

Universidad Técnica del Norte



Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

Proyecto de tesis previo a la obtención del título de Ingeniero en
Sistemas Computacionales

Tema:

SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE
TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE
PASAJEROS INTER CANTONAL URCUQUÍ.

Autor:

Fernando Xavier Cangás Cangás

Director:

Ing. Hernán Patricio Castro Andrade

Ibarra, Mayo del 2015

Certificación

En calidad de Director del Trabajo de Grado presentado por el Sr. Fernando Xavier Cangás Cangás, para optar por el Título de Ingeniero en Sistemas Computacionales cuyo tema es: “SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUI.”, considero que el presente trabajo reúne los requisitos correspondientes para ser sometido a la presentación pública y evaluación por parte del tribunal examinador que se designe.

Ing. Hernán Patricio Castro Andrade

Director de Tesis

CERTIFICACIÓN

Señores

UTN

Presente

De mi consideración

Siendo auspiciantes del proyecto de tesis del Sr. Fernando Xavier Cangás Cangás, con cedula de ciudadanía Nro.: 100210867-6, quien desarrollo su trabajo con el tema: “SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUI.”, me es grato informar que se han superado con satisfacción las pruebas técnicas y la revisión de cumplimiento de los requerimientos funcionales, por lo que se recibe el proyecto como culminado y realizado por el Sr. Fernando Xavier Cangás Cangás.

Una vez que hemos recibido la capacitación y documentación respectiva, nos comprometemos a continuar utilizando el mencionado aplicativo en beneficio de nuestra institución

El Sr. Fernando Xavier Cangás Cangás puede hacer uso de este documento para los fines pertinentes en la Universidad Técnica del Norte.

Sr. Franklin Cruz
Presidente de la Cooperativa de Transportes “Urcuquí”

DECLARACIÓN DE AUTORÍA

Yo, Fernando Xavier Cangás Cangás, declaro bajo juramento que el trabajo de grado titulado: **“SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUI”**, es de mi autoría, y que no ha sido presentado en ningún otro grado, ni calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

Fernando Xavier Cangás Cangás

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Yo, Fernando Xavier Cangás Cangás, con cédula de ciudadanía N° 100210867-6 manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la ley de propiedad intelectual del Ecuador artículos 4, 5 y 6, en calidad de autor del trabajo de grado denominado:

“SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUI” que ha sido desarrollado para optar por el título de Ingeniero en Sistemas Computacionales en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la biblioteca de la Universidad Técnica del Norte.

Fernando Xavier Cangás Cangás

CC: 100210867-6

Ibarra, Mayo del 2015

AUTORIZACIÓN DE USO Y DE PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La Universidad Técnica del Norte dentro del proyecto Repositorio Digital Institucional, determino la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad. Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	100210867-6		
APELLIDOS Y NOMBRES:	CANGÁS CANGÁS FERNANDO XAVIER		
DIRECCIÓN:	RIOBAMBA 10-121		
EMAIL:	steveny1981@yahoo.com		
TELÉFONO FIJO:		TÉLEFONO MÓVIL	0988113903
DATOS DE LA OBRA			
TÍTULO:	“SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUÍ”		
AUTOR:	CANGÁS CANGÁS FERNANDO XAVIER		
FECHA:AAAA-MM-DD	2015-05-30		
SOLO PARA TRABAJOS DE GRADO			
PROGRAMA:	<input checked="" type="checkbox"/> PREGRADO	<input type="checkbox"/> POSTGRADO	
TITULO POR EL QUE OPTA:	INGENIERO EN SISTEMAS COMPUTACIONALES		
ASESOR/ DIRECTOR:	Ing. Hernán Patricio Castro Andrade		

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, Fernando Xavier Cangás Cangás, con cédula de ciudadanía Nro. 100210867-6, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en formato digital y autorizo a la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la ley de Educación Superior Artículo 144.

3. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló; sin violar derechos de autor de terceros, por lo tanto la obra es original y que es el titular de los derechos patrimoniales por lo que asumen la responsabilidad sobre el contenido de la misma y saldrán en defensa de la universidad en caso de reclamación por parte de terceros.

Ibarra, a los 30 días del mes de Mayo del 2015.

AUTOR:

.....
Fernando Xavier Cangás Cangás

C.C: 1002108676

ACEPTACIÓN:

.....
Ing. Betty Chávez

JEFE DE BIBLIOTECA

Facultado por resolución del Honorable Consejo Universitario

DEDICATORIA

A mis padres

Sus principios, sembraron en mí la tenacidad para enfrentar cualquier reto.

AGRADECIMIENTO

Agradezco:

A la Cooperativa de Transporte de pasajeros intercantonal Urcuquí, por haberme abierto las puertas de su distinguida institución, y al Ing. Hernán Patricio Castro Andrade, por su acertado criterio profesional sobre la elaboración de la presente Tesis.

RESUMEN

El presente proyecto de tesis consiste en desarrollar e implementar una aplicación Web, encargada de respaldar la gestión de información de la Cooperativa de Transporte Inter cantonal Urcuquí, utilizando la tecnología innovadora Node.js, la cual ofrece prestaciones incomparables en cuanto a mantenimiento y reducción de costos de inversión de infraestructura y tiempo.

En el capítulo I, se describen las funcionalidades de JavaScript tanto del lado del cliente, como del lado del servidor con Node.js, y las fases de la metodología de desarrollo de software Programación Extrema XP con la cual se implementará la aplicación Web Transur.

El capítulo II, presenta el estudio de factibilidad, estableciendo los elementos en los que se basa la productividad interna de la Cooperativa de transporte inter cantonal Urcuquí, con el análisis de las encuestas y sus respectivas conclusiones.

En el capítulo III se desarrolla la aplicación Web Transur, aplicando las fases de la metodología de Programación Extrema XP. En la fase de exploración se definen las historias de usuario y la arquitectura de la aplicación, la fase de planeación establece un cronograma de actividades del programador, la fase de iteraciones implementa las tareas y en la fase de producción se publica la aplicación web Transur.

El capítulo IV cubre el análisis de costo / beneficio en base a las herramientas de software utilizadas y al análisis de la productividad interna de la cooperativa de transporte inter cantonal Urcuquí.

En capítulo V, contiene las conclusiones y recomendaciones de la aplicación Web Transur.

SUMMARY

This thesis project is to develop and deploy a Web application suitable for the support of information management Cooperativa de transporte inter cantonal Urcuquí, using Node.js innovative technology , which offers unmatched performance in terms of maintenance and reducing infrastructure investment costs and time.

In Chapter I, JavaScript functionalities of both the client side and the server side with Node.js are described, and the phases of the software development methodology Extreme Programming XP with which the Transur Web application will be deployed.

Chapter II presents the feasibility study, establishing the facts upon which the internal productivity of the Cooperativa de Transporte intercantonal Urcuquí, with the analysis of surveys and their findings are based.

In Chapter III the Transur Web application is developed by applying phases of XP Extreme Programming methodology. In the exploration phase user stories and application architecture are defined , the planning phase establishes a schedule of activities of the developer , the phase of iterations implements tasks and in the production phase the Transur web application is published .

Chapter IV covers the cost / benefit based on software tools used and the analysis of the internal productivity of the bus cooperative inter cantonal Urcuquí.

In Chapter V contains the conclusions and recommendations of the Transur Web application.

ÍNDICE DE CONTENIDOS

PORTADA.....	I
CERTIFICACIÓN DEL TUTOR	III
CERTIFICACIÓN INSTITUCIONAL.....	IV
DECLARACIÓN DE AUTORÍA.....	V
CESIÓN DE DERECHOS DE AUTOR.....	VI
AUTORIZACIÓN DE USO Y DE PUBLICACIÓN	VII
DEDICATORIA	IX
AGRADECIMIENTO	X
RESUMEN	XI
SUMMARY	XII
CAPITULO I	8
1.1 Introducción a JavaScript.....	9
ECMAScript	9
1.2 JavaScript del lado del cliente.....	10
1.3 JavaScript del lado del servidor	10
1.4 Herramientas JavaScript en el Servidor	11
1.4.1 EJScript.....	11
1.4.2 RingoJS	12
1.4.3 Node.js	12
1.5 Análisis comparativo entre herramientas JavaScript en el servidor.....	14
1.5.1 Criterios de comparación	14
1.5.1.1 Arquitectura	14
1.5.1.2 Integración	16
1.5.1.3 Velocidad	17
1.5.1.4 Acceso a la documentación.....	18
1.6 Resultado del análisis comparativo entre herramientas JavaScript.....	19
1.7 MEAN.....	20

1.7.1 MongoDB.....	20
1.7.2 Framework Express.....	21
1.7.3 AngularJS.....	22
1.7.4 NodeJS.....	23
Socket.IO.....	23
1.8 Callbacks manejadas por evento NodeJS.....	23
Modelo de procesamiento Node.js.....	25
1.9 Servidor HTTP básico.....	26
1.10 Funciones NodeJS.....	27
1.11 Módulos NodeJS.....	28
Módulo URL.....	29
1.12 Ruteo de peticiones NodeJS.....	30
1.13 Metodología XP.....	35
1.13.1 Roles XP.....	35
1.13.2 Valores XP.....	36
1.13.3 Fases de la metodología XP.....	38
Primera Fase: Planificación del proyecto.....	38
Segunda Fase: Diseño.....	38
Tercera fase: Codificación.....	39
Cuarta Fase: Pruebas.....	39
1.13.4 Ventajas y desventajas de la metodología XP.....	40
CAPÍTULO II.....	41
2.1 Antecedentes.....	42
2.2 Objetivos.....	42
2.2.1 Objetivo diagnóstico.....	42
2.2.2 Objetivos Específicos.....	42
2.3 Variables.....	42
2.4 Indicadores.....	43
Técnicas.....	43
Fuentes de información.....	43
2.5 Matriz relación.....	43

2.6 Selección de la muestra.....	44
Muestra	45
2.7 Análisis e interpretación de resultados.....	46
Encuesta Socios - conductores.....	46
Entrevista Personal – administrativo.....	53
2.8 Conclusiones del estudio.....	55
CAPÍTULO III.....	57
3.1 Fase de exploración.....	58
3.1.1 Historias de usuario.....	58
3.1.1.1 Iteración I.....	58
Gestión del proyecto	58
Iteración 1	58
3.1.2 Arquitectura Funcional del sistema TRANSUR.....	66
3.2 Fase de planeación	66
3.2.1 Cronograma de actividades.....	66
3.2.2 Módulos de sistema TRANSUR.....	67
3.2.2.1 Descripción de los módulos del sistema TRANSUR.....	67
3.3. Iteración I.....	68
3.3.1. Cronograma de actividades	69
3.3.2. Tareas por historia Iteración I.....	69
3.3.2.1 Registro y Autenticación.....	69
3.3.2.2 Registro de Usuarios TRANSUR.....	75
3.3.2.3 Registro de información institucional	79
3.3.2.4 Registro de Socios.....	83
3.3.2.5 Registro de conductores	87
3.3.3. Pruebas Iteración I	91
3.3.3.1. Especificación de Prueba: Registro y Autenticación	92
3.3.3.2 Especificación de Prueba: Registro de Usuarios TRANSUR.....	93
3.3.3.3 Especificación de Prueba: Registro de Información Institucional	94
3.3.3.4 Especificación de Prueba: Registro de Socios	95

3.3.3.5 Especificación de Prueba: Registro de Conductores	97
3.3.4. Resultado de la 1ª Iteración	99
3.3.5. Demo de la versión, Iteración I	99
3.3.6. Incidencias.....	106
3.3.7 Diagrama de base de datos	107
3.3.8 Producción.....	108
3.3.9 Publicación de la aplicación TRANSUR	109
CAPÍTULO IV	110
4.1 Análisis Costo Beneficio	111
4.1.1. Objetivo.....	111
4.1.2. Costos	111
Estimación de gastos operativos.....	112
Estimación de los gastos operativos con TRANSUR.....	113
Conclusión de gastos operativos con TRANSUR	113
Rentabilidad del proyecto TRANSUR	113
4.2 Conclusiones	116
4.3 Recomendaciones.....	117
Bibliografía.....	118
GLOSARIO.....	120
ANEXOS.....	121

ÍNDICE DE TABLAS

Tabla 1: Análisis comparativo entre herramientas JavaScript del Servidor.....	18
Tabla 2: Resultado del análisis comparativo entre las herramientas JavaScript del servidor	19
Tabla 3: Matriz relación.....	44
Tabla 4. Población estimada	44
Tabla 5:. Población	45
Tabla 6: Pregunta1 Socios-Conductores	46
Tabla 7: Pregunta 2 Socios-Conductores	47
Tabla 8: Pregunta 3 Socios-Conductores	48
Tabla 9: Pregunta 4 Socios-Conductores	50
Tabla 10: Pregunta 5 Socios-Conductores	51
Tabla 11: Pregunta 6 Socios-Conductores	52
Tabla 12: Historia 1	58
Tabla 13: Historia 2	59
Tabla 14: Historia 3	60
Tabla 15: Historia 4	60
Tabla 16: Historia 5	61
Tabla 17: Historia 6	62
Tabla 18: Historia 7	63
Tabla 19: Historia 8	63
Tabla 20: Historia 9	64
Tabla 21: Historia 10	65
Tabla 22: Cronograma de actividades.....	66
Tabla 23: Cronograma de actividades Iteración I	69
Tabla 24: Tareas de Registro y Autenticación	69
Tabla 25: Registro y Autenticación – Tarea 1	70
Tabla 26: Registro y Autenticación – Tarea 2	70
Tabla 27: Registro y Autenticación – Tarea 3	71
Tabla 28: Registro y Autenticación – Tarea 4	71
Tabla 29: Registro y Autenticación – Tarea 5	72
Tabla 30: Registro y Autenticación – Tarea 6	72

Tabla 31: Registro y Autenticación – Tarea 7.....	73
Tabla 32: Registro y Autenticación – Tarea 8.....	73
Tabla 33: Registro y Autenticación – Tarea 9.....	74
Tabla 34: Registro y Autenticación – Tarea 10.....	74
Tabla 35: Tareas de Registro de Usuarios TRANSUR	75
Tabla 36: Registro de Usuarios TRANSUR - Tarea 1	75
Tabla 37: Registro de Usuarios TRANSUR - Tarea 2	76
Tabla 38: Registro de Usuarios TRANSUR - Tarea 3	76
Tabla 39: Registro de Usuarios TRANSUR - Tarea 4	77
Tabla 40: Registro de Usuarios TRANSUR - Tarea 5	77
Tabla 41: Registro de Usuarios TRANSUR - Tarea 6	78
Tabla 42: Registro de Usuarios TRANSUR - Tarea 7	78
Tabla 43: Registro de Información Institucional.....	79
Tabla 44: Registro de Información Institucional - Tarea 1	79
Tabla 45: Registro de Información Institucional - Tarea 2	80
Tabla 46: Registro de Información Institucional - Tarea 3	80
Tabla 47: Registro de Información Institucional - Tarea 4	81
Tabla 48: Registro de Información Institucional - Tarea 5	81
Tabla 49: Registro de Información Institucional - Tarea 6	82
Tabla 50: Registro de Información Institucional - Tarea 7	82
Tabla 51: Tareas de Registro de Socios	83
Tabla 52: Registro de Socios - Tarea 1	83
Tabla 53: Registro de Socios - Tarea 2	84
Tabla 54: Registro de Socios - Tarea 3	84
Tabla 55: Registro de Socios - Tarea 4	85
Tabla 56: Registro de Socios - Tarea 5	85
Tabla 57: Registro de Socios - Tarea 6	85
Tabla 58: Registro de Socios - Tarea 7	86
Tabla 59: Tareas de Registro de Conductores.....	87
Tabla 60: Registro de Conductores - Tarea 1	87
Tabla 61: Registro de Conductores - Tarea 2.....	88
Tabla 62: Registro de Conductores - Tarea 3.....	89

Tabla 63: Registro de Conductores - Tarea 4.....	89
Tabla 64: Registro de Conductores - Tarea 5.....	89
Tabla 65: Registro de Conductores - Tarea 6.....	90
Tabla 66: Registro de Conductores - Tarea 7.....	90
Tabla 67: Iteración I, Historial de revisiones Nro. 1.....	92
Tabla 68: Iteración I, Historial de revisiones Nro. 2.....	93
Tabla 69: Iteración I, Historial de revisiones Nro. 3.....	94
Tabla 70: Iteración I, Historial de revisiones Nro. 4.....	96
Tabla 71: Iteración I, Historial de revisiones Nro. 5.....	97
Tabla 72: Costos TRANSUR.....	111
Tabla 73: Lista de costos de servicios de telefonía CNT	112
Tabla 74: Estimación de gastos operativos	112
Tabla 75: Estimación de gastos operativos con la aplicación Transur.....	113
Tabla 76: Comparativa de gastos operativos	113
Tabla 77: Rentabilidad del proyecto TRANSUR	114
Tabla 78: Indicadores de estudio de factibilidad TRANSUR.....	114
Tabla 79: Indicadores con la implementación de la aplicación TRANSUR.....	115

CAPÍTULO I

Marco Teórico

- 1.1 Introducción a JavaScript
 - ECMAScript
- 1.2 JavaScript del lado del cliente
- 1.3 JavaScript del lado del servidor
- 1.4 Servidores de JavaScript
- 1.5 Análisis comparativo entre servidores de JavaScript
- 1.6 Resultado del análisis herramientas JavaScript
- 1.7 MEAN
- 1.8 Callbacks manejadas por evento NodeJS
- 1.9 Servidor HTTP básico
- 1.10 Funciones NodeJS
- 1.11 Módulos NodeJS
- 1.12 Ruteo de peticiones NodeJS
- 1.13 Metodología XP

1.1 Introducción a JavaScript

A principios de los años 90, empezaban a desarrollarse las primeras aplicaciones web, las cuales incluían formularios complejos, con una velocidad de navegación que alcanzaba los 28.8 kbps.

En estas condiciones, surgió la necesidad de la creación de un lenguaje de programación que se ejecutara en el navegador del cliente, liberando así al servidor, de peticiones tales como validación de formularios.

JavaScript fue creado por Brendan Eich que trabajaba para Netscape¹ en los años 90, inicialmente el lenguaje se lo denominó LiveScript. Posteriormente, con la alianza con Sun Microsystems se decidió cambiar el nombre por el de JavaScript.

Con la aparición del navegador web Netscape Navigator 3.0 se incorpora la versión 1.1 de JavaScript, convirtiéndose en un verdadero éxito, al mismo tiempo, Microsoft lanzó Internet Explorer 3 con JScript que era una copia de JavaScript.

Con la intención de evitar conflictos legales, Netscape decidió estandarizar el lenguaje JavaScript. Así, en 1997 se envió la especificación JavaScript 1.1 al organismo ECMA (European Computer Manufacturers Association).

ECMAScript

ECMAScript es una especificación de lenguaje de programación presentada por ECMA² International (European Computer Manufacturers Association), basada en el lenguaje JavaScript, en la actualidad representa el estándar ISO 16262.

La especificación ECMAScript define un lenguaje de tipos dinámicos inspirado en Java y otros lenguajes, soportando algunas características de la programación orientada a objetos.

El estándar ECMAScript, y el acceso al Document Object Model (DOM)³, residen en la mayoría de los navegadores web, incluyen funcionalidades para manipular páginas

¹ **Netscape:** Es una empresa de software creador del navegador web Netscape Navigator.

² **ECMA:** Organización Internacional encargada de validar la entrega de membresías y estándares de comunicación e información.

³ **DOM:** es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante JavaScript para cambiar dinámicamente los contenidos y aspecto de la página.

web. Algunos navegadores como Internet Explorer de Microsoft usan un estándar diferente en el cual reside el lenguaje JScript. Opera tiene su propio intérprete de ECMAScript con características de JavaScript y JScript para navegadores de Internet Explorer.

1.2 JavaScript del lado del cliente

Entre los principales lenguajes de programación web del lado del cliente se pueden citar Java (applets), VBScript, Dart(Google) y JavaScript, el cual es el más utilizado y aceptado por todos los navegadores. (Wikipedia, Javascript, 2013)

Estos lenguajes de programación, realizan localmente (en el cliente) procesos de respuesta rápida y efectos controlados como validación de formularios, efectos visuales o cálculos en las aplicaciones web, aliviando así la cantidad de peticiones o servicios solicitados al servidor.

JavaScript se integra en los ámbitos de desarrollos web, junto con otros lenguajes del lado del servidor como PHP, JSP ASP. En la práctica, JavaScript reside principalmente en el lado del cliente, sin dejar a un lado la posibilidad de que se integre junto al código del lado cliente y del servidor de las aplicaciones web.

1.3 JavaScript del lado del servidor

JavaScript es un lenguaje de programación interpretado, débilmente tipado y dinámico, utilizado en el desarrollo web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

Los navegadores interpretan el código JavaScript integrado en las páginas web y para interactuar con ella se provee de una implementación DOM.

En la actualidad con sucesivas innovaciones tecnológicas, JavaScript está compitiendo en diferentes ámbitos de aplicaciones de software.

Los frameworks⁴ como JQuery⁵ y otros fueron sus grandes revelaciones de JavaScript, la creación de JSON⁶ (JavaScript Object Notation) abrió una rivalidad con XML⁷ para

⁴ **Frameworks:** En el desarrollo de software, un framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software.

la transmisión de datos y luego con la aparición de HTML 5, permiten generar gráficos, juegos y animaciones.

Otra tendencia que está tomando más relevancia en este poderoso lenguaje, es JavaScript del lado del Servidor, compitiendo esta vez con los típicos y establecidos lenguajes de programación Web como JSP, PHP y ASP.

Existen una variedad de alternativas en las que se encuentra JavaScript del lado del servidor, entre estas podemos citar EJScrip, RingoJS, y Node.js, las cuales son evaluadas permanentemente con pruebas de laboratorio y experimentación.

1.4 Herramientas JavaScript en el Servidor

JavaScript del lado del servidor puede ser un concepto novedoso para los desarrolladores que trabajaron exclusivamente con JavaScript del lado del cliente, por qué no se podría utilizar el lenguaje de programación del lado del cliente, en el lado del servidor. Las herramientas de JavaScript del lado del servidor son en realidad proyectos de laboratorio dirigidos a la experimentación y no cuentan con una documentación extensa. Entre las principales alternativas de herramientas de JavaScript del lado del servidor se pueden citar:

1.4.1 EJScrip

La ejecución de EJScrip es diferente a las otras herramientas JavaScript del lado del servidor, pero muy similar en estructura a diseños de servidores clásicos como Struts⁸, utilizando el paradigma MVC Modelo-Vista-Controlador. La herramienta básica es una línea de comandos que puede ejecutar JavaScript o crear un servidor Web. El proceso de desarrollo, de aplicaciones, es similar a la codificación en Java al incluir una manera de definir clases y subclases.

Esta herramienta sería ideal para programadores con conocimientos en Java y JavaScript, con un ligero inconveniente el cual requiere de la duplicación de código que se ejecutara en el cliente y la lógica de negocio en el servidor.

⁵**jQuery:** jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML

⁶**JSON:** Es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML

⁷**XML:** Es un lenguaje de marcas utilizado para almacenar datos en forma legible

⁸ **Struts:** Es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC bajo la plataforma Java EE.

Características

- Estructura similar a servidores clásicos como Struts
- Construido utilizando el paradigma Modelo-Vista-Controlador
- Incluye funcionalidad para definir clases y subclasses.

1.4.2 RingoJS

Es la herramienta de JavaScript del lado del servidor, resultado de la combinación del motor Mozilla Rhino con el núcleo del servidor Web Java, que ofrece un rendimiento aceptable. Sin embargo ahora existe una serie de nuevos motores de JavaScript como V8 de Google, que es mucho más rápido y utiliza muchas de las ideas de compilación just-in-time⁹ que sostenían las máquinas virtuales de Java.

RingoJS y EJScrip están orientados por el paradigma Modelo Vista Controlador (MVC) y se hace eco de muchas de las estructuras comunes utilizados por Java y Rubí.

Características

- El lenguaje JavaScript no tiene las convenciones orientadas a objetos como en EJScrip
- La construcción de un sitio es un proceso similar basado en JSP
- Utiliza el motor Mozilla Rhino con el núcleo del servidor Web Java

1.4.3 Node.js

Entre las aplicaciones web de creciente complejidad, es fundamental el intercambio de código, si se estableciera un entorno de servidor con JavaScript, sería la contante razonable y funcional residente en casi todos de navegadores, teniendo la posibilidad del intercambio de código, lo cual no es posible con otros lenguajes como PHP o Java.

JavaScript es un lenguaje de programación ampliamente utilizado para la programación de las interfaces de los sitios web, por otro lado Node.js permite la aplicación de JavaScript en más contextos y particularmente en servidores web.

⁹ just-in-time: Es la compilación en tiempo de ejecución.

Al igual que el código fuente de JavaScript del lado del cliente es interpretado y ejecutado por un motor JavaScript, Node.js utiliza un intérprete rápido V8 JavaScript¹⁰, escrito en C++ creado por Google, subyacente en su browser Chrome¹¹.

El intérprete V8 JavaScript, podría ser descargado e incorporado en las aplicaciones, sin tener ningún tipo de restricción en la ejecución del mismo, en otros navegadores.

Node.js es un contenedor del alto rendimiento, el mismo que proporciona APIs¹² adicionales optimizadas para los casos de uso específicos, como la manipulación de datos binarios, que no son soportados por JavaScript.

Node.js no sólo proporcionara acceso directo a la ejecución de las aplicaciones en su interprete V8 JavaScript, sino que además hace que JavaScript sea más útil en otros contextos, como el de residir del lado del servidor.

