```
  <property name="fontSize" value="17" />
```

```
  <property name="fontColor" value="blue"/>
```

```
</media>
```

```
<media id="municipios.txt" src="municipios.txt" descriptor="descriptor-municipios.txt">
```

```
  <property name="fontSize" value="17" />
```

```
  <property name="fontColor" value="blue"/>
```

```
</media>
```

```
<media id="button.png" src="button.png" descriptor="descriptor-button.png">
```

```
</media>
```

Donde :

id: identificador único del nodo multimedia.

Src: fuente del objeto multimedia. Se trata del camino para el archivo multimedia.

Descriptor: establece el descriptor que controla la presentación de objeto multimedia. El valor de este atributo deberá ser el identificador del descriptor.

Property: establece las propiedades que se le puede dar a la media

Name: es el nombre de la propiedad, como ser: "fontSize", "fontColor", "fontFamily", "background" que son para darle tamaño, color, estilo y fondo al contenido de la media.

Value: es el valor de que se le asigna al nombre de propiedad, en este caso número, color y tipo de la letra.

Paso 6: Definición de los enlaces de sincronización, para iniciar y terminar la exhibición de las medias

Finalmente habilitamos cada una de las acciones que se desean realizar con los botones de interactividad:

Los archivos multimedia se detendrán al presionar el botón rojo del control remoto o después de haber transcurrido 8 horas.

En el primer link se especificara la inicialización de la aplicación, llamando al connector "onBeginStart".

```
<link id="LinkInicializacion" xconnector="onBeginStart">
  <bind role="onBegin" component="plantilla.png">
  </bind>
  <bind role="start" component="button.png">
  </bind>
  <bind role="start" component="alerta.txt">
  </bind>
  <bind role="start" component="fecha.txt">
  </bind>
  <bind role="start" component="municipios.txt">
  </bind>
</link>
```

En el segundo Link se especificara la culminación de la aplicación, llamando al conector “onKeySelectionStop”, en caso de presionar el botón rojo del control remoto.

```
<link id="LinkFinalizacion" xconnector="onKeySelectionStop">
  <bind role="onSelection" component="button.png">
    <bindParam name="keyCode" value="RED"/>
  </bind>
  <bind role="stop" component="plantilla.png">
  </bind>
  <bind role="stop" component="alerta.txt">
  </bind>
  <bind role="stop" component="fecha.txt">
  </bind>
  <bind role="stop" component="municipios.txt">
  </bind>
  <bind role="stop" component="button.png">
  </bind>
</link>
```

Y por ultimo el tercer Link se especificara la inicialización de la aplicación llamando al conector “onKeySelectionStart” en caso de presionar el botón verde del control remoto.

```
<link id="LinkIniciacion" xconnector="onKeySelectionStart">
  <bind role="onSelection" component="button.png">
    <bindParam name="keyCode" value="GREEN"/>
  </bind>
  <bind role="start" component="plantilla.png">
  </bind>
  <bind role="start" component="alerta.txt">
  </bind>
  <bind role="start" component="fecha.txt">
```

```

</bind>
<bind role="start" component="municipios.txt">
</bind>
</link>

```

Donde:

<link> responsable por la definición de los eslabones, que utilizan conectores. Donde: **xconnector** son los conectores ya creados en la parte <head>.

<bind> es un elemento hijo de link que permite asociar nudos a papeles (roles) del conector.

Donde:

Role: se utiliza para hacer referencia a un papel del conector, para nuestro código utilizaremos “start” para dar inicio y “stop” para finalizar.

Component: se utiliza para identificar la media.

<bindParam> define valores para los parámetros del conector.

Name: nombre de un parámetro del conector.

Value: valor a ser atribuido al respectivo parámetro, en este caso RED y GREEN, que son para acceder a los botones del control remoto.

2.6 Resultados

Para que se pueda visualizar el prototipo es necesario añadir a Eclipse el software Ginga4Windows, y se lo realiza de la siguiente manera:

- Run
- External Tools
- External Tool Configuration

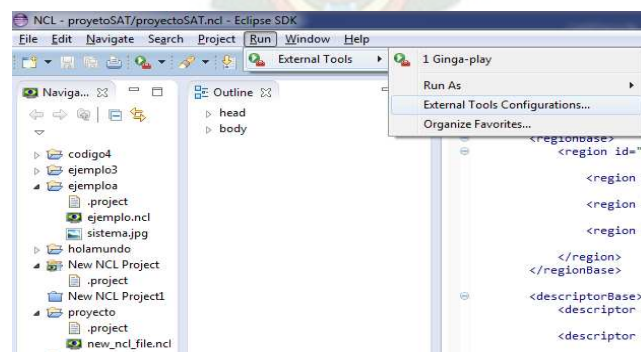


Figura 73. Adhiriendo Ginga4Windows a Eclipse

Fuente: Elaboración Propia

Ir hasta “New Configuration” e ingresar la dirección de Ginga4Windows

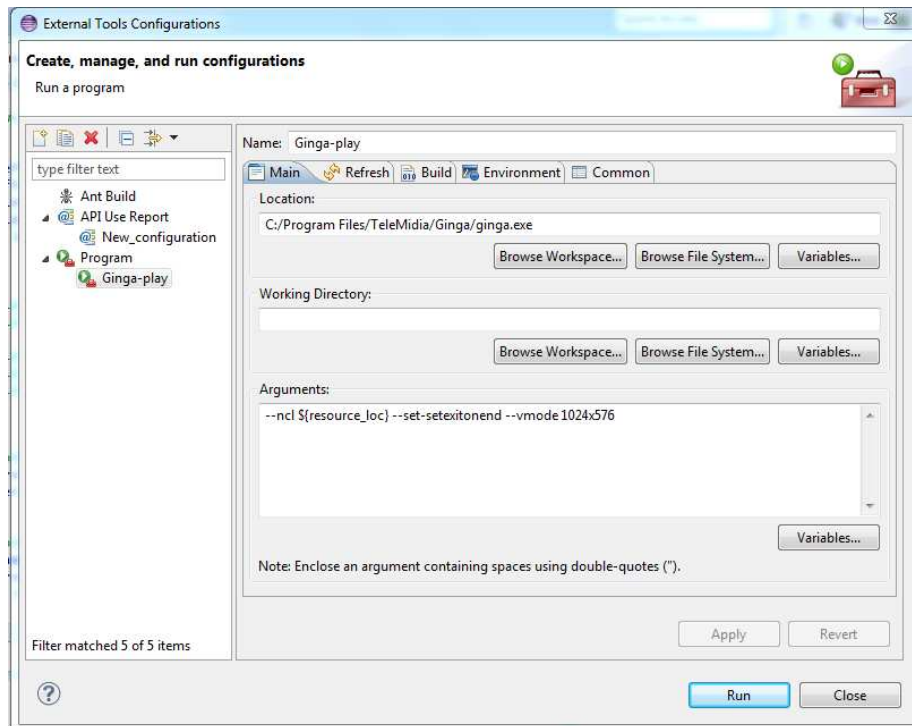


Figura 74. Ingresando la dirección de Ginga4Windows en Eclipse

Fuente: Elaboración Propia

y por ultimo para demostración del prototipo le damos play al icono Ginga-play.



Figura 75. Ejecución de la Aplicación con Ginga-Play

Fuente: Elaboración Propia

A continuación se podrá visualizar un ejemplo de los mensaje que se pretende difundir.

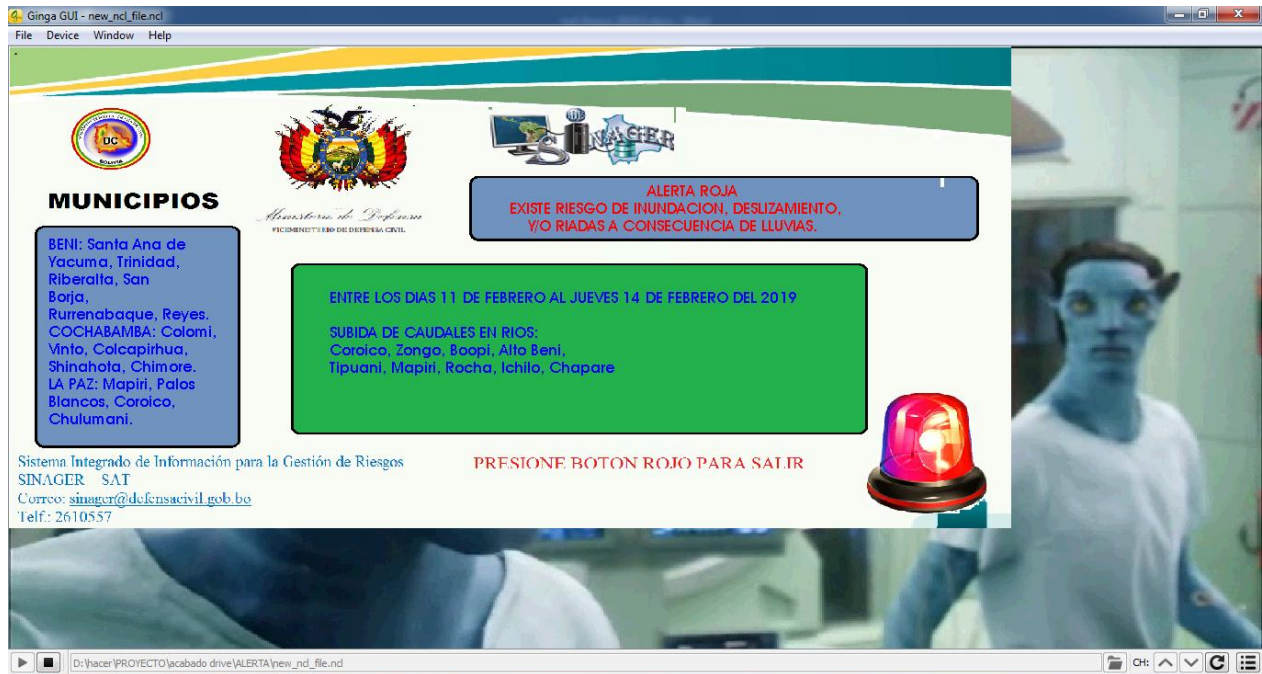


Figura 76. Visualización de la imagen principal del prototipo

Fuente: Elaboración Propia



Figura 77. Visualización de la imagen secundaria del prototipo

Fuente: Elaboración Propia

CAPITULO VI: COSTOS

1 Análisis de costos para el diseño de la red VPN

Para el análisis de costos se toma en cuenta los requerimientos de software y recurso humano que deben adquirirse para la implementación de la VPN.

En el análisis de costos de la VPN, no se considera el requerimiento de equipos que hagan de servidor y cliente, ni las conexiones a Internet, tomando en cuenta que ambas instituciones poseen estos elementos.

En los siguientes cuadros se presenta un presupuesto de los gastos que se deben realizar para la implementación de la VPN.

1.1 Costos de software

La siguiente tabla es considerada si las oficinas de SNATD y Bolivia Tv necesitaran de licencia de Windows, en cualquier otro caso que tuviesen su propio sistema operativo de paga o gratuito, ya no sería necesaria. Es importante recordar que OpenVPN es una herramienta multiplataforma, soportadas en sistemas operativos como Linux, Windows, OpenBSD, FreeBSD, NetBSD, Mac OS X y Solaris.

Tabla 21. Tabla de Costos de Software

Fuente: Elaboración Propia

SISTEMA OPERATIVO	LICENCIAS	COSTO UNITARIO (\$)	COSTO TOTAL (\$)
Windows 10 Pro	2	289	578

1.2 Costos del recurso humano

Tabla 22. Tabla de Costos Recurso Humano

Fuente: Elaboración Propia

RECURSO HUMANO	CANTIDAD	TIEMPO (meses)	COSTO/MENSUAL (\$)	COSTO TOTAL (\$)
Ingeniero en Sistemas	1	1	900	900

2 Análisis de costos para la elaboración del prototipo de sistema de alerta temprana

Considerando que los softwares utilizados son de licencia libre, solo habría que mencionar el costo de recurso humano.

Tabla 23. Tabla de Costos para elaboración del programa NCL

Fuente: Elaboración Propia

DETALLE	CANTIDAD	TIEMPO (meses)	COSTO/MENSUAL (\$)	COSTO TOTAL (\$)
Ingeniero en Sistemas	1	1	900	900

3 Costos totales para la hipotética implementación del proyecto

Tabla 24. Tabla de Costo Total

Fuente: Elaboración Propia

DESCRIPCION	PRECIO TOTAL (\$)
Costo de Software Windows 10 Pro	578
Costo de Recurso Humano (VPN)	900
Costo de Recurso Humano (Programa-NCL)	900
TOTAL	2378

CAPITULO VII: CONCLUSIONES Y RECOMENDACIONES

1 Conclusiones

Para la realización del prototipo se recolecto datos relevantes de antiguos boletines que envía SNATD. Se generó un código de programación sencillo y flexible, que permitirá visualizar los parámetros característicos de cada evento de desastre del sistema SNATD, entre los cuales son: tipo de desastre, nivel de alerta, fechas a ocurrir, lugares donde ocurrirá.

La realización del diseño de este prototipo que optimizará el sistema de alerta temprana para la prevención de desastres naturales de SINAGER-SNATD utilizando el sistema middleware Ginga-NCL de la norma ISDB-Tb, podrá ser incorporada a las redes existentes de televisión, y podrá ser visualizada en receptores fijos y móviles.

Este prototipo permitiría que las autoridades municipales y pobladores puedan tomar decisiones frente a la probabilidad de un evento de desastre, lo cual en muchos casos permitirá salvar vidas, proteger a la ganadería y proteger a la infraestructura productiva. Asimismo, le permitirá coordinar acciones de respuesta de ser necesarias.

Se resaltó la importancia del estándar de televisión digital terrestre ISDB-Tb para la implementación de sistemas de alerta temprana mediante la utilización del middleware Ginga, mencionando también la opción de envío de señales EWBS característico de este estándar.

Se creó y configuro los archivos del software OpenVPN para que a futuro se copien dichos archivos al servidor y cliente, y así poder comunicar la oficina de SNATD con la estación de televisión.

Una conexión por Internet es mucho más barata que una conexión permanente a través de una línea alquilada. Por lo general, las líneas alquiladas tienen un costo fijo de conexión y otro costo que depende de la distancia; sin embargo, con este tipo de líneas, a medida que aumenta el tamaño de la red, los costes se disparan. Por el contrario, el coste de las conexiones por Internet depende sólo de la capacidad de la línea, y no de la distancia por lo que los costes de explotación de cada nodo de VPN se reducen considerablemente.

2 Recomendaciones

Debido que el estándar brasileño ISDB-Tb está basado en el desarrollo de contenido interactivo sobre software libre para la simulación de las aplicaciones al igual que para el desarrollo, siempre se debe trabajar sobre las últimas versiones que se publican en las páginas web oficiales de Ginga, principalmente las de Brasil, ya que las versiones de Chile, Argentina, y Perú están basadas sobre el mismo y solamente poseen modificaciones menores, para cada país.

Aunque nuestro país todavía está muy atrasado respecto al tema de TVD en Latinoamérica respecto a los otros países que también adoptaron esta norma, es de mucha ayuda el poder acceder a las páginas oficiales de Ginga de cada país, ya que se puede interactuar con otros desarrolladores que en ocasiones con sus experiencias pueden ayudar en ciertos problemas que se presentan al introducirse en el tema, además se puede compartir y ayudar a otros a solucionarlos también, cumpliendo con el objetivo principal de Ginga que pretende a través del código libre la evolución constante y la mejora del estándar, realizada no solo por los creadores del estándar sino también por personas particulares interesadas en el tema.

Para el diseño del prototipo es necesario tomar en cuenta que la presentación final es de modo visual, por lo que es aconsejable que el prototipo tenga una imagen llamativa que contenga información clara.

GLOSARIO

AAC:	Advanced Audio Coding - Codificación Avanzada de Audio.
ABNT:	Asociación Brasileña de Normas Técnicas.
AES:	Advanced Encryption Standard - Estándar de Cifrado Avanzado.
Apagón Analógico:	Finalización de las transmisiones de televisión analógica.
API:	Application Programming Interface - Interfaz de programación de aplicaciones.
Aplicación declarativa:	Aplicación que utiliza principalmente, y como punto de partida, información declarativa para expresar su comportamiento.
Aplicación procedural:	Aplicación que utiliza principalmente, y como punto de partida, informaciones procedurales para expresar su comportamiento
Aplicación interactiva:	Programa adicional al contenido televisivo que pueden ejecutarse en un set-top box.
ARIB:	Association of Radio Industries and Businesses – Asociación de Industrias y Empresas de Radio.
Atributo:	parámetro para representar la naturaleza de una propiedad.
ATSC:	Advanced Television Systems Committee - Comité de Sistemas de Televisión Avanzada.
CA:	Certificate Authority - Autoridad Certificadora.
CBC:	Cipher Block Chaining - Cifrado por Bloques en Cadena.
Conexión Remota:	Operación realizada en una computadora remota a través de una red de computadoras, como si se tratase de una conexión local.
DES:	Estándar de Cifrado de Datos. Algoritmo de cifrado de datos el cual utiliza bloques de datos de 64 bits y una clave de 56 bits.
DH:	Diffie-Hellman, es un algoritmo de establecimiento de claves entre partes que no han tenido contacto previo
DMB-T:	Digital Multimedia Broadcast-Terrestrial - Radiodifusión multimedia digital-terrestre.
DVB:	Digital Video Broadcasting - Difusión Digital de Video Terrestre.
Encriptación/ Cifrado:	Tratamiento de un conjunto de datos, contenidos o no en un paquete, a fin

de impedir que nadie excepto el destinatario de los mismos pueda leerlos.

- ES:** Elementary Stream - Flujo Elemental.
- Ethernet:** Es el nombre de una tecnología de redes de computadoras de área local (LANs) basada en tramas de datos.
- EWBS:** Emergency Warning Broadcasting System - Sistema de Transmisión de Alerta de Emergencia.
- Ginga:** Middleware abierto del sistema nipo-brasileño de televisión digital.
- Ginga-NCL:** Subsistema lógico de Ginga que interpreta archivos NCL y permite la presentación de aplicaciones interactivas.
- GPL:** Licencia Pública General de GNU – GNU General Public License.
- HDTV:** High Denition Televisión - Televisión de Alta Definición.
- HMAC:** Hash Message Authentication Code - Código de Autenticación de Mensaje Basados en Resúmenes.
- HTML:** HyperText Markup Language - Lenguaje de marcado de hipertexto.
- IANA:** Internet Assigned Numbers Authority - Agencia de Asignación de Números de Internet.
- IDE:** Integrated Development Environment - Ambiente de Desarrollo Integrado.
- Interfaz:** Zona de comunicación o acción de un sistema sobre otro.
- IP:** Internet Protocol, es un protocolo utilizado para enviar datos a través de una red.
- ISDB-T:** Terrestrial Integrated Services Digital Broadcasting - Radiodifusión Digital de servicios integrados terrestres.
- Java:** Lenguaje de programación orientado a objetos.
- JVM:** Máquina virtual de Java.
- LAN:** Local Area Network. Red de área local. Red de computadoras personales ubicadas dentro de un área geográfica limitada que se compone de servidores, estaciones de trabajo, sistemas operativos de redes y un enlace encargado de distribuir las comunicaciones.
- LUA:** Lenguaje de programación imperativo, estructurado y bastante ligero, que fue diseñado como un lenguaje interpretado con una semántica

	extensible.
LZO:	Lempel Ziv Oberhumer - Algoritmo de Compresión de Datos.
MD5:	Message Digest algorithm 5 - Algoritmo de Resumen de Mensaje 5.
Middleware:	Software que vincula aplicaciones de diferentes tipos de software.
MPEG:	Moving Picture Experts Group - Grupo de Expertos de Imágenes en Movimiento.
NCL:	Nested Context Language variante de programación en Ginga. Es un lenguaje de aplicación XML que proporciona soporte para la especificación de sincronización espacio-temporal de los objetos multimedia.
NCM:	Nested Context Model – Modelo de contexto anidado.
PES:	Packetized Elementary Stream - Flujo Elemental Paquetizado.
PMT:	Program Map Table - Tabla del Mapa del Programa.
Protocolo:	Descripción formal de formatos de mensaje y de reglas que dos computadoras deben seguir para intercambiar dichos mensajes.
PS:	Program Stream - Flujo de Programa.
RC4:	River Cipher 4 stream cipher Algorithm - Algoritmo de Cifrado en Flujo Rivest Cipher 4.
Script:	Archivo de procesamiento por lotes.
SDTV:	Standard definition television - Televisión de definición estándar.
Senamhi:	Servicio Nacional de Meteorología e Hidrología.
Set-top box:	Equipo que decodifica la señal de la antena del receptor y la muestra en la televisión.
SHA:	Secure Hash Algorithm - Algoritmo de Hash Seguro.
Sinager:	Sistema Nacional de Gestión de Riesgos.
Snatd:	Sistema Nacional de Alerta Temprana de Desastres.
Software:	Conjunto de programas que permiten a una computadora realizar una determinada. tarea.
SSL:	Seguridad de la Capa de Transporte – Secure Socket Layer.
STB:	Set-Top-Box.
TCP:	Transmission Control Protocol - Protocolo de Control de Transmisión.

- TDT:** Televisión Digital Terrestre.
- TLS:** Transport Layer Security - Seguridad de la Capa de Transporte.
- TMCC;** Transmission and Multiplexing Configuration Control - Control de Configuración de Transmisión y Multiplexado.
- UDP:** User Datagram protocol - Protocolo de Capa de Transporte Basado en Datagramas.
- URI:** Uniform resource identifier - Identificador uniforme de recursos.
- VPN:** Red Privada Virtual – Virtual Private Network.
- XML:** Extensible Markup Language - Lenguaje de marcado extensible.
- 100Base-TX:** Es la forma predominante de Fast Ethernet a 100Mbit/s. Utiliza cables de cat5 con dos pares de hilos.



REFERENCIAS

- ABNT NBR 15606-2. (2009). *Televisión digital terrestre — Codificación de datos y especificaciones de transmisión para radiodifusión digital. Parte 2: Ginga-NCL para receptores fijos y móviles – Lenguaje de aplicación XML para codificación de aplicaciones*. Asociación Brasileira de Normas Técnicas.
- Asamblea Legislativa Plurinacional . (2014). *Ley de Gestion de Riesgos*.
- Asociación Brasileña de Normas Técnicas. (2007). *ABNT NBR 15606-2. Televisión Digital Terrestre - Sistema de Transmisión*. Brasil .
- Autoridad de Regulación y Fiscalización de Telecomunicaciones y transporte. (s.f.). *Marco Legal* . Obtenido de <https://www.att.gob.bo/>
- Barba Cherez, D. J. (2018). *INFLUENCIA DE UN SISTEMA DE ALERTA TEMPRANA ANTE ERUPCIONES VOLCÁNICAS Y SISMOS COMPATIBLE CON EL ESTÁNDAR ISDB-Tb EN LA DISMINUCIÓN DE DAÑOS EN LA PROVINCIA DE TUNGURAHUA*. (Proyectos de Investigación y Desarrollo en Magister en Sistemas de Telecomunicaciones). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO, Riobamba-Ecuador.
- Bolivia. Consejo de Ministros . (2019). *Decreto Supremo N° 3896: Modifica el Plan de Implementación de Television Digital Terrestre, aprobado en el anexo del Decreto Supremo N° 3152*. Gaceta Oficial .
- Cubero, M. (2009). *La televisión Digital. Fundamentos y Teorias* . Mexico, Mexico: Alfaomega Grupo Editor S.A.
- DiBEG. (junio de 2008). *Digital Broadcasting Expert Group*. Obtenido de Feature of ISDB-T system: http://www.dibeg.org/techp/feature_of_isdb-t.html
- Dueñas Flores, A. S. (2017). *Ginga Studio, Software para la Creación y Edición de Aplicaciones Interactivas Ginga-NCL de Televisión Digital Mediante una Interfaz Grafica de Usuario*. (Trabajo de Titulación Previo a la Obtencion del Título de Ingeniero en Electrónica) Universidad de las Fuerzas Armadas, Carrera de Ingenieria en Electrónica y Telecomunicaciones, Sangolqui .
- Espin Lucas, A. P. (2014). *Análisis de un Sistema de Transmisión de Alertas contra Desastres Naturales mediante TDT*. (Trabajo de Titulación previo a la Obtención del Título de: INGENIERO EN TELECOMUNICACIONES CON MENCIÓN EN GESTIÓN

- EMPRESARIAL EN TELECOMUNICACIONES). Universidad Católica de Santiago, Facultad de Educación Técnica para el Desarrollo, Guayaquil.
- Foro Internacional ISDB-T. (2013). *ISDB-T Documento de Armonización, Parte 3 Sistemas de Alerta de Emergencias EWBS*.
- Frenzel, L. (2003). *Electronica Aplicada a los Sistemas de Comunicaciones*. Mexico: Alfaomega Grupo Editor.
- Galabay Teolongo, P. T., & Vivar Espinoza, F. R. (2012). *Manejo del software Ginga para el desarrollo de aplicaciones interactivas para televisión digital, basado en el estándar Brasileño ISDB-Tb*. (Trabajo de grado previo a la obtención del Título de Ingeniero Electrónico). UNIVERSIDAD POLITÉCNICA SALESIANA, CARRERA DE INGENIERIA ELECTRONICA, Cuenca.
- Los Tiempos . (6 de Abril de 2017). *Bolivia, muy vulnerable al cambio climático*. Obtenido de Los Tiempos: <https://www.lostiempos.com/tendencias/medio-ambiente/20170406/bolivia-muy-vulnerable-al-cambio-climatico>
- Mariaca, C., Trujillo, F., Rossi, L., & Mendoza, O. (2013). *Implementación de un sistema integrado en tiempo real para la proyección, monitoreo y alerta temprana de riesgos de desastres naturales en Bolivia*. FAO Bolivia.
- Ministerio de Defensa. (2019). *Boletín de Riesgo Nacional N° 11/19*. La Paz: Sinager-SAT.
- Ñacato Gualotuña, M. A. (2007). *Diseño e implementación de una Red Privada Virtual (VPN) para la Empresa Hato Telecomunicaciones*. (Proyecto previo a la obtención del título de Tecnólogo en análisis de sistemas informáticos). Escuela Politécnica Nacional, Quito.
- Ochoa Domínguez, H. d., Mireles García, J., & Cota Ruíz, J. (2007). *Descripción del nuevo estándar de video H.264 y comparación de su eficiencia de codificación con otros estándares*. Obtenido de Recuperado el Septiembre de 2017, de SciELO: http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-77432007000300004&lng=es&tlng=es.
- OpenVPN. (s.f.). *OpenVPN*. Obtenido de <https://openvpn.net/>
- Pisciotta, N., Liendo, C., & Lauro, R. (2013). *Transmisión de Televisión Digital Terrestre en la norma ISDB-Tb*. Buenos Aires, Argentina: Cengage Learning.
- Rodríguez Ortega, O. D. (2017). *MIDDLEWARE GINGA EN EL DESARROLLO DE APLICACIONES INTERACTIVAS PARA LA TELEVISIÓN DIGITAL TERRESTRE*.

APLICATIVO: PROTOTIPO DE UN PORTAL INTERACTIVO PARA LA COMPRA Y VENTA DE PRODUCTOS POR TELEVISIÓN. . (Trabajo de Grado previo a la obtención del título de Ingeniero en Sistemas Computacionales). Universidad Técnica del Norte, Facultad de Ingeniería en Ciencias Aplicadas, Ibarra.

Senamhi. (s.f.). *Estaciones Meteorológicas (Ubicación)*. Obtenido de <http://www.senamhi.gob.bo/mapaestaciones/index.php>

Sitio Oficial de Middleware Ginga. (2007). *Apertura*. Recuperado el octubre de 2019. Obtenido de Tv interactiva Ginga : <http://www.ginga.org.br/>

Tomás Cánovas, J. J. (2008). *Servicio VPN de Acceso remoto basado en SSL mediante OpenVPN*. (Proyecto Fin de Carrera Ingeniería Técnica de Telecomunicación). Universidad Politécnica de Cartagena, Departamento de Tecnología de la información y las comunicaciones, Cartagena .

Vice-ministerio de Defensa Civil. (2016). *Guía de Preparación para la Atención de Desastres y/o Emergencias*. La Paz.

Vice-ministerio Defensa Civil. (s.f.). Obtenido de Vice-ministerio Defensa Civil: <http://www.defensacivil.gob.bo>

Vice-ministerio Defensa Civil. (s.f.). *¿ Que es SINAGER - SAT ?* Obtenido de <http://www.defensacivil.gob.bo/web/pagina/que-es-sinager-sat.html>

ANEXO A: CÓDIGO DEL PROGRAMA

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Generated by NCL Eclipse -->
<ncl id="alerta" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>

    <regionBase>
      <region id="cuatro" left="0%" top="0%" height="15%" width="80%" zIndex="0"/>
<region id="cero" left="0%" top="0%" height="80%" width="80%" zIndex="1">
<region id="uno" left="4%" top="39%" height="43%" width="18%" zIndex="2" />
      <region id="dos" left="32%" top="50%" height="29%" width="52%" zIndex="2"/>
      <region id="tres" left="50%" top="28%" height="15%" width="50%" zIndex="2"/>
    </region>
  </regionBase>

  <descriptorBase>
    <descriptor id="descriptor-plantilla.png" explicitDur="30s" region="cero"/>
    <descriptor id="descriptor-alerta.txt" explicitDur="30s" region="tres"/>
    <descriptor id="descriptor-fecha.txt" explicitDur="30s" region="dos"/>
    <descriptor id="descriptor-municipios.txt" explicitDur="30s" region="uno"/>
  <descriptor id="descriptor-button.png" explicitDur="2880s" region="cuatro"/>
  </descriptorBase>

  <connectorBase>

    <causalConnector id="onBeginStart">
      <simpleCondition role="onBegin"/>
      <simpleAction role="start" max="unbounded" qualifier="par"/>
    </causalConnector>

    <causalConnector id="onKeySelectionStop">
      <connectorParam name="keyCode"/>
      <simpleCondition role="onSelection" key="RED"/>
      <simpleAction role="stop" max="unbounded" qualifier="par"/>
    </causalConnector>

    <causalConnector id="onKeySelectionStart">
      <connectorParam name="keyCode"/>
      <simpleCondition role="onSelection" key="GREEN"/>
      <simpleAction role="start" max="unbounded" qualifier="par"/>
    </causalConnector>

  </connectorBase>
</head>

<body >
  <port id="Port-plantilla.png" component="plantilla.png"/>
  <media id="plantilla.png" src="plantilla.png" descriptor="descriptor-
  plantilla.png">
  </media>
```

```

<media id="alerta.txt" src="alerta.txt" descriptor="descriptor-
alerta.txt">
    <property name="fontSize" value="17" />
    <property name="fontColor" value="red"/>
    <property name="fontFamily" value="calibri"/>
</media>

<media id="fecha.txt" src="fecha.txt" descriptor="descriptor-fecha.txt">
    <property name="fontSize" value="17" />
    <property name="fontColor" value="blue"/>
    <property name="fontColor" value="calibri"/>
</media>

<media id="municipios.txt" src="municipios.txt" descriptor="descriptor-
municipios.txt">
    <property name="fontSize" value="17" />
    <property name="fontColor" value="blue"/>
    <property name="fontFamily" value="calibri"/>
</media>

<media id="button.png" src="button.png" descriptor="descriptor-
button.png">
</media>

<link id="LinkInicializacion" xconnector="onBeginStart">
    <bind role="onBegin" component="plantilla.png">
    </bind>
    <bind role="start" component="button.png">
    </bind>
    <bind role="start" component="alerta.txt">
    </bind>
    <bind role="start" component="fecha.txt">
    </bind>
    <bind role="start" component="municipios.txt">
    </bind>
</link>

<link id="LinkFinalizacion" xconnector="onKeySelectionStop">
    <bind role="onSelection" component="button.png">
        <bindParam name="keyCode" value="RED"/>
    </bind>
    <bind role="stop" component="plantilla.png">
    </bind>
    <bind role="stop" component="alerta.txt">
    </bind>
    <bind role="stop" component="fecha.txt">
    </bind>
    <bind role="stop" component="municipios.txt">
    </bind>
    <bind role="stop" component="button.png">
    </bind>
</link>

<link id="LinkIniciacion" xconnector="onKeySelectionStart">
    <bind role="onSelection" component="button.png">

```

```

        <bindParam name="keyCode" value="GREEN"/>
    </bind>
    <bind role="start" component="plantilla.png">
    </bind>
    <bind role="start" component="alerta.txt">
    </bind>
    <bind role="start" component="fecha.txt">
    </bind>
    <bind role="start" component="municipios.txt">
    </bind>
</link>
</body>
</ncl>

```

ANEXO B: BASE DE CONECTORES

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<! Generated by NCL Eclipse >
<ncl id="connectorBase" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProle">
<head>
<connectorBase>
    <!-- OnBegin -->
    <causalConnector id="onBeginStart">
        <simpleCondition role="onBegin"/>
        <simpleAction role="start"/>
    </causalConnector>
    <causalConnector id="onBeginStop">
        <simpleCondition role="onBegin"/>
        <simpleAction role="stop"/>
    </causalConnector>
    <causalConnector id="onBeginPause">
        <simpleCondition role="onBegin"/>
        <simpleAction role="pause"/>
    </causalConnector>
    <causalConnector id="onBeginResume">
        <simpleCondition role="onBegin"/>
        <simpleAction role="resume"/>
    </causalConnector>

```

```
</causalConnector>
<causalConnector id="onBeginSet">
  <connectorParam name="var"/>
  <simpleCondition role="onBegin"/>
<simpleAction role="set" value="$var"/>
</causalConnector>
```

<! OnEnd >

```
<causalConnector id="onEndStart">
  <simpleCondition role="onEnd"/>
  <simpleAction role="start"/>
</causalConnector>
<causalConnector id="onEndStop">
  <simpleCondition role="onEnd"/>
  <simpleAction role="stop"/>
</causalConnector>
<causalConnector id="onEndPause">
  <simpleCondition role="onEnd"/>
  <simpleAction role="pause"/>
</causalConnector>
<causalConnector id="onEndResume">
  <simpleCondition role="onEnd"/>
  <simpleAction role="resume"/>
</causalConnector>
<causalConnector id="onEndSet">
  <connectorParam name="var"/>
  <simpleCondition role="onEnd"/>
  <simpleAction role="set" value="$var"/>
</causalConnector>
```

<! OnMouseSelection >

```

<causalConnector id="onSelectionStart">
    <simpleCondition role="onSelection"/>
    <simpleAction role="start" />
</causalConnector>
<causalConnector id="onSelectionStop">
    <simpleCondition role="onSelection"/>
    <simpleAction role="stop" />
</causalConnector>
<causalConnector id="onSelectionPause">
    <simpleCondition role="onSelection"/>
    <simpleAction role="pause" />
</causalConnector>
<causalConnector id="onSelectionResume">
    <simpleCondition role="onSelection"/>
    <simpleAction role="resume" />
</causalConnector>
<causalConnector id="onSelectionSetVar">
    <connectorParam name="var" />
    <simpleCondition role="onSelection"/>
    <simpleAction role="set" value="$var"/>
</causalConnector>

```

<! OnKeySelection >

```

<causalConnector id="onKeySelectionStart">
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <simpleAction role="start"/>
</causalConnector>
<causalConnector id="onKeySelectionStop">
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>

```

```

        <simpleAction role="stop"/>
    </causalConnector>
    <causalConnector id="onKeySelectionPause">
        <connectorParam name="keyCode"/>
        <simpleCondition role="onSelection" key="$keyCode"/>
        <simpleAction role="pause"/>
    </causalConnector>
    <causalConnector id="onKeySelectionResume">
        <connectorParam name="keyCode"/>
        <simpleCondition role="onSelection" key="$keyCode"/>
        <simpleAction role="resume"/>
    </causalConnector>
    <causalConnector id="onKeySelectionSetVar">
        <connectorParam name="keyCode"/>
        <connectorParam name="var"/>
        <simpleCondition role="onSelection" key="$keyCode"/>
        <simpleAction role="set" value="$var"/>
    </causalConnector>

```

<! OnBeginAttribution >

```

    <causalConnector id="onBeginAttributionStart">
        <simpleCondition role="onBeginAttribution"/>
        <simpleAction role="start"/>
    </causalConnector>
    <causalConnector id="onBeginAttributionStop">
        <simpleCondition role="onBeginAttribution"/>
        <simpleAction role="stop"/>
    </causalConnector>
    <causalConnector id="onBeginAttributionPause">
        <simpleCondition role="onBeginAttribution"/>
        <simpleAction role="pause"/>
    </causalConnector>

```



```
</causalConnector>
<causalConnector id="onBeginAttributionResume">
  <simpleCondition role="onBeginAttribution"/>
  <simpleAction role="resume"/>
</causalConnector>
<causalConnector id="onBeginAttributionSet">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <simpleAction role="set" value="$var"/>
</causalConnector>
```

<! OnEndAttribution >

```
<causalConnector id="onEndAttributionStart">
  <simpleCondition role="onEndAttribution"/>
  <simpleAction role="start"/>
</causalConnector>
<causalConnector id="onEndAttributionStop">
  <simpleCondition role="onEndAttribution"/>
  <simpleAction role="stop"/>
</causalConnector>
<causalConnector id="onEndAttributionPause">
  <simpleCondition role="onEndAttribution"/>
  <simpleAction role="pause"/>
</causalConnector>
<causalConnector id="onEndAttributionResume">
  <simpleCondition role="onEndAttribution"/>
  <simpleAction role="resume"/>
</causalConnector>
<causalConnector id="onEndAttributionSet">
  <connectorParam name="var"/>
  <simpleCondition role="onEnd"/>
```

```
    <simpleAction role="set" value="$var"/>
</causalConnector>
```

<! OnBegin multiple actions >

```
<causalConnector id="onBeginStartStop">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="start"/>
        <simpleAction role="stop"/>
    </compoundAction>
</causalConnector>

<causalConnector id="onBeginStartPause">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="start"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>

<causalConnector id="onBeginStartResume">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="start"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>

<causalConnector id="onBeginStartSet">
    <connectorParam name="var"/>
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="start"/>
        <simpleAction role="set" value="$var"/>
    </compoundAction>
</causalConnector>
```

```

        </compoundAction>
</causalConnector>
<causalConnector id="onBeginStopStart">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginStopPause">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginStopResume">
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginStopSet">
    <connectorParam name="var"/>
    <simpleCondition role="onBegin"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="set" value="$var"/>
    </compoundAction>
</causalConnector>

```

```

<causalConnector id="onBeginSetStart">
  <connectorParam name="var"/>
  <simpleCondition role="onBegin"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginSetStop">
  <connectorParam name="var"/>
  <simpleCondition role="onBegin"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginSetPause">
  <connectorParam name="var"/>
  <simpleCondition role="onBegin"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginSetResume">
  <connectorParam name="var"/>
  <simpleCondition role="onBegin"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="resume"/>
  </compoundAction>

```

</causalConnector>

<! OnEnd multiple actions >

```
<causalConnector id="onEndStartStop">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
```

</causalConnector>

```
<causalConnector id="onEndStartPause">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
```

</causalConnector>

```
<causalConnector id="onEndStartResume">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
```

</causalConnector>

```
<causalConnector id="onEndStartSet">
  <connectorParam name="var"/>
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
```

```

</causalConnector>
<causalConnector id="onEndStopStart">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndStopPause">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndStopResume">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndStopSet">
  <connectorParam name="var"/>
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndSetStart">

```

```

    <connectorParam name="var"/>
    <simpleCondition role="onEnd"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndSetStop">
    <connectorParam name="var"/>
    <simpleCondition role="onEnd"/>
    <compoundAction operator="seq">
        <simpleAction role="stet" value="$var"/>
        <simpleAction role="stop"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndSetPause">
    <connectorParam name="var"/>
    <simpleCondition role="onEnd"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndSetResume">
    <connectorParam name="var"/>
    <simpleCondition role="onEnd"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>

```

<! OnMouseSelection multiple actions >

```
<causalConnector id="onSelectionStartStop">
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStartPause">
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStartResume">
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStartSet">
  <connectorParam name="var"/>
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
```



```

<causalConnector id="onSelectionStopStart">
  <simpleCondition role="onEnd"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStopPause">
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStopResume">
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionStopSet">
  <connectorParam name="var"/>
  <simpleCondition role="onSelection"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onSelectionSetStart">
  <connectorParam name="var"/>

```

```

    <simpleCondition role="onSelection"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onSelectionSetStop">
    <connectorParam name="var"/>
    <simpleCondition role="onSelection"/>
    <compoundAction operator="seq">
        <simpleAction role="stet" value="$var"/>
        <simpleAction role="stop"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onSelectionSetPause">
    <connectorParam name="var"/>
    <simpleCondition role="onSelection"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onSelectionSetResume">
    <connectorParam name="var"/>
    <simpleCondition role="onSelection"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>

```

<! OnKeySelection multiple actions >

```
<causalConnector id="onKeySelectionStartStop">
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStartPause">
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStartResume">
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStartSet">
  <connectorParam name="var"/>
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
```

```

        <simpleAction role="set" value="$var"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStopStart">
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStopPause">
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStopResume">
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStopSet">
    <connectorParam name="var"/>
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>

```

```

    <compoundAction operator="seq">
      <simpleAction role="stop"/>
      <simpleAction role="set" value="$var"/>
    </compoundAction>
  </causalConnector>
  <causalConnector id="onKeySelectionSetStart">
    <connectorParam name="var"/>
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
      <simpleAction role="set" value="$var"/>
      <simpleAction role="start"/>
    </compoundAction>
  </causalConnector>
  <causalConnector id="onKeySelectionSetStop">
    <connectorParam name="var"/>
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
      <simpleAction role="set" value="$var"/>
      <simpleAction role="stop"/>
    </compoundAction>
  </causalConnector>
  <causalConnector id="onKeySelectionSetPause">
    <connectorParam name="var"/>
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
      <simpleAction role="set" value="$var"/>
      <simpleAction role="pause"/>
    </compoundAction>

```

```

</causalConnector>
<causalConnector id="onKeySelectionSetResume">
  <connectorParam name="var"/>
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>

```

<! OnBeginAttribution multiple actions >

```

<causalConnector id="onBeginAttributionStartStop">
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStartPause">
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStartResume">
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="resume"/>
  </compoundAction>

```

```

        </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStartSet">
    <connectorParam name="var"/>
    <simpleCondition role="onBeginAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="start"/>
        <simpleAction role="set" value="$var"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStopStart">
    <simpleCondition role="onBeginAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStopPause">
    <simpleCondition role="onBeginAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionStopResume">
    <simpleCondition role="onBeginAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="resume"/>
    </compoundAction>
</causalConnector>

```

```

<causalConnector id="onBeginAttributionStopSet">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionSetStart">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionSetStop">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onBeginAttributionSetPause">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="pause"/>
  </compoundAction>

```



```

</causalConnector>
<causalConnector id="onBeginAttributionSetResume">
  <connectorParam name="var"/>
  <simpleCondition role="onBeginAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>

```

<! OnEndAttribution multiple actions >

```

<causalConnector id="onEndAttributionStartStop">
<simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="stop"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStartPause">
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStartResume">
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="resume"/>
  </compoundAction>

```

```

</causalConnector>
<causalConnector id="onEndAttributionStartSet">
  <connectorParam name="var"/>
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="start"/>
    <simpleAction role="set" value="$var"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStopStart">
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStopPause">
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="pause"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStopResume">
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionStopSet">

```

```

    <connectorParam name="var"/>
    <simpleCondition role="onEndAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="set" value="$var"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionSetStart">
    <connectorParam name="var"/>
    <simpleCondition role="onEndAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionSetStop">
    <connectorParam name="var"/>
    <simpleCondition role="onEndAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="stop"/>
    </compoundAction>
</causalConnector>
<causalConnector id="onEndAttributionSetPause">
    <connectorParam name="var"/>
    <simpleCondition role="onEndAttribution"/>
    <compoundAction operator="seq">
        <simpleAction role="set" value="$var"/>
        <simpleAction role="pause"/>
    </compoundAction>
</causalConnector>

```

```

<causalConnector id="onEndAttributionSetResume">
  <connectorParam name="var"/>
  <simpleCondition role="onEndAttribution"/>
  <compoundAction operator="seq">
    <simpleAction role="set" value="$var"/>
    <simpleAction role="resume"/>
  </compoundAction>
</causalConnector>

```

<!Miscellaneous>

```

<causalConnector id="onKeySelectionStopResizePauseStart">
  <connectorParam name="width"/>
  <connectorParam name="height"/>
  <connectorParam name="left"/>
  <connectorParam name="top"/>
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="stop"/>
    <simpleAction role="setWidth" value="$width"/>
    <simpleAction role="setHeight" value="$height"/>
    <simpleAction role="setLeft" value="$left"/>
    <simpleAction role="setTop" value="$top"/>
    <simpleAction role="pause"/>
    <simpleAction role="start"/>
  </compoundAction>
</causalConnector>

```

```

<causalConnector id="onEndResizeResume">
  <connectorParam name="left"/>
  <connectorParam name="top"/>
  <connectorParam name="width"/>

```

```

<connectorParam name="height"/>
<simpleCondition role="onEnd"/>
<compoundAction operator="seq">
    <simpleAction role="setLeft" value="$left"/>
    <simpleAction role="setTop" value="$top"/>
    <simpleAction role="setWidth" value="$width"/>
    <simpleAction role="setHeight" value="$height"/>
    <simpleAction role="resume"/>
</compoundAction>
</causalConnector>
<causalConnector id="onKeySelectionStopSetPauseStart">
    <connectorParam name="bounds"/>
    <connectorParam name="keyCode"/>
    <simpleCondition role="onSelection" key="$keyCode"/>
    <compoundAction operator="seq">
        <simpleAction role="stop"/>
        <simpleAction role="set" value="$bounds"/>
        <simpleAction role="pause"/>
        <simpleAction role="start"/>
    </compoundAction>
</causalConnector>
</connectorBase>
</head>
</ncl>

```

ANEXO C: ARCHIVOS DE CONFIGURACION OPENVPN

SEVIDOR

```

#####
# Sample OpenVPN 2.0 config file for      #

```

```

# multi-client server.          #
# This file is for the server side      #
# of a many-clients <-> one-server      #
# OpenVPN configuration.              #
# OpenVPN also supports                #
# single-machine <-> single-machine     #
# configurations (See the Examples page #
# on the web site for more info).      #
# This config should work on Windows   #
# or Linux/BSD systems. Remember on    #
# Windows to quote pathnames and use   #
# double backslashes, e.g.:            #
# "C:\\Program Files\\OpenVPN\\config\\foo.key" #
# Comments are preceded with '#' or ';' #
#####
# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one. You will need to
# open up this port on your firewall.
port 1194

# TCP or UDP server?
;proto tcp
proto udp

# "dev tun" will create a routed IP tunnel,

```

```
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one. On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key). Each client
# and the server must have their own cert and
# key file. The server and all clients will
# use the same ca file.
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
```

```
# and private keys. Remember to use
# a unique Common Name for the server
# and each of the client certificates.
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert server.crt
key server.key
# This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
# openssl dhparam -out dh2048.pem 2048
dh dh2048.pem

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0
```



```
# Maintain a record of client <-> virtual IP address
# associations in this file. If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist ipp.txt
```

```
# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface. Then you must manually set the
# IP/netmask on the bridge interface, here we
# assume 10.8.0.4/255.255.255.0. Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients. Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100
```

```
# Configure server mode for ethernet bridging
# using a DHCP-proxy, where clients talk
# to the OpenVPN server-side DHCP server
# to receive their IP address allocation
# and DNS server addresses. You must first use
# your OS's bridging capability to bridge the TAP
# interface with the ethernet NIC interface.
# Note: this mode only works on clients (such as
# Windows), where the client-side TAP adapter is
# bound to a DHCP client.
;server-bridge
```

```
# Push routes to the client to allow it
# to reach other private subnets behind
# the server. Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.
;push "route 192.168.10.0 255.255.255.0"
;push "route 192.168.20.0 255.255.255.0"

# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).

# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
# Then create a file ccd/Thelonious with this line:
# iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN. This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.
```

```
# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
# ifconfig-push 10.9.0.1 10.9.0.2

# Suppose that you want to enable different
# firewall access policies for different groups
# of clients. There are two methods:
# (1) Run multiple OpenVPN daemons, one for each
# group, and firewall the TUN/TAP interface
# for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
# modify the firewall in response to access
# from different clients. See man
# page for more info on learn-address script.
;learn-address ./script

# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# or bridge the TUN/TAP interface to the internet
# in order for this to work properly).
;push "redirect-gateway def1 bypass-dhcp"

# Certain Windows-specific network settings
```

```
# can be pushed to clients, such as DNS
# or WINS server addresses. CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
# The addresses below refer to the public
# DNS servers provided by.opendns.com.
;push "dhcp-option DNS 208.67.222.222"
;push "dhcp-option DNS 208.67.220.220"
```

```
# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
;client-to-client
```

```
# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names. This is recommended
# only for testing purposes. For production use,
# each client should have its own certificate/key
# pair.
# if you have not generated individual
# certificate/key pairs for each client,
# each having its own unique "common name",
# uncomment this line out.
;duplicate-cn
```

```
# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
```

```
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
# Generate with:
#  openssl --genkey --secret ta.key
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
tls-auth ta.key 0 # This file is secret

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"
```

```
# For compression compatible with older clients use comp-lzo
# If you enable it here, you must also
# enable it in the client config file.
;comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
# You can uncomment this out on
# non-Windows systems.
;user nobody
;group nobody

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status openvpn-status.log

# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
```

```
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it. Use one
# or the other (but not both).
;log    openvpn.log
;log-append openvpn.log
```

```
# Set the appropriate level of log
# file verbosity.
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 3
```

```
# Silence repeating messages. At most 20
# sequential messages of the same message
# category will be output to the log.
;mute 20
```

```
# Notify the client that when the server restarts so it
# can automatically reconnect.
explicit-exit-notify 1
```

CLIENTE

```
#####
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.  #
# This configuration can be used by multiple #
# clients, however each client should have #
# its own cert and key files.          #
```

```
# On Windows, you might want to rename this #
# file so it has a .ovpn extension      #
#####

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
Client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one. On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
;proto tcp
proto udp

# The hostname/IP and port of the server.
```



```
# You can have multiple remote entries
# to load balance between the servers.
remote my-server-1 1194
;remote my-server-2 1194

# Choose a random host from the remote
# list for load-balancing. Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server. Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific local port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
;user nobody
;group nobody

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
```

```
# port number here. See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets. Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description. It's best to use
# a separate .crt/.key file pair
# for each client. A single ca
# file can be used for all clients.
ca ca.crt
cert client.crt
key client.key
# Verify server certificate by checking that the
# certificate has the correct key usage set.
# This is an important precaution to protect against
# a potential attack discussed here:
# http://openvpn.net/howto.html#mitm
# To use this feature, you will need to generate
# your server certificates with the keyUsage set to
# digitalSignature, keyEncipherment
# and the extendedKeyUsage to
# serverAuth
# EasyRSA can do this for you.
```

remote-cert-tls server

If a tls-auth key is used on the server
then every client must also have the key.
tls-auth ta.key 1

Select a cryptographic cipher.
If the cipher option is used on the server
then you must also specify it here.
Note that v2.4 client/server will automatically
negotiate AES-256-GCM in TLS mode.
See also the ncp-cipher option in the manpage
cipher AES-256-CBC

Enable compression on the VPN link.
Don't enable this unless it is also
enabled in the server config file.
#comp-lzo

Set log file verbosity.
verb 3

Silence repeating messages
;mute 20



**DIRECCIÓN DE DERECHO DE AUTOR
Y DERECHOS CONEXOS
RESOLUCIÓN ADMINISTRATIVA NRO. 1-323-D/2020
La Paz, 16 de Octubre del 2020**

VISTOS:

La solicitud de Inscripción de Derecho de Autor presentada en fecha **14 de Octubre del 2020** vía online, por **BISMARCK PONCE LIMACHI** con C.I. N° **9101096 LP.**, con número de trámite **DA-128-DIG/2020**, señala la pretensión de inscripción del Proyecto de Grado titulado: **"DISEÑO DE SISTEMA DE ALERTA TEMPRANA COMPLEMENTARIA PARA PREVENCIÓN DE DESASTRES. Aplicación: sistema nacional de alerta temprana para desastres SINAGER-SNATD y canal de televisión, utilizando la norma ISDB-Tb"**, cuyos datos y antecedentes se encuentran adjuntos y expresados en el Formulario de Declaración Jurada.

CONSIDERANDO

Que, en observación al Artículo 4º del Decreto Supremo N° 27938 modificado parcialmente por el Decreto Supremo N° 28152 el *"Servicio Nacional de Propiedad Intelectual SENAPI, administra en forma desconcentrada e integral el régimen de la Propiedad Intelectual en todos sus componentes, mediante una estricta observancia de los regímenes legales de la Propiedad Intelectual, de la vigilancia de su cumplimiento y de una efectiva protección de los derechos de exclusiva referidos a la propiedad industrial, al derecho de autor y derechos conexos; constituyéndose en la oficina nacional competente respecto de los tratados internacionales y acuerdos regionales suscritos y adheridos por el país, así como de las normas y regímenes comunes que en materia de Propiedad Intelectual se han adoptado en el marco del proceso andino de integración"*.

Que, el Artículo 16º del Decreto Supremo N° 27938 establece *"Como núcleo técnico y operativo del SENAPI funcionan las Direcciones Técnicas que son las encargadas de la evaluación y procesamiento de las solicitudes de derechos de propiedad intelectual, de conformidad a los distintos regímenes legales aplicables a cada área de gestión"*. En ese marco, la Dirección de Derecho de Autor y Derechos Conexos otorga registros con carácter declarativo sobre las obras del ingenio cualquiera que sea el género o forma de expresión, sin importar el mérito literario o artístico a través de la inscripción y la difusión, en cumplimiento a la Decisión 351 Régimen Común sobre Derecho de Autor y Derechos Conexos de la Comunidad Andina, Ley de Derecho de Autor N° 1322, Decreto Reglamentario N° 23907 y demás normativa vigente sobre la materia.

Que, la solicitud presentada cumple con: el Artículo 6º de la Ley N° 1322 de Derecho de Autor, el Artículo 26º inciso a) del Decreto Supremo N° 23907 Reglamento de la Ley de Derecho de Autor, y con el Artículo 4º de la Decisión 351 Régimen Común sobre Derecho de Autor y Derechos Conexos de la Comunidad Andina.

Que, de conformidad al Artículo 18º de la Ley N° 1322 de Derecho de Autor en concordancia con el Artículo 18º de la Decisión 351 Régimen Común sobre Derecho de Autor y Derechos Conexos de la Comunidad Andina, referentes a la duración de los Derechos Patrimoniales, los mismos establecen que: *"la duración de la protección concedida por la presente ley será para toda la vida del autor y por 50 años después de su muerte, a favor de sus herederos, legatarios y cesionarios"*.

Que, se deja establecido en conformidad al Artículo 4º de la Ley N° 1322 de Derecho de Autor, y Artículo 7º de la Decisión 351 Régimen Común sobre Derecho de Autor y Derechos Conexos de la Comunidad Andina que: *"...No son objeto de protección las ideas contenidas en las obras literarias, artísticas, o el contenido ideológico o técnico de las obras científicas ni su aprovechamiento industrial o comercial"*.

Que, el Decreto Supremo, N° 4218 del 14 de Abril de 2020, regula el teletrabajo como una modalidad especial de prestación de servicios caracterizada por la utilización de Tecnologías de la Información y Comunicación - TIC, en los sectores públicos y privados, estableciendo a través de su Artículo 12 que *"con el objeto de implementar y promover el teletrabajo, las entidades públicas, deben desarrollar e implementar una estrategia de digitalización para la atención de trámites y servicios en línea en el marco del Plan de Implementación del Gobierno Electrónico ..."*.

Que, mediante Resolución Administrativa N° 14/2020 del 22 de Abril de 2020, el Director General Ejecutivo del SENAPI, Resuelve: *"... Aprobar el Reglamento para trámites On-Line de la Dirección de Derecho de Autor y Derechos Conexos del Servicio Nacional de Propiedad Intelectual ..."*.

Que, el artículo 4, inciso e) de la ley 2341 de Procedimiento Administrativo, instituye que: *"... en la relación de los particulares con la Administración Pública, se presume el principio de buena fe. La confianza, la cooperación y la lealtad en la actuación de los servidores públicos y de los ciudadanos ..."*, por lo que se presume la buena fe de los administrados respecto a las solicitudes de registro y la declaración jurada respecto a la originalidad de la obra.

POR TANTO

La Directora de Derecho de Autor y Derechos Conexos, sin ingresar en mayores consideraciones de orden legal, en ejercicio de las atribuciones conferidas.

RESUELVE:

INSCRIBIR en el Registro de Tesis, Proyectos de Grado, Monografías y Otras Similares de la Dirección de Derecho de Autor y Derechos Conexos, el Proyecto de Grado titulado: **"DISEÑO DE SISTEMA DE ALERTA TEMPRANA COMPLEMENTARIA PARA PREVENCIÓN DE DESASTRES. Aplicación: sistema nacional de alerta temprana para desastres SINAGER-SNATD y canal de televisión, utilizando la norma ISDB-Tb"**, a favor del autor y titular: **BISMARCK PONCE LIMACHI** con C.I. N° **9101096 LP**, quedando amparado su derecho conforme a Ley, salvando el mejor derecho que terceras personas pudieren demostrar.

Regístrese, Comuníquese y Archívese.



Abog. Gabriela Arancibia Peredo
**DIRECTORA DE DERECHO DE AUTOR
Y DERECHOS CONEXOS
SERVICIO NACIONAL DE PROPIEDAD INTELECTUAL**

poncebismark55@gmail.com

79570108