



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

MAESTRÍA EN TELECOMUNICACIONES

TÍTULO DE LA TESIS:

Sistema de Seguimiento de Proyectos de Construcción y Mantenimiento para
Empresas de Telecomunicaciones a través de la Web”

Previa la obtención del Grado Académico de Magíster en Telecomunicaciones

ELABORADO POR:

Ing. Christian León Cercado

Guayaquil, a los 12 días del mes Junio año 2012



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por Magíster Christian León Cercado como requerimiento parcial para la obtención del Grado Académico de Magíster en Telecomunicaciones.

Guayaquil, a los 12 días del mes Junio año 2012

DIRECTOR DE TESIS

Nombre

REVISORES:

MsC. Edwin Palacios Meléndez

MsC. Luis Cordova Rivadeneira

DIRECTOR DEL PROGRAMA

MsC. Manuel Romero Paz



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

DECLARACIÓN DE RESPONSABILIDAD

YO, Christian León Cercado

DECLARO QUE:

La tesis “Sistema de Seguimiento de Proyectos de Construcción y Mantenimiento para Empresas de Telecomunicaciones a través de la Web”, previa a la obtención del grado Académico de Magíster, ha sido desarrollada en base a una investigación exhaustiva, respetando derechos intelectuales de terceros conforme las citas que constan al pie de las páginas correspondientes. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico de la tesis del Grado Académico en mención.

Guayaquil, a los 12 días del mes Junio año 2012

EL AUTOR

Christian León Cercado



UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

AUTORIZACIÓN

YO, Christian León Cercado

Autorizo a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución de la Tesis de Maestría titulada: “Sistema de Seguimiento de Proyectos de Construcción y Mantenimiento para Empresas de Telecomunicaciones a través de la Web”, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 12 días del mes Junio año 2012

EL AUTOR

Christian León Cercado

AGRADECIMIENTO

Agradezco en primer lugar a DIOS, que me da todo los días la oportunidad de crecer profesionalmente y es el que me ilumina a seguir por el buen camino, a mi esposa e hijas que son la parte más sacrificada en este sacrificio valga la redundancia ya que ellas son la que soportaban el que por mis estudios no podamos gozar de un fin de semana como familia, pero como siempre le decía a mi esposa y ella hoy en día me lo recalca, los beneficiados de mis esfuerzos siempre serán mi familia y es una enseñanza para mis hijas el ver que su padre se dedica o se esfuerza en avanzar en la selva de cemento, a mi madre que ha sido esa mujer de tesón y ñeque que siempre está ahí para dar sus palabras sabias y aparte inculcarnos el avanzar pues ella más que nadie nos demostró que no existe edad para estudiar y avanzar, y a mis hermanos que junto a mi madre y familia se vuelven los pilares fundamentales de mi vida y de la lucha de este ser por ser una persona que le suma a la sociedad de mi país .

También le agradezco a mi ALMA MATER formadora de seres importantes de este país y de mi persona.

DEDICATORIA

Padre, sé que estás en el cielo y siempre me guías con tus bendiciones, aunque no estés aquí físicamente, te dedico este esfuerzo pues siempre fuiste ese hombre que aunque no demostrabas afecto, te sentías dichoso de tus hijos pues junto a mi madre habían formados hombres de bien, leales, sinceros y sin malicias, características que hoy en día casi ya no existen en los hombres, te dedico esta victoria como que fuese una tuya, un paso tuyo en la vida y sigue iluminando mi caminar en esta batalla de llevar tu apellido y legado a lo más grande, a la inmortalidad .

ÍNDICE GENERAL

CAPÍTULO 1: INTRODUCCIÓN	12
1.1. ANTECEDENTES	14
1.2. JUSTIFICACIÓN DEL PROBLEMA DE INVESTIGACIÓN.....	16
1.3. DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN:.....	17
1.4. HIPÓTESIS:	17
1.5. OBJETIVOS	17
1.5.1. OBJETIVO GENERAL:	17
1.5.2. OBJETIVOS ESPECÍFICOS.....	17
1.6. METODOLOGÍA DE INVESTIGACIÓN:	18
MÉTODOS TEÓRICOS:.....	18
CAPÍTULO 2: TECNOLOGÍAS DE PROGRAMACIÓN WEB.	18
2.1. INTRODUCCIÓN A PHP.....	18
2.2. CARACTERÍSTICAS DE PHP.....	19
2.3. FUNCIONALIDAD DE PHP	21
2.4. PHP ORIENTADO A OBJETOS.	22
2.4.1. <i>Objetos</i>	22
2.4.2. <i>Clases</i>	23
2.4.3. <i>Declaración de clases</i>	25
2.4.4. <i>La palabra reservada new y los constructores</i>	25
2.4.5. <i>Los destructores</i>	27
2.4.6. <i>Accediendo a los métodos y atributos a través de la variable \$this</i>	28
2.4.7. <i>Encapsulación</i>	28
2.4.8. <i>Métodos public, protected y private</i>	31
2.4.9. <i>Herencia</i>	32
2.4.10. <i>Sintaxis de la herencia</i>	33
2.4.11. <i>Polimorfismo</i>	35
2.4.12. <i>El modelo objeto en PHP 5. Otras características</i>	39
2.4.13. <i>Posibilidad de uso de interfaces</i>	39
2.4.14. <i>Operador instanceof</i>	40
2.4.15. <i>Métodos y clases final</i>	40

2.4.16. Atributos y métodos <i>static</i>	41
2.4.17. Clases y métodos abstractos.....	42
2.4.18. Clases reservadas <i>self::</i> y <i>parent::</i>	43
2.4.19. Constantes de clase.....	44
2.4.20. Función <i>__autoload()</i>	45
2.4.21. Clonado de objetos.....	46
2.5. BASES DE DATOS EN LA WEB	48
2.5.1. BIBLIOTECAS DE ACCESO A DATOS EN PHP	49
2.6. ACCESO A BASES DE DATOS MYSQL EN PHP	50
2.7. SOPORTE PARA PROGRAMACIÓN WEB CON PHP Y MYSQL	54
2.7.1. SERVIDOR WAMP.	54
2.7.2. APPSERV.....	55
2.7.3. PHP	55
2.7.4. MYSQL	56
2.7.5. PHPMYADMIN	56
CAPÍTULO 3: DESARROLLO DE LA SOLUCIÓN PROPUESTA	57
3.1. ARQUITECTURA DE LA SOLUCIÓN.....	57
3.2. MODELADO DE LA BASE DE DATOS.	57
3.3. DISEÑO DE OBJETOS.....	59
3.4. ALCANCES Y RESTRICCIONES DEL SOFTWARE	70
3.4.1. RESTRICCIONES.....	71
3.4.2. GRAFO SIN CICLOS CON REGIÓN ÚNICA	72
3.4.3. GRAFOS CÍCLICOS	73
3.4.4. GRAFOS MULTI-CAMINO	74
3.5. SEGURIDAD Y CONTROL DE ACCESO	75
3.5.1. <i>Seguridad general de acceso a la base de datos</i>	76
3.5.2. <i>Seguridad de acceso al sistema</i>	76
3.5.3. <i>Seguridad de acceso a la información</i>	77
3.6. PARAMETRIZACIÓN GENERAL	78
3.6.1. <i>Administración de Feriados y Jornada laboral</i>	79
3.6.2. <i>Definición de Categorías de proyectos</i>	79

3.7. PLANTILLAS	80
3.7.1. <i>Definición de etapas de un proyecto</i>	82
3.7.2. <i>Definición de Actividades Generales de proyectos</i>	84
3.7.3. <i>Registro de plantillas por categoría de proyectos</i>	84
3.8. REGISTRO DE PROYECTOS	85
3.8.1. <i>Asignación de tiempos</i>	86
3.8.2. <i>Responsables de las actividades</i>	87
<i>Fuente: El autor</i>	88
3.8.3. <i>Elaboración de redes</i>	88
3.9. SEGUIMIENTO DE PROYECTOS	89
3.10. CÁLCULO DE COSTOS DEL PROYECTO	89
3.11. ÍNDICES DE EFICIENCIA	89
3.12. GRÁFICOS Y REPORTE.....	90
3.12.1. <i>Informe de Actividades pendientes</i>	90
3.12.2. <i>Informe de Actividades Finalizadas</i>	91
3.12.3. <i>Informe de Órdenes finalizadas</i>	93
3.12.4. <i>Comparativo de costos en Órdenes Finalizadas</i>	94
3.12.5. <i>Reporte de Orden de Trabajo</i>	95
3.12.6. <i>Reporte de Seguimiento de Órdenes</i>	96
CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES.....	99
4.1. CONCLUSIONES	99
4.2. RECOMENDACIONES.....	99
ANEXO A: GUÍA DE INSTALACIÓN.....	100
ANEXO II MANUAL DE USUARIO	101
ANEXO III GLOSARIO DE TÉRMINOS	117
ANEXO IV DICCIONARIO DE DATOS.....	118
BIBLIOGRAFÍA.....	153