JavaScript es un lenguaje orientado a eventos, y Node.js utiliza esta orientación para producir servidores altamente escalables, reduciendo la complejidad de la concurrencia para el desarrollo de aplicaciones.

Los lenguajes tradicionales de aplicaciones Web generan un elevado costo de memoria en conexiones concurrentes, siendo necesario en ocasiones agregar más servidores para abastecer la demanda de conexiones.

Node.js¹³ resuelve este elevado consumo de recursos cambiando la forma de conexión con el servidor, disparando cada ejecución de evento, dentro del proceso del motor de Node.js en lugar de generar un hilo por cada conexión.

Proporciona un conjunto de bibliotecas "sin bloqueo", en esencia, se trata de interfaces a recursos tales como un sistema de archivos o bases de datos, que funcionan de una manera orientada a eventos, cuando se hace una petición al sistema de archivos, en lugar de esperar a que el disco duro responda y recuperar el archivo, la interfaz sin bloqueo simplemente hace una notificación cuando se tiene acceso al recurso, de la misma manera forma que los navegadores web notifican a su código sobre algún evento. Las conexiones establecidas con el servidor nunca se quedarán en estado liquidado, porque

¹⁰ **V8 JavaScript:** V8 es un motor de código abierto para JavaScript creado por Google, siendo su creador Lars Bak.

¹² **APIs:** Es el conjunto de subrutinas o procedimientos de una biblioteca, la cual puede ser utilizada por otra aplicación o software.

¹² **Chrome:** Google Chrome es un navegador web desarrollado por Google y compilado con base en varios componentes e infraestructuras de desarrollo de aplicaciones (frameworks) de código abierto.

¹³ **Node.js :** Node.js es un entorno de programación en la capa del servidor basado en el lenguaje de programación JavaScript

no se permiten bloqueos para las llamadas Entrada /Salida, afirmando que el servidor que lo ejecute puede soportar miles de conexiones concurrentes.

Los servidores web como Apache¹⁴ o Tomcat¹⁵ son productos listos para instalarse y alojar aplicaciones instantáneamente, mientras que Node.js es un programa de servidor. Node.js afirma poseer las funcionalidades de agregar módulos de forma similar como lo hace PHP en Apache, para la implementación de aplicaciones web dinámicas. (Apache, 2015)

Node.js es extremadamente extensible, con un gran volumen de módulos disponibles en su comunidad, construidos en un tiempo relativamente corto, en relación con el lanzamiento del proyecto, estos módulos están al alcance del programador en contadas líneas de comandos. Aunque existen otras plataformas que soportan JavaScript del lado del servidor; Node.js se está convirtiendo en la plataforma dominante.

Node.js con el motor V8 JavaScript, cristaliza la posibilidad de utilizar el lenguaje de programación JavaScript en el lado del servidor, en este contexto se podría definir a Node.js como un intérprete JavaScript del lado del servidor orientado a la construcción de aplicaciones de red escalables.

Características

- Node.js crea solo un hilo de procesos para todos los clientes lo que hace que el servidor soporte muchas más conexiones.
- Velocidad de Respuesta en sistemas Cliente-Servidor
- Software con licencia OpenSource.

1.5 Análisis comparativo entre herramientas JavaScript en el Servidor

Para establecer un análisis comparativo entre las herramientas de JScripnt del lado del servidor, se enlista un conjunto de características orientadas a comparar las ventajas y desventajas de EJScript, RingoJS y Node.js.

1.5.1 Criterios de comparación

1.5.1.1 Arquitectura: La arquitectura en las herramientas JavaScript, definen en un alto nivel el diseño y de implementación de su estructura de Software. Es el resultado de

¹⁴ **Apache:** El servidor HTTP Apache es un servidor web HTTP de código abierto

¹⁵ **Tomcat:** Apache Tomcat (también llamado Jakarta Tomcat o simplemente Tomcat) funciona como un contenedor de servlets .desarrollado bajo el proyecto Jakarta en la Apache Software Foundation..

Callbacks : Devolución de llamadas

ensamblar sus componentes arquitectónicos de forma adecuada para satisfacer requerimientos de desempeño y confiabilidad de la herramienta de software.

EJScript

Ejscript consiste en un compilador de optimización, una máquina virtual integrable, una biblioteca de sistema y un grupo de herramientas. Diseñados e implementados para proporcionar un alto rendimiento y un compacto entorno de programación JavaScript del lado del servidor.

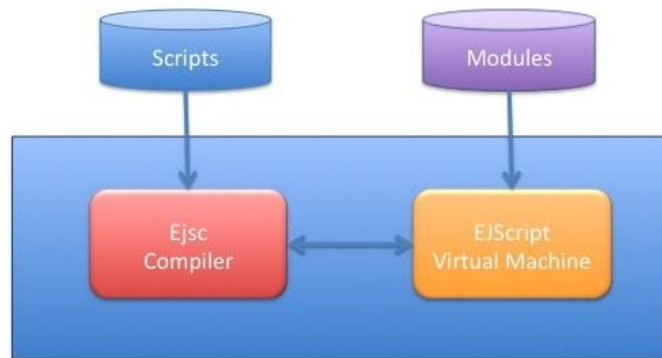


Fig.1 Arquitectura EJScript
Fuente: EJScript

Ringojs

Es el servidor JavaScript resultado de la combinación del motor Mozilla Rhino¹⁶ con el núcleo del servidor Web Java, que ofrece un rendimiento aceptable. Una de sus características, es tener las bibliotecas Java a su alcance, las cuales podrían residir dentro de su aplicación.

NodeJS

Arquitectura Blocking

El significado del término Blocking significa que cuando una línea de código se está ejecutando, el resto de las líneas de código están bloqueadas a la espera de terminar la ejecución.

¹⁶ **Rhino:** Rhino es una implementación de una máquina virtual de JavaScript codificada en Java con licencia Open source

Arquitectura Non-blocking

Por otro lado, en la arquitectura Non-blocking no existe ningún bloqueo para las líneas de código, las cuales que con la intervención de devoluciones de llamada (callbacks) pueden ser llamadas cuando se ocasione un evento.

Los lenguajes de programación como Java, Ruby, Python, PHP utilizan una arquitectura Blocking, teniendo como principal alternativa para superar la concurrencia la utilización de múltiples hilos de ejecución, mientras que JavaScript lo realiza usando un bucle de eventos de un solo hilo sin bloqueos.

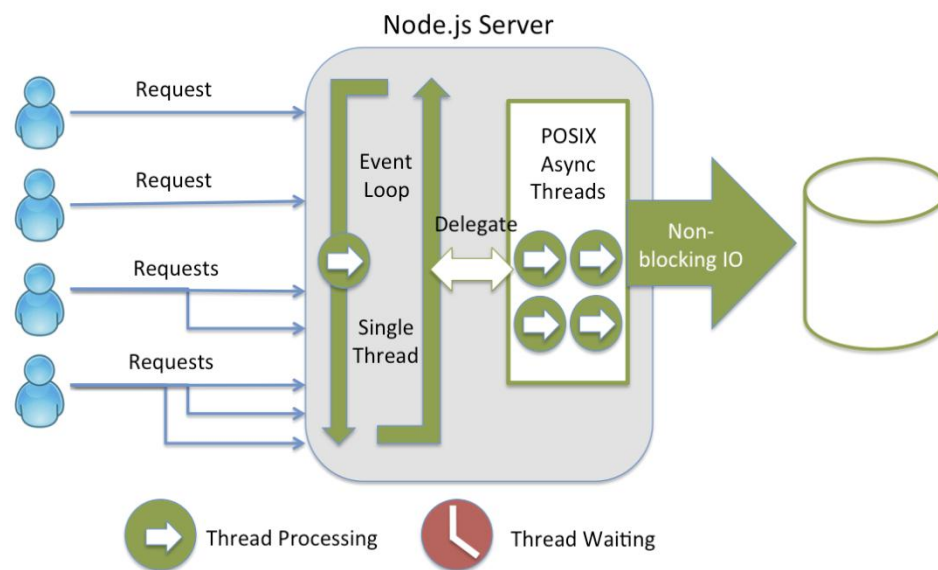


Fig. 2. Arquitectura NodeJs
Fuente: NodeJS

1.5.1.2 Integración

La integración consiste en la realización de pruebas (conocidas como integración y testeo) es la fase de la prueba de software en la cual módulos individuales de software son combinados y probados como un ente único.

EJScript

EJScript ofrece un entorno completo para aplicaciones web de alto rendimiento. Es un framework web Modelo-Vista-Controlador con la base de datos SQLite integrada y bases de datos Objeto Relacionales Mapper (ORM). Además cuenta con bibliotecas de

Ajax¹⁷, JQuery para la integración del lado del cliente y soporte de migración de base de datos.

RingoJs

Con RingoJs existen tres formas de integrar las clases Java y las bibliotecas (.jar) con RingoJs:

Integrando la clase Java o biblioteca en la ruta de clases

Integrando la biblioteca de Java (jar) en el directorio /lib Ringo.

Insertando la biblioteca de Java (.jar) o paquete (directorio) a la ruta de Clases.

NodeJS

Node.js dispone de un gestor de módulos npm (Node Package Manager), encargado de ampliar su funcionalidad y facilitar tareas.

Integración con Base de Datos

Compatibilidad de las herramientas JavaScript con base de datos.

EJScript

EJScript integra bases de datos SQLite integrada y Objeto Relacionales Mapper (ORM)

RingoJS

La integración de RingoJS con las bases de datos se la realiza con Ifuncionalidad proporcionada por Java.

NodeJS

Node.js con su gestor de módulos npm (Node Package Manager), permite ampliar su funcionalidad con base de datos SQL y NoSQL como MongoDB.

1.5.1.3 Velocidad

EJScript

EJScript es rápido, compacto y está optimizado para las aplicaciones web del lado del servidor. Un recolector de basura directa y generacional proporciona una óptima ejecución de las aplicaciones.

RingoJS No existe documentación bajo este parámetro

¹⁷ **Ajax:** Es una tecnología asíncrona, que actúa cuando se solicitan datos adicionales al servidor, estos se cargan en segundo plano sin influir con la visualización, o el comportamiento de la pagina

NodeJS

La conclusión citada según Alejandro Fernández: “su respuesta tan lineal: el doble de peticiones por segundo resulta en el doble de carga” (Fernandez, 2011). Esta conclusión está sustentada en un estudio de rendimiento entre las herramientas más representativas de JavaScript del lado del servidor.

1.5.1.4 Acceso a la documentación

Analizada en relación a la cantidad y calidad de la documentación disponible para el aprendizaje y desarrollo de aplicaciones.

EJScript

EJScript ofrece un aceptable nivel de documentación teórica y técnica sobre el servidor JavaScript alojado en: <https://embedthis.com/ejscrip/>

RingoJS

Presenta un desmejorado nivel de documentación técnica y la documentación teórica es inexistente su sitio web se encuentra alojado en: <http://ringojs.org/>

NodeJS

Ofrece un aceptable nivel de documentación teórica y técnica, en relación a EJScript y RingoJS su documentación está disponible en: <https://nodejs.org/>

Una vez citadas las principales herramientas JavaScript del lado del servidor con sus respectivos criterios de comparación como Arquitectura, Integración, velocidad y documentación. Para la elección de la alternativa idónea entre las herramientas JavaScript del lado del servidor para la implementación del proyecto, en la Tabla 1, se muestran los criterios de comparación con su respectiva puntuación: Bajo: 0-3 Medio: 4-6 Bueno: 7-8 Alto: 9-10.

Tabla 1: Análisis comparativo entre herramientas JavaScript del Servidor

Criterios de comparación	Herramientas JavaScript del Servidor		
	<u>EJScript</u>	<u>RingoJS</u>	<u>Node.js</u>
Arquitectura	7	5	8
Integración	7	4	8
Integración BD	8	4	8
Velocidad	7	5	8
Documentación	7	3	7
TOTAL	36	21	39

Fuente: Propia

1.6 Resultado del análisis comparativo entre herramientas JavaScript

Tabla 2: Resultado del análisis comparativo entre las herramientas JavaScript del servidor

	Herramientas JavaScript	Puntaje	Descripción
1	EJScript	36/5 = 7.2	Bueno
2	RingoJS	21/5 = 4.2	Medio
3	Node.js	39/5 = 7.8	Bueno

Fuente: Propia

En relación a la puntuación mostrada en la Tabla 2, se puede destacar que la tecnología Node.js, que cuenta con el mejor puntaje, lleva ventaja sobre la tecnología RingoJS, en cuanto a los parámetros de comparación como: Arquitectura, Integración y Velocidad.

Por lo tanto la tecnología Node.js será utilizada para la implementación de la aplicación detallada en el capítulo III.

Ante la diversidad de herramientas de software, para el desarrollo web con Node.js se destacan los IDEs¹⁸ Sublime Text, Web Matrix 3, Enide Studio 0.5 y WebStorm.

Una vez efectuadas varias practicas sobre Sublime Text¹⁹, es destacable mencionar que es una herramienta con adaptabilidad para el desarrollo en diferentes plataformas, pero posee una deficiente interfaz gráfica.

La plataforma OpenSource de Microsoft Web Matrix 3²⁰, ofrece un despliegue gráfico amigable con el desarrollador pero deficiente en cuanto a pruebas de depuración sobre el código fuente, excesivas peticiones de actualización y un deficiente mantenimiento para los módulos NPM²¹, que son de vital importancia para el desarrollo con Node.js.

Enide Studio 0.5²², ofrece entre sus principales características una funcionalidad similar a las herramientas Eclipse con alta disposición para la realización de pruebas tipo depuración y un soporte actualizado para los paquetes NPM de Node.js, en modo consola. JetBrains Web Storm 8.0.3²³, posee características de adaptabilidad con GIT²⁴, en diferentes plataformas, con una interfaz gráfica amigable y fácil integración con MEAN Stack.

¹⁸ **IDE:** Entorno de desarrollo integrado es una aplicación de software, que ofrece servicios integrales para el desarrollo de software.

¹⁹ **Sublime Text:** IDE de desarrollo para aplicaciones Nodejs.

²⁰ **Web Matrix 3:** IDE de desarrollo para aplicaciones Nodejs.

²¹ **NPM:** Gestor de paquetes para Node.js.

²² **Enide Studio 0.5:** IDE de desarrollo para aplicaciones Node.js.

²³ **Jet Brains Web Storm 8.0.3:** IDE de desarrollo para aplicaciones Node.js.

²⁴ **Git:** Git es un sistema distribuido de control de versiones de código abierto, que puede ser ejecutado en pequeños y grandes proyectos con rapidez y eficiencia.

Por las características citadas anteriormente se puede concluir que Node.js en su versión 0.10.26 y MongoDB 2.2.7 alojadas en un IDE JetBrains Web Storm 8.0.3 y con un controlador de Versiones GIT para la integración con MEAN stack, forman el conjunto de herramientas de desarrollo idóneas para el desarrollo de la aplicación Web TRANSUR citada en el capítulo III.

1.7 MEAN



Con el lanzamiento de herramientas de desarrollo NodeJS, con JavaScript del lado del servidor, es necesaria la optimización de la construcción de aplicaciones web, robustas y mantenibles.

MEAN es un acrónimo de las tecnologías MongoDB, Express, AngularJS y Node.js, las cuales conforman el llamado “stack MEAN”, cuya característica en común es la presencia del lenguaje de programación JavaScript, de lo que se esperaría tener todas las herramientas necesarias para la construcción de aplicaciones web end-to-end (Frontend, Backend y Persistencia de datos).

1.7.1 MongoDB



La base de datos no relacional MongoDB es uno de los proyectos ligados al movimiento NoSQL²⁵. NoSQL, provee soluciones alternativas a las bases de datos en donde se emplea SQL²⁶ como lenguaje de consulta.

Con SQL se realizan peticiones o consultas a las bases de datos donde la información se encuentra estructurada en base a modelos de datos relacionales, las entidades

²⁵ **NoSQL:** Es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico del sistema de gestión de bases de datos relacionales (RDBMS) en aspectos importantes, el más destacado que no usan SQL como el principal lenguaje de consultas.

²⁶ **SQL:** El lenguaje de consulta estructurado o SQL (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

representadas en esos modelos se almacenan en tablas entre las que se establecen relaciones.

NoSQL es más flexible, existiendo varias alternativas de organizar la información con las combinaciones: clave-valor, clave-documento. Esta flexibilidad se refleja en la realización de las consultas, haciendo uso de la lógica de la aplicación para la realización de las operaciones sobre los datos, con la intención de tener más control sobre el rendimiento de la consulta.

MongoDB, pertenece a la categoría de bases de datos orientadas a documentos, en este caso las entidades no son las filas en una tabla sino más bien un conjunto de pares clave-valor o campos, que a su vez se organizan en colecciones. Las consultas o peticiones se las realiza solicitando documentos cuyos campos deben de cumplir con una serie de criterios establecidos, entre las características más relevantes de MongoDB, se pueden destacar:

Flexibilidad: Utiliza un modelo de datos, del tipo documentos en notación JavaScript (JSON), estos documentos no poseen un esquema fijo, como los documentos XML, los cuales son más rígidos en el sentido que los elementos deben ajustarse al esquema previamente establecido.

Potencia: Incorpora mecanismos propios de bases de datos relacionales, incorporando varias funcionalidades como: índices secundarios, consultas dinámicas, ordenación y operaciones “upsert²⁷”

Velocidad: Almacena la totalidad de la información relacionada en documentos, en lugar de en tablas proveyendo de escalabilidad horizontal, con el uso de mecanismos que garantizaran un servicio seguro y sin interrupciones.

En la actualidad los tipos de información de las aplicaciones web demandan alta flexibilidad, menor coherencia y mayor capacidad de escalar, surgiendo la tendencia tecnológica NoSQL. De la persistencia de almacenes de datos no-relacionales se destaca MongoDB, representa la "M" del stack MEAN.

1.7.2 Framework Express



Node.js es el encargado de crear la lógica de las aplicaciones, dispone de un gestor de paquetes (npm) como también el módulo HTTP, y aun así sería costoso crear una aplicación sin la ayuda de Express. Express es el framework creado en JavaScript para

²⁷ **Upsert:** Update si el documento existe, insert si no existe

Node.js, cuyo objetivo principal es ofrecer soporte para los requerimientos de estas aplicaciones tales como gestión de servicios en el servidor como: request, response, route y views. Express ofrece varias características entre las que cabe destacar:

- Eficaz sistema de enrutamiento de peticiones
- Soporta la generación de HTML dinámicas con la capacidad de utilizar motores de renderizado de vistas.

Con la intervención de express en el servidor, es posible crear aplicaciones web de manera práctica, representa la "E" del stack MEAN.

1.7.3 AngularJS



En las aplicaciones web actuales, el front-end se encuentra libre de incrustaciones de código back-end (PHP-Java), separando su funcionalidad de las otras capas, lo realiza usando plantillas con algunas etiquetas incrustadas que hacen las llamadas o bindings²⁸ hacia el back-end.

Características de AngularJS:

Data-Binding

En las aplicaciones web, si en el backend cambia el valor de una variable, se verá reflejado en el frontend. Si el cambio surge en una variable del frontend, el valor de la variable automáticamente se actualizará sin tener que hacer llamadas extras.

MVC (Modelo Vista Controlador)

La estructura de AngularJS, obliga de manera ortodoxa a tener el código ordenado, haciendo uso de controladores para enviar datos a la vista, modelos para base de datos, directivas, servicios o filtros, para la inyección de dependencias²⁹.

AngularJS es un framework de código abierto en JavaScript, creado y soportado por Google, contiene un conjunto de librerías útiles para el desarrollo de aplicaciones web del lado del cliente, representa la "A" del stack MEAN.

²⁸ **Binding:** Es la sincronización automática de los datos entre los componentes del modelo y vista.

²⁹ **Inyección de dependencias:** Es un patrón de diseño, en el que se suministran o inyectan objetos a una clase en lugar de ser la propia clase quien cree el objeto.

1.7.4 NodeJS



Node.js es una herramienta JavaScript del lado del servidor utiliza el motor de JavaScript de Google, conocido como V8, provee de una arquitectura orientada a eventos, y un conjunto de librerías asíncronas (no-bloqueantes) que proporcionan un elevado rendimiento y escalabilidad. Representa la “N” del stack MEAN.

Socket.IO



Entre las características de Node.js, se destaca el procesamiento de conexiones masivas, alcanzada con la presencia de la librería Socket.IO³⁰, para aplicaciones de tiempo real, en otras palabras Socket.IO es una capa intermedia en la comunicación Cliente - Servidor que abstrae la manera en que se realiza la conexión (mecanismo de transporte) de la comunicación. La librería socket.IO pretende ofrecer un soporte universal de comunicación, independiente del agente de usuario integrando sus propias características. Socket.IO es residente en el cliente, al conectarse, decide qué modo de transporte es el que va a utilizar, entendiéndolo por modo de transporte a la tecnología o técnica empleada para realizar la conexión Cliente-Servidor, soportada por el agente de usuario, entre los que se pueden mencionar viejos navegadores o navegadores para móviles.

1.8 Callbacks manejadas por evento NodeJS

La programación estructurada persuade a los desarrolladores a pensar en el todo, en términos de procedimientos o funciones, y en las estructuras de datos que esos procedimientos manejan, es decir solo se escriben funciones que procesan datos.

Los programadores que emplean programación orientada a objetos, en cambio, primero definen objetos para luego enviarles mensajes solicitándoles que realicen sus métodos por sí mismos, es un medio de programación abstracto, más cercano a la vida real en comparación a otros tipos de programación, porque observamos las entidades como objetos con atributos y métodos los cuales nos permitirían modificar su estado.

³⁰ **Socket.IO:** Es una librería de JavaScript para aplicaciones web en tiempo real, consta de dos partes: una librería del lado del cliente ejecutándose en el navegador y una librería en el servidor para Node.js

La programación orientada a objetos, difiere de la programación estructurada, en que los datos y los procedimientos están separados y sin relación, ya que lo único que se busca es el procesamiento de unos datos de entrada para obtener otros de salida.

Por otro lado el modelo de programación por eventos usado por Node.js está orientada a desencadenar acciones, únicamente cuando se dispare un evento, como un click; entonces en este objeto se podría desplegar un mensaje o ejecutar diversas actividades. JavaScript del lado del servidor realmente no es tan diferente del lado del cliente, en verdad, no se están presionando botones, o ingresando texto en los campos de formulario, pero a un nivel superior, están sucediendo eventos. La Fig.3 muestra la codificación de servidor http básico.

```
1. var http = require ("http");
2. http.createServer( function (request, response){
3. response.writeHead(200,{"Content-Type":"text/html"});
4. response.write("Saludo Node.js");
5. response.end ();
6. }).listen(8001);
```

Fig. 3 Servidor HTTP básico
Fuente: (Kiessling & Jungle, 2012)

En la línea 2, la instrucción **http.createServer(function(request, response){**, crea una función allí mismo donde debería de estar el primer parámetro de la función `http.createServer()`.

function(request, response){, es el primer y único parámetro que le vamos a dar a las llamadas a la función `createServer()`, esta particularidad ofrece JavaScript, las funciones pueden ser pasadas de un lado a otro como si se tratase de un valor. La Fig.4, se muestra el paso de funciones.

```
1. function decir(palabra) {
2. console.log (palabra);
3. }
4. function ejecutar(algunaFuncion, valor)
5. algunaFuncion (valor);
6. }
7.ejecutar(decir, "Saludo");
```

Fig. 4 Paso de funciones
Fuente: (Kiessling & Jungle, 2012)

En la línea 1, se declara la función `decir()`, con un parámetro `palabra`, la función `decir` imprimirá con la instrucción `console.log()`, el valor pasado como parámetro.

En la línea 4, se declara la función `ejecutar`, la misma que lleva dos parámetros el primero es `algunaFuncion` y el segundo es `valor`.

En la línea 5, se muestra que el parámetro algunaFuncion() que ahora se trata de una función, lleva el parámetro valor.

Finalmente en la línea 7, se muestra la llamada a la función ejecutar (), la misma que lleva dos parámetros, el primero se trata de la función decir () de la línea 1, se trata de la propia función decir (), mas no de su valor de retorno.

Con esto la función decir (), se convierte en la variable local de algunaFuncion() dentro de ejecutar y ejecutar puede llamar a la función en esta variable usando algunaFuncion(){}.

JavaScript, permite pasar una función por su nombre como parámetro a otra función, sin tener que definir previamente la función, para luego pasarla.

JavaScript, brinda la posibilidad de pasar una función como un parámetro, cuando llamamos a otra función, asignando la función a una variable. Esta es la naturaleza misma de Node.js, está orientado por eventos.

Modelo de procesamiento Node.js

En el servidor http básico, representado por el siguiente código, se puede evidenciar , que en la línea 2, se crea el servidor http, y pasamos una función al método que lo crea en este caso function(req, res){...}, y cada vez que el servidor , reciba una petición, la función que le pasamos será llamada, tal como lo muestra la Fig. 3

```

1. var http = require ("http");
2. http.createServer( function (req, res){
3. res.writeHead(200,{"Content-Type":"text/html"});
4. res.write("Saludo Node.js");
5. res.end ();
6. }).listen(8001);

```

En la Fig. 5, se muestra el modelo de procesamiento de Node.js, desde las peticiones realizadas por el cliente, hasta la respuesta dada por el servidor.

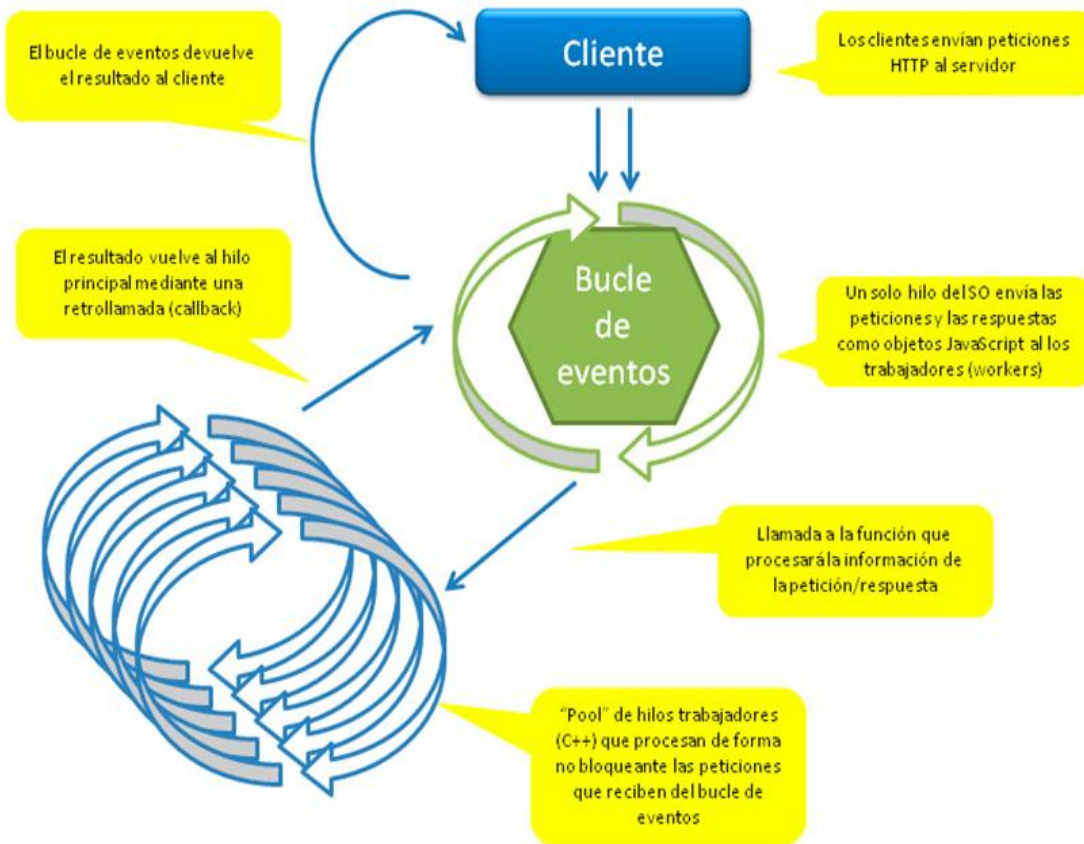


Fig. 5. Modelo de procesamiento de Node.js
Fuente: (Guillermo Mir, 2013)

Las peticiones HTTP son enviadas desde el cliente al servidor, un bucle de eventos es despertado por el sistema operativo, llama a la función que procesa la información de peticiones y las envía como objetos JavaScript a los workers (trabajadores).

El “pool” de hilo trabajadores (worker threads) scripts de C++, procesan de forma no bloqueante las peticiones recibidas del bucle de eventos.

El resultado es devuelto (objeto JavaScript), al hilo principal mediante una devolución de llamada (callback). El bucle de eventos devuelve el resultado al cliente.

1.9 Servidor HTTP básico

1. `var http = require ("http"31);`
2. `http.createServer(function (req, res){`
3. `res.writeHead(200, {"Content-Type": "text/html"});`

³¹ **HTTP**: Hypertext Transfer Protocol o HTTP (en español protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web.

```

4.   res.write("Saludo Node.js");
5.   res.end ();
6. }).listen(8001);

```

La línea 1, se crea una variable `http` que requiere el modulo “`http`”, incluido con `Node.js`, haciéndolo accesible a través de la variable local `http`.

En la línea 2, se llama a la función `createServer` que el modulo `http` ofrece, esta función retorna un objeto, y este objeto posee un método llamado `listen` (escuchar), tomando como parámetro el valor numérico que indica el número de puerto que el servidor `Http`, va a escuchar las peticiones.

En la línea 2, se crea una función anónima (sin nombre), `function(req, res){}`, la misma que tiene dos parámetros `req` y `res`. ***Req***, es el parámetro encargado de gestionar las peticiones, mientras que ***res*** es el encargado de las respuestas.

En la línea 3, la instrucción `response.writeHead(200,{"Content-Type":"text/html"})`, es la forma de responder en la escritura de cabecera con un status `HTTP 200`, respuesta normal y un tipo de contenido `text / html`.

En la línea 4, `res.write("Saludo Node.js")`, el navegador web responde con una respuesta en pantalla con “`Saludo Node.js`”

En la línea 5, `res.end ()`, se da por finalizada la respuesta. En la línea 6, `}).listen(8001)`, finalmente el servidor escuchara las peticiones por el puerto: `8001`

1.10 Funciones NodeJS

En las implementaciones de aplicaciones medianamente extensas, determinados procesos que se pueden concebir de forma independiente. Una función es una serie de instrucciones que englobamos dentro de un mismo proceso, este proceso se podrá luego ejecutar desde cualquier otro sitio con solo llamarlo. Las funciones se utilizan constantemente, no sólo las creadas, sino también las que ya están definidas en `JavaScript`. Una función se debe definir con la siguiente sintaxis:

```

function nombrefuncion (){
    instrucciones de la función
    .....
}

```

Primero se escribe la palabra `function`, la misma que es una palabra reservada. Seguida del nombre de la función, que deberá cumplir con la misma normativa de una variable, el mismo que puede contener números, letras y algún carácter adicional.

Para ejecutar una función se la puede invocar en cualquier parte de la página, consiguiendo ejecutar todas las instrucciones que tiene la función estarán entre las

llaves, para ejecutar la función utilizamos su nombre seguido de los paréntesis: **nombreDeLaFuncion()**

1.11 Módulos NodeJS

Modulo HTTP

Una vez realizada la instalación de Node.js en su versión 0.10.26, el módulo HTTP trabajara con el modulo del mismo nombre, el cual es utilizado en Internet para transferir datos en la Web.

Para efectos de una explicación didáctica se sugiere ir al path de instalación de la aplicación Node.js, dentro de la cual se deberá crear un archivo de nombre server, con la extensión .js de JavaScript, el cual deberá contener el siguiente script.

```
1. var http = require ("http");
2. http.createServer( function (req, res){
3.     res.writeHead(200,{"Content-Type":"text/html"});
4.     res.write("Saludo Node.js");
5.     res.end ();
6. }).listen(8001);
```

La línea 1, la siguiente instrucción: **var http= require("http")**, la cual hace una llamada al módulo "http". Ahora el archivo server.js, será transformado en un módulo, del cual se deberá exportar las partes de su funcionalidad, para otros módulos o scripts que requieran de su funcionalidad, su código es el siguiente:

```
1. var http = require("http");
2. function iniciar(){
3.     function onRequest(req, res){
4.         console.log("Petición recibida");
5.         res.writeHead(200,{"Content-Type":"text/html"});
6.         res.write("Primer saludo");
7.         res.end();
8.     }
9. http.createServer(onRequest).listen(8001);
10. console.log("Servidor iniciado");
11.}
12.exports.iniciar=iniciar;
```

En la línea 2, del código anterior se crea la función iniciar, la misma que tiene todas las instrucciones necesarias, para iniciar el servidor HTTP

La línea 9, se invoca la función createServer del módulo http, el mismo que tiene como único parámetro la función onRequest, seguida del puerto de escucha del servidor.

En la línea 12, se muestra la exportación de la función iniciar para que sea utilizada por otros módulos. Una vez creado el script server.js, se creara el archivo index.js, deberá ser guardado en el origen de la aplicación Node.js

1. var server = require (“./server”);
2. server.iniciar();

El script anterior, muestra la declaración de una variable server, que utiliza el modulo servidor tal como cualquier otro modulo interno, requiere el archivo donde este el contenido y lo asigna a la variable server, con sus respectivas funciones exportadas o disponibles.

Con los dos archivos .js, guardados en el origen de la aplicación Node.js, podemos arrancar la aplicación con nuestro archivo principal index.js.

Se deberá abrir una consola y posicionarse en path de origen de Node.js, donde se escribe la siguiente instrucción:

```
1.> node index.js
```

Con estas indicaciones se puede evidenciar, que se puede poner diferentes partes de una aplicación en archivos diferentes y enlazarlas juntas a través de la creación de módulos.

Módulo URL

El módulo url, posee utilidades para la resolución y análisis del localizador uniforme de recursos (URL). Para el análisis de objetos URL intervienen uno o varios campos, de los que están formados la cadena URL, para tener presente una apreciación práctica se analizara la presente dirección URL.

<http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5>

protocol: El protocolo de la petición **request**.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

host: Una parte del host de la URL, incluye la información del puerto y autenticación.

http://**user:pass@host.org:80001**/p5/a5/t5/h5?query=string#hash5

auth: Parte de la información de autenticación de la URL.

http://**user:pass**@host.org:80001/p5/a5/t5/h5?query=string#hash5

hostname: Sólo la parte del nombre del host.

http://user:pass@**host.org**:80001/p5/a5/t5/h5?query=string#hash5

port: El número de puerto del host.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

pathname: La sección de la ruta de la URL, que viene después del host y antes de la consulta, incluyendo la barra inicial en caso de existir.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

search: La sección de la *cadena de consulta* de la URL, incluyendo el signo de interrogación principal.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

query: La sección de cualquier *parámetro* de la cadena de consulta, o un parser (analizador) de un objeto de cadena de consulta.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

hash: La parte del **fragmento** de la URL incluyendo el símbolo #.

http://user:pass@host.org:80001/p5/a5/t5/h5?query=string#hash5

1.12 Ruteo de peticiones NodeJS

Una vez creado el servidor HTTP básico, podemos recibir peticiones HTTP, ahora es necesario que las peticiones se dirijan a diferentes partes de nuestro código, se lo denomina ruteo.

Entre las condiciones iniciales el ruteo el servidor debe ser capaz de entregar la URL³² requerida y los posibles parámetros GET³³ o POST³⁴, adicionales al router, y basados en estos el router deberá ser capaz de decidir que código ejecutar

Una vez recibidas las peticiones HTTP, se deberá extraer la URL, requerida así como también los parámetros GET / POST.

Toda la información necesaria esta en el objeto request, el que es pasado como primer parámetro a nuestra función callback onRequest(). Para la interpretación de esta información son necesarios algunos módulos adicionales de Node.js.

El módulo url, provee métodos que nos permiten extraer las diferentes partes de la URL, como la ruta requerida o el string de consulta.

El módulo querystring, puede ser usado para parsear el string de consulta, para uno o varios parámetros solicitados. (Kiessling & Jungle, 2012)

³² **URL:** Un localizador de recursos uniforme, denominado URL, es una secuencia de caracteres, de acuerdo a un formato modélico y estándar, se usa para nombrar recursos en Internet para su localización o identificación, como documentos, imágenes o vídeos.

³³ **GET:** Es el método HTTP utilizado en el submit de los formularios, pide al servidor que le envíe un recurso

³⁴ **POST:** Es el método HTTP utilizado en el submit de los formularios, envía datos al servidor para ser procesados por el recurso especificado en la petición, los datos se incluyen en el cuerpo de la petición.

Ahora se debe de agregar a la funcion `onRequest()`, la lógica requerida para encontrar la ruta URL, que el browser solicito.

```

1. var http = require("http");
2. var url = require ("url");
3. function iniciar(){
4.     function onRequest(request, response) {
5.         var pathname=url.parse(request.url).pathname;
6.         console.log("Petición para: "+ pathname + " recibida.");
7.         request.writeHead(200,{"Content-Type":"text/html"});
8.         request.write("Bienvenido al navegador");
9.         request.end();
10.    }
11. http.createServer(onRequest).listen(8001);
12. console.log("Servidor iniciado");
13. exports.iniciar= iniciar;

```

Ahora la aplicación puede distinguir peticiones basadas en la URL requerida. Para la implementación del router que se acoplara débilmente como una inyección por dependencia.

El contenido del `router.js` será el siguiente:

```

1. function route(pathname){
2.     console.log("Rutear una petición para: "+pathname);
3. }
4. exports.route= route;

```

El siguiente script demuestra el código de servidor: `server.js`

```

1. var http = require ("http");
2. var url= require("url");
3. function iniciar (route){
4.     function onRequest(request, response){
5.         var pathname= url.parse(request.url).pathname;
6.         console.log("Petición: "+pathname+ " recibida");
7.         route(pathname);
8.         response.writeHead(200,{"Content-Type":"text/html"});
9.         response.write("saludo del navegador");
10.        response.end();
11.    }
12. http.createServer(onRequest).listen(8001);
13. console.log("servidor iniciado");
14. }
15. exports.iniciar= iniciar;

```

En las línea 3, del código anterior se observa que la función iniciar tiene como único parámetro a route, el mismo que es una función, que es invocada en la línea 7, la cual la pasamos como parámetro el atributo **pathname** de la petición.

Script del archivo index.js

```
1. var server = require (“./server”);  
2. var router = require (“./router”);  
3. server.iniciar (router.route);
```

En la línea 1 y 2 del script index.js, se realiza la invocación de los módulos necesarios para ejecutar la aplicación y para extender el index.js adecuadamente, se inyecta la función de ruteo del router en el servidor.

Para la ejecución simplemente se abre una consola y se posiciona directamente en el path de origen de Node.js y se agrega las siguientes instrucciones.

```
>node index.js
```

A lo cual Node.js responde de la siguiente manera:

```
>node index.js  
>servidor iniciado  
>Petición: /subir recibida  
>Rutear una petición para: /subir
```

El servidor HTTP y el router de peticiones, ahora pueden comunicarse, pero rutear, significa manipular las peticiones a distintas URLs de manera diferente.

Por ahora el ruteo termina en el router, si se incrementara la complejidad de la aplicación se deberán de implementar funciones donde estas peticiones sean direccionadas, dichas funciones serán denominadas de ahora en adelante manipuladores de peticiones o requestHandlers.

Script requestHandlers.js

```
1. function arrancar ( ){  
2.     console.log(“Handler de petición arrancar ha sido invocado”);  
3. }  
4. function montar ( ){  
5.     console.log(“Handler de petición montar ha sido invocado”);  
6. }  
7. exports.arrancar = arrancar;  
8. exports.montar = montar;
```

Ahora se cuenta solo con dos manipuladores de petición, pero en una aplicación real, este valor se incrementara variablemente.

Para que los manipuladores de petición no estén aislados, se los deberá pasar desde el servidor al router y para mapear las peticiones a manipuladores cada vez que una URL sea agregado, sería muy tedioso.

Para cada petición se debe tener en cuenta que un número variable de ítems, cada uno de ellos mapeados a un string, siendo necesario el uso de arrays. JavaScript a diferencia de otros lenguajes de programación orientados a objetos, trata a los objetos como colecciones de pares id/ valor, donde el valores pueden ser números, string o funciones.

Script de index.js (Manipuladores como objetos del router)

```

1. var server = require (“./server”);
2. var router = require (“./router”);
3. var requestHandlers = require (“./requestHandlers”);
4. var handle = { };
5. handle [“/”]= requestHandlers.iniciar;
6. handle [“/iniciar”] = requestHandlers.iniciar;
7. handle[“/subir”]= requestHandlers.subir;
8. server.iniciar(router.route,handle);

```

Después de definir el objeto (id/ valor) se lo deberá pasar al servidor como un parámetro adicional, pero se debe redefinir el router (router.js).

Script de router.js

```

1. function route (handle,pathname){
2.     console.log(“Pronto será ruteada la petición;” + pathname);
3.     if(handle [pathname]){
4.         return handle[pathname]();
5.     }
6.     else{
7.         console.log(“Manejador: ”+pathname+”desconocido”);
8.         return “error 404 Manejador no Encontrado”;
9.     }
10.}

```

En el router la línea 1, se muestran ahora dos parámetros handle y pathname, este último es una cadena de caracteres que corresponde al pathname y el objeto handle.

En la línea 3, se pregunta si handle[pathname], cuenta con algún valor , en el caso de ser verdadero simplemente se retorna el valor de handle[pathname](), de lo contrario se retornara un mensaje indicando que no se ha encontrado un manejador de peticiones para ese pathname.

Script de server.js

```

1. var http = require(“http”);
2. var url = require (“url”);

```

```
3. function iniciar (route , handle) {
4.     function onRequest(request, response){
5.         var pathname = url.parse(request.url).pathname;
6.         console.log("Petición para: "+pathname+" recibida");
7.         route(handle,pathname);
8.         response.writeHead(200,{"Content-Type":"text/html"});
9.         response.write("Bienvenido al navegador");
10.        response.end();
11.    }
12. http.createServer(onRequest).listen(8001);
13. console.log("servidor iniciado");
14.}
15.exports.iniciar=iniciar;
```

Ahora se abre una consola, se deberá posesionar en la ubicación de origen de la aplicación node.js y se deberá tipear.

>node index.js

Se mostrara en pantalla

>servidor iniciado

Se abre una ventana del navegador y se coloca en la dirección url lo siguiente

localhost: 8001/iniciar

En consola se mostraran los siguientes mensajes

>servidor iniciado

>Petición: /iniciar recibida

>A punto de rutearse petición: /iniciar

>Manejador de petición INICIAR invocada

Una vez citados los conceptos de callbacks, el servidor HTTP Básico, funciones, módulos, ruteos y manejadores (requestHandlers), se puede concluir que se han estudiado los principios fundamentales de Node.js, destinados a la implementación de la aplicación TRANSUR, descrita en el capítulo III.

1.13 Metodología XP



Extreme Programming

Es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck³⁵, es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos procesos de desarrollo, la programación extrema se diferencia de las metodologías tradicionales principalmente en enfatizar constantemente la adaptabilidad sobre la previsibilidad de requerimientos de software.

Los seguidores de Extreme Programming, consideran que los cambios de requisitos sobre la marcha es un aspecto natural e inevitable del desarrollo de proyectos. Creen que la capacidad de adaptarse a los cambios de requisitos en la vida del proyecto es una aproximación realista que los intentos de definir todos los requisitos al inicio del proyecto e invertir esfuerzos dirigidos al control de cambios de requisitos.

Se puede considerar a la metodología de programación extrema como la adopción de las mejores prácticas de las metodologías de desarrollo, aplicadas de manera dinámica durante el ciclo de vida del software.

1.13.1 Roles XP

Programador

- Pieza básica en desarrollos con la metodología XP.
- Responsable sobre el código de las aplicaciones.
- Responsable sobre el diseño (refactorización, simplicidad).
- Responsable sobre la integridad del sistema (pruebas).
- Código colectivo (Aceptación de críticas).

Cliente

- Pieza básica en desarrollos con la metodología XP.
- Define especificaciones de la aplicación.
- Influye sin controlar (Usuario de la aplicación).
- Define las pruebas funcionales.

Encargado de Pruebas (Tester)

- Apoya al cliente en la preparación / realización de las pruebas funcionales.
- Ejecuta las pruebas funcionales y publica los resultados.

³⁵ **Kent Beck:** Es un ingeniero de software estadounidense, uno de los creadores de las metodologías de desarrollo de software de programación extrema (eXtreme Programming o XP)

- Encargado de Seguimiento (Tracker).
- Analiza y publica información sobre la marcha del proyecto sin afectar el proceso de desarrollo.
- Vigila el cumplimiento de las estimaciones en cada iteración.
- Informa sobre la marcha de la iteración en curso.
- Controla la ejecución y marcha de las pruebas funcionales, los errores reportados, las responsabilidades aceptadas y las pruebas agregadas por los errores encontrados.

Entrenador (Coach)

- Experto en la metodología de desarrollo XP.
- Responsable del proceso en su entorno conjunto.
- Identifica las desviaciones y reclama atención sobre las mismas.

Consultor

- Apoya al equipo XP en soluciones objetivas de proyectos de mediano y alto alcance.

Jefe del Proyecto

- Fortalece la relación entre usuarios y desarrolladores.
- Afianza el alcance de los objetivos.

1.13.2 Valores XP

Los valores de la metodología de programación extrema son: simplicidad, comunicación y retroalimentación; el respeto, fue añadido en la segunda edición de Extreme Programming.

Simplicidad

Es necesario simplificar el diseño para agilizar el desarrollo y el mantenimiento del software, un diseño complejo del código acompañado de sucesivas modificaciones de diferentes desarrolladores hacen que la complejidad aumente progresivamente.

Para mantener la simplicidad, es necesaria la refactorización³⁶ del código fuente³⁷, ésta es la manera de mantener el código simple a medida que crece, garantizando el conocimiento de todo el sistema por parte del equipo de desarrollo.

³⁶ **Refactorización:** Es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

³⁷ **Código fuente:** El código fuente de un programa informático (o software) es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa

Comunicación

Si el código fuente es sencillo ofrecerá una comunicación fluida entre los desarrolladores, mientras que si el código es complejo hay que esforzarse para hacerlo inteligible³⁸. El código autodocumentado es más fiable que los comentarios en el código ya que éstos últimos, quedan desfasados cuando el código es modificado. Para la comunicación el cliente decide qué características tienen prioridad y siempre debe estar disponible para solucionar dudas.

Retroalimentación (feedback)

El cliente siempre deberá estar integrado en el proyecto, su opinión refleja el estado real del proyecto, al realizarse ciclos cortos que ofrecen resultados reales, se deja de mantener partes del código que no cumplan con los requerimientos de la aplicación.

El código es una fuente de retroalimentación gracias a las herramientas de desarrollo. Por ejemplo, las pruebas unitarias informan sobre la situación del código, con la ejecución de pruebas unitarias se descubren fallos ocasionados por cambios recientes en el código.

Coraje

Todas las prácticas de la metodología de desarrollo implican valentía, una de ellas es “siempre diseñar y programar para hoy y no para mañana”. (Castillo, Figueroa, & Sevilla). Es la inversión del esfuerzo para evitar empantanarse en el diseño que requiere de demasiado tiempo y trabajo para implementar el proyecto en su totalidad. Permitiendo a los desarrolladores sentirse cómodos reconstruyendo su código cuando sea necesario, lo que implica revisar el sistema existente y modificarlo de manera que los cambios futuros se implementaran fácilmente.

Respeto

Las partes del equipo se respetan, por el trabajo colectivo, una apreciación base sería entre los encargados de las pruebas funcionales y los desarrolladores porque los programadores no pueden realizar cambios que hacen que las pruebas existentes fallen o que retrasen el trabajo de sus compañeros. Los miembros respetan su trabajo al estar buscando constantemente la alta calidad en el producto del cliente.

³⁸ **Inteligible:** Es una propiedad de las variedades lingüísticas o dialectos por la cual dos hablantes de variedades diferentes pueden comprenderse mutuamente sin haber estudiado o aprendido previamente la variedad ajena.

1.13.3 Fases de la metodología XP

Primera Fase: Planificación del proyecto

Historias de usuario: El primer paso de la metodología Programacion Extrema, es definir las historias de usuario con el cliente, para este caso historias de usuario poseen la misma finalidad que los casos de uso pero con algunas disimilitudes, constando principalmente de pocas líneas de texto escritas por el cliente en un lenguaje no técnico, siendo utilizadas para estimar los tiempos de desarrollo de la parte de la aplicación descrita.

Al implementar las historias de usuario, el cliente y los desarrolladores se reúnen para concretar y detallar lo que debe hacer cada historia, determinando el tiempo estimado de desarrollo para una historia de usuario esta entre una y tres semanas.

Release planning: Con las historias de usuario definidas es necesario crear un plan de publicaciones, (Release plan), donde se indicaran las historias de usuario que se crearán para cada versión del programa y las fechas en las que se publicarán estas versiones.

Iteraciones: Los proyectos se dividirán en iteraciones estimadas de diez semanas de duración, al iniciar cada iteración los clientes deben seleccionar las historias de usuario definidas en el "Release planning" que serán implementadas. También se seleccionan las historias de usuario que no pasaron el test de aceptación. Al terminar la iteración anterior, estas historias de usuario serán divididas en tareas estimadas entre uno y seis días de duración y serán asignadas a los programadores.

Velocidad del proyecto: Es una medida que representa la rapidez con la que se desarrolla el proyecto; para estimarla basta con contar el número de historias de usuario que se pueden implementar en una iteración; de esta forma, se sabrá el cupo de historias que se pueden desarrollar en las distintas iteraciones.

Programación en pareja: Es aconsejable la programación en parejas para incrementar la productividad y la calidad del software desarrollado; mientras uno de los desarrolladores codifica una función o método, el otro analiza si ese método o función es adecuado, consiguiendo un código y diseño de alta calidad.

Reuniones diarias. Es necesario que los participantes del desarrollo se reúnan diariamente y expongan sus problemas, soluciones e ideas de forma conjunta.

Segunda Fase: Diseño

Diseños simples: La generación de diseños simples y sencillos, entendibles e implementables a la larga reducirán el tiempo y esfuerzo en el desarrollo de la aplicación.

Glosarios de términos: El uso de glosario de términos y una correcta especificación de nombres de métodos y clases aportaran con una fácil comprensión del diseño, las posibles ampliaciones de la aplicación y la reutilización del código.

Refactorizar: Es modificar la estructura y codificación del código existente, sin alterar su funcionalidad actual, lo que supone revisar de nuevo estos códigos para procurar optimizar su funcionamiento.

Tercera fase: Codificación

El cliente es parte del equipo de desarrollo; su presencia es indispensable en las distintas fases de desarrollo con la metodología de Programación Extrema.

Antes de codificar cada historia de usuario, el cliente debe especificar su funcionalidad y verificar por medio de los test que la historia implementada cumpla con las funcionalidades indicadas. La creación de los test, indican con exactitud las actividades que deberá realizar el código a implementar, creando inicialmente los test para cada unidad consiguiendo el cumplimiento de todos los requisitos especificados.

La metodología sugiere la utilización de repositorios de código donde las duplas de programadores publiquen diariamente sus códigos implementados y corregidos junto a los test superados, de manera que los otros programadores que necesiten códigos ajenos trabajen con las últimas versiones publicadas manteniendo un código consistente. El permitir a los programadores ajenos modificar códigos, no supone ningún riesgo ya que para que un código pueda ser publicado en el repositorio tiene que pasar los test de funcionamiento.

Cuarta Fase: Pruebas

El uso de test para comprobar la funcionalidad de los códigos es esencial en la metodología de desarrollo de programación extrema. Se crearan las aplicaciones que realizarán los test en un entorno de desarrollo específico para cada test, sin poseer ningún tipo de dependencia con el código que en el futuro se evaluará, asegurando la independencia del test respecto al código que evalúa.

Ningún código puede ser publicado en el repositorio sin que haya pasado su test de funcionamiento, asegurando el uso colectivo del código. El uso de los test es adecuado para la refactorización, permitiendo verificar que un cambio en la estructura de un código no lleva consecuencia alguna sobre su funcionamiento.

Test de aceptación

Los test sirven para evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para garantizar el funcionamiento final de una determinada historia de usuario

existen los "Test de aceptación"; los mismos que son usados para comprobar que las historias de usuario cumplan su cometido (Castillo, Figueroa, & Sevilla).

Al ser las distintas las funcionalidades de la aplicación, se realizarán pruebas para las funcionalidades generales que debe consumir el programa especificado en la descripción de requisitos.

1.13.4 Ventajas y desventajas de la metodología XP

Según: Daniel Zambrano las principales ventajas y desventajas de la metodología de desarrollo de Programación Extrema son:

Ventajas

- La codificación de las aplicaciones es organizada.
- Mínima tasa de errores de codificación.
- Altos niveles de desempeño en programadores.
- Implementa una forma de trabajo adaptable a diversas circunstancias.

Desventajas

- Es recomendable emplearlo solo en proyectos a corto plazo.
- Producen altas comisiones en caso de fallos (Danielzs75.blogspot.com, 2012).
- Imposible prever todo antes de programar.

CAPÍTULO II

Estudio de factibilidad

- 2.1 Antecedentes
- 2.2 Objetivos
 - 2.2.1 Objetivo diagnóstico
 - 2.2.2 Objetivos Específicos
- 2.3 Variables
- 2.4 Indicadores
- 2.5 Matriz relación
- 2.6 Selección de la muestra
- 2.7 Análisis e interpretación de resultados
- 2.8 Conclusiones del estudio

2.1 Antecedentes

La cooperativa de transporte de pasajeros Urcuquí opera bajo la modalidad de pasajeros y tipo de transporte inter cantonal, haciéndose acreedora a su personería jurídica mediante el acuerdo ministerial Nro. 000795 del 27 de Julio de 1983.

La institución ofrece a la colectividad los servicios de movilización de pasajeros y encomiendas entre los cantones Ibarra y Urcuquí de la provincia de Imbabura, fortaleciendo de esta manera el intercambio comercial, cultural y social de sus habitantes.

La cooperativa de transporte inter cantonal Urcuquí, con el afán de mejorar su productividad en sus servicios internos, prevé la necesidad de implementar una solución informática que sea de fácil acceso y de baja inversión para optimizar el proceso logístico de revisiones vehiculares, administración de socios, conductores, unidades de transporte y servicio de las unidades de transporte.

2.2 Objetivos

2.2.1 Objetivo diagnóstico

Conocer la situación actual de la productividad interna de la cooperativa de transporte de pasajeros inter cantonal Urcuquí.

2.2.2 Objetivos Específicos

Efectuar un análisis de la capacidad de registro de unidades de transporte, rutas, frecuencias, socios, conductores y boletería de la cooperativa de transporte de pasajeros inter cantonal Urcuquí.

Establecer los elementos en los que se basa la productividad interna de la cooperativa de Transporte de pasajeros inter cantonal Urcuquí.

2.3 Variables

Tomando en consideración la variable que influye directamente en la implementación de una aplicación informática encargada de optimizar el proceso de registro de las unidades de transporte, rutas, frecuencias, socios, conductores y boletería de la

cooperativa de transporte de pasajeros intercantonal Urcuquí, se destaca la variable de productividad interna.

2.4 Indicadores

Generalmente los indicadores se encuentran asociados a problemas de tráfico o la distribución física de las unidades de transporte, los mismos que están orientados a satisfacer de las necesidades de los usuarios.

Para el caso particular de la Cooperativa de transporte intercantonal Urcuquí, la administración actual, esta eventualmente preocupada por mejorar su propia productividad, es decir, por mejorar al interior de la institución.

En este contexto según: John Dolce³⁹, ofrece una lista de indicadores de productividad de una flota de transporte, en los que, según una división funcional de la empresa transportista se tienen:

- Indicadores globales
- Indicadores en la oficina
- Indicadores en andenes de maniobras
- Indicadores en ruta. (Dolce, 1984)

De los anteriores indicadores el más relevante, para la continuidad del presente proyecto es el indicador en la oficina. (Islas, Torres, & Rivera, 2000)

Para valorar la productividad de las oficinas de la empresa de transporte dicho Jhon Dolce sugiere el indicador de Solicitudes atendidas / hora.

Técnicas : Encuestas, Entrevistas

Fuentes de información

Primaria: Encuestas

Secundaria: Libros, documentación en la web

2.5 Matriz relación

Con el propósito de identificar las diferentes relaciones existentes se describe en el cuadro a continuación: objetivos, variables, indicadores, técnicas y fuentes de investigación, para establecer las interrelaciones directas que se dan entre estos aspectos.

³⁹ **John Dolce:** Es el autor del libro: Fleet Management. McGraw-Hill Book Co.

Tabla 3: Matriz relación

OBJETIVOS DIAGNÓSTICOS	VARIABLES	INDICADORES	TÉCNICAS	FUENTES DE INFORMACIÓN
Establecer los elementos en los que se basa la productividad interna en las oficinas de la cooperativa de Transporte de pasajeros inter cantonal Urcuquí.	P R O D U C T I V I D A D	Capacidad de respuesta a las Solicitudes de información por hora	Encuestas y entrevistas	Presidencia Gerencia Socios
Efectuar un análisis de la capacidad de registro de unidades de transporte, rutas, frecuencias, socios y conductores de la cooperativa de transporte de pasajeros inter cantonal Urcuquí.		Capacidad de registro de información por hora	Encuestas y entrevistas	Secretaría

Fuente: Personal

2.6 Selección de la muestra

Para la elección representativa del conjunto de individuos implicados en el proceso administrativo de la cooperativa es decir la población, será sometida a una evaluación estadística con la intervención del muestreo. La población estimable para el presente proyecto esta mostrada en la Tabla 4.

Tabla 4: Población estimada

No	Cargo	Cantidad
1	Presidente (socio)	1
2	Gerente (socio)	1
3	Asesoría jurídica(contratado)	1
4	Contador (Contratado)	1
5	Socios	36
6	Personal administrativo	2
7	Conductores	11
Total		53

Fuente: Personal

Para la población real se estima que la asesoría jurídica y el contador no forman parte, de la cooperativa y se asume que la presidencia y gerencia están integradas por socios de la cooperativa, de manera que la población real se muestra en la Tabla 4.

Tabla 5: Población

No	Cargo	Cantidad
1	Presidente (socio)	1
2	Gerente (socio)	1
3	Socios	34
4	Personal administrativo	2
5	Conductores	11
Total		49

Fuente: Personal

Muestra

Para determinar la muestra para el presente proyecto se cita el siguiente criterio técnico citado por el M.S.c. Walter Jácome, “Si la población es menor o igual a 50, se trabajara con censo, en el cual se evaluarán a todos los integrantes de la población”. (Msc Walter Jacome, 2009). Por lo cual se realizara un censo entre todas las personas afines de la institución.

2.7 Análisis e interpretación de resultados

Se elaboran los contenidos de los formularios de las encuestas y entrevistas los cuales están orientados a definir, la capacidad de respuesta a las peticiones de información, y la influencia de los medios de comunicación actualmente utilizados para la difusión de resultados de información de la Cooperativa de transporte de pasajeros intercantonal Urcuquí.

Los formularios de las encuestas y entrevistas están formadas de 6 preguntas cada una las cuales interpretaran las inquietudes y expectativas de las personas afines a la institución sobre la difusión de información institucional.

Con el análisis de los resultados se determina la consolidación de los ítems de los formularios y la interpretación determina el resultado de cada análisis, lo cual determinara una conclusión general del estudio de factibilidad.

Encuesta Socios - conductores

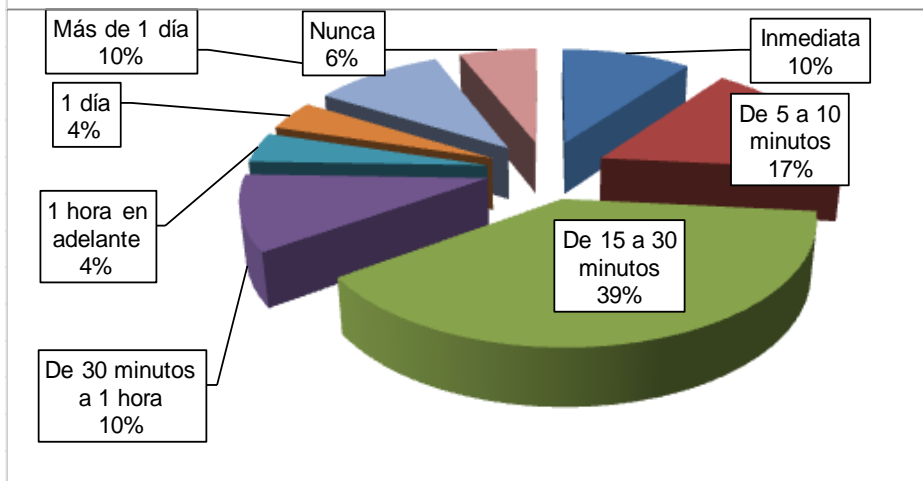
1. Cuando usted realiza una petición de información a secretaría, sobre información interna de la cooperativa como: socios, conductores, rutas y frecuencias de unidades de transporte, estas peticiones son contestadas de forma:

- a) Inmediata
- b) De 5 a 10 minutos
- c) De 15 a 30 minutos
- d) 30 minutos a 1 hora
- e) 1 hora en adelante
- f) 1 día
- g) Más de 1 día
- h) Nunca

Tabla 6: Pregunta1 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
Inmediata	5	10,20%
De 5 a 10 minutos	8	16,33%
De 15 a 30 minutos	19	38,78%
De 30 minutos a 1 hora	5	10,20%
1 hora en adelante	2	4,08%
1 día	2	4,08%
Más de 1 día	5	10,20%
Nunca	3	6,12%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 1. Pregunta 1 Socios-Conductores

Fuente: Personal

Análisis: De los 49 socios – conductores que constituyen la muestra, 19 que corresponde al 38,78%, reciben la respuesta a su petición de información en un tiempo estimado de 15 a 30 minutos, 8 que corresponde al 16,33 % , en un tiempo estimado de 5 a 10 minutos, 5 que corresponde al 10,20 % reciben una respuesta inmediata es decir entre 1 y 3 minutos, 5 que corresponden al 10,20%, reciben una respuesta de tiempo estimado entre 30 minutos a 1 hora, 2 que corresponde al 4,08%, reciben una respuesta en un periodo de una hora en adelante, 2 que corresponden al 4,08 % , reciben una repuesta en un periodo de 1 día en adelante, 5 que corresponden al 10,20% , reciben una respuesta en un periodo mayor de 1 día, y 3 que corresponden al 6,12% , no reciben respuesta a su petición.

Interpretación:

Con relación a la pregunta Nro. 1, de la encuesta realizada a socios-conductores se determina que las respuestas a las peticiones de información realizadas por socios o conductores en su mayoría tardan de 15 a 30 minutos, en el mejor de los casos ya que en varias oportunidades las respuestas llevan mayor tiempo y en ocasiones no son contestadas.

2. Cuando usted necesita realizar una petición de información interna de la cooperativa sobre: conductores, rutas y frecuencias de unidades de transporte, en horas NO LABORABLES O FERIADOS, recurre a:

a) Otros socios

b) Otra forma

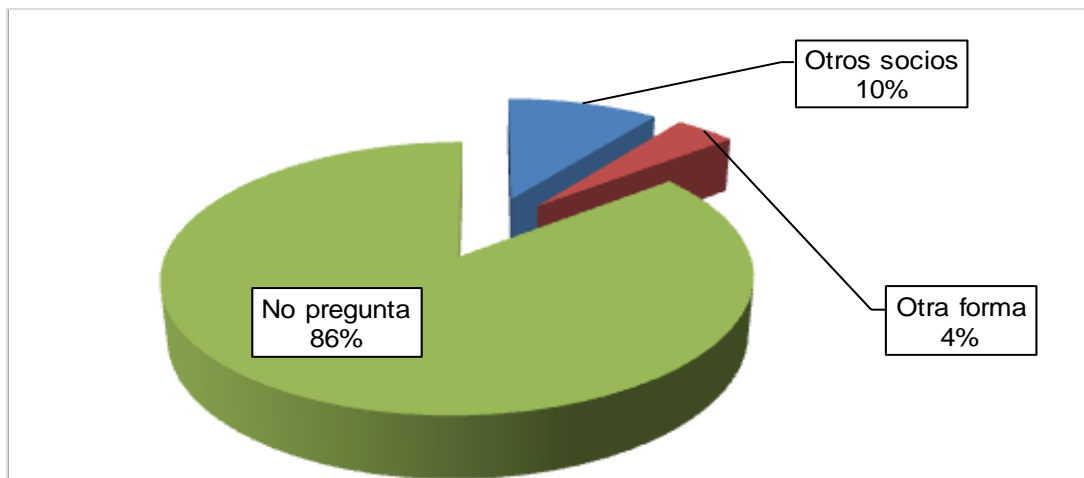
¿Cuál?.....

c) NO pregunta

Tabla 7: Pregunta 2 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
Otros socios	5	10,20%
Otra forma	2	4,08%
No pregunta	42	85,71%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 2. Pregunta 2 Socios-Conductores

Fuente: Personal

Análisis:

De los 49 socios – conductores que constituyen la muestra, 42 de ellos que corresponde al 85,71%, NO realizan ninguna pregunta, 5 que corresponden al 10,20 %, solicitan información a otros compañeros, 2 que corresponde al 4,08 % realizan las peticiones de información de otra forma.

Interpretación:

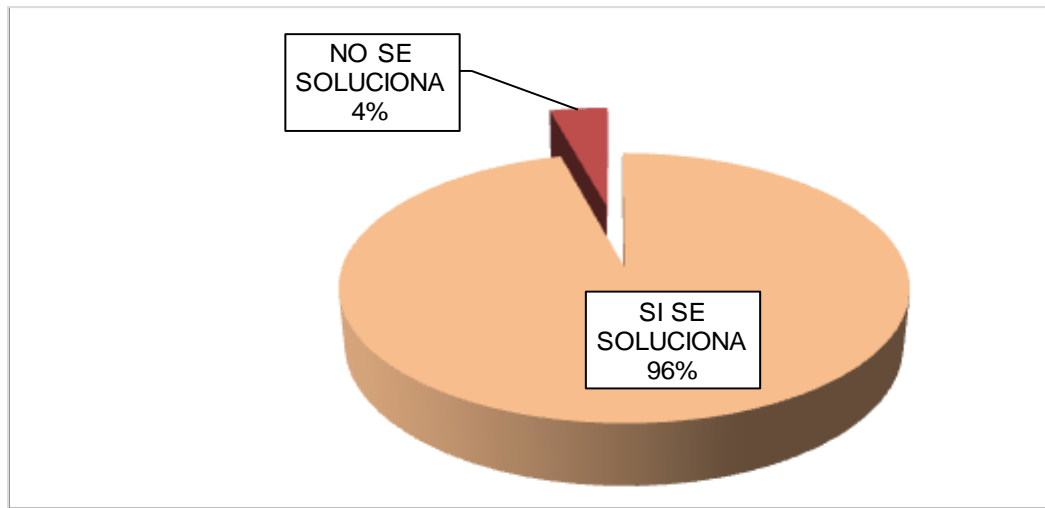
Con relación a la pregunta Nro. 2, de la encuesta realizada a socios-conductores se determina la mayor parte de socios-conductores NO realiza ninguna consulta, aun cuando sea de vital importancia para la cooperativa.

3. ¿Este inconveniente se solucionaría, si se tuviera la información a su alcance? SI-NO.

Tabla 8: Pregunta 3 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
SI SE SOLUCIONA	47	95,92%
NO SE SOLUCIONA	2	4,08%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 3. Pregunta 3 Socios-Conductores

Fuente: Personal

Análisis:

De los 49 socios – conductores que constituyen la muestra, 47 de ellos que corresponde al 95,92%, afirman que si tuvieran la información a su alcance no recurrirían a otro medio de información, 2 que corresponden al 4,08 %, afirman que no se solucionarían la necesidad de información aun cuando la información estuviese a su alcance.

Interpretación:

Con relación a la pregunta Nro. 3, de la encuesta realizada a socios-conductores se determina que la mayor parte de socios-conductores afirman que se eliminaría la necesidad de información de la cooperativa, si la información estuviese a su alcance.

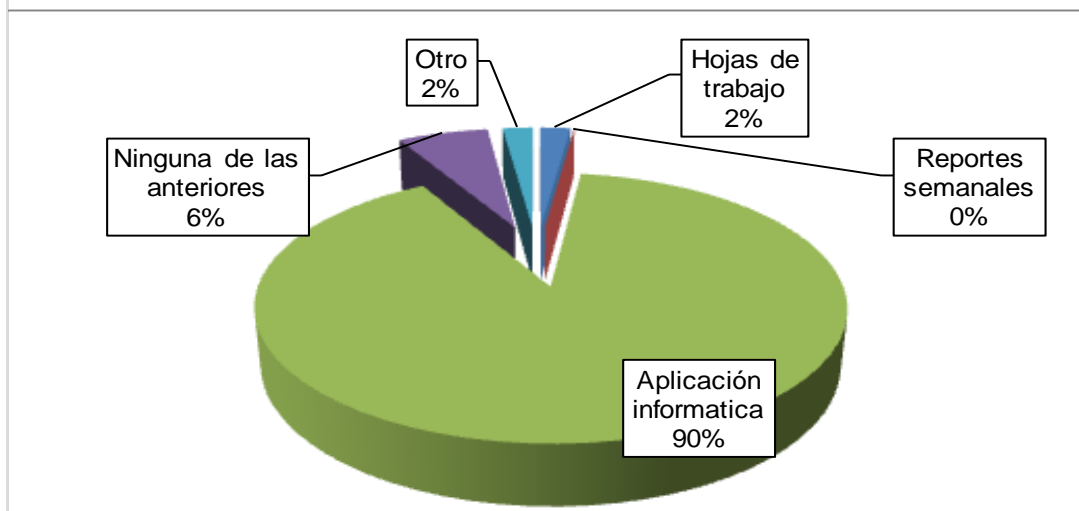
4. ¿Qué medio de información, sería el más fiable para garantizar información actualizada y permanente (24 horas)?.

- a) Hojas de trabajo
- b) Reportes semanales de secretaría
- c) Aplicación informática en Internet (Software)
- d) Ninguna de las anteriores
- e) Otro ¿Cuál medio?

Tabla 9: Pregunta 4 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
Hojas de trabajo	1	2,04%
Reportes semanales	0	0,00%
Aplicación informatica	44	89,80%
Ninguna de las anteriores	3	6,12%
Otro	1	2,04%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 4. Pregunta 4 Socios-Conductores

Fuente: Personal

Análisis:

De los 49 socios – conductores que constituyen la muestra, 44 de ellos que corresponde al 89,80%, afirman que el medio de información más fiable para mantener la información actualizada y permanente es la aplicación informática en la web, 3 que corresponden al 6,12 %, descartan que los medios más fiables para mantener la información actualizada, sean las hojas de trabajo, reportes semanales y las aplicaciones informáticas en la web, 1 que corresponde al 2,04 %, afirma que las hojas de trabajo, son el medio más fiable para mantener la información, mientras que 1 socio-conductor que corresponde al 2,04 %, afirma que otros son los medios más fiables para mantener la información de forma permanente y actualizada.

Interpretación:

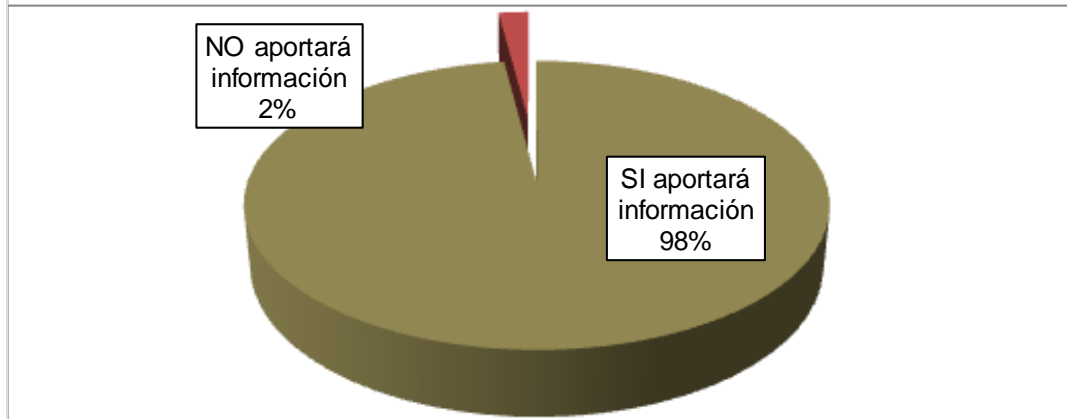
Con relación a la pregunta 4, el 89,80 % de los socios afirman que el medio para mantener la información actualizada y permanente sería una aplicación web alojada en internet.

5. En el caso de que la respuesta anterior fuese: c) Aplicación informática en Internet. ¿Usted puede afirmar que una Aplicación Informática (TRANSUR) alojada en internet con información interna de la cooperativa sobre unidades de transporte, socios conductores, rutas y frecuencias, con reportes o presentaciones de información actualizada, sustentara o apoyara la gestión de respuestas, a sus peticiones de información de forma permanente a cualquier hora y día de la semana? SI o NO.

Tabla 10: Pregunta 5 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
SI aportará información	48	97,96%
NO aportará información	1	2,04%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 5. Pregunta 5 Socios-Conductores

Fuente: Personal

Análisis:

De los 49 socios – conductores que constituyen la muestra, 48 de ellos que corresponde al 97,96%, afirman que una aplicación web aloja en internet apoyara/sustentara la peticiones de información interna de la cooperativa, mientras que 1 socio conductor que

corresponde al 2,04%, afirma que una aplicación web alojada en internet, no aportara en nada a las peticiones de información realizadas.

Interpretación:

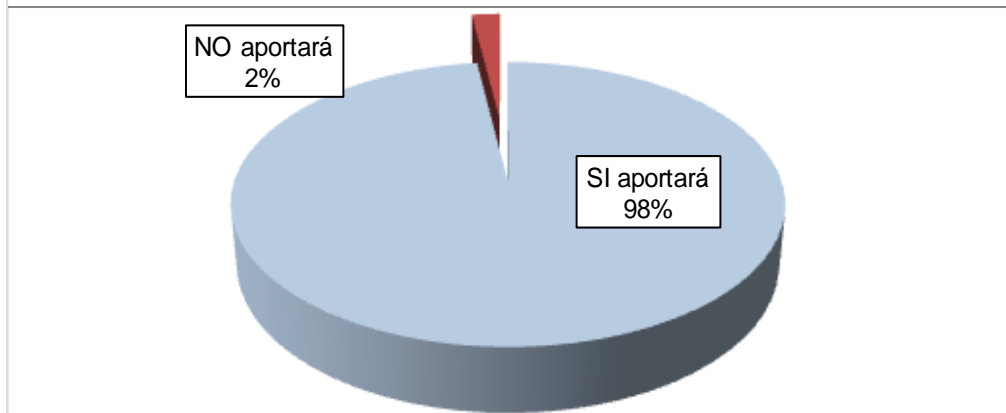
Con relación a la pregunta Nro. 5, de la encuesta realizada a socios-conductores se determina que la mayor parte de socios-conductores afirma que una aplicación web alojada en internet, aportara a satisfacer la necesidad de información de los socios conductores, sobre información interna de la cooperativa.

6. ¿Usted puede afirmar que la implementación (creación e instalación) de la aplicación informática TRANSUR, contribuirá con el desarrollo de la cooperativa mejorando su productividad interna? SI o NO

Tabla 11: Pregunta 6 Socios-Conductores

OPCIONES	FRECUENCIA	PORCENTAJE
SI aportará	48	97,96%
NO aportará	1	2,04%
TOTAL	49	100,00%

Fuente: Encuesta socios-conductores cooperativa de transporte inter cantonal Urcuquí



Gráfica 6. Pregunta 6 Socios-Conductores

Fuente: Personal

Análisis:

De los 49 socios – conductores que constituyen la muestra, 48 de ellos que corresponde al 97,96%, afirman que la implementación de una aplicación web, mejorara la

producción interna de la cooperativa, 1 socio – conductor que corresponde al 2,04%, afirma que la implementación de una aplicación web NO mejorara la producción interna de la cooperativa.

Interpretación:

Con relación a la pregunta Nro. 6, de la encuesta realizada a socios-conductores se determina que la mayor parte de socios-conductores afirman que la aplicación TRANSUR, contribuirá a la productividad interna de la cooperativa.

Entrevista Personal – Administrativo

El personal administrativo de la cooperativa está compuesto de 2 personas, las mismas que en ocasiones están respaldadas por la gerencia y la secretaria de boletería a quienes se les realizo una entrevista en la cual se pudo obtener los siguientes resultados.

1. ¿Cuándo se solicita información interna de la cooperativa sobre: socios, conductores, rutas y frecuencias de unidades de transporte, la realiza con el apoyo de alguna herramienta informática (Aplicación de software) u otro medio? SI o NO.

¿Cuáles son las herramientas que utiliza?

- a) Word
- b) Excel
- c) Aplicación informática
- d) Hojas de trabajo
- e) Otros

¿Cuáles?.....

Entre las herramientas utilizadas ¿Cuáles son las que usa con más frecuencia y porque?.....

¿Con la utilización de las herramientas anteriores, la información se encuentra al alcance permanente (24 horas) de la presidencia, gerencia, socios y entes afines de la institución? SI o NO.

Cuando existen peticiones de información, el personal administrativo procesa las peticiones con ayuda de herramientas como Word y Excel, y manifiestan que la información NO SE ENCUENTRAN al alcance de las dependencia de forma permanente y peor aún las 24 horas.

¿Con la utilización de las herramientas anteriores, considera que la información se encuentra?

Segura (protegida)

Centralizada (Base de datos)

Fácil acceso (Internet)

Ninguna

La utilización de las herramientas como Word o Excel, para el procesamiento de información NO GARANTIZA la seguridad, la centralidad, el fácil acceso, es decir NINGUNA.

2. Según su criterio ¿Es relativamente fácil, responder a las peticiones de información de presidencia, gerencia, socios, conductores y medios extra institucionales? SI o NO.

El personal administrativo manifiesta que SI es relativamente fácil, responder a la demanda de información de las dependencias de la cooperativa.

3. ¿Si, se incrementara la cantidad de socios de la cooperativa, la respuesta a peticiones de la presidencia, gerencia, socios y entes afines de la institución, sería inmediata? SI o NO.

El personal administrativo manifiesta que con el incremento de socios, la respuesta a peticiones de las dependencias de la cooperativa NO sería inmediata y tendrían inconvenientes en la entrega de información.

4. Según su criterio ¿Qué medio de información, sería el más fiable (confiable) para garantizar información actualizada y permanente (24 horas)?

a) Hojas de trabajo

b) Reportes semanales de secretaría

c) Aplicación informática en Internet (Software)

d) Tarjetas de rutas de unidades

d) Otros

¿Cuáles?.....

El personal administrativo manifiesta que el medio de información MAS CONFIABLE, para mantener la información actualizada y permanente es UNA APLICACIÓN INFORMATICA EN INTERNET.

5. En el caso de que la respuesta anterior fuese: c) Aplicación informática en Internet. ¿Usted puede afirmar que una Aplicación Informática (TRANSUR) alojada en internet con información interna de la cooperativa sobre unidades de transporte, socios conductores, rutas y frecuencias, con reportes o presentaciones de información actualizada, sustentara o apoyara la gestión de respuestas, a las peticiones de información de organizaciones foráneas, presidencia, gerencia, socios y conductores de forma permanente a cualquier hora y día de la semana? SI o NO.

El personal administrativo manifiesta que la Aplicación Informática (TRANSUR) alojada en internet con información interna de la cooperativa, APORTARA LA GESTIÓN DE RESPUESTAS A LAS PETICIONES DE INFORMACIÓN DE LA DE ORGANIZACIONES FORÁNEAS, DEPENDENCIAS INTERNAS Y CONDUCTORES DE FORMA PERMANENTE A CUALQUIER HORA Y DÍA DE LA SEMANA.

6. ¿Usted puede afirmar que la implementación (creación e instalación) de la aplicación informática TRANSUR, contribuirá con el desarrollo de la cooperativa mejorando su productividad interna? SI o NO.

El personal administrativo afirma que la implementación de la aplicación informática TRANSUR, CONTRIBUIRÁ CON EL DESARROLLO DE LA COOPERATIVA MEJORANDO SU PRODUCTIVIDAD INTERNA.

2.8 Conclusiones del estudio

La cooperativa de Transporte de pasajeros inter cantonal Urcuquí actualmente cuenta con un limitado personal administrativo que realiza las actividades de registro de personal, unidades de transporte y planificación de rutas.

Las instituciones de control de transporte terrestre realizan revisiones periódicas a las flotas de transporte, requiriendo información actualizada de legalización de las unidades de transporte y conductores.

El personal administrativo registra la información de la cooperativa en herramientas de software como hojas de cálculo, lo que conduce a tener la información dispersa en archivos aislados, teniendo que consolidar información y planificar rutas semanales en hojas de trabajo.

La capacidad de respuesta del personal administrativo tiene un tiempo estimado entre 15 y 30 minutos, al realizarse las peticiones de información en días y horas laborables, de lo contrario no se realizan peticiones de información.

Un incremento mínimo en la demanda de actividades en el personal administrativo, eliminaría la posibilidad de procesar tal cantidad de información, produciendo pérdida de credibilidad y prestigio de la institución afectando directamente a la difusión de resultados informativos.

Tales inconvenientes opacarían la imagen institucional, ante los organismos de control de transporte y ofrecería un servicio deficiente a la colectividad urcuquirense.

Siendo necesaria la implementación de una aplicación Informática en Internet TRANSUR, con información confiable de la cooperativa sobre unidades de transporte, socios, conductores, rutas y frecuencias, de forma segura, permanente y actualizada, que aportara al desarrollo y fortalecimiento institucional de la producción interna de la prestigiosa cooperativa de transporte de pasajeros inter cantonal Urcuquí.

CAPÍTULO III

Desarrollo de la Solución

- 3.1 Fase de exploración
 - 3.1.1 Historias de usuario
 - 3.1.2 Arquitectura funcional del sistema
- 3.2 Fase de planeación
 - 3.2.1 Cronograma de actividades
 - 3.2.2 Iteración I
 - Tareas por historia
 - Pruebas
 - 3.2.3 Iteración II
 - Tareas por historia
 - Pruebas
 - Incidencias
 - 3.2.4 Diagrama de base de datos

3.1 Fase de exploración

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo los desarrolladores se familiarizan con las herramientas tecnológicas a utilizarse en el proyecto.

Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

3.1.1 Historias de usuario

Las historias de usuario son un instrumento fundamental para el levantamiento de requerimientos de un proyecto de software, capturando la esencia de los requerimientos del cliente las cuales se detallan en un lenguaje cotidiano.

Inicialmente se listan las historias de usuario encargadas de aportar con los requerimientos iniciales del sistema TRANSUR, constituyéndose como el paso inicial de la implementación de la metodología de desarrollo de software XP.

3.1.1.1 Iteración I

Fase I: Planificación Inicial

Gestión del proyecto

Planificación inicial

Iteración 1

Tabla 12: Historia 1

Historia de Usuario Nro. 1	
Número: 1	Usuario: Gerente - Secretaria
Nombre historia: Registro y Autenticación	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 1
Programador responsable: Fernando Cangás	
Descripción: Formulario encargado de solicitar información de usuario con su respectivo (Login y Password) para acceder a la aplicación TRANSUR.	
Eventos botón Submit	

<p>Se verificara que todos los campos de texto estén debidamente validados y llenos, para que la librería Passport gestione la autenticación.</p> <p>Eventos el botón Salir</p> <p>Cerrará el formulario de autenticación</p>
<p>Observaciones: No se dispondrá la opción de recuperación de contraseña olvidada o perdida por razones de seguridad.</p>

Fuente: Gerente –Secretaria- Programador

Tabla 13: Historia 2

Historia de Usuario Nro. 2	
Número: 2	Usuario: Gerente
Nombre historia: Registro de Usuarios TRANSUR	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 1
Programador responsable: Fernando Cangás	
<p>Descripción: Los usuarios registrados en el formulario de registro y autenticación, se les deberá asignar roles de autenticado y administrador para que tengan acceso a determinados módulos del sistema TRANSUR, dichos roles podrán ser editados.</p> <p>Características :</p> <p>El password de los usuarios para acceder a la aplicación TRANSUR será gestionado bajo la librería Passport integrada en la MEAN.IO stack.</p> <p>Passport con sus características de encriptación en internet administrara eficazmente las contraseñas y el control de acceso en el caso de presentarse ataques de robots de software.</p>	
<p>Observaciones: Existirán únicamente 2 roles, admin y authenticated, No se dispondrá la opción de recuperación de contraseña olvidada o perdida por razones de seguridad.</p>	

Fuente: Gerente

Tabla 14: Historia 3

Historia de Usuario Nro.3	
Número: 3	Usuario: Gerente
Nombre historia: Registro de información institucional	
Prioridad en negocio: Media	Riesgo en desarrollo: Media
Puntos estimados: 3	Iteración asignada: 1
Programador responsable: Fernando Cangás	
Descripción: Ingresar información institucional, Misión institucional, visión institucional, reglamento interno. La información institucional deberá ser subida en un repositorio web, por lo cual se publicaran los links para los usuarios registrados del sistema TRANSUR.	
Observaciones: Se deberían registrar las resoluciones de las juntas directivas indicando sus fechas. De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones.	

Fuente: Gerente

Tabla 15: Historia 4

Historia de Usuario Nro. 4	
Número: 4	Usuario: Gerente - Secretaria
Nombre historia: Registro de Socios	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 1
Programador responsable: Fernando Cangás	
Descripción: Registrar la información de los socios de la empresa : <p style="margin-left: 40px;">Datos personales CI, Nombre, Apellidos, estado civil, dirección, email, teléfono</p> <p style="margin-left: 40px;">Datos de socio/Institución <u>Unidad de transporte,</u> fecha inicio/Termino de funciones,</p>	

<p>Licencia de conducir</p> <p>Tipo licencia , Fecha de expedición, Fecha caducidad</p> <p>Créditos</p> <p>Se realizaran préstamos únicamente a los socios que se encuentren en la lista blanca es decir Sin Deudas.</p> <p>Se registraran datos de fecha de aprobación, razón del préstamo, una tasa de interés simbólica que cubre 1, 3, y 5 % de interés, la fecha de pago del crédito, de esta manera se podrá presentar un reporte de los socios deudores y los socios libres que pueden acceder a préstamos .</p>
<p>Observaciones</p> <p>De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones según correspondiere, teniendo en cuenta que, para realizar la eliminación o baja de un cliente, se deberá eliminar previamente todo su historial crediticio.</p>

Fuente: Gerente-Secretaria

Tabla 16: Historia 5

Historia de Usuario Nro.5	
Número: 5	Usuario: Gerente - Secretaria
Nombre historia: Registro de conductores	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 1
Programador responsable: Fernando Cangás	
<p>Descripción:</p> <p>Registrar la información de conductores de las unidades de transporte</p> <p>Datos personales</p> <p>CI, Nombre, Apellidos, estado civil, dirección, email, teléfono</p> <p>Datos de conductor /Institución</p> <p>Fecha inicio de funciones</p> <p>Fecha de terminación de funciones</p> <p>Licencia de conducir</p> <p>Tipo licencia , Fecha de expedición, Fecha caducidad</p>	

<p>Sanciones</p> <p>Se realizara el registro de sanciones únicamente a los conductores, mas no a los Socios, ingresando datos de fecha de sanción, razón de la sanción y el estado, el cual deberá ser: “Libre” o “Sancionado”.</p>
<p>Observaciones</p> <p>De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones según correspondiere, teniendo en cuenta; para realizar la eliminación de un conductor, se deberá eliminar previamente todo su historial de sanciones, en el caso que existiera alguna sanción.</p>

Fuente: Gerente – Secretaria

Tabla 17: Historia 6

Historia de Usuario Nro. 6	
Número: 6	Usuario: Gerente - Secretaria
Nombre historia: Registro de Personal administrativo	
Prioridad en negocio: Media	Riesgo en desarrollo: Media
Puntos estimados: 3	Iteración asignada: 1
Programador responsable: Fernando Cangás	
<p>Descripción: Registrar la información del personal administrativo :</p> <p>Datos personales CI, Nombre, Apellidos, estado civil, dirección, email, teléfono</p> <p>Datos de socio/Institución Fecha inicio de funciones, Fecha de terminación de funciones.</p> <p>Observaciones En la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones /bajas.</p>	

Fuente: Gerente-Secretaria

Tabla 18: Historia 7

Historia de Usuario Nro. 7	
Número: 7	Usuario: Gerente - Secretaria
Nombre historia: Registro de Sectores/Rutas	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 5
Programador responsable: Fernando Cangás	
Descripción: Registrar la información de los sectores/lugares de las rutas adjudicadas a la Cooperativa de transporte de pasajeros inter cantonal “Urcuquí”, se deberán ingresar la provincia, el cantón y el sector de la ruta.	
Observaciones: Es un prerrequisito para el registro de Rutas. De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones.	

Fuente: Gerente – Secretaria

Tabla 19: Historia 8

Historia de Usuario Nro. 8	
Número: 8	Usuario: Gerente - Secretaria
Nombre historia: Registro de Rutas	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 5
Programador responsable: Fernando Cangás	
Descripción: <p>Previamente ingresados los sectores/lugares con su respectivo cantón y provincia se procederá a ingresar la información de las rutas adjudicadas a la Cooperativa de transporte de pasajeros inter cantonal “Urcuquí”.</p> <p>El formulario solicitará la fecha de la ruta, el lugar de origen, lugar de destino, la hora de inicio de la ruta y la frecuencia de las unidades en minutos con un ámbito máximo de una hora lo que corresponde a 60 minutos.</p>	

Observaciones:

Una vez ingresados los datos anteriormente mencionados, se generaran las rutas con su respectiva hora de inicio, lugar origen, lugar destino, y los registros para ingresar alguna eventual observación o la cantidad de atrasos de alguna unidad de transporte en caso de existir.

Fuente: Gerente – Secretaria

Tabla 20: Historia 9

Historia de Usuario Nro. 9	
Número: 9	Usuario: Gerente - Secretaria
Nombre historia: Registro de Unidades de transporte	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 1
Programador responsable: Fernando Cangás	
<p>Descripción: Registrar la información de las unidades de transporte:</p> <p>Placa, Nro. Motor, Nro. Chasis, Modelo, Cilindraje, Años vida útil, Marca de vehículo, País, Año de fabricación, Nro. Asientos, Fecha de compra, Fecha expedición matrícula Fecha de caducidad matrícula, Color 1, Color2, Código permiso de operación, Estado exoneración, Fecha permiso de operación, Fecha última revisión Observaciones. En el documento de la matricula existen más ítems informativos por cada unidad de transporte, pero una vez socializado con el gerente de la institución se llegó a un común acuerdo, el registrar únicamente los ítems señalados anteriormente.</p> <p>Observaciones : De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones según correspondiere, teniendo en cuenta que, para realizar la eliminación o baja de una unidad, incurrirá en inconsistencias en los módulos de socios, rutas y boletería, por lo que las eliminaciones/ bajas, únicamente las realizara el administrador de TRANSUR, teniendo las precauciones necesarias para no generar inconsistencias.</p>	

Fuente: Gerente – Secretaria

Tabla 21: Historia 10

Historia de Usuario Nro. 10	
Número: 10	Usuario: Gerente - Secretaria
Nombre historia: Gestión Boletería	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 5	Iteración asignada: 5
Programador responsable: Fernando Cangás	
<p>Descripción: Registrar la información de los boletos separados o reservados así como los de los boletos vendidos en la sucursal de la cabecera cantonal de la ciudad de Ibarra.</p> <p>Reservación de boletos El usuario interesado en la reservación de boletos, previamente se registrara en la aplicación, al ingresar al módulo de Gestión de boletería deberá de registrar nuevamente sus datos personales como: CI, Nombres, Apellidos, Nro. Teléfono y cantidad de boletos a comprar/reservar. Una vez ingresados sus datos personales, deberá elegir la ruta (hora de salida, lugar de origen, lugar de destino), dando lugar a que en la venta de boletos los asientos reservados no puedan ser vendidos.</p> <p>Ventas de boletos La persona/ personas encargadas de la venta de boletos deberán registrar la venta de boletos como usuario anónimo o con CI, nombres y apellidos del usuario, los asientos que venderá deberán estar libres de reservaciones.</p> <p>Observaciones : Debido a la congestión en determinadas horas y al no existir una política que reglamente el uso estricto de boletos el abordaje de las unidades de transporte, el registro de la venta de boletos, quedara a discreción de la persona o personas encargadas de esta actividad.</p>	

Fuente: Gerente – Secretaria

3.1.2 Arquitectura Funcional del sistema TRANSUR

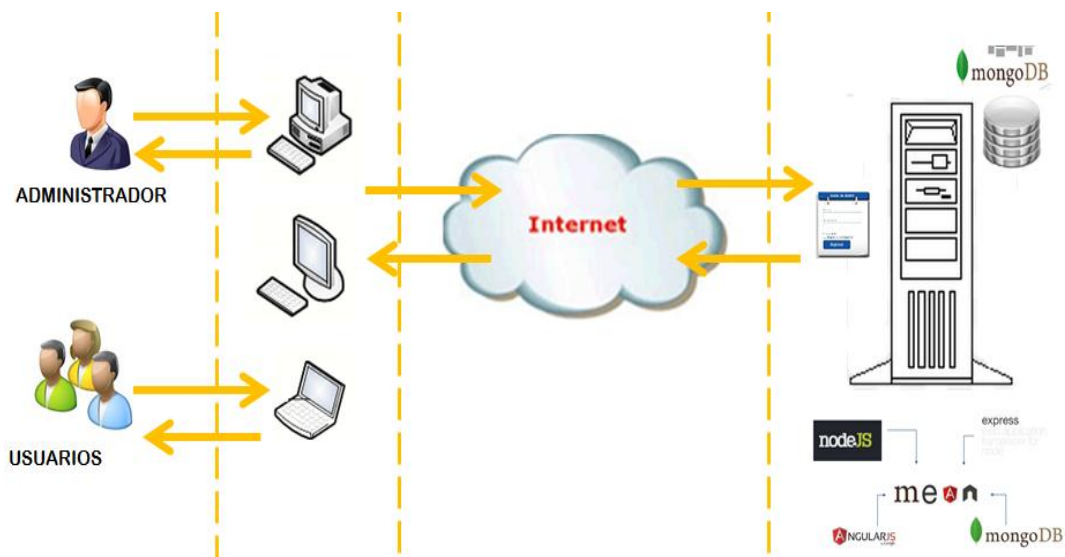


Fig. 6 Arquitectura Funcional del sistema TRANSUR
Fuente: personal

3.2 Fase de planeación

En base a las historias de usuario, se obtuvo los módulos del sistema TRANSUR, estableciéndose un cronograma de actividades a realizarse en 2 iteraciones.

3.2.1 Cronograma de actividades

Tabla 22: Cronograma de actividades

Historia		Tiempo estimado		
		F. inicio	F. final	Horas Est.
H1	Registro y Autenticación	01/08/2014	21/08/2014	160
H2	Registro de Usuarios TRANSUR	22/08/2014	30/08/2014	64
H3	Registro de información institucional	31/08/2014	09/09/2014	72
H4	Registro de Socios	10/09/2014	20/09/2014	80
H5	Registro de conductores	21/09/2014	28/09/2014	56
H6	Registro de Personal administrativo	29/09/2014	07/10/2014	64
H7	Registro de Sectores/Rutas	08/10/2014	13/10/2014	40
H8	Registro de Rutas	14/09/2014	30/09/2014	128
H9	Registro de Unidades de Transporte	01/10/2014	12/10/2014	88

H10	Gestión de Boletería	13/10/2014	02/11/2014	152
-----	----------------------	------------	------------	-----

Fuente: Propia

3.2.2 Módulos del sistema TRANSUR

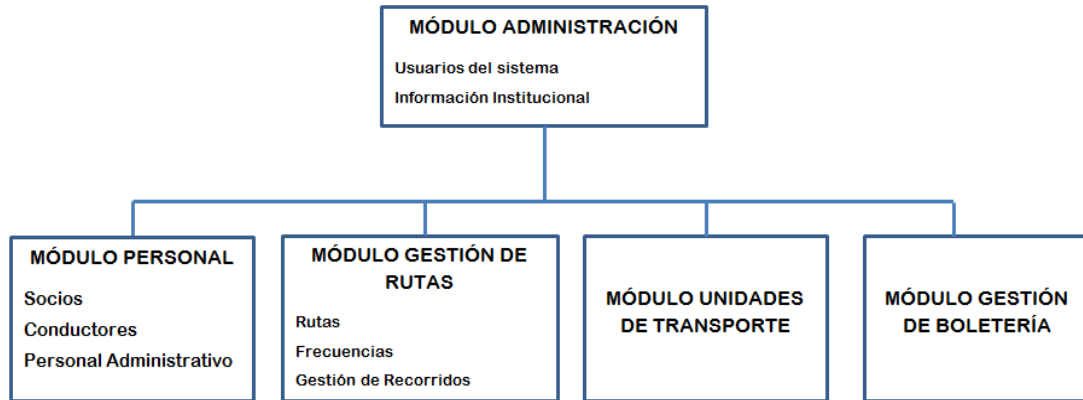


Fig. 7: Módulos del Sistema TRANSUR

Fuente: Personal

3.2.2.1 Descripción de los módulos del sistema TRANSUR

Módulo Administración

Es el módulo encargado de validar y registrar el ingreso de usuarios al sistema TRANSUR y a cada uno de sus módulos, así como la publicación de información institucional, para los usuarios autenticados.

Módulo Personal

Es el modulo, destinado a registrar socios, conductores y personal administrativo de la institución, con sus datos personales.

Los socios deberán ser registrados con la unidad de transporte de su propiedad y los créditos adjudicados.

Los conductores deberán ser registrados con sus respectivos datos personales y sanciones en caso de que existieran.

Tanto en socios y conductores se deberá registrar el tipo de licencia y fechas de expedición y caducidad de cada permiso de conducir.

El personal Administrativo deberá ser registrado con sus respectivos datos personales.

Módulo Gestión de Rutas

Es el modulo encargado de registrar las rutas de las unidades de transporte, con los datos de la fecha de la ruta, lugar salida, hora de salida, frecuencia de unidades, lugar de destino.

Cada registro de ruta, además de los datos citados anteriormente deberá contener los siguientes ítems: observaciones, unidades de transporte y los posibles atrasos.

Módulo Unidades de Transporte

El modulo unidades de transporte es el encargado de registrar las unidades de transporte con datos referentes a la matrícula y permiso de operación de cada unidad de transporte.

Módulo Gestión de Boletería

Es el modulo encargado de registrar reservaciones de asientos en las unidades de transporte, así como la venta de boletos de la sucursal Ibarra, el registro de venta de boletos no es obligatorio ya que los usuarios pueden abordar las unidades de transporte, sin boletos.

3.3. Iteración I

3.3.1. Cronograma de actividades

El cronograma de actividades de la iteración I, contempla las historias de usuario, con fecha de inicio, fecha de finalización y su respectiva cantidad estimada de horas.

Tabla 23: Cronograma de actividades Iteración I

Iteración I				
Historia		Tiempo estimado		
		F. inicio	F. final	Horas Est.
H1	Registro y Autenticación	01/08/2014	21/08/2014	160
H2	Registro de Usuarios TRANSUR	22/08/2014	30/08/2014	64
H3	Registro de información institucional	31/08/2014	09/09/2014	72
H4	Registro de Socios	10/09/2014	20/09/2014	80
H5	Registro de conductores	21/09/2014	28/09/2014	56
Total Horas				432

Fuente: Propia

3.3.2. Tareas por historia Iteración I

3.3.2.1 Registro y Autenticación

Tabla 24: Tareas de Registro y Autenticación

H1 Registro y Autenticación		
	Actividad	Horas
1	Descarga e Instalación de software para la implementación	16
2	Estudio de alternativas de Hosting MEAN.IO para alojamiento de la app	24
3	Adquisición y alta en el servicio de Hosting MEAN.IO	16
4	Diseño de colecciones (Tablas), para el Registro y Autenticación.	24
5	Implementación de restricciones de validación en las colecciones	8
6	Implementación de Controlador de lado del Servidor	16
7	Implementación del Ruteo de lado del Servidor	8
8	Implementación del Servicio (Factory) de lado del Cliente	16
9	Implementación del Controlador de Angular (lado del Cliente)	8
10	Implementación de Formulario de Registro	24
Total		160

Fuente: Personal

1. Descarga e Instalación de software para la implementación

Tabla 25: Registro y Autenticación – Tarea 1

Nro. de tarea: T1-H1	Nro. de Historia: H1
Nombre tarea: Descarga e Instalación de software para la implementación	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 01/08/2014	Fecha fin: 02/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Descarga e Instalación de software para la implementación de la aplicación TRANSUR. Todas las herramientas de software utilizadas para el desarrollo e implementación de la aplicación TRANSUR, poseen licencia OpenSource.</p> <p>Software a instalarse Se instalaran las siguientes herramientas de software: Node.JS v 0.10.20 MongoDB 2.2.7 Jet Brains Web Storm 8.0.3 Git 1.9.5</p>	

Fuente: Propia

2. Estudio de alternativas de Hosting MEAN.IO para alojamiento de la app

Tabla 26: Registro y Autenticación – Tarea 2

Nro. de tarea: T2-H1	Nro. de Historia: H1
Nombre tarea: Estudio de alternativas de Hosting MEAN.IO para alojamiento de la app	
Tipo de tarea: General	Tiempo estimado: 24 Horas
Fecha inicio: 03/08/2014	Fecha fin: 05/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Determinar la alternativa idónea para el alojamiento de la aplicación TRANSUR, teniendo en cuenta parámetros estratégicos como la integración con MEAN stack.</p>	

Fuente: Propia

4. Adquisición y alta en el servicio de Hosting MEAN.IO

Tabla 27: Registro y Autenticación – Tarea 3

Nro. de tarea: T3-H1	Nro. de Historia: H1
Nombre tarea: Adquisición y alta en el servicio de Hosting MEAN.IO	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 06/08/2014	Fecha fin: 07/08/2014
Programador responsable: Fernando Cangás	
Descripción: Al determinar el servicio de Hosting idóneo, se realiza la adquisición del mismo con una forma de pago electrónica o vía Paypal, Concluida la adquisición del servicio, se realizan las pruebas de evaluación como la instalación de un nuevo servidor de alojamiento, la zona del servicio, el registro de usuario administrador, los servicios repositorio de imágenes Wixmedia , y administración mean-admin.	

Fuente: Propia

5. Diseño de colecciones (Tablas), para el Registro y Autenticación

Tabla 28: Registro y Autenticación – Tarea 4

Nro. de tarea: T4-H1	Nro. de Historia: H1
Nombre tarea: Diseño de colecciones (Tablas), para el Registro y Autenticación	
Tipo de tarea: General	Tiempo estimado: 24 Horas
Fecha inicio: 08/08/2014	Fecha fin: 10/08/2014
Descripción: Se definen los campos de la colección, los cuales contendrán documentos definidos por pares de tipo clave/valor, los documentos pueden contener variables de tipo Strings, Number, Float y contener objetos de tipo Ref , es decir instancias de otra tabla del modelo. Estas colecciones serán gestionadas con la librería Passport de MEAN stack para la encriptación /seguridad de las claves y usuarios de la aplicación TRANSUR.	

Fuente: Propia

5. Implementación de restricciones de validación en las colecciones

Tabla 29: Registro y Autenticación – Tarea 5

Nro. de tarea: T5-H1	Nro. de Historia: H1
Nombre tarea: Implementación de restricciones de validación en las colecciones	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 11/08/2014	Fecha fin: 11/08/2014
Programador responsable: Fernando Cangás	
Descripción: Definidos los campos de la colección, serán estrictamente validados, siendo indispensable mantener restricciones con el propósito de mantener consistencia de información en la base de datos, teniendo en cuenta el tipo de datos a ingresar y la atomicidad de algún campo específico (documento único).	

Fuente: Propia

6. Implementación de Controlador de lado del Servidor

Tabla 30: Registro y Autenticación – Tarea 6

Nro. de tarea: T6-H1	Nro. de Historia: H1
Nombre tarea: Implementación de Controlador de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 12/08/2014	Fecha fin: 13/08/2014
Programador responsable: Fernando Cangás	
Descripción: El controlador del lado del servidor, gestionara las operaciones CRUD solicitadas por el front-end, al servidor y a su base de datos.	

Fuente: Propia

7. Implementación del Ruteo de lado del Servidor

Tabla 31: Registro y Autenticación – Tarea 7

Nro. de tarea: T7-H1	Nro. de Historia: H1
Nombre tarea: Implementación del Ruteo de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 12/08/2014	Fecha fin: 12/08/2014
Programador responsable: Fernando Cangás	
Descripción: El ruteador del lado del servidor, gestionara el acceso o permisos a las operaciones CRUD o servicios soportados por el servidor y su base de datos.	

Fuente: Propia

8. Implementación del Servicio (Factory) de lado del Cliente

Tabla 32: Registro y Autenticación – Tarea 8

Nro. de tarea: T8-H1	Nro. de Historia: H1
Nombre tarea: Implementación del Servicio (Factory) de lado del Cliente	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 13/08/2014	Fecha fin: 14/08/2014
Programador responsable: Fernando Cangás	
Descripción: El servicio Factory con la gestión de ID (Inyección de dependencias), permitirá el acceso a los servicios brindados por el servidor y las operaciones CRUD.	

Fuente: Propia

9. Implementación del Controlador de Angular (lado del Cliente)

Tabla 33: Registro y Autenticación – Tarea 9

Nro. de tarea: T9-H1	Nro. de Historia: H1
Nombre tarea: Implementación del Controlador de Angular (lado del Cliente)	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 15/08/2014	Fecha fin: 16/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: El Controlador del lado del cliente será el encargado de parametrizar, validar y proveer de los recursos necesarios al/ los formularios de los recursos del lado del servidor. La característica más relevante es de proporcionar de data binding entre las capas front –end y back –end es decir que los cambios ocasionados en los formularios , se reflejaran en la base de datos y viceversa sin necesidad de recargar la página actual de navegación</p>	

Fuente: Propia

10. Implementación de Formulario de Registro

Tabla 34: Registro y Autenticación – Tarea 10

Nro. de tarea: T10-H1	Nro. de Historia: H1
Nombre tarea: Implementación de Formulario de Registro	
Tipo de tarea: General	Tiempo estimado: 24 Horas
Fecha inicio: 16/08/2014	Fecha fin: 18/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Se implementa el Formulario encargado de solicitar información de usuario con su respectivo (Login y Password) para acceder a la aplicación TRANSUR.</p> <p>Eventos botón Submit Se verificara estrictamente que todos los campos de texto estén debidamente</p>	

Validados y llenos, para que la librería Passport gestione la autenticación.

Eventos el botón Salir

Cerrará el formulario de autenticación

Observaciones: No se dispondrá la opción de recuperación de contraseña olvidada o perdida por razones de seguridad.

Fuente: Propia

3.3.2.2 Registro de Usuarios TRANSUR

Tabla 35: Tareas de Registro de Usuarios TRANSUR

H2 Registro de Usuarios TRANSUR		
	Actividad	Horas
1	Diseño de colecciones (Tablas), para el Registro y Autenticación.	8
2	Implementación de restricciones de validación en las colecciones	8
3	Implementación de Controlador de lado del Servidor	8
4	Implementación del Ruteo de lado del Servidor	8
5	Implementación del Servicio (Factory) de lado del Cliente	8
6	Implementación del Controlador de Angular (lado del Cliente)	16
7	Implementación de Formulario de Registro	8
Total		64

Fuente: Propia

1. Diseño de colecciones (Tablas), para el Registro de Usuarios TRANSUR

Tabla 36: Registro de Usuarios TRANSUR - Tarea 1

Nro. de tarea: T1-H2	Nro. de Historia: H2
Nombre tarea: Diseño de colecciones (Tablas), para el Registro de Usuarios TRANSUR	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 22/08/2014	Fecha fin: 22/08/2014
Programador responsable: Fernando Cangás	

Descripción: Se definen los campos de la colección, los cuales contendrán documentos definidos por pares de tipo clave/valor, los documentos pueden contener variables de tipo Strings, Number, Float y contener objetos de tipo Ref , es decir instancias de otra tabla del modelo. Estas colecciones serán gestionadas con la librería Passport de MEAN stack para la encriptación /seguridad de las claves y usuarios de la aplicación TRANSUR.

Fuente: Propia

2. Implementación de restricciones de validación en las colecciones

Tabla 37: Registro de Usuarios TRANSUR - Tarea 2

Nro. de tarea: T2-H2	Nro. de Historia: H2
Nombre tarea: Implementación de restricciones de validación en las colecciones	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 23/08/2014	Fecha fin: 23/08/2014
Programador responsable: Fernando Cangás	
Descripción: Definidos los campos de la colección, serán estrictamente validados, siendo indispensable mantener restricciones con el propósito de mantener consistencia de información en la base de datos, teniendo en cuenta el tipo de datos a ingresar y la atomicidad de algún campo específico (documento único).	

Fuente: Propia

3. Implementación de Controlador de lado del Servidor

Tabla 38: Registro de Usuarios TRANSUR - Tarea 3

Nro. de tarea: T3-H2	Nro. de Historia: H2
Nombre tarea: Implementación de Controlador de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 24/08/2014	Fecha fin: 24/08/2014
Programador responsable: Fernando Cangás	

Descripción:

El controlador del lado del servidor, gestionara las operaciones CRUD solicitadas por el front-end, al servidor y a su base de datos.

Fuente: Propia

4. Implementación del Ruteo de lado del Servidor

Tabla 39: Registro de Usuarios TRANSUR - Tarea 4

Nro. de tarea: T4-H2	Nro. de Historia: H2
Nombre tarea: Implementación del Ruteo de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 25/08/2014	Fecha fin: 25/08/2014
Programador responsable: Fernando Cangás	
Descripción: El ruteador del lado del servidor, gestionara el acceso o permisos a las operaciones CRUD o servicios soportados por el servidor y su base de datos.	

Fuente: Propia

5. Implementación del Servicio (Factory) de lado del Cliente

Tabla 40: Registro de Usuarios TRANSUR - Tarea 5

Nro. de tarea: T5-H2	Nro. de Historia: H2
Nombre tarea: Implementación del Servicio (Factory) de lado del Cliente	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 26/08/2014	Fecha fin: 26/08/2014
Programador responsable: Fernando Cangás	
Descripción: El servicio Factory con la gestión de ID (Inyección de dependencias), permitirá el acceso a los servicios brindados por el servidor y las operaciones CRUD.	

Fuente: Propia

6. Implementación del Controlador de Angular (lado del Cliente)

Tabla 41: Registro de Usuarios TRANSUR - Tarea 6

Nro. de tarea: T6-H2	Nro. de Historia: H2
Nombre tarea: Implementación del Controlador de Angular (lado del Cliente)	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 27/08/2014	Fecha fin: 28/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: El Controlador del lado del cliente será el encargado de parametrizar, validar y proveer de los recursos necesarios al/ los formularios de los recursos del lado del servidor.</p> <p>La característica más relevante es de proporcionar de data binding entre las capas front –end y back –end es decir que los cambios ocasionados en los formularios , se reflejaran en la base de datos y viceversa sin necesidad de recargar la página actual de navegación</p>	

Fuente: Propia

7. Implementación de Formulario de Registro

Tabla 42: Registro de Usuarios TRANSUR - Tarea 7

Nro. de tarea: T7-H2	Nro. de Historia: H2
Nombre tarea: Implementación de Formulario de Registro de Usuarios TRANSUR	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 31/08/2014	Fecha fin: 29/08/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Los usuarios registrados en el formulario de registro y autenticación, se les deberá asignar roles de autenticado y administrador para que tengan acceso a determinados módulos del sistema TRANSUR, dichos roles podrán ser editados.</p> <p>Características : El password de los usuarios para acceder a la aplicación TRANSUR será gestionado bajo la librería Passport integrada en la MEAN.IO stack.</p>	

Passport con sus características de encriptación en internet administrara eficazmente las contraseñas y el control de acceso en el caso de presentarse ataques de robots de software.

Fuente: Propia

3.3.2.3 Registro de información institucional

Tabla 43: Tareas de Registro de Información Institucional

H3 Registro de información institucional		
	Actividad	Horas
1	Diseño de colecciones (Tablas), para el Registro y Autenticación.	8
2	Implementación de restricciones de validación en las colecciones	8
3	Implementación de Controlador de lado del Servidor	8
4	Implementación del Ruteo de lado del Servidor	8
5	Implementación del Servicio (Factory) de lado del Cliente	8
6	Implementación del Controlador de Angular (lado del Cliente)	16
7	Implementación de Formulario de Registro	16
Total		72

Fuente: Propia

1. Diseño de colecciones (Tablas), para el Registro de información institucional

Tabla 44: Registro de Información Institucional - Tarea 1

Nro. de tarea: T1-H3	Nro. de Historia: H3
Nombre tarea: Diseño de colecciones (Tablas), para el Registro de información institucional	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 31/08/2014	Fecha fin: 31/08/2014
Programador responsable: Fernando Cangás	
Descripción: Se definen los campos de la colección, los cuales contendrán documentos	

Definidos por pares de tipo clave/valor, los documentos pueden contener variables de tipo Strings, Number, Float y contener objetos de tipo Ref , es decir instancias de otra tabla del modelo.

Fuente: Propia

2. Implementación de restricciones de validación en las colecciones

Tabla 45: Registro de Información Institucional - Tarea 2

Nro. de tarea: T2-H3	Nro. de Historia: H3
Nombre tarea: Implementación de restricciones de validación en las colecciones	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 01/09/2014	Fecha fin: 01/09/2014
Programador responsable: Fernando Cangás	
Descripción: Definidos los campos de la colección, serán estrictamente validados, siendo indispensable mantener restricciones con el propósito de mantener consistencia de información en la base de datos, teniendo en cuenta el tipo de datos a ingresar y la atomicidad de algún campo específico (documento único).	

Fuente: Propia

3. Implementación de Controlador de lado del Servidor

Tabla 46: Registro de Información Institucional - Tarea 3

Nro. de tarea: T3-H3	Nro. de Historia: H3
Nombre tarea: Implementación de Controlador de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 02/09/2014	Fecha fin: 02/09/2014
Programador responsable: Fernando Cangás	
Descripción:	

El controlador del lado del servidor, gestionara las operaciones CRUD solicitadas por el front-end, al servidor y a su base de datos.

Fuente: Propia

3. Implementación del Ruteo de lado del Servidor

Tabla 47: Registro de Información Institucional - Tarea 4

Nro. de tarea: T4-H3	Nro. de Historia: H3
Nombre tarea: Implementación del Ruteo de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 03/09/2014	Fecha fin: 03/09/2014
Programador responsable: Fernando Cangás	
Descripción: El ruteador del lado del servidor, gestionara el acceso o permisos a las operaciones CRUD o servicios soportados por el servidor y su base de datos.	

Fuente: Propia

4. Implementación del Servicio (Factory) de lado del Cliente

Tabla 48: Registro de Información Institucional - Tarea 5

Nro. de tarea: T5-H3	Nro. de Historia: H3
Nombre tarea: Implementación del Servicio (Factory) de lado del Cliente	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 04/09/2014	Fecha fin: 04/09/2014
Programador responsable: Fernando Cangás	
Descripción: El servicio Factory con la gestión de ID (Inyección de dependencias), permitirá el acceso a los servicios brindados por el servidor y las operaciones CRUD.	

Fuente: Propia

6. Implementación del Controlador de Angular (lado del Cliente)

Tabla 49: Registro de Información Institucional - Tarea 6

Nro. de tarea: T6-H3	Nro. de Historia: H3
Nombre tarea: Implementación del Controlador de Angular (lado del Cliente)	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 05/09/2014	Fecha fin: 06/09/2014
Programador responsable: Fernando Cangás	
<p>Descripción: El Controlador del lado del cliente será el encargado de parametrizar, validar y proveer de los recursos necesarios al/ los formularios de los recursos del lado del servidor.</p> <p>La característica más relevante es de proporcionar de data binding entre las capas front –end y back –end es decir que los cambios ocasionados en los formularios , se reflejaran en la base de datos y viceversa sin necesidad de recargar la página actual de navegación</p>	

Fuente: Propia

7. Implementación de Formularios CRUD de información institucional

Tabla 50: Registro de Información Institucional - Tarea 7

Nro. de tarea: T7-H3	Nro. de Historia: H3
Nombre tarea: Implementación de Formularios CRUD de información institucional	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 07/09/2014	Fecha fin: 09/09/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Ingresar información institucional, Misión institucional, visión institucional, reglamento interno. La información institucional deberá ser subida en un repositorio web, por lo cual se publicaran los links para los usuarios registrados del sistema TRANSUR.</p>	

Observaciones: Se deberían registrar las resoluciones de las juntas directivas indicando sus fechas. De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones.

Fuente: Propia

3.3.2.4 Registro de Socios

Tabla 51: Tareas de Registro de Socios

H4 Registro de Socios		
	Actividad	Horas
1	Diseño de colecciones (Tablas), para el Registro de Socios	16
2	Implementación de restricciones de validación en las colecciones	8
3	Implementación de Controlador de lado del Servidor	8
4	Implementación del Ruteo de lado del Servidor	8
5	Implementación del Servicio (Factory) de lado del Cliente	8
6	Implementación del Controlador de Angular (lado del Cliente)	16
7	Implementación de Formulario de Registro	16
Total		80

Fuente: Propia

1. Diseño de colecciones (Tablas), para el Registro de Socios

Tabla 52: Registro de Socios - Tarea 1

Nro. de tarea: T1-H4	Nro. de Historia: H4
Nombre tarea: Diseño de colecciones (Tablas), para el Registro de Socios	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 10/09/2014	Fecha fin: 11/09/2014
Programador responsable: Fernando Cangás	
Descripción: Se definen los campos de la colección, los cuales contendrán documentos definidos por pares de tipo clave/valor, los documentos pueden contener variables de tipo Strings, Number, Float y contener objetos de tipo Ref , es decir instancias de otra tabla del modelo.	

Estas colecciones serán gestionadas con la librería Passport de MEAN stack para la encriptación /seguridad de las claves y usuarios de la aplicación TRANSUR.

Fuente: Propia

2. Implementación de restricciones de validación en las colecciones

Tabla 53: Registro de Socios - Tarea 2

Nro. de tarea: T2-H4	Nro. de Historia: H4
Nombre tarea: Implementación de restricciones de validación en las colecciones	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 12/09/2014	Fecha fin: 12/09/2014
Programador responsable: Fernando Cangás	
Descripción: Definidos los campos de la colección, serán estrictamente validados, siendo indispensable mantener restricciones con el propósito de mantener consistencia de información en la base de datos, teniendo en cuenta el tipo de datos a ingresar y la atomicidad de algún campo específico (documento único).	

Fuente: Propia

3. Implementación de Controlador de lado del Servidor

Tabla 54: Registro de Socios - Tarea 3

Nro. de tarea: T3-H4	Nro. de Historia: H4
Nombre tarea: Implementación de Controlador de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 13/09/2014	Fecha fin: 13/09/2014
Programador responsable: Fernando Cangás	
Descripción: El controlador del lado del servidor, gestionara las operaciones CRUD solicitadas por el front-end, al servidor y a su base de datos.	

Fuente: Propia

4. Implementación del Ruteo de lado del Servidor

Tabla 55: Registro de Socios - Tarea 4

Nro. de tarea: T4-H4	Nro. de Historia: H4
Nombre tarea: Implementación del Ruteo de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 14/09/2014	Fecha fin: 14/09/2014
Programador responsable: Fernando Cangás	
Descripción: El ruteador del lado del servidor, gestionara el acceso o permisos a las operaciones CRUD o servicios soportados por el servidor y su base de datos.	

Fuente: Propia

5. Implementación del Servicio (Factory) de lado del Cliente

Tabla 56: Registro de Socios - Tarea 5

Nro. de tarea: T5-H4	Nro. de Historia: H4
Nombre tarea: Implementación del Servicio (Factory) de lado del Cliente	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 15/09/2014	Fecha fin: 15/09/2014
Programador responsable: Fernando Cangás	
Descripción: El servicio Factory con la gestión de ID (Inyección de dependencias), permitirá el acceso a los servicios brindados por el servidor y las operaciones CRUD.	

Fuente: Propia

6. Implementación del Controlador de Angular (lado del Cliente)

Tabla 57: Registro de Socios - Tarea 6

Nro. de tarea : T6-H4	Nro. de Historia: H4
------------------------------	-----------------------------

Nombre tarea: Implementación del Controlador de Angular (lado del Cliente)	
Tipo de tarea: General	Tiempo estimado: 16 Horas
Fecha inicio: 16/09/2014	Fecha fin: 17/09/2014
Programador responsable: Fernando Cangás	
Descripción: <p>El Controlador del lado del cliente será el encargado de parametrizar, validar y proveer de los recursos necesarios al/ los formularios de los recursos del lado del servidor.</p> <p>La característica más relevante es de proporcionar de data binding entre las capas front –end y back –end es decir que los cambios ocasionados en los formularios , se reflejaran en la base de datos y viceversa sin necesidad de recargar la página actual de navegación</p>	

Fuente: Propia

7. Implementación de Formulario de Registro de Socios

Tabla 58: Registro de Socios - Tarea 7

Nro. de tarea: T7-H4	Nro. de Historia: H4
Nombre tarea: Implementación de Formularios CRUD de para Socios	
Tipo de tarea: General	Tiempo estimado: 24 Horas
Fecha inicio: 18/09/2014	Fecha fin: 20/09/2014
Programador responsable: Fernando Cangás	
Descripción: <p>Registrar la información de los socios de la empresa :</p> <p>Datos personales CI, Nombre, Apellidos, estado civil, dirección, email, teléfono</p> <p>Datos de socio/Institución <u>Unidad de transporte,</u> fecha inicio/Termino de funciones,</p> <p>Licencia de conducir Tipo licencia , Fecha de expedición, Fecha caducidad</p>	

<p>Créditos</p> <p>Se realizaran préstamos únicamente a los socios que se encuentren en la lista blanca es decir Sin Deudas.</p> <p>Se registraran datos de fecha de aprobación, razón del préstamo, una tasa de interés simbólica que cubre 1, 3, y 5 % de interés, la fecha de pago del crédito, de esta manera se podrá presentar un reporte de los socios deudores y los socios libres que pueden acceder a préstamos .</p> <p>Observaciones</p> <p>De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones según correspondiere, teniendo en cuenta que, para realizar la eliminación o baja de un cliente, se deberá eliminar previamente todo su historial crediticio.</p>

Fuente: Propia

3.3.2.5 Registro de conductores

Tabla 59: Tareas de Registro de Conductores

H5 Registro de Conductores		
	Actividad	Horas
1	Diseño de colecciones (Tablas), para el Registro y Autenticación.	8
2	Implementación de restricciones de validación en las colecciones	8
3	Implementación de Controlador de lado del Servidor	8
4	Implementación del Ruteo de lado del Servidor	8
5	Implementación del Servicio (Factory) de lado del Cliente	8
6	Implementación del Controlador de Angular (lado del Cliente)	8
7	Implementación de Formulario de Registro	8
Total		56

Fuente: Propia

1. Diseño de colecciones (Tablas), para el Registro de Socios

Tabla 60: Registro de Conductores - Tarea 1

Nro. de tarea: T1-H5	Nro. de Historia: H5
Nombre tarea: Diseño de colecciones (Tablas), para el Registro de Socios	

Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 21/09/2014	Fecha fin: 21/09/2014
Programador responsable: Fernando Cangás	
<p>Descripción: Se definen los campos de la colección, los cuales contendrán documentos definidos por pares de tipo clave/valor, los documentos pueden contener variables de tipo Strings, Number, Float y contener objetos de tipo Ref , es decir instancias de otra tabla del modelo. Estas colecciones serán gestionadas con la librería Passport de MEAN stack para la encriptación /seguridad de las claves y usuarios de la aplicación TRANSUR.</p>	

Fuente: Personal

2. Implementación de restricciones de validación en las colecciones

Tabla 61: Registro de Conductores - Tarea 2

Nro. de tarea: T2-H5	Nro. de Historia: H5
Nombre tarea: Implementación de restricciones de validación en las colecciones	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 22/09/2014	Fecha fin: 22/09/2014
Programador responsable: Fernando Cangás	
<p>Descripción:</p> <p>Definidos los campos de la colección, serán estrictamente validados, siendo indispensable mantener restricciones con el propósito de mantener consistencia de información en la base de datos, teniendo en cuenta el tipo de datos a ingresar y la atomicidad de algún campo específico (documento único).</p>	

Fuente: Propia

3. Implementación de Controlador de lado del Servidor

Tabla 62: Registro de Conductores - Tarea 3

Nro. de tarea: T3-H5	Nro. de Historia: H5
Nombre tarea: Implementación de Controlador de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 23/09/2014	Fecha fin: 23/09/2014
Programador responsable: Fernando Cangás	
Descripción: El controlador del lado del servidor, gestionara las operaciones CRUD solicitadas por el front-end, al servidor y a su base de datos.	

Fuente: Personal

4. Implementación del Ruteo de lado del Servidor

Tabla 63: Registro de Conductores - Tarea 4

Nro. de tarea: T4-H5	Nro. de Historia: H5
Nombre tarea: Implementación del Ruteo de lado del Servidor	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 24/09/2014	Fecha fin: 24/09/2014
Programador responsable: Fernando Cangás	
Descripción: El ruteador del lado del servidor, gestionara el acceso o permisos a las operaciones CRUD o servicios soportados por el servidor y su base de datos.	

Fuente: Propia

5. Implementación del Servicio (Factory) de lado del Cliente

Tabla 64: Registro de Conductores - Tarea 5

Nro. de tarea: T5-H5	Nro. de Historia: H5
Nombre tarea: Implementación del Servicio (Factory) de lado del Cliente	
Tipo de tarea: General	Tiempo estimado: 8 Horas

Fecha inicio: 25/09/2014	Fecha fin: 25/09/2014
Programador responsable: Fernando Cangás	
Descripción: El servicio Factory con la gestión de ID (Inyección de dependencias), permitirá el acceso a los servicios brindados por el servidor y las operaciones CRUD	

Fuente: Propia

6. Implementación del Controlador de Angular (lado del Cliente)

Tabla 65: Registro de Conductores - Tarea 6

Nro. de tarea: T6-H5	Nro. de Historia: H5
Nombre tarea: Implementación del Controlador de Angular (lado del Cliente)	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 26/09/2014	Fecha fin: 26/09/2014
Programador responsable: Fernando Cangás	
Descripción: El Controlador del lado del cliente será el encargado de parametrizar, validar y proveer de los recursos necesarios al/ los formularios de los recursos del lado del servidor. La característica más relevante es de proporcionar de data binding entre las capas front –end y back –end es decir que los cambios ocasionados en los formularios , se reflejaran en la base de datos y viceversa sin necesidad de recargar la página actual de navegación	

Fuente: Propia

7. Implementación de Formularios CRUD para Socios

Tabla 66: Registro de Conductores - Tarea 7

Nro. de tarea: T7-H2	Nro. de Historia: H2
Nombre tarea: Implementación de Formularios CRUD para socios	
Tipo de tarea: General	Tiempo estimado: 8 Horas
Fecha inicio: 27/09/2014	Fecha fin: 28/09/2014

Programador responsable: Fernando Cangás

Descripción:

Registrar la información de conductores de las unidades de transporte

Datos personales

CI, Nombre, Apellidos, estado civil, dirección, email, teléfono

Datos de conductor /Institución

Fecha inicio de funciones

Fecha de terminación de funciones

Licencia de conducir

Tipo licencia , Fecha de expedición, Fecha caducidad

Sanciones

Se realizara el registro de sanciones únicamente a los conductores, mas no a los Socios, ingresando datos de fecha de sanción, razón de la sanción y el estado, el cual deberá ser: “Libre” o “Sancionado”.

Observaciones

De la información citada anteriormente se deberán hacer inserciones, modificaciones y eliminaciones según correspondiere, teniendo en cuenta; para realizar la eliminación de un conductor, se deberá eliminar previamente todo su historial de sanciones, en el caso que existiera alguna sanción.

Fuente: Propia

3.3.3. Pruebas Iteración I

3.3.3.1. Especificación de Prueba: Registro y Autenticación

Historia de usuario 1

Versión <1.0>

Historial de Revisiones

Tabla 67: Iteración I, Historial de revisiones Nro. 1

Fecha	Versión	Descripción	Autor
21/08/2014	1.0	Revisión	Fernando Cangás
21/08/2014	1.0.1	Modificación	Fernando Cangás

Fuente: Propia

En la historia H1 se hace la verificación del ingreso de datos para loguearse. En el logueo se valida, la autenticidad del ingreso del usuario, verificando los campos cubren las especificaciones de ingreso como un email/username valido, en el caso de que el usuario no este registrado se le enviara un aviso en pantalla.

El usuario al loguearse como administrador, se le presentara un menú en el lado izquierdo de la pantalla donde, se podrán administrar, los perfiles de los usuarios registrados, no estára habilitada la función recuperar password por motivos de seguridad y por petición institucional.

Registro y Autenticación

Descripción

La información ingresada en el formulario de logueo deberá cumplir, con las validaciones y restricciones pre establecidas para acceder a la aplicación TRANSUR.

Condiciones de ejecución:

El usuario deberá previamente estar registrado, para acceder a la aplicación TRANSUR.

Entrada

El usuario ingresa su email (usuario) y password.

Resultado esperado

Si el usuario logueado cumple con las reglas de validación y restricciones, tendrá acceso a la aplicación TRANSUR, con roles de autenticado y/o administrador.

Si el usuario logueado No cumple con las reglas de validación y restricciones, obtendrá mensajes en pantalla de usuario No registrado.

Evaluación de la prueba

Prueba satisfactoria

Registro de información incorrecta

3.3.3.2 Especificación de Prueba: Registro de Usuarios TRANSUR

Historia de usuario 2

Versión <1.1>

Historial de Revisiones

Tabla 68: Iteración I, Historial de revisiones Nro. 2

Fecha	Versión	Descripción	Autor
30/08/2014	1.1	Revisión	Fernando Cangás
30/08/2014	1.1.1	Modificación	Fernando Cangás

Fuente: Propia

En la historia H2 Registro de Usuarios TRANSUR, se hace el ingreso de datos personales de los usuarios, entre los campos a ingresarse se tienen nombres completos, email, usuario, password, repetir password. En registro de usuarios verificara las validaciones en los campos del formulario, verificando con especial énfasis el email y el password. La H2 Registro de Usuarios TRANSUR, es un prerequisite para la H1 Registro y Autenticidad.

Registro y Autenticación

Descripción

La información ingresada en el formulario deberá cumplir, con las validaciones y restricciones pre establecidas para acceder a la aplicación TRANSUR.

Condiciones de ejecución:

El usuario deberá previamente deberá tipear los datos personales, correctamente validados.

Entrada

El usuario tipea el ingreso de los siguientes campos del formulario: nombres completos (full name), email, usuario, password, repetir password.

Resultado esperado

Si los datos ingresados cumplen con las reglas de validación y restricciones, el usuario tendrá acceso a la aplicación TRANSUR, a través del logueo con el rol de autenticado.

Si los datos ingresados No cumplen, con las reglas de validación y restricciones, obtendrá mensajes en la pantalla de usuario.

Evaluación de la prueba

Prueba satisfactoria

Registro de Usuarios Incorrecto

3.3.3.3 Especificación de Prueba: Registro de Información Institucional

Historia de usuario 3

Versión <1.2>

Historial de Revisiones

Tabla 69: Iteración I, Historial de revisiones Nro. 3

Fecha	Versión	Descripción	Autor
09/09/2014	1.2	Revisión	Fernando Cangás
09/09/2014	1.2.1	Modificación	Fernando Cangás

Fuente: Propia

En la historia H3 Registro de Información Institucional, se hace el ingreso de datos institucionales, entre los campos a ingresarse se tienen la razón social, titular, ruc,

misión, visión, resoluciones, comisiones. En el registro de información institucional se verificara la validación de campos de formulario así como la dirección de ubicación de los recursos (documentos pdf) alojados en los repositorios.

Información Institucional

Descripción

La información ingresada en el formulario deberá cumplir, con las validaciones y restricciones establecidas en la aplicación TRANSUR.

Condiciones de ejecución:

El usuario deberá ingresar la información en los formularios previamente validados.

Entrada

El usuario tipea el ingreso de los siguientes campos en el formulario: la razón social, titular, ruc, misión, visión, resoluciones, comisiones

Resultado esperado

Si los datos ingresados cumplen con las reglas de validación y restricciones, el usuario podrá guardar / grabar la información en la base de datos.

Si los datos ingresados No cumplen, con las reglas de validación y restricciones, obtendrá mensajes de errores de validación en la pantalla de usuario.

Evaluación de la prueba

Prueba satisfactoria

Registro de Información Institucional Incorrecto

3.3.3.4 Especificación de Prueba: Registro de Socios

Historia de usuario 4

Versión <1.3>

Historial de Revisiones

Tabla 70: Iteración I, Historial de revisiones Nro. 4

Fecha	Versión	Descripción	Autor
09/09/2014	1.3	Revisión	Fernando Cangás
09/09/2014	1.3.1	Modificación	Fernando Cangás

Fuente: Propia

En la historia H4 Registro de Socios, se hace el ingreso la información de los socios institucionales, entre los campos a ingresarse se tienen:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección

Datos institucionales

Placa /Disco Nro., Inicio / Actividades, Término / Actividades

Licencia de conducir

Tipo/Licencia, Fecha expedición, Fecha caducidad.

Los campos de los formulario serán validados y controlados, revisando una posible duplicidad del Nro. CI de cada socio.

Información del Socio

Descripción

La información ingresada en el formulario deberá cumplir, con las validaciones y restricciones establecidas en la aplicación TRANSUR.

Condiciones de ejecución:

El usuario deberá ingresar la información en los formularios previamente validados.

Entrada

El usuario tipea el ingreso de los siguientes campos en el formulario:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección

Datos institucionales

Placa /Disco Nro., Inicio / Actividades, Término / Actividades

Licencia de conducir

Tipo/Licencia, Fecha expedición, Fecha caducidad.

Resultado esperado

Si los datos ingresados cumplen con las reglas de validación y restricciones, el usuario podrá guardar / grabar la información en la base de datos.

Si los datos ingresados No cumplen, con las reglas de validación y restricciones, obtendrá mensajes de errores de validación en la pantalla de usuario.

Evaluación de la prueba

Prueba satisfactoria

Registro de Información del Socio Incorrecta

3.3.3.5 Especificación de Prueba: Registro de Conductores

Historia de usuario 5

Versión <1.4>

Historial de Revisiones

Tabla 71: Iteración I, Historial de revisiones Nro. 5

Fecha	Versión	Descripción	Autor
09/09/2014	1.3	Revisión	Fernando Cangás
09/09/2014	1.3.1	Modificación	Fernando Cangás

Fuente: Propia

En la historia H5 Registro de Conductores, se hace el ingreso la información de los conductores, entre los campos a ingresarse se tienen:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección.

Licencia de conducir

Tipo/Licencia, Fecha expedición, Fecha caducidad.

Los campos del formulario serán validados y controlados al ingreso, para verificar la duplicidad del Nro. CI de cada conductor.

Información del Conductor

Descripción

La información ingresada en el formulario deberá cumplir, con las validaciones y restricciones establecidas en la aplicación TRANSUR.

Condiciones de ejecución:

El usuario deberá ingresar la información en los formularios previamente validados.

Entrada

El usuario tipea el ingreso de los siguientes campos en el formulario:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección

Licencia de conducir

Tipo/Licencia, Fecha expedición, Fecha caducidad.

Resultado esperado

Si los datos ingresados cumplen con las reglas de validación y restricciones, el usuario podrá guardar / grabar la información en la base de datos.

Si los datos ingresados No cumplen, con las reglas de validación y restricciones, obtendrá mensajes de errores de validación en la pantalla de usuario.

Evaluación de la prueba

Prueba satisfactoria

Registro de Información del Conductor Incorrecta

3.3.4. Resultado de la 1ª Iteración

Plan de entrega inicial

Historias

H1: Control y Acceso a sistema.

H2: Gestión de usuarios.

H3: Asignación de Roles a un usuario.

H4: Asignación de Responsabilidades a un usuario.

H5: Asignación de permisos hacia pantallas

Versión del sistema TRANSUR 1.4

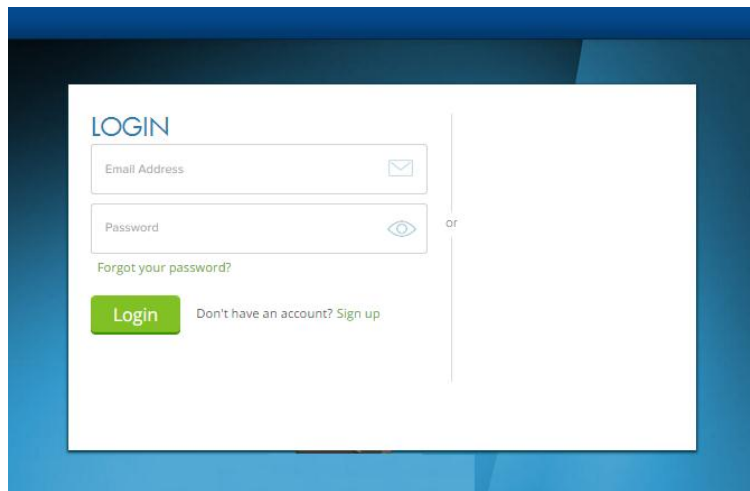
Modificación de requerimientos

- 1.- Registro y Autenticación
- 2.- Registro de Usuarios TRANSUR
- 3.- Registro de información institucional
- 4.- Registro de Socios
- 5.- Registro de Conductores

3.3.5. Demo de la versión, Iteración I

Historia 1: Registro y Autenticación

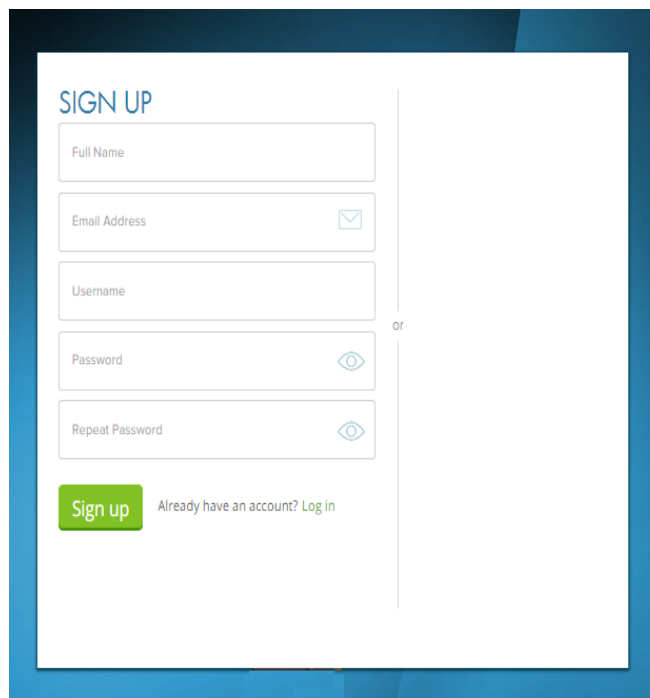
El formulario Login, solicita el ingreso de información, para la validación de campos email /user y password, al realizar el submit, se verificara la información en el servidor , el cual responderá de forma favorable o enviara un mensaje de error.



The image shows a login form titled "LOGIN" on a white background with a blue border. It features two input fields: "Email Address" with an envelope icon and "Password" with an eye icon. Below the password field is a link "Forgot your password?". A green "Login" button is positioned to the left of the text "Don't have an account? Sign up". A vertical line with the word "or" is on the right side of the form.

Fig.8: Formulario Login
Fuente: Propia

Historia 2: Registro de Usuarios TRANSUR



The image shows a sign-up form titled "SIGN UP" on a white background with a blue border. It features five input fields: "Full Name", "Email Address" with an envelope icon, "Username", "Password" with an eye icon, and "Repeat Password" with an eye icon. A green "Sign up" button is positioned to the left of the text "Already have an account? Log in". A vertical line with the word "or" is on the right side of the form.

Fig. 9: Formulario Registro
Fuente: Propia

Historia 4: Registro de Socios

Socios / Nuevo - Socio

Datos personales

CI / Socio:

Nombres:

Apellidos:

Estado civil:

Nro. telefono:

Correo electrónico:

Dirección

Información institucional/ Socio

Placa /Disco Nro:

Inicio / Act.:

Término / Act.:

Licencia de conducir

Tipo/Licencia:

Fecha expedición / Lic.:

Fecha caducidad / Lic.:

Fig.10: Formulario Registro de Socios
Fuente: Propia

Socios / listar - Socios

Buscar Socios ...

CI	Apellidos	Nombres	Unidad		
1002366564	Huera Fuel	Carlos Andres	PLACA15	Editar	Remover
1008881234	Alfaro	Juaquin Patricio	PLACA14	Editar	Remover
1004564455	Barragan	Juan	PLACA14	Editar	Remover
1004567894	Caicedo	Juan	PLACA14	Editar	Remover
1001234569	Reinoso	Juan	PLACA13	Editar	Remover
1002005665	Benavides	Juan	PLACA11	Editar	Remover
1001234568	Almeida	Arturo	PLACA14	Editar	Remover
1002004567	Heredia	Helber	PLACA11	Editar	Remover

Fig.11: Formulario Registro de Socios
Fuente: Propia

Socios / Ver - Socio

1002366564 Huera Fiel Carlos Andres [Editar](#) [Remover](#) [Exportar](#)

	CI	Nombres	Apellidos
	1002366564	Carlos Andres	Huera Fiel
	Estado civil	Nro. Telefono	Email
	Casado	0945646	carlos@yahoo.es
	Dirección		
	Quito		
	Placa / Disco Nro	Fecha Inicio Act	Fecha Termino Act
	PLACA15	Apr 28, 2015	Apr 28, 2015
	Tipo / Licencia	Fecha expedicion/Lic	Fecha caducidad/Lic
	E1	Apr 28, 2015	Apr 28, 2015

Fig.12: Formulario Registro de Socios
Fuente: Propia

Socios / Actualiza-Socio

[http://media.wixapps.net/wix-523f7349-203e-4eab-86c9-b7ae9e987bdf/images/77900a2835b34df7ab53b413b2c24124/v1/fbi/w_3f](#)

Datos personales

CI / Socio: 1002366564 Nombres: Carlos Andres Apellidos: Huera Fiel

Estado civil: Casado Nro. telefono: 0945646 Correo electrónico: carlos@yahoo.es

Dirección: Quito

Información institucional/ Socio

Placa /Disco Nro: PLACA15 Inicio / Act.: 2015-04-28 Término / Act.: 2015-04-28

Licencia de conducir

Tipo/Licencia: E1 Fecha expedición / Lic.: 2015-04-28 Fecha caducidad / Lic.: 2015-04-28

[Submit](#)

Fig.13: Formulario Registro de Socios
Fuente: Propia

Historia 5: Registro de Conductores

MEAN Instituciones Gestión Unidades Gestión Boletería Gestión-personal Gestión Rutas pepe naranjo

Choferes / Nuevo - Chofer

create choferes

Datos Conductores

CI: Ingrese CI

Nombres: Nombres

Apellidos: Apellidos

Estado civil: [dropdown]

Nro. telefono: Telefono

Corre electrónico: Correo electronico

Dirección: Dirección

Licencia de conducir

TipoLicencia: [dropdown]

Fecha expedición / Lic: Thu May 14 2015 14:57:36 GI

Fecha caducidad / Lic: Thu May 14 2015 14:57:36 GI

Submit

Fig. 14: Formulario Registro de Conductores
Fuente: Propia

MEAN Instituciones Gestión Unidades Gestión Boletería Gestión-personal Gestión Rutas pepe naranjo

Choferes / listar - Choferes

list choferes

Buscar Choferes... Export

CI	Apellidos	Nombres	Tipo / Licencia	
2001234567	Calderon	Juan	E	Editar Remover
2007894562	Bedon Almeida Taez	Ernesto Francisco	E	Editar Remover
2003214323	Calderon	Luis	D1	Editar Remover
2003214567	Calcedo	Luis	E	Editar Remover
2002314567	Andrade	Julian	D1	Editar Remover
2003004004	Navamete Umulia	Camilo Ernesto	E	Editar Remover
2003004003	Mendez	Lucio Tarciso	D1	Editar Remover
2002013023	Mendoza	Juan	E	Editar Remover

Fig.15: Formulario Registro de Socios
Fuente: Propia

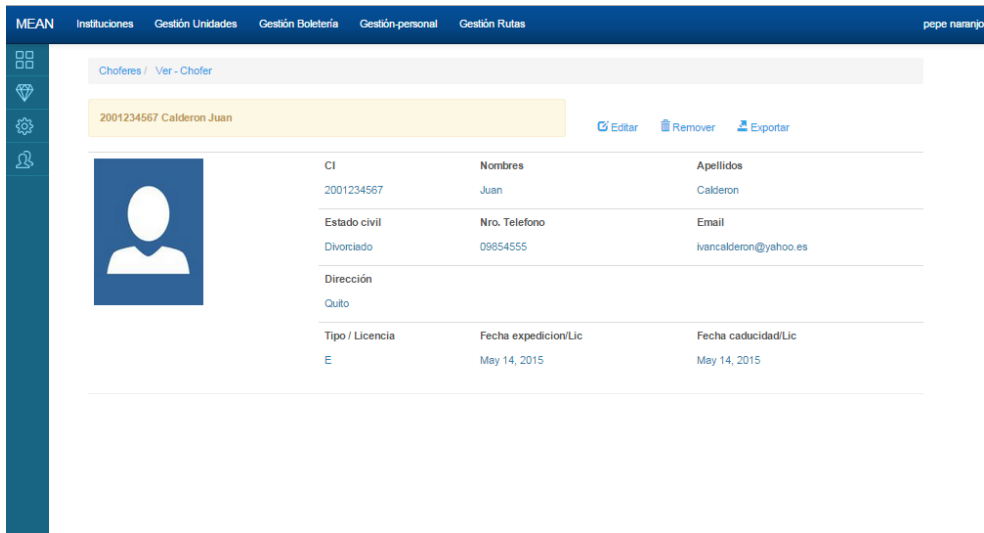


Fig.16: Formulario Registro de Socios
Fuente: Propia

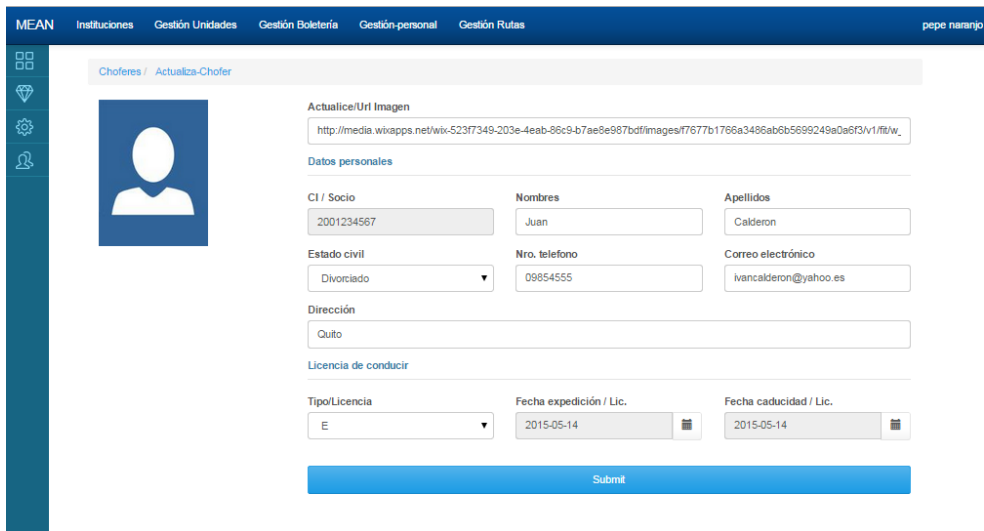


Fig. 17: Formulario Registro de Socios
Fuente: Propia

Pruebas de aceptación

H1 Registro y Autenticación

El formulario Login, solicita el ingreso de información, para la validación de los campos: email /user y password, al realizar el submit, se verificara la información en el servidor, el cual responderá de forma favorable o enviara un mensaje de error.

H2 Registro de Usuarios TRANSUR

El formulario de registro de Usuarios TRANSUR, solicita el ingreso de información, para la validación de los campos: nombre completo, email, username, password y repetir password. Al realizar el submit, se verificara la información en el servidor, el cual responderá de forma favorable o enviara un mensaje de error.

H3 Registro de información institucional

El formulario de información institucional, solicita el ingreso de información, para la validación de los campos: razón social, ruc, titular, misión, visión, resoluciones de la junta directiva y comisiones. Con el ingreso de información en cada campo se verificara su validez y al realizar el submit, se verificara la información en el servidor, el cual responderá de forma favorable o enviara un mensaje de error.

H4 Registro de Socios

El formulario de registro de socios, solicita el ingreso de información, para la Validación de los campos:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección

Información institucional/ Socio

Placa /Disco Nro, Inicio / Act, Término / Act.

Licencia de conducir

Tipo/Licencia, Fecha expedición / Lic, Fecha caducidad / Lic.

Con el ingreso de información en cada campo se verificara su validez y al realizar el submit, se verificara la información en el servidor, el cual responderá de forma favorable o enviara un mensaje de error.

H5 Registro de conductores

El formulario de registro de conductores, solicita el ingreso de información, para la Validación de los campos:

Datos personales

CI / Socio, Nombres, Apellidos, Estado civil, Nro. Teléfono, Correo electrónico, Dirección

Licencia de conducir

Tipo/Licencia, Fecha expedición / Lic, Fecha caducidad / Lic.

Con el ingreso de información en cada campo se verificara su validez y al realizar el submit, se verificara la información en el servidor, el cual responderá de forma favorable o enviara un mensaje de error.

Las pruebas o test, se los realiza al término de cada tarea de las historias de usuario, para ir mitigando posibles inconvenientes originados en el transcurso del desarrollo de la aplicación.

3.3.6. Incidencias

Las validaciones en los campos de los formularios deben ser estrictamente impecables, para evitar el ingreso de información anómala en la base de datos.

El modelaje inicial de las colecciones y de los documentos se somete a constantes actualizaciones, debido a las restricciones de validación en el formulario, resultando compleja la adaptación a este concepto de persistencia.

Al finalizar la segunda iteración con las pruebas de aceptación, se muestra la estructura del modelo de la base de datos.

3.3.8 Producción

Para la puesta en marcha de una aplicación MEAN stack, se necesita de los pasos mencionados a continuación:

Registro de una cuenta de Google

Contar con una cuenta en Pay Pal, o una cuenta de tarjeta de crédito /compra

Crear una cuenta en Google Cloud Platform, una vez creada la cuenta, se realiza las siguientes configuraciones:

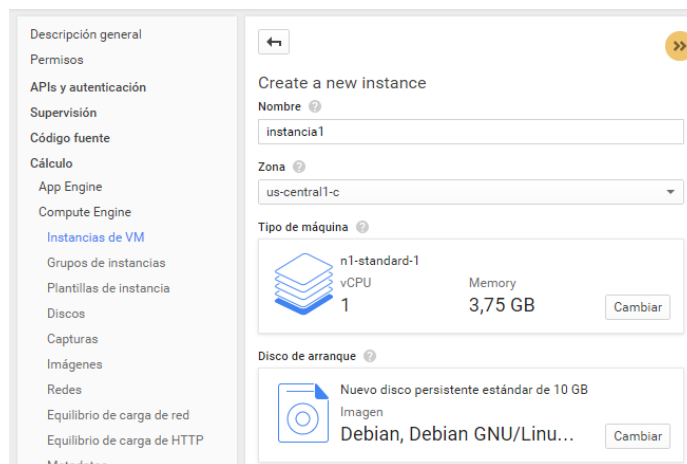


Fig. 19: Instancia del Servidor DEBIAN GNU/Linux
Fuente: Propia

Se inicia una nueva instancia y se elige la zona horaria del servidor, DEBIAN GNU/Linux



Fig. 20 Monitoreo de Instancia del Servidor
Fuente: Propia

En la figura anterior se muestra la instancia creada con los siguientes parámetros:
Nombre, Zona, Disco, Red, Usadas por: IP Externa, Conectar.

El icono IP Externa se selecciona, y muestra un menú de tareas, donde se crea una nueva tarea con el nombre: mean y puerto output: tcp: 3000, se guarda la configuración y la página de bienvenida del hosting se refleja en el navegador

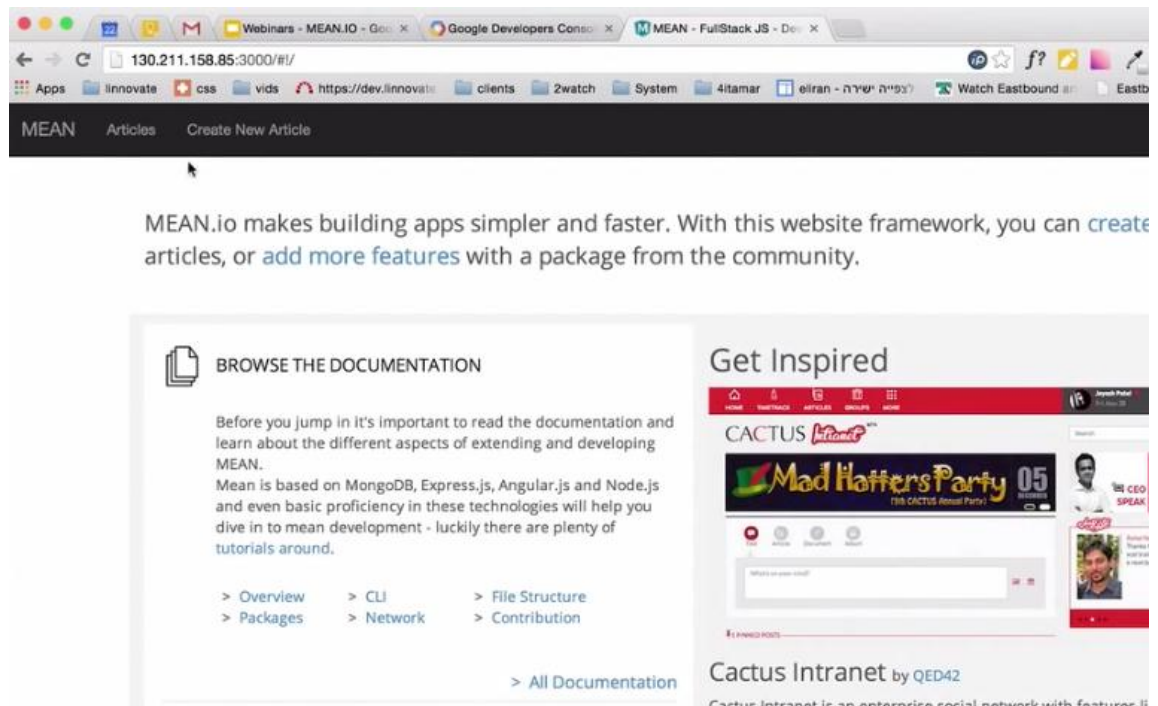


Fig. 21 Pantalla de Bienvenida del Servidor

Fuente: Propia

3.3.9 Publicación de la aplicación TRANSUR

Con la instalación de Git bash 1.9.5, seteamos al servidor y subimos los recursos de la aplicación al directorio remoto del hosting DEBIAN GNU/Linux

Para la aplicación TRANSUR , el directorio donde se encuentra el código fuente de la aplicación se encuentra en : \elapp\packages\unidades

El directorio \unidades contiene el código fuente de la aplicación TRANSUR.

CAPÍTULO IV

Análisis Costo Beneficio, Conclusiones y Recomendaciones

- 4.1 Análisis Costo Beneficio
- 4.2 Conclusiones
- 4.3 Recomendaciones

4.1 Análisis Costo Beneficio

El análisis costo-beneficio es una herramienta financiera encargada de determinar o medir la relación entre los costos y beneficios asociados a un proyecto de inversión, con el fin de evaluar su rentabilidad. (InfoSimilares, 2013) Entendiéndose por proyecto de inversión, no solo a la creación de un nuevo negocio, sino también, a posibles inversiones u adquisiciones que se realicen en un negocio establecido.

Para citar un ejemplo este podría ser la adquisición de nueva maquinaria o la instalación de una aplicación informática para la empresa o institución.

4.1.1 Objetivo

Determinar la viabilidad de un proyecto, mediante el análisis de costos.

4.1.2 Costos

El análisis Costo-Beneficio, tiene como objetivo fundamental determinar la rentabilidad de un proyecto, mediante la comparación de los beneficios esperados contra los resultados de la producción del proyecto.

Tabla 72: Costos de implementación del proyecto TRANSUR

Hardware			
1	Computador de escritorio	700	0,00
Subtotal			0,00
Alojamiento de aplicación (Hosting)			
1	Servicio de Google Stack (MEAN.IO)	50	50
Subtotal			50
Software			
1	Node.js	0,00	0,00
1	Mongo DB	0,00	0,00
7	Programador Stack MEAN.IO(\$ 400,00)	2800,00	2800,00
Subtotal			2800,00
Servicios			
7	Electricidad \$ 15,00 mensual	105,00	105,00
7	Servicio de Internet 25,00 mensual	175,00	175,00
Subtotal			280,00
Varios			
1	Transporte	25,00	25,00
1	Materiales de oficina	60,00	60,00
Subtotal			85,00
Total			\$ 3215,00

Fuente: Propia

Gastos operativos institucionales

La Cooperativa de transporte inter cantonal Urcuquí, alquila mensualmente el servicio CNT de telefonía fija y móvil, según datos consultados en: (Bulletin Solutions, 2014) las tarifas de servicio de llamadas telefónicas son:

Tabla 73: Lista de costos de servicios de telefonía CNT

Servicio Básico de Telefonía		\$ 6,20
Extras del servicio básico		
Origen	Destino	Precio
CNT Fijo	CNT Fijo	\$ 0,04 + IVA
CNT Fijo	Otros Fijos	\$ 0.10 + IVA
CNT Fijo	Móviles	\$ 0,15 + IVA

Fuente: Personal

Estimación de gastos operativos

Con relación a la tabla anterior se realiza la estimación de gastos en las actividades de petición - respuesta de información. En la institución en días y horas laborables se reciben un promedio de 16 peticiones de información en la secretaría, lo cual incurre en gastos por parte de los solicitantes, dichas peticiones generalmente se las realiza con el uso del servicio de telefonía móvil teniendo un costo de \$ 0.15 + IVA por minuto.

Este es un gasto que no afecta a la institución pero si a los socios y personas afines de la institución, tiene un costo promedio de \$ 0.25 dólares por solicitud, haciendo un total de \$ 4,00 dólares diarios, correspondiente a \$ 120,00 dólares mensuales y \$ 1440,00 dólares anuales. Una vez recibida la petición en secretaría, la contestación se la realiza con el uso de llamadas de telefonía fija a móvil con una duración estimada entre 1 y 2 minutos con un gasto diario que corresponden a 16 respuestas por \$ 0,15 equivale a \$ 2.40 dólares, correspondiente a \$ 72,00 dólares mensuales y \$ 864,00 dólares anuales.

Tabla 74: Estimación de gastos operativos

Estimación de gastos operativos			
Usuarios	Actividad	Gasto mensual	Gasto Anual
Personal institucional	Peticiones de información	\$120,00	\$1440,00
Personal Administrativo	Respuesta de peticiones	\$72,00	\$864,00
Total			\$2304,00

Fuente: Propia

Estimación de los gastos operativos con la implementación de la aplicación TRANSUR

Con la implementación de la aplicación TRANSUR, se reducen drásticamente las peticiones de información del promedio inicial de 16 peticiones a 5, con un costo promedio de \$ 0,25 dólares por solicitud por parte del personal a fin de la institución, haciendo un total de \$ 1,25 dólares diarios, correspondiente a \$ 37,50 dólares mensuales y \$ 450,00 dólares anuales.

A1 recibir la petición en secretaría, la contestación se la realiza con el uso de llamadas de telefonía fija a móvil con una duración estimada entre 1 y 2 minutos con un gasto diario que corresponden a 5 respuestas por \$ 0,15 equivale a \$ 0.75 dólares, correspondiente a \$ 22,5 dólares mensuales y \$ 270,00 dólares anuales.

Tabla 75: Estimación de gastos operativos con la aplicación TRANSUR

Estimación de gastos operativos con la aplicación TRANSUR			
Usuarios	Actividad	Gasto mensual	Gasto Anual
Personal institucional	Peticiones de información	\$37,50	\$450,00
Personal Administrativo	Respuesta de peticiones	\$22,50	\$270,00
Total			\$720,00

Fuente: Propia

Conclusión de gastos operativos con la implementación de la aplicación TRANSUR

Tabla 76: Comparativa de gastos operativos

Comparativa de gastos operativos	
Estimación de gastos operativos	\$2304,00
Estimación de gastos operativos con la implementación de la aplicación TRANSUR	\$720,00
Ahorro de gasto operativo anual	\$1584,00

Fuente: Propia

Los datos mostrados en la tabla anterior, concluyen que el gasto operativo institucional anual, se reduciría de \$ 2304,00 dólares a \$ 720,00 dólares, con la implementación de la aplicación web TRANSUR, con un ahorro de \$ 1584,00 dólares en gastos operativos.

Rentabilidad del proyecto TRANSUR

Con el análisis de la relación costo-beneficio en un periodo de dos años, se determinara la rentabilidad de la aplicación.

Ahorro de gasto operativo mensual = Ahorro gasto operativo anual /12 (meses)

Ahorro de gasto operativo mensual: \$ 132,00 dólares

Tabla 77: Rentabilidad del proyecto TRANSUR

Periodo	Periodo/meses	Ahorro gastos operativos	Rentabilidad
		Periodo	
1 periodo	12 meses	\$ 1584,00	\$ 0,00
2 periodo	24 meses	\$ 3168,00	\$ 0,00
3 periodo	1 mes	\$ 3300,00 (\$3215,00)	\$ 85,00
	2 mes	\$ 3432,00	\$ 217,00
	3 mes	\$ 3564,00	\$ 349,00
	4 mes	\$ 3696,00	\$ 481,00
	5 mes	\$ 3828,00	\$ 613,00
	6 mes	\$ 3960,00	\$ 745,00

Fuente: Propia

Los datos mostrados en la tabla anterior reflejan la rentabilidad del proyecto Transur. Una vez superado los \$ 3215,00 de costos de implementación de la aplicación, al inicio del mes 26, se refleja una rentabilidad de \$ 85,00 dólares.

En el estudio de factibilidad del proyecto TRANSUR, se describieron los objetivos diagnósticos, las variables y los indicadores los cuales son representados en la siguiente tabla:

Tabla 78: Indicadores de estudio de factibilidad Transur

OBJETIVOS DIAGNÓSTICOS	VARIABLES	INDICADORES
Establecer los elementos en los que se basa la productividad interna en las oficinas de la cooperativa de Transporte de pasajeros inter cantonal Urcuquí.	P R O D U C T I V I D A D	Capacidad de respuesta a las Solicitudes de información por hora.
Efectuar un análisis de la capacidad de registro de unidades de transporte, rutas, frecuencias, socios y conductores de la cooperativa de transporte de pasajeros inter cantonal Urcuquí.		Capacidad de registro de información por hora

Fuente: Propia

Entre las conclusiones del estudio de factibilidad se citó que: La capacidad de respuesta del personal administrativo tiene un tiempo estimado entre 15 y 30 minutos, al realizarse las peticiones de información en días y horas laborables, de lo contrario no se realizan peticiones de información.

Con los indicadores se realiza la relación entre la capacidad de respuesta y la capacidad de registro antes y después de la implementación de la aplicación Transur.

Tabla 79: Indicadores con la implementación de la aplicación Transur

Indicador	Hojas de Trabajo	App. Transur
Capacidad de respuesta a las Solicitudes de información por hora	Lunes - Viernes	Todos los días
	15-30 min	1 – 2 min
	4	30
Indicador	Hojas de Trabajo	App. Transur
Capacidad de registro de información por hora(Rutas)	Lunes-Viernes	Todos los días
	60 min	60 min
	10	50

Fuente: Propia

La tabla anterior muestra un incremento significativo en la capacidad de respuesta a solicitudes y registro de información en Rutas.

4.2 Conclusiones

La tecnología NodeJS, es una infraestructura compleja formada por diversos recursos alojados en internet, los cuales están respaldados por una gran comunidad de desarrolladores, tales recursos son alcanzables con escasas líneas de comandos.

Si bien la flexibilidad de MongoDB, es una de sus principales virtudes, se debería tener en cuenta un estricto control en la validación de la información a ingresar, para que el servidor se encuentre libre de realizar una revalidación de la información ingresada.

La implementación de una aplicación web con JavaScript residente en el lado del cliente y del servidor implica como requerimiento mínimo poseer un nivel intermedio de conocimientos sobre tal lenguaje de programación.

MEAN.IO es una pila de software análoga a LAMP (Linux, Apache, MySQL, PHP), la cual permite alojar aplicaciones con Node.js acompañadas de las tecnologías MongoDB, Mongoose, Express y Angular.

El servicio de alojamiento Hosting para aplicaciones MEAN.IO, es limitado, por lo que la plataforma de google se establece como la alternativa idónea para la publicación de la aplicación Transur detallada en el Capítulo III.

Los resultados de rentabilidad, inician desde el mes 26 de la puesta en producción del aplicativo, una vez que han sido superados los costos de la implementación y desarrollo de la aplicación Transur.

Los indicadores de productividad interna, se sustentan satisfactoriamente por medio de la evaluación de la capacidad de respuesta y registro de información, que tiene la Cooperativa de transporte Intercantonal Urcuquí.

4.3 Recomendaciones

Es recomendable para los programadores del Stack MEAN.IO, estar familiarizados con las tecnologías que ofrece esta pila de aplicaciones, por lo que la tecnología Angular, exige un estricto orden en la estructura de las aplicaciones.

Se recomienda estudiar la tecnología Mongoose, para gestionar la comunicación entre el controlador del lado del servidor y la base de datos, para de esta manera liberar al programador de escribir cientos de líneas de código.

Es recomendable analizar los servicios de alojamiento de aplicaciones web (Hosting) para las aplicaciones MEAN.IO, por lo que la oferta más costosa no siempre es la más fiable.

Se recomienda antes de realizar una contratación de los servicios de Hosting, crear una cuenta en Paypal, para que intervenga de manera oportuna en el/los pagos por servicios.

Es recomendable, revisar los backups, descolgados de la base de datos y revisarlos exhaustivamente, de manera que al volverlos a subir al sistema Transur, no se encuentren inconsistencias en la aplicación.

Las eventuales peticiones de información de las instituciones reguladoras de tránsito y transporte público, incurrirán en actualizaciones de la aplicación Transur, siendo recomendable un continuo mantenimiento con programadores especializados en MEAN.IO.

Bibliografía

- Adolfo Segovia Ampuero. (3 de Julio de 2013). <http://prezi.com/>. Recuperado el 20 de Octubre de 2013, de <http://prezi.com/ln6zkgun4cao/introduccion-nodejs-y-socketio/>
- Antioquia, U. d. (5 de Enero de 2011). <http://aprendeenlinea.udea.edu.co/>. Recuperado el 20 de Octubre de 2013, de <http://aprendeenlinea.udea.edu.co/lms/ova/mod/resource/view.php?id=1599>
- Castillo, O., Figueroa, D., & Sevilla, H. (s.f.). <http://programacionextrema.tripod.com/>. Recuperado el 10 de Diciembre de 2013, de <http://programacionextrema.tripod.com/>: <http://programacionextrema.tripod.com/fases.htm>
- Danielzs75.blogspot.com. (1 de Octubre de 2012). <http://danielzs75.blogspot.com/>. Recuperado el 11 de Diciembre de 2013, de <http://danielzs75.blogspot.com/>: <http://danielzs75.blogspot.com/2012/10/metodologias-agiles-extreme-programming.html>
- Dolce, J. (1984). *Fleet management*. New York : McGraw-Hill.
- Eric Ries. (5 de Enero de 2012). <http://www.dontknow.net/>. Recuperado el 20 de Octubre de 2013, de <http://www.dontknow.net/decision/montar-start-tecnologica>
- Freud Jhon. (1977). *Estadística elemental moderna*. La Habana: Pueblo y educación.
- German Marcano. (s.f.). http://curso_sin2.blogia.com/. Recuperado el 5 de Diciembre de 2013, de http://curso_sin2.blogia.com/: http://curso_sin2.blogia.com/2005/070905-extreme-programming-x.p.-por-german-marcano.php
- Guillermo Mir. (17 de Junio de 2013). <http://www.securityartwork.es/>. Recuperado el 26 de Noviembre de 2013, de <http://www.securityartwork.es/>: <http://www.securityartwork.es/2013/06/17/nuevas-tecnologias-viejas-vulnerabilidades-node-js-i/>
- Islas, V., Torres, G., & Rivera, C. (2000). *PRODUCTIVIDAD EN EL TRANSPORTE MEXICANO*. Mexico: Sanfandila.
- Ivan de Dios. (14 de Marzo de 2011). <http://www.aunclidelastic.com/>. Recuperado el 21 de Octubre de 2013, de <http://www.aunclidelastic.com/cloud-computing/casos-de-uso-de-cloud-computing/>
- Kiessling, M., & Jungle, H. (2012). *El Libro para Principiantes en Node.js*.

MongoDB Inc. (s.f.). <http://www.mongodb.org/>. Recuperado el 4 de Diciembre de 2013, de <http://www.mongodb.org/>: <http://www.mongodb.org/>

Msc Walter Jacome. (2009). *Bases teóricas y practicas para el diseño y evaluación de proyectos productivos y de inversión*. Ibarra: Editorial UTN.

Socket.IO. (s.f.). <http://socket.io/>. Recuperado el 5 de Diciembre de 2013, de <http://socket.io/>: <http://socket.io/>

TJ Holowaychuk. (4 de Junio de 2010). <http://tjholowaychuk.com/>. Recuperado el 4 de Diciembre de 2013, de <http://tjholowaychuk.com/>: <http://tjholowaychuk.com/post/664516126/connect-middlewares-for-nodejs>

GLOSARIO

AJAX.- JavaScript asíncrono y XML, es una técnica de desarrollo web para crear aplicaciones interactivas.

API.- Interfaz de programación de aplicaciones, está formado por un conjunto de comandos, funciones y protocolos que los programadores pueden utilizar en la construcción de software.

CSS.- hojas de estilo usado para describir la presentación semántica de un documento escrito en lenguaje HTML.

DBMS.- Sistema de Gestión de Bases de Datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos.

DOM.- El Modelo de Objetos del Documento es una interfaz de la plataforma y del lenguaje neutro que permitirá a los programas y scripts acceder y actualizar dinámicamente el contenido, la estructura y el estilo de los documentos.

HTML.- (lenguaje de marcas de hipertexto) Es un lenguaje de marcado para la elaboración de páginas web.

ISS.- Servidor web y conjunto de servicios para el sistema operativo Microsoft Windows.

JSON.- Acrónimo de JavaScript Object Notation, es un formato ligero para el intercambio de datos.

URL.- Localizador de recursos uniforme, la primera parte de la dirección indica qué protocolo utilizar, la segunda parte especifica la dirección IP o nombre de dominio donde se localiza el recurso.

W3C.- El Consorcio World Wide Web es una organización encargada de regular los estándares web.

WSDL.- describe la interfaz pública a los servicios Web. Está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo.

XML.- Es un lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible.

ANEXOS

**Universidad Técnica del Norte
Carrera de Ingeniería en Sistemas computacionales**

**SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA
COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUÍ.**

Socios – conductores de la cooperativa de Transporte de pasajeros inter cantonal Urcuquí

Nombre:

Señale el tipo de actividad que realiza en la cooperativa:

Socio Conductor

Encuesta

El siguiente cuestionario tiene incidencia directa sobre la continuidad del presente proyecto, por lo que se le solicita, leer detenidamente cada enunciado y señalar la respuesta correcta con una (X).

1. Cuando usted realiza una petición de información a secretaría, sobre información interna de la cooperativa como: socios, conductores, rutas y frecuencias de unidades de transporte, estas peticiones son contestadas de forma:

- | | | | |
|------------------------|--------------------------|-----------------------|--------------------------|
| a) Inmediata | <input type="checkbox"/> | e) 1 hora en adelante | <input type="checkbox"/> |
| b) De 5 a 10 minutos | <input type="checkbox"/> | f) 1 día | <input type="checkbox"/> |
| c) De 15 a 30 minutos | <input type="checkbox"/> | g) Más de 1 día | <input type="checkbox"/> |
| d) 30 minutos a 1 hora | <input type="checkbox"/> | h) Nunca | <input type="checkbox"/> |

2. Cuando usted necesita realizar una petición de información interna de la cooperativa sobre: conductores, rutas y frecuencias de unidades de transporte, en horas NO LABORABLES O FERIADOS, recurre a:

- a) Otros socios
- b) Otra forma ¿Cuál?:
.....

3. ¿Este inconveniente se solucionaría, si se tuviera la información a su alcance?

SI NO

4. ¿Qué medio de información, sería el más fiable para garantizar información actualizada y permanente (24 horas)?.

- | | | | |
|--|--------------------------|------------|--------------------------|
| a) Hojas de trabajo | <input type="checkbox"/> | d) Ninguna | <input type="checkbox"/> |
| b) Reportes semanales de secretaría | <input type="checkbox"/> | e) Otro | <input type="checkbox"/> |
| c) Aplicación informática en Internet (Software) | <input type="checkbox"/> | | |

¿Cuál medio?

Sistema web TRANSUR

5. En el caso de que la respuesta anterior fuese: c) Aplicación informática en Internet. ¿Usted puede afirmar que una Aplicación Informática (TRANSUR) alojada en internet con información interna de la cooperativa sobre unidades de transporte, socios conductores, rutas y frecuencias, con reportes o presentaciones de información actualizada, sustentara o apoyara la/las respuestas, a sus peticiones de información de forma permanente a cualquier hora y día de la semana?

SI NO

6. ¿Usted puede afirmar que la implementación (creación e instalación) de la aplicación informática TRANSUR, contribuirá con el desarrollo de la cooperativa mejorando su productividad interna?

SI NO

Universidad Técnica del Norte
Carrera de Ingeniería en Sistemas computacionales

SISTEMA WEB TRANSUR CON NODE.JS PARA LA GESTIÓN DE TRANSPORTE DE LA COOPERATIVA DE TRANSPORTE DE PASAJEROS INTER CANTONAL URCUQUÍ.
Personal administrativo de la cooperativa de Transporte de pasajeros inter cantonal Urcuquí

Nombre:

Encuesta

El siguiente cuestionario tiene incidencia directa sobre la continuidad del presente proyecto, por lo que se le solicita, leer detenidamente cada enunciado y señalar la respuesta correcta con una (X).

1. ¿Cuándo se solicita información interna de la cooperativa sobre: socios, conductores, rutas y frecuencias de unidades de transporte, la realiza con el apoyo de alguna herramienta informática (Aplicación de software) u otro medio?

SI NO

¿Cuáles son las herramientas que utiliza?

a) Word d) Hojas de trabajo
b) Excel e) Otros
c) Aplicación informática

¿Cuáles?.....

Entre las herramientas utilizadas ¿Cuáles son las que usa con más frecuencia y porque?

¿Con la utilización de las herramientas anteriores, la información se encuentra al alcance permanente (24 horas) de la presidencia, gerencia, socios y entes afines de la institución?

SI NO

¿Con la utilización de las herramientas anteriores, considera que la información se encuentra?

Segura (protegida) Centralizada (Base de datos) Fácil acceso (Internet)
Ninguna

2. Según su criterio ¿Es relativamente fácil, responder a las peticiones de información de presidencia, gerencia, socios, conductores y medios extra institucionales?

SI NO

3. ¿Si, se incrementara la cantidad de socios de la cooperativa, la respuesta a peticiones de la presidencia, gerencia, socios y entes afines de la institución, sería inmediata?

SI NO

4. Según su criterio ¿Qué medio de información, sería el más fiable (confiable) para garantizar información actualizada y permanente (24 horas)?

Sistema web TRANSUR

- a) Hojas de trabajo c) Tarjetas de rutas de unidades
b) Reportes semanales de secretaría d) Otros
c) Aplicación informática en Internet (Software)
¿Cuáles?.....

5. En el caso de que la respuesta anterior fuese: c) Aplicación informática en Internet. ¿Usted puede afirmar que una Aplicación Informática (TRANSUR) alojada en internet con información interna de la cooperativa sobre unidades de transporte, socios conductores, rutas y frecuencias, con reportes o presentaciones de información actualizada, sustentara o apoyara la/las respuestas, a las peticiones de información de la de organizaciones foráneas, presidencia, gerencia, socios y conductores de forma permanente a cualquier hora y día de la semana?

SI NO

6. ¿Usted puede afirmar que la implementación (creación e instalación) de la aplicación informática TRANSUR, contribuirá con el desarrollo de la cooperativa mejorando su productividad interna?

SI NO