ÍNDICE DE FIGURAS

Capítulo 2:

Figura 2. 1: Funcionamiento de PHP.....	21
Figura 2. 2: Diagrama de la clase persona.....	24
Figura 2. 3: Representación de la herencia entre las clases.....	33
Figura 2. 4: Esquema básico de un sitio Web soportado por bases de datos	48

Capítulo 3:

Figura 3. 1: Modelo de Órdenes o Proyectos	58
Figura 3. 2: Parámetros y Configuración	59
Figura 3. 3: Plantillas de Tipos de Proyectos.....	59
Figura 3. 4: Flujo de actividades con 1 sola región, grafo a cíclico.....	73
Figura 3. 5: Flujo de actividades cíclica, en este ejemplo el grafo posee 2 regiones	73
Figura 3. 6: Dos caminos diferentes desde el mismo origen y con el mismo destino	74
Figura 3. 7: Dos caminos diferentes hacia el mismo destino, desde orígenes distintos	75
Figura 3. 8: Tipos de roles predefinidos	76
Figura 3. 9: Parametrización por catálogo de tablas	79
Figura 3. 10: Definición de Tipos de Proyectos	80
Figura 3. 11: Definición de plantillas de proyectos	81
Figura 3. 12: Definición de usuarios que recibirán avisos por email de proyectos ...	82
Figura 3. 13: Etapas generales de la administración de Proyectos.....	83
Figura 3. 14: Definición de Actividades Generales	84
Figura 3. 15: Plantilla de Actividades para un tipo de Proyecto.....	85
Figura 3. 16: Registro de orden de trabajo para un proyecto	85
Figura 3. 17: Asignación de tiempos para las actividades de un tipo de proyecto....	86
Figura 3. 18: Registro de usuarios asignados a una actividad	87
Figura 3. 19: Flujo de actividades con 1 sola región, grafo acíclico.....	88
Figura 3. 20: Matriz de adyacencia, para flujo de actividades	88
Figura 3. 21: Datos para generar informe de actividades pendientes	90
Figura 3. 22: Ejemplo de informe de actividades pendientes	91

Figura 3. 23: Datos para generar informe de actividades finalizadas	92
Figura 3. 24: Ejemplo de informe de actividades finalizadas	93
Figura 3. 25: Datos para la generación de informe de órdenes finalizadas.....	93
Figura 3. 26: Ejemplo del informe de órdenes finalizadas	94
Figura 3. 27: Datos para la generación de informe de comparativo de costos.....	94
Figura 3. 28: Resumen de costos.....	95
Figura 3. 29: Reporte de ordenes de trabajos.....	96
Figura 3. 30: Seguimiento de órdenes.....	97
Figura 3. 31: Reporte de seguimiento de actividades por órdenes.....	97
Figura 3. 32: Reporte de seguimiento de ordenes de trabajos.....	98

Capítulo 1: Introducción

En los manuales y guías para la elaboración de proyectos, difundidos por las dependencias de la Organización de las Naciones Unidas (ONU), el Banco Mundial (BM) y el Banco Interamericano de Desarrollo (BID), aparecen, entre otras, las siguientes definiciones de proyecto:

1. "Proyecto es el conjunto de antecedentes que permiten estimar ventajas y desventajas económicas de la asignación de insumos para la producción de determinados bienes y servicios" (ONU, 1973).
2. "Un proyecto es una empresa planificada consistente en un conjunto de actividades interrelacionadas y coordinadas con el fin de alcanzar objetivos específicos dentro de los límites de un presupuesto y un periodo de tiempo dados" (ONU, 1984).
3. "Proyecto es una empresa que cuenta con recursos financieros, materiales y humanos y está orientada a la consecución de un objetivo en un periodo de tiempo dado" (BID, 1984).

Pasando a algunas de las definiciones propuestas por los especialistas, en los manuales sobre diseño, formulación y desarrollo de proyectos, se encuentran algunas como las siguientes:

4. "Proyecto es la menor unidad de actividades que se pueden planear, analizar y administrar aisladamente" (Salomón y Edin).
5. "Proyecto es un conjunto de actividades que tiene un principio y un fin definibles y que se puede administrar aisladamente para alcanzar el objetivo por el cual dicha actividad fue instituida" (R. L. Martino: Administración y control de proyectos, 1970).
6. "Proyecto es un proceso cuyo objetivo es transformar una idea en un producto terminado, constituido por bienes y servicios que serán los medios para producir otros bienes o servicios... Por lo general un proyecto nace de una institución existente que quiere hacer una modificación cualitativa o cuantitativa de la producción de su sistema de régimen. También puede ser

que el proyecto consista en crear una institución nueva, pero siempre habrá un organismo que formula y ejecuta el proyecto que es su entidad madre” (Luis Melgar y José Lefiero, 1979).

7. “El proyecto representa la unidad más operativa dentro del proceso de planificación y constituye el eslabón final de dicho proceso. Está orientado a la producción de determinados bienes o a prestar servicios específicos” (A.Pichardo, 1984).
8. Algunos autores consideran que “un proyecto es un instrumento para la acción que presupone la sistematización y ordenamiento de una serie de actividades específicas y la asignación de responsabilidades para alcanzar determinados objetivos”.
9. Otros autores, como Espinoza y Ander-Egg (1989) definen al proyecto en relación con un programa específico: es "la unidad menor en que se pueden separar las acciones concurrentes para el cumplimiento de los objetivos y metas de un programa, que involucra un estudio específico que permite estimar las ventajas o desventajas de asignar recursos para la realización de dicha acción" (Espinoza, 1989). Por su parte Ander-Egg (1989) define al proyecto como un "diseño o pensamiento de ejecutar algo; cualquier previsión, ordenación o predeterminación que se hace para la ejecución de una obra u operación.
10. “Componente o unidad más pequeña que forma parte de un programa. Conjunto de actividades que se proponen realizar de una manera articulada entre sí, con el fin de producir determinados bienes o servicios capaces de satisfacer necesidades o resolver problemas”.

El concepto de proyecto ha evolucionado particularmente durante el presente siglo, pasando del campo de la ingeniería al de la planeación del desarrollo económico y social, de modo que en la instrumentación de planes y programas de desarrollo los proyectos constituyen un elemento necesario e imprescindible. En un intento de síntesis integradora de las definiciones presentadas por los especialistas, “un proyecto es un proceso que describe la idea dinámica de una acción organizada

para lograr determinados fines u objetivos, que se puede planear, administrar y evaluar por sí mismo, constituyendo un ámbito de gestión peculiar que no puede ser confundido con el de la entidad madre o del sistema en donde surge". Pero no parece conveniente admitir que un proyecto se pueda planear y administrar aisladamente de cualquier referencia institucional o de determinados planes y programas que operan en el contexto en que ese proyecto surge y se desarrolla.

Un proyecto constituye un campo de acción, de responsabilidad y de gestión propio que, por su naturaleza peculiar, requiere de suficiente delegación por parte de la autoridad institucional y que no puede confundirse con los niveles ordinarios de autoridad institucional.

En congruencia con el concepto de planeación propuesto, un proyecto se podría definir como *"la descripción y desarrollo de un proceso metodológico clasificado en etapas y actividades, que permite organizar la resolución de un problema relevante y complejo, orientando los procesos de cambio constante, mediante la instrumentación de estrategias apropiadas, dentro de un límite de tiempo y espacio determinado y con los recursos adecuados"*.

1.1. Antecedentes

El mundo en la actualidad está cambiando a velocidades inusitadas y las organizaciones deben reaccionar rápidamente abordando proyectos que las ayuden a alcanzar nuevos objetivos. La administración de proyectos provee una metodología ordenada, sistemática y rigurosa para administrar los proyectos que enfrentan cada día las empresas y sus administradores. El adecuado conocimiento y aplicación de estas metodologías permiten crear un ambiente de trabajo propicio y con menor variabilidad para obtener resultados efectivos.

Varios estudios realizados por especialistas sostienen que sólo el 20% de los proyectos finalizan obteniendo el objetivo planteado, en el tiempo y con los recursos estimados. Esta problemática se da en todo tipo de proyectos, y está particularmente acentuada en proyectos tecnológicos y que no constan con una adecuada

documentación procedimental.

De esto último surge entonces que además de los costos directos, que son fácilmente contabilizables, existen los costos indirectos - *que seguramente son mucho más importantes que lo que pueda suponerse* - de no ser así entonces la falla estuvo en promover un proyecto que no aportaba demasiado valor a la organización. Esto fundamentalmente impacta en una baja de productividad de algún área de la organización y en un COSTO de OPORTUNIDAD al no disponer de un resultado que seguramente será un eslabón importante para la cadena de factores críticos de éxito previstos en la estrategia global.

Esto conlleva a darle una importancia superior a los motivos que generan estos fracasos y desarrollar lineamientos para corregirlos. En Ecuador particularmente, y dada la situación de los últimos años, muchos proyectos de Telecomunicaciones han seguido la misma suerte que la estadística general anota. La permanente búsqueda en reducir los costos de RRHH, más que cualquier otro, han llevado a las empresas a generar proyectos de alta criticidad y exigencia con el fin de que los mismos aporten algún beneficio tangible a la organización que necesita aumentar ingresos y bajar costos, pero al introducir esa baja de costos de recursos, casualmente en la área que deben ejecutar dichos proyectos, se generan dos malos escenarios; o se tiene gente preparada para liderar dichos proyectos pero con sobrecargada de trabajo lo cual implica no poder ejecutarlos como se debe y por lo tanto se ingresa en el 80% o se recambia personal con menor costo y menor experiencia para la función, lo cual produce el mismo resultado y en algunos casos “peores a lo imaginado”.

En un aspecto más conceptual, la dirección o liderazgo de proyectos, no ha tenido la categorización de una especialidad en sí misma, lo cual implica que se *improvisa muchísimo*. Este escenario, conduce al replanteamiento en la forma de cómo se ejecutan los proyectos. Se necesitan más líderes, para conseguir concretar los proyectos de cambio en las organizaciones de manera tal de aumentar la

productividad de las mismas.

1.2. Justificación del Problema de Investigación

En lo referente al diseño y ejecución de proyectos de telecomunicaciones, el factor crítico es el empleo de **metodologías**. Muchas organizaciones pequeñas y medianas el factor brilla por su ausencia, y en muchas organizaciones grandes el factor existe y en general se trabaja bien, pero no son pocos los casos en los cuales – en general por falta de tiempo – la metodología termina siendo utilizada como una máscara formal para el cumplimiento de normas y etapas, y no como lo que verdaderamente es el eje del proyecto tomando el contenido de la metodología y no solo la forma.

Uno de los puntos en los cuales es muy débil la utilización de metodologías, es el diseño de la estructura de un proyecto y en la estimación de esfuerzos y tiempos. A tal efecto, lo que se propone en esta tesis es una estructura jerárquica basada en el concepto de costo por actividad, entendiéndose por costo no solo lo referente al dinero. Esto permite disponer de una herramienta de tablero de control que no solo mide el avance permanente del proyecto sino que permite evaluar impactos en los cambios que se produzcan y en las estimaciones de los tiempos.

Llevar a este punto a los proyectos de Mantenimiento no es una tarea sencilla, para lograrlo el proyecto debe iniciar con una estructura homogénea, que debe medir en forma estandarizada tamaños de productos a desarrollar, a partir de lo cual se podrán realizar estimaciones coherentes basadas en indicadores de productividad. Una de las claves es disponer, o comenzar a construir, una base de conocimiento de la organización, que se alimentará con las experiencias de proyectos que se vayan desarrollando, lo que denominamos "*plantillas*".

En lo que se refiere a problemas humanos, es crítico, y así lo expresan los números. Las metodologías formales son de fundamental importancia, para la tarea de los líderes de proyectos, pero no siempre resultan suficientes para lograr el éxito en el

cumplimiento del objetivo previsto.

1.3. Definición del Problema de investigación:

Necesidad de contar con una herramienta de control basada en TIC's, para medir el avance permanente de proyectos y que inmediatamente permita evaluar impactos en los cambios que se produzcan, así como también las estimaciones en los tiempos de ejecución.

1.4. Hipótesis:

La posibilidad de implementación de un sistema con aplicaciones con licencia GNU/GPL y basado en un modelo de clases de objetos programado en PHP 5.2.6, cuyo motor de *WorkFlow* sea diseñado, codificado y embebido en una Base de Datos MySQL 5 que permita gestionar las etapas de ejecución y control de cualquier tipo de proyecto, hasta su finalización o cierre.

1.5. Objetivos

1.5.1. Objetivo General:

Implementación de un sistema GNU/GPL basado en un modelo de clases de objetos programado en PHP 5.2.6, cuyo motor de *WorkFlow* sea diseñado, codificado y embebido en una Base de Datos MySQL 5 permitiendo gestionar las etapas de ejecución y control de cualquier tipo de proyecto de telecomunicaciones, hasta su finalización o cierre.

1.5.2. Objetivos Específicos

- Estudio del marco teórico y conceptual del tema de investigación, así como del estado del arte del mismo.
- Estudio de las tecnologías necesarias y factibles para la construcción del sistema que se propone.
- Realizar el diseño y modelación del sistema.
- Realización de pruebas al sistema.

Se pretende, finalmente obtener un producto de software relacionado a las telecomunicaciones a la altura de las exigencias actuales de producción de software en la sociedad actual, acorde con los estándares internacionales de catalogación y de diseño.

1.6. Metodología de Investigación:

Métodos Teóricos:

- 1. Histórico-Lógicos:** Para el análisis del objeto de estudio y sus antecedentes históricos, y las tendencias al desarrollo de la aplicación con tecnología que se propone.
- 2. Análisis-Síntesis:** Para analizar y sintetizar toda la información relacionada con el tema de la investigación y en la determinación de los hechos que han servido de base para fundamentar la necesidad del sistema que se propone.
- 3. Hipotético Deductivo:** Para la elaboración de la hipótesis y deducir de ella consecuencias directamente verificables en la realidad.
- 4. Modelación:** Elaborar una aplicación con tecnología de programación Web que permita el seguimiento y evaluación de proyectos de telecomunicaciones.

Métodos empíricos:

- 1. Observación:** Para diagnosticar la situación actual que presenta el seguimiento y evaluación de proyectos de telecomunicaciones.
- 2. Encuestas y Entrevistas:** Realizadas para el diagnóstico y la validación por criterio de los especialistas.
- 3. Análisis documental:** Para la caracterización y desarrollo de la tesis y la elaboración de la Aplicación.

Capítulo 2: Tecnologías de programación Web.

2.1. Introducción a PHP.

La programación en PHP se inicia como simple lenguaje de scripts, para cada versión siempre se incluían algunas nuevas características que orientaban la

programación mediante objetos. De tal modo que con la aparición de PHP 5, los ingenieros desarrolladores de páginas webs PHP orientada a objetos vieron como una verdadera plataforma de programación por su eficaz y potente engine Zend 2.0, el cual permite mayor rapidez y eficiencia en la ejecución de cualquier tipo de programa.

A la hora de programar se puede pensar en *objetos* como entidades que maneja el programa y que funcionan de una determinada manera. Estos objetos poseen determinadas características (variables) y con ellos se pueden realizar diversas tareas (funciones). En un esquema de programación orientada a objetos, las variables y las funciones no son considerados como elementos independientes sino como componentes de una misma entidad: el objeto; tal como ocurre con los objetos de la vida real. (Cobo, Gómez, Pérez, & Rocha, 2005)

2.2. Características de PHP.

La herramienta de programación PHP posee varias características que les ha permitido mantenerse como una tecnología confiable y eficiente, para lo cual existen tres campos generales en PHP usados como scripts:

- **Scripts en la parte del servidor.**

Es el campo más tradicional y el principal campo de trabajo, necesita tres cosas para que su operatividad: (1) Parseador PHP (CGI ó módulo), (2) un servidor web y (3) un navegador.

- **Scripts en línea de comandos.**

Para este script solo se requiere usar el parseador PHP, siendo ideal su uso para scripts ejecutados regularmente desde cron (Unix) ó el Planificador de tareas (Windows 7). Estos scripts también pueden ser usados para tareas simples de procesado de texto.

- **Escribir aplicaciones gráficas clientes.**

Aunque existen plataformas de programación gráficas más robustas, que para las aplicaciones PHP no es considerada robusta en gráficas, pero se puede usar algunas características avanzadas en programas clientes, como *PHP-GTK*¹ para escribir dichos programas.

Los módulos de funciones proporcionan a PHP muchas de sus útiles prestaciones, aunque el motor de procesamiento engine Zend 2.0 se encarga de analizar, traducir y ejecutar el código entrante. PHP se compila en el momento que se ejecuta, en el servidor, por lo que evita tener que compilar previamente el código específico para cada tipo de equipo en el que se espera que sea ejecutado.

Permite modularizar una página web en varios módulos PHP desarrollados en forma independiente, realizando “includes” dentro de un script PHP, de esta forma se puede además desarrollarse en PHP componentes reusables que luego se utilizan en otras páginas o incluso en otros sitios web. Una forma común de trabajo usando PHP para generar sitios dinámicos es definir “*templates*”² en los cuales se divide la página en “zonas” o “módulos” que serán desarrollados.

La plataforma PHP opera bajo los sistemas operativos principales del mercado: Linux, Unix, Windows, Mac OS X, RISC OS. Asimismo, compatible con la mayoría de servidores web de hoy en día (Apache, Microsoft Internet Information Server, Personal Web Server, O'Reilly Website Pro Server y muchos otros). Varios módulos de PHP se encuentran disponibles para la mayoría de los servidores, utilizado como un procesador CGI, en otras palabras PHP tiene la libertad de operar en cualquier sistema operativo y el servidor web que desee.

Otra característica de PHP es el gran número de funcionalidades que presenta, entre las principales se puede citar las siguientes:

¹ PHP – GTK.- Es un proyecto cuyo objetivo es probar que PHP también puede ser usado para crear aplicaciones gráficas.

² Templates.- Diseños o interfaces que se implementa de forma fácil en un sitio web o CMS, también son conocidas como plantillas.

- Funciones de encriptación de datos
- Funciones para manejo de FTP
- Funciones de hashing
- Generación dinámica de imágenes
- Manejo de cuentas de mail IMAP y POP3
- Funciones de networking usando sockets
- Serialización de estructuras de datos
- Manejo de sesiones.

2.3. Funcionalidad de PHP

En la figura 2.1 se muestra el funcionamiento de PHP para determinada aplicación web, donde se deben cumplir ciertos requisitos básicos, tales como:

- ✓ El servidor solicita páginas web de cualquier formato.
- ✓ Identifica si es una página con extensión .php.
- ✓ PHP reconocible para convertirse en código HTML.
- ✓ Página HTML se envía al usuario (PHP cuya sintaxis sea correcta).

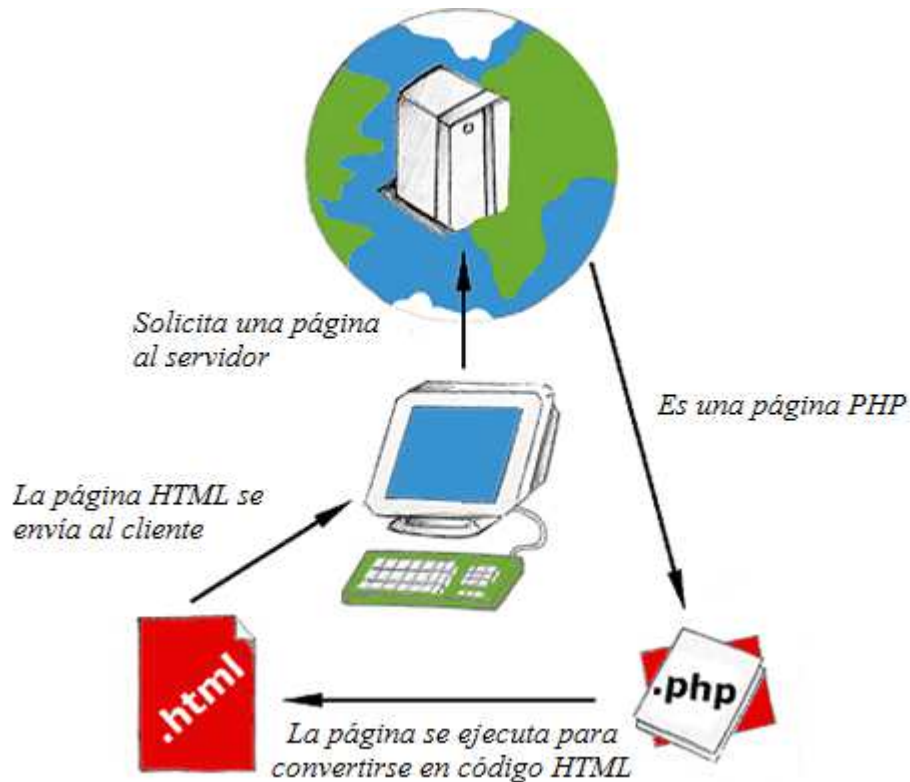


Figura 2. 1: Funcionamiento de PHP

Fuente: <http://www.desarrolloweb.com/articulos/392.php>

2.4. PHP orientado a Objetos.

La tendencia actual de los lenguajes es la programación orientada a objetos (POO) (Lavin, 2006). Desde la versión 3 de PHP se introducen un conjunto de estructuras para llegar al afamado paradigma, claro está que en esta versión la POO era muy simplista y no fue hasta la próxima versión, PHP 4, que se llega a perfeccionar notablemente. A causa de una demanda popular para perfeccionar el soporte orientado a objetos, se rediseñó completamente el modelo de objetos que se venía siguiendo y con esto nació el actual PHP 5. Se adicionó una cantidad considerable de ventajas y cambios en el comportamiento de los objetos que conforman la base del lenguaje.

Al igual que ocurre en la vida real, los objetos se agrupan en familias o clases de objetos que comparten características comunes. Así por ejemplo, dos clientes diferentes tendrán las mismas variables, aunque lógicamente con diferentes valores para cada uno de ellos, y podrán realizar las mismas acciones; por tanto, serían dos objetos de una misma clases. La POO se basa en definir clases, crear objetos de esas clases y realizar acciones con ellos. La clase establecerá las características (variables y métodos) que compartirán todos los objetos de la misma; en este sentido la clase puede entenderse como un prototipo para lo objetos. (Cobo, Gómez, Pérez, & Rocha, 2005)

El mayor problema de la programación orientada a objetos (POO) en las versiones 3 y 4 de PHP se basaba en que, cada vez que se asignaba una variable que contenía un objeto a otra variable, o se pasaba un objeto por parámetro en una función, se realizaba una copia de ese objeto y quedaba a disposición del programa en la nueva variable o parámetro.³

2.4.1. Objetos

La primera pregunta que salta a la vista es: ¿Qué es un objeto? (Gilmore, 2008) y

³Recuperado de la página web: <http://www.desarrolloweb.com/articulos/1697.php>

muchos se responderán que son *entidades tangibles* que tienen un conjunto de... y mi siguiente pregunta es la que sigue: ¿Un sueño es un objeto? Asiento con la cabeza ante todos y respondo yo mismo. Un sueño perfectamente puede ser un objeto, porque un sueño tiene propiedades, tiene un comportamiento y además alrededor de él suceden un conjunto de eventos que responden a comportamientos de los demás objetos existentes en el entorno en que se encuentra. Aterricemos ahora a la realidad del mundo y definamos a cualquier otro objeto.

Las aplicaciones orientadas a objeto frecuentemente poseen varios objetos que interactúan entre sí. (Lavin, 2006) Donde el objeto es una entidad del problema cuyo identificador es único, mismo que tiene un conjunto de propiedades denominadas atributos, que mediante los atributos mejora su comportamiento. Dichos atributos son variables básicas que describen las características del objeto, mientras que los métodos son un conjunto de funciones que solucionan los requerimientos del sistema, cuyas responsabilidades del objeto para con los demás se encuentran en el sistema.

2.4.2. Clases

Una clase es un plantilla para los objetos que intervienen en un sistema, es el tipo de dato de los objetos que son instancias de ella. Las clases en programación orientada a objetos son definiciones de los elementos que forman un sistema, en este caso, definiciones de los objetos que van a intervenir en nuestros programas. Es además la que define los atributos y métodos que tendrán los objetos.

El diagrama que se muestra en la figura 2.2 representa a la clase persona, que tiene como atributo la variable *\$name* y como métodos el *setNombre(\$name)* y el *getNombre()*. El primero se utiliza para inicializar o cambiar el atributo *\$name* de una persona en específico y el segundo devuelve al programador el valor de ese mismo atributo.

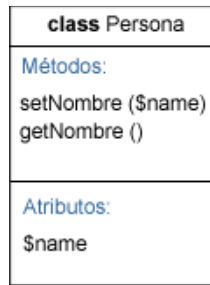


Figura 2. 2: Diagrama de la clase persona.

Es necesario aclarar que basado en el problema que estemos enfrentando se crearán los objetos que intervendrán como solución, específicamente crearemos dos instancias de *Persona*, un objeto pudiera ser **persona1** y otro **persona2**, que se detalla de la siguiente manera:

```
<?php
class Persona
{
private $name;
functionsetNombre($name)
{
    $this->name = $name;
}
functiongetNombre()
{
return $this->name;
}
};
$persona1 = new Persona();
$persona1->setNombre("Lidiexy");
$persona2 = new Persona();
$persona2->setNombre("Dayren");
print $persona1->getNombre()."\n";
print $persona2->getNombre()."\n";
?>
```


2.4.3. Declaración de clases.

Declarar una clase, es bastante simple, usando primeramente la palabra reservada `class`, seguido de un identificador o nombre de la clase y por último una lista con los atributos y métodos que tendrán los objetos instancias de la clase. (Gosselin, Kokoska, & Easterbrooks, 2010)

```
class MyClass
{
    ... // Lista de atributos
    ...
    ... // Lista de métodos
};
```

Se puede notar que antes de hacer la declaración del atributo `$name`, se usó la palabra reservada `private`. Más adelante se explicará en detalle esta palabra, pero básicamente significa que solo los métodos de esta clase pueden acceder al valor de `$name`, esto obliga a que todas las instancias de la clase `Persona` a usar los métodos `setNombre($name)` y `getNombre()` para poder modificar y recibir la información del atributo.

2.4.4. La palabra reservada `new` y los constructores

La palabra reservada `new` se utiliza para crear instancias de una clase determinada (objetos). En el ejemplo visto anteriormente se creó una instancia `persona1` de la clase `Persona` de la siguiente forma: `$persona1 = new Persona();`. La palabra reservada `new` crea un nuevo objeto que contiene una copia de los atributos de la clase y se llama entonces al constructor que fue definido. El constructor de una clase es un método llamado `__construct()`, se ejecuta automáticamente cuando se crea un objeto con `new`. Notemos que puede existir o no la declaración explícita en la clase, siempre se llama a un constructor, aunque sea el constructor vacío. Ellos son funciones como las tratadas en temas anteriores, que pueden tener uno o más argumentos. En PHP se utiliza también el nombre de la clase como método para

definir el constructor como se hace en C++, pero es buena practica de programación el uso de `__construct()` en nuestras aplicaciones. Aclarar que son dos guiones bajos antes de la palabra "construct". (Gilmore, 2008)

Reescribamos entonces el ejemplo anterior con un constructor en el cual pasaremos el atributo como parámetro:

```
<?php
class Persona
{
private $name;
function __construct($name)
{
    $this->name = $name;
}

function setNombre($name)
{
    $this->name = $name;
}
function getNombre()
{
return $this->name;
}
};

$persona1 = new Persona("Lidiexy");
$persona2 = new Persona("Dayren");

print $persona1->getNombre()."\n";
print $persona2->getNombre()."\n";
```

?>

Pudiéramos haber eliminado la función `setNombre($name)`, pero en algún momento la necesitaremos para los objetos que necesiten cambiarse el nombre por una razón u otra.

2.4.5. Los destructores.

Como su nombre lo indica, los destructores son las funciones que destruyen los objetos, es decir, hacen la función contraria a los constructores. Estos se llaman de forma automática cuando se quiere destruir o liberar a un objeto (cuando no se necesitan más referencias al objeto). PHP al igual que otros lenguajes se encarga de la liberación de los recursos al final de cada llamada, por lo cual la importancia de los destructores pierde en alguna medida el sentido para los cuales fueron creados. No por esto dejan de ser necesarios para liberar la memoria de algunos recursos utilizados o para guardar información en el servidor que quede como constancia del trabajo con la aplicación (logging). La definición de un destructor es tan simple como adicionar una función `__destruct()` (Lavin, 2006) en nuestra clase:

```
<?php
class MyClass
{
function __construct()
{
print "Se creo un objeto de MyClass";
}
function __destruct()
{
print "Un objeto de la clase MyClass se ha destruido";
}
};

$objeto = new MyClass();
```

```
$objeto = NULL;
```

```
?>
```

La salida del ejemplo sería:

Se creo un objeto de MyClass

Un objeto de la clase MyClass se ha destruido

Al ejecutar la sentencia `$objeto = NULL;` se libera el único manipulador (handle) que tiene el objeto por tanto se llama al destructor correspondiente y el objeto deja de existir.

El momento exacto en que se llama al destructor no está garantizado en PHP, puede ser unas sentencias después de haber realizado la última referencia al objeto que por lógica fue la referencia donde se libera al objeto de la memoria, por tanto es responsabilidad del programador tener en cuenta esta aclaración cuando realiza una aplicación.

2.4.6. Accediendo a los métodos y atributos a través de la variable *\$this*.

Cuando ejecutamos cualquier método de un objeto o accedemos a uno de sus atributos, se llama automáticamente a una variable que se define como parte de la clase por defecto, *\$this*. (Lavin, 2006) Esta variable o atributo de la clase es una referencia a sí misma. Para esto *\$this* usa la notación \rightarrow . Se pudiera entonces acceder al atributo *\$name* usando la siguiente sentencia, *\$this* \rightarrow *name* (nótese que no es necesario poner el signo de \$ delante del atributo cuando se referencia de esta manera), de igual forma se accede a cualquier método que se haya definido, ejemplo: *\$this* \rightarrow *getNombre()*;

2.4.7. Encapsulación.

La clave del paradigma POO es la encapsulación y la protección de acceso a los atributos y métodos de la clase. Los lenguajes orientados a objetos más comunes presentan tres palabras reservadas para restringir el acceso: `public`, `protected` y `private`. A partir de ahora podemos utilizar los modificadores de acceso habituales de la POO. Estos modificadores sirven para definir qué métodos y propiedades de las

clases son accesibles desde cada entorno. La encapsulación, también llamada "ocultación de la información", esto asegura que los objetos no pueden cambiar el estado interno de otros objetos de maneras inesperadas; solamente los propios métodos internos del objeto pueden acceder a su estado. Cada tipo de objeto expone una *interfaz* a otros objetos que especifica cómo otros objetos pueden interactuar con él. (Lavin, 2006)

En el modelo objeto de PHP 3 o 4, se definía con la palabra reservada *var* un atributo (miembro) de la clase, lo que es equivalente a *public* en PHP 5. Un ejemplo que muestra el uso de estas palabras reservadas es el que sigue:

```
classMyClass
{
public $attrpublico = "Atributo publico";
protected $attrprotegido = "Atributo protegido";
private $attrprivado = "Atributo privado";
functionmiMetodo()
{
    //...
}
}
$objeto = new MyClass();
```

Definiciones de cada uno de los modificadores de acceso:

- ❖ *public*: Los miembros públicos de una clase se pueden acceder desde cualquier lugar de la clase con la variante dada anteriormente: $\$this \rightarrow \text{miembropúblico}$ y también las instancias de esa clase (objetos) utilizando la siguiente sentencia: $\$objeto \rightarrow \text{miembropúblico}$. Es el nivel de acceso más permisivo. Sirve para indicar que el método o atributo de la clase es público. En este caso se puede acceder a ese atributo, para visualizarlo o editarlo, por cualquier otro elemento de nuestro programa. Es el modificador que se aplica por defecto.

- ❖ `protected`: Este indica un nivel de acceso medio y un poco más especial que los anteriores. Sirve para que el método o atributo sea público dentro del código de la propia clase y de cualquier clase que herede de aquella donde está el método o propiedad `protected`. Es privado y no accesible desde cualquier otra parte. Es decir, un elemento `protected` es público dentro de la propia clase y en sus heredadas. La *herencia* la veremos más adelante y tendremos ejemplos del uso de este modificador.
- ❖ `private`: Es el nivel de acceso más estricto. Sirve para indicar que esa variable sólo se va a poder acceder desde el propio objeto, nunca desde fuera. Si intentamos acceder a un método o atributo declarado `private` desde fuera del propio objeto, obtendremos un mensaje de error indicando que no es posible acceder a ese elemento.

Podemos pensar que para los atributos de una clase que se quieren acceder desde fuera de los objetos usamos el modificador de acceso `public` y para los que no se quieren acceder desde fuera, pues usar `private`. Hay que tener en consideración la importancia del modificador `protected` para los conceptos que estudiaremos en acápite posteriores.

Pudiéramos poner un ejemplo para mostrar la utilización de los modificadores de acceso que hemos tratado:

```
class Usuario
{
public $nombre;
protected $sitio = "www.curso_php.com";
private $clave;
function encriptarClave($pass)
{
    //...
}
```

```

function getClave()
{
return $this->encriptarClave($this->clave);
}
}
$objeto = new Usuario();
print $objeto->nombre." tiene la clave: ".$this->getClave();

```

2.4.8. Métodos public, protected y private.

Los modificadores de acceso pueden ser usados para restringir también a los métodos de las clases siguiendo las mismas reglas que se trataron anteriormente:

- ❖ **public:** Los métodos que pueden ser invocados desde cualquier lugar, dentro o fuera de la clase.
- ❖ **protected:** Los que solo pueden ser llamados desde dentro de la clase o de las clases que heredan de la principal.
- ❖ **private:** Los métodos que solo pueden ser invocados desde dentro de la misma clase o redeclarados en las clases hijas, por tanto cada clase tiene una versión de lo que realiza este método.

```

class Usuario
{
public $nombre;
protected $sitio = "www.curso_php.com";
private $clave;
private function encriptarClave($pass)
{
//...
}
public function getClave()
{
return $this->encriptarClave($this->clave);
}
}

```

```
}  
}
```

Cuando se obvia el modificador de acceso para los métodos se asume `public`, mientras que para los atributos no se asume ningún modificador, por lo que hay que aclarar en cada caso.

2.4.9. Herencia.

La herencia es uno de los mecanismos fundamentales de la programación orientada a objetos. Por medio de la herencia, se pueden definir clases a partir de la declaración de otras clases. Las clases que heredan incluyen tanto los métodos como las propiedades de la clase a partir de la que están definidos. (Lavin, 2006) Pensemos en un ejemplo de la vida real, en una universidad pudiéramos definir la clase "Persona" que puede incluir las características referentes a todo el personal que trabaja o estudia en la universidad (atributos de la clase), al igual que las responsabilidades o funcionalidades que tienen (métodos de la clase), así como identificarse o moverse.

Como es lógico el personal en una universidad es muy variado y cumplen diferentes funciones. Tomemos por ejemplo a los profesores, a los estudiantes y a los dirigentes que son los más comunes. Todos tienen las mismas características que puede tener una persona, pero en particular se diferencian en algunos aspectos. Por tanto puede existir en nuestro sistema una clase Estudiante, Profesor y otra clase Dirigente que sean una herencia de la clase Persona definida anteriormente. La clase estudiante tendría atributos como: `cantidad_asignaturas`, `semestre_que_cursa` y otros atributos además de los que hereda de la clase Persona, así como funcionalidades como `recibir_evaluacion()`, `promedio_semestre()` o `acreditar_asignatura`.

Lo normal en sistemas de herencia es que las clases que heredan de otras incluyan nuevas características y funcionalidades, aparte de los atributos y métodos

heredados. Pero esto no es imprescindible, de modo que se pueden crear objetos que hereden de otros y no incluyan nada nuevo.

La herencia organiza y facilita el *polimorfismo* y la *encapsulación* (Gilmore, 2008), permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que reimplementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases y las clases en *árboles* o *enrejados* que reflejan un comportamiento común. (Lavin, 2006)

2.4.10. Sintaxis de la herencia.

El modelo de objeto de PHP nos brinda una serie de estructuras para definir la herencia en nuestras aplicaciones, de modo que se puedan crear jerarquías de objetos que heredan unos de otros. Realizamos un esquema mostrado por la figura 2.3 tomando en cuenta el ejemplo que definimos anteriormente, cuyo modelo es puramente ficticio y solo sirve como medio de enseñanza.

```
class Mi_Clase_Hija extends Clase_Padre
{
    ...
}
```

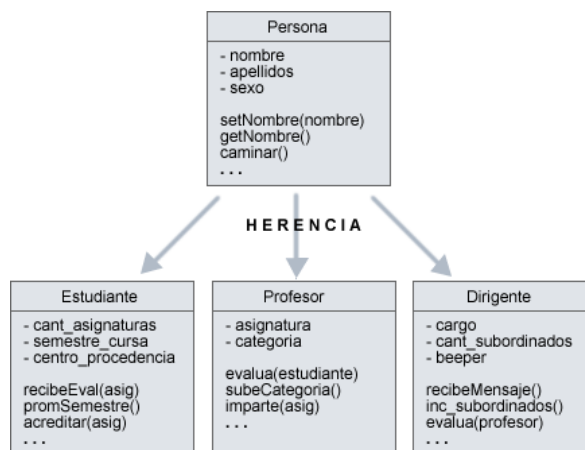


Figura 2. 3: Representación de la herencia entre las clases.

```
class Persona
```

```

{
public $nombre;
protected $sexo;
protected $apellidos;
function _construct($name, $lastname, $sex = "Masculino")
{
    $this->nombre = $name;
    $this->apellidos = $lastname;
    $this->sexo = $sex;
}
privatefunction caminar()
{
print "De forma común";
}
}
class Estudiante extends Persona
{
private $cant_asignaturas;
private $semestre_cursa;
private $centro_procedencia;
function __construct($name, $last, $sex, $asig, $sem, $centro)
{
    parent::__construct($name, $last, $sex);
    $this->cant_asignaturas = $asig;
    $this->semestre_cursa = $sem;
    $this->centro_procedencia = $centro;
}
//...
functionpromSemestre()
{
    //$suma_notas se calcula de alguna manera...
}

```

```

        return $suma_notas/$this->cant_asignaturas;
    }
}

```

```

$estudiante1 = new Estudiante("Dayren", "Martinez", "F", "PHP",
                             5, "IPUEC -Levantamiento Jucaral");
print "El estudiante: ".$estudiante1->nombre. "tiene un atributo público y es el
nombre";
print "Sería un error tratar de acceder a: ".$estudiante1->sexo. ", así como hacerlo
para ".$estudiante1->sexo;

```

Por tanto se puede decir que la herencia y los modificadores de acceso juegan un papel fundamental en la construcción de la solución de las aplicaciones en PHP.

2.4.11. Polimorfismo.

Cuando se habla de polimorfismo, posiblemente estemos tratando uno de los logros más significativos de la POO. Queda demostrado que con el uso de clases y la herencia se hace más fácil describir situaciones de la vida real, en vez de usar un grupo de datos y funciones; suele ser más cómodo extender o agrandar los proyectos a través de la reutilización del código que brinda la herencia. Además para lograr una programación robusta y extensible es necesario el *polimorfismo*. (Williams & Lane, 2004)

Polimorfismo es una palabra de origen griego que significa aproximadamente "*muchas formas*", (poly = muchas, morphos = forma). Veamos el siguiente ejemplo:

```

class Gato
{
function miau()
{
    print "miau";
}
}

```

```

    }
class Vaca
{
functionmuu()
    {
        print "Muu";
    }
}
functionimprimirSonidoCorrecto($obj)
{
if($objinstanceof Gato)
    {
        $obj->miau();
    }
elseif($objinstanceof Vaca)
    {
        $obj->muu();
    }
else
    {
        print "ERROR: El objeto no es un animal apropiado";
    }
print "<br>";
}

$obj1 = new Gato();
$obj2 = new Vaca();
imprimirSonidoCorrecto($obj1);
imprimirSonidoCorrecto($obj2);

```

La salida de este ejemplo:

Miau

Muu

Es evidente ver que este ejemplo no es para nada extensible, si tratamos de adicionar más sonidos de animales tendremos que poner más sentencias `elseif` y `else` en la función `imprimirSonidoCorrecto()` chequeando que el objeto pasado por parámetros es instancia del nuevo animal para poder invocar el método correspondiente.

Este problema se resuelve con el uso del polimorfismo y la herencia. Si sacamos del problema que pudiera existir una clase *Animal* y que de ella heredaran todos los animalitos que queramos, entonces existirían una serie de características y en especial una función común para todos los animales que denominaríamos, `emiteSonido()`. Por tanto una clase hija de *Animal* pudiera ser la clase *Gato*, *Vaca* y otros.

```
class Animal
{
functionemiteSonido()
{
    print "ERROR: Impleméntalo en la clase hija";
}
}
class Vaca extends Animal
{
functionemiteSonido()
{
    print "Muu";
}
}

class Gato extends Animal
```

```
{
functionemiteSonido()
{
    print "Miau";
}
}

functionimprimirSonidoCorrecto($obj)
{
if($objinstanceof Animal)
{
    $obj->emiteSonido();
}
else
{
    print "ERROR: El objeto es de otro tipo";
}
print "<br>";
}
```

```
$obj1 = new Gato();
$obj2 = new Vaca();
imprimirSonidoCorrecto($obj1);
imprimirSonidoCorrecto($obj2);
```

De igual manera se imprimiría:

Miau

Muu

En el transcurso de la clase aprenderemos algunas sentencias que nos ayudaran a resolver un conjunto de problemas que todavía quedan en la POO que hemos visto

hasta ahora. Conoceremos que significa, aunque sea evidente, la palabra reservada `instanceof`, así como `abstract`, que obliga al programador a implementar la función en la clase hija, por ejemplo `emiteSonido()` de la clase *Animal*, debería llevar delante la palabra `abstract`, ya que no tiene sentido implementar una función cuando será re-implementada en las clases hijas, porque además, los Animales no tienen un sonido en común, sino que cada una de las especies emiten el suyo.

2.4.12. El modelo objeto en PHP 5. Otras características.

Existe un conjunto de funciones, operadores y palabras reservadas en PHP 5 que lo hace alcanzar un nivel determinante en el agrado de todo programador Web. En los inicios de PHP 3 se hicieron intentos de adicionar sintaxis orientada a objetos, pero dejaba mucho que desear. No fue hasta la versión 4, cuando Zeev Suraski y Andi Gutmans (personalidades en la programación) reescriben el nuevo sistema (engine) que serviría de base para lograr una mayor velocidad, estabilidad y además un sistema que alardeaba de mejores posibilidades y capacidades que los existentes en el momento. Así y todo, todavía quedaba vigente el pobre modelo orientado a objeto que poseía PHP 3. No obstante comenzó a utilizarse con más fuerza el novedoso modelo de PHP 4, hasta la llegada del “mejoradísimo” modelo para la programación orientada a objetos que tiene PHP 5 y que vamos a ver sus principales características.

2.4.13. Posibilidad de uso de interfaces.

Las interfaces se utilizan en la POO para definir un conjunto de métodos que implementa una clase. Una clase puede implementar varias interfaces o conjuntos de métodos. En la práctica, el uso de interfaces es utilizado muy a menudo para suplir la falta de herencia múltiple de lenguajes como PHP o Java.

Interfaces: Son clases que no tienen implementación. (Williams & Lane, 2004)

- Sirven como tipos de otras clases.
- Todos sus métodos son *abstractos* y todos sus atributos son *final*.
- Una clase puede implementar muchas interfaces.

```

interfaceMostrar_en_pantalla
{
functionmostrar_en_pantalla();
}
class Circulo implementsMostrar_en_pantalla
{
functionmostrar_en_pantalla()
{
    print "Mostrando el circulo...";
}
}

```

2.4.14. Operador instanceof.

Se utiliza para saber si un objeto es una instancia de una clase determinada, es un operador que sustituye a la función *is_a()* de PHP 4 que se encargaba de chequear las relaciones entre clases. (Lavin, 2006)

```

if($obj instanceof Circulo)
{
print "El objeto creado es una instancia de la clase Circulo";
}

```

2.4.15. Métodos y clases final.

En PHP 5 se puede indicar que un método es final. Con ello no se permite sobrescribir ese método, en una nueva clase que lo herede. Si la clase es final, lo que se indica es que esa clase no permite ser heredada por otra clase.

```

classMetodo_Final
{
    final functiongetNombreClase()
    {
        return __CLASS__;
    }
}

```



```
}
```

Este ejemplo devuelve un error debido al concepto de clase final:

```
final class Clase_Final
{
function metodo1()
{
    //...
}
}
```

```
class Clase_Erronea extends Clase_Final
{
    //...
}
```

2.4.16. Atributos y métodos static.

En PHP5 podemos hacer uso de atributos y métodos static. Son las propiedades y funcionalidades a las que se puede acceder a partir del nombre de clase, sin necesidad de haber instanciado un objeto de dicha clase.

```
class MiClase
{
static $attrEstatico;
static $attrInitEstatico = 0;
static function metodo_estatico()
{
    print "Hola Universidad";
}
}
```

```
//llamada a los atributos
MiClase::$attrEstatico++;
$result = MiClase::$attrInitEstatico + 2;
echoMiClase::$attrEstatico. "<br>";
echo $result. "<br>";
MiClase::metodo_estatico();
```

Este ejemplo imprime los siguientes valores:

```
1
2
Hola Universidad
```

2.4.17. Clases y métodos abstractos.

También es posible crear clases y métodos abstractos. Las clases abstractas no se pueden instanciar, se suelen utilizar para heredarlas desde otras clases que no tienen porque ser abstractas. Los métodos abstractos no se pueden llamar, se utilizan más bien para ser heredados por otras clases, donde no tienen porque ser declarados abstractos. Hay que aclarar que las clases que incluyen métodos abstractos tienen que ser declaradas abstractas.

```
abstractclassClase_abstracta
{
functionmostrarTexto()
{
    print "Texto que muestra la clase abstracta";
}
}
```

/* Lo que viene a continuación es un error porque las clases abstractas no se pueden instanciar */

```
$obj = new Clase_abstracta(); // ERROR
```

A continuación mostramos un ejemplo con la utilización de los métodos abstractos:

```

abstractclass Figura
{
protected $x;
protected $y;

abstractfunction dibujar()

functionsetCentro($x, $y)
{
    $this->x = $x;
    $this->y = $y;
}
}

class Circulo extends Figura
{
function dibujar()
{
    //...
}
}

```

Como se pueden fijar la función *dibujar()* de la clase *Figura* no se implementa porque estamos en presencia de un método abstracto que será re implementado en las clases hijas de *Figura*.

2.4.18. Clases reservadas `self::` y `parent::`

PHP en su nueva versión agregó dos clases reservadas para mejorar el código cuando de programación orientada a objeto se refiere. `self::` es una clase que se refiere a los atributos y métodos estáticos, así como a las constantes definidas. Es una referencia a la clase que la utiliza. (Lavin, 2006) Por su lado `parent::` se refiere a

la clase ancestro o padre y se usa a menudo cuando se quiere invocar el constructor o un método de la clase padre.

```
class Clase_Padre
{
    static nombre = "Padre";
    function __construct()
    {
        print "Constructor en la clase: ".self::$nombre." \n";
    }
}
class Clase_Hija
{
    static nombre = "Hija";
    function __construct()
    {
        print parent::__construct();
        print "Constructor en la clase: ".self::$nombre." \n";
    }
}
$obj = new Clase_Hija();
```

El ejemplo tendría la siguiente salida:

Constructor en la clase: Padre

Constructor en la clase: Hija

2.4.19. Constantes de clase.

Se pueden definir constantes dentro de la clase. Luego se pueden acceder dichas constantes a través de la propia clase.

```
class MiClase
{
    const UCI = "Universidad de las Ciencias Informáticas";
```

```

const MIJ = "Ministerio de Justicia";
functionorganismo_Cuba()
{
    printself::UCI;
}
functionorganismo_Venezuela()
{
    printself::MIJ;
}
}
printMiClase::UCI." en conjunto con el ".MiClase::MIJ."<br>";
$obj = new MiClase();
$obj->organismo_Cuba;
echo "<br>";
$obj->organismo_Venezuela;
echo "<br>";

```

El código imprime los siguientes mensajes:

Universidad de las Ciencias Informáticas en conjunto con el Ministerio de Justicia.

Universidad de las Ciencias Informáticas

Ministerio de Justicia

Es buena práctica de programación escribir las constantes en mayúscula.

2.4.20. Función `__autoload()`.

Es habitual que los desarrolladores escriban un archivo por cada clase que realizan, como técnica para organizar el código de las aplicaciones. Por esa razón, a veces resulta tedioso realizar los include de cada uno de los códigos de las clases que se utilizan en un script. La función `__autoload()` sirve para intentar incluir el código de una clase que se necesite, y que no haya sido declarada todavía en el código que se está ejecutando.

[miclase.php](#)

```

<?php
class MiClase
{
function imprimeTexto()
{
    echo "Cursos del Alba. UCi – MIJ";
}
}
?>

```

general.inc

```

<?php
function __autoload($nombre_clase)
{
    require_once($_SERVER["DOCUMENT_ROOT"].
"/clases/$nombre_clase.php");
}
?>

```

index.php

```

<?php
require_once "general.inc";
$obj = new MiClase();
$obj->imprimeTexto();
?>

```

2.4.21. Clonado de objetos.

Si se desea, se puede realizar un objeto a partir de la copia exacta de otro objeto. Para ello se utiliza la instrucción `clone`. También se puede definir el método `__clone()` para realizar tareas asociadas con la clonación de un objeto. Muchas veces, al hacer una *copia* de un objeto, no nos interesa tener una copia exacta del mismo, sino que lo que buscamos es un objeto del *mismo tipo* que el copiado, pero con algunos recursos o características propias del nuevo objeto, y no "copiadas" del

primero. Con el nuevo modelo de objetos de PHP 5, al hacer un `$objeto2 = $objeto1;` ya no estamos copiando un objeto, sino creando una nueva referencia al mismo, por lo que es necesario un mecanismo para copiar objetos, sin tener que llamar al operador `new`. Aquí es dónde entra en juego, el método `__clone()`.

```
class Clonable
{
    public $variable;
    function __construct()
    {
        $this->variable = 'Variable Clonada';
    }
}
$objeto1 = new Clonable();
$objeto2 = $objeto1->__clone();
```

El método `__clone()` puede ser sobre escrito para hacer lo que el usuario desee, y de esta manera, se puede controlar mejor el clonado de objetos:

```
class Clonable
{
    static $id = 0;
    public $nombre;
    public $apellidos;
    function __construct()
    {
        $this->id = self::$id++;
    }
    function __clone()
    {
        $this->nombre = $that->nombre;
        $this->apellidos = $that->apellidos;
        $this->id = self::$id++;
    }
}
```

```

}
}
$objeto1 = new Clonable();
$objeto1->nombre = "Lidiexy";
$objeto1->apellidos = "Alonso Hernandez";
$objeto2 = $objeto1->__clone();
print $objeto2->id. "\n";
print $objeto2->nombre. "\n";
print $objeto2->apellidos. "\n";

```

2.5. Bases de datos en la Web

Las bases de datos permiten almacenar de una forma estructurada y eficiente toda la información de un sitio Web (ver figura 2.4). Además el uso de bases de datos en la Web nos brinda una serie de ventajas tales como:

- Dinamismo del sitio Web.
- Proporcionar información actualizada.
- Facilitar la realización de búsquedas.
- Disminuir los costes de mantenimiento.
- Implementar sistemas de control de acceso.
- Almacenar preferencias de los usuarios.

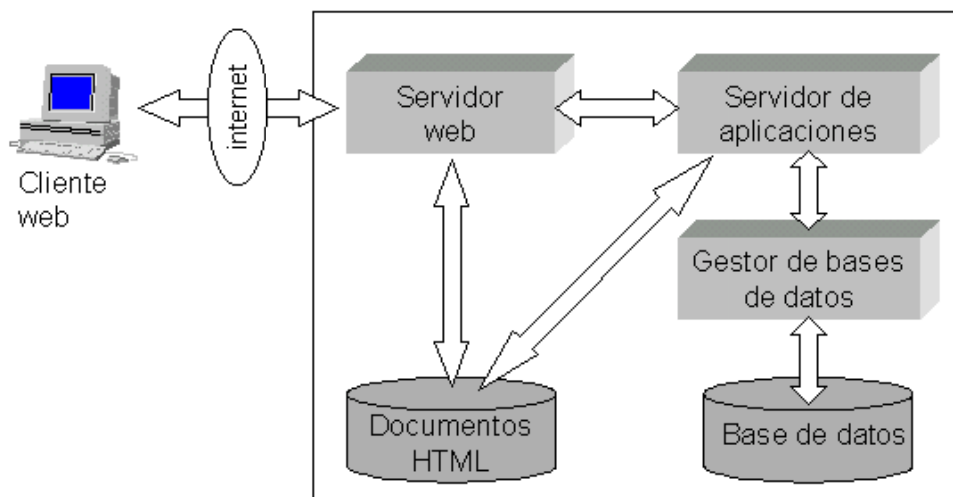


Figura 2. 4: Esquema básico de un sitio Web soportado por bases de datos

2.5.1. Bibliotecas de acceso a datos en PHP

Quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos. Escribir una interfaz Web para mantener una base de datos es una tarea simple con PHP; ya que el mismo cuenta con un conjunto de módulos, extensiones, e incluso bibliotecas de clases para el acceso a múltiples tipos de bases de datos, las que se muestra en la tabla 2.1.

Tabla 2. 1: Bases de datos Disponible para PHP

Adabas D	Ingres	Oracle (OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro (read-only)	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

También se cuenta con una extensión DBX de abstracción de base de datos que permite usar de forma transparente cualquier base de datos soportada por la extensión. Adicionalmente, PHP soporta ODBC (del inglés Open Database Connectivity Standard, en español Estándar Abierto de Conexión con Bases de Datos), así que puede conectarse a cualquier base de datos que soporte tal estándar. Entre ellas se incluyen los productos de Microsoft y muchos otros.

Para la conexión a los diferentes tipos de bases de datos el PHP cuenta con un conjunto de extensiones de la biblioteca PECL (PHP Extension Community Library), las cuales se pueden incluir directamente en la configuración del PHP en el archivo php.ini o dinámicamente durante la ejecución del script, es decir dentro del código de una página php con la función **dl('extension.dll')**.

En la versión de PHP para Windows las extensiones son DLLs que se incluyen al mismo. Algunas de estas extensiones son:

➤ **php_mysql.dll**

Biblioteca de funciones para el acceso a bases de datos MySQL.

➤ **php_mysqli.dll**

Biblioteca de clases para el acceso a bases de datos MySQL

➤ **php_pgsql.dll**

Biblioteca de funciones para el acceso a bases de datos PostgreSQL.

➤ **php_mssql.dll**

Biblioteca de funciones para el acceso a bases de datos MS SQL Server.

➤ **php_dbx.dll**

Biblioteca de funciones para el acceso a bases de datos de tipo

- FrontBase
- Microsoft SQL Server
- MySQL
- ODBC
- PostgreSQL
- Sybase-CT
- Oracle (oci8)
- SQLite, entre otras

2.6. Acceso a bases de datos MySQL en PHP

MySQL es uno de los gestores más utilizados junto a PHP. Como se dijo anteriormente para el trabajo con MySQL el PHP tiene dos extensiones, `php_mysql` y `php_mysqli`, esta última como la versión mejorada con un modelo orientado a objeto. Por lo que usaremos esta para el desarrollo de la clase. La extensión `php_mysqli` contiene tres clases de objetos fundamentales, que son:

mysqli: Representa una conexión entre PHP y la base de datos MySQL.

mysqli_stmt: Representa una sentencia preparada.

mysqli_result: Representa el resultado obtenido de la consulta hecha en la base de datos.

De estas tres clases la primera y la última son las más usadas comúnmente. De las cuales estudiaremos sus métodos y propiedades principales.

mysqli

Constructor:

mysqli - constructor de un nuevo objeto mysqli

Métodos:

- **change_user** - cambia el usuario del identificador de enlace especificado.
- **close** - cierra una conexión previamente abierta.
- **commit** - completa la transacción actual.
- **connect** - abre una nueva conexión al servidor de MySQL.
- **init** - inicializa el objeto mysqli.
- **kill** - solicita al servidor destruir un hilo de MySQL.
- **multi_query** - realiza consultas de SQL múltiples.
- **more_results** - chequea si existen más resultados de la consulta múltiple actualmente ejecutada.
- **next_result** - lee el siguiente resultado de la consulta múltiple actualmente ejecutada.
- **ping** - llama una conexión con el servidor o reconecta si no hay conexión.
- **prepare** - prepara una sentencia SQL.
- **query** - ejecuta una sentencia SQL.
- **rollback** - deshace la transacción actual.
- **select_db** - selecciona la base de datos por defecto.
- **store_result** - transfiere un resultado de la última consulta.

Propiedades:

- **affected_rows** - obtiene el número de filas afectadas en la operación MySQL previa.
- **errno** - regresa el código de error para la llamada a función más reciente.
- **error** - regresa la cadena de error para la llamada a función más reciente.
- **field_count** - regresa el número de columnas para la consulta más reciente.
- **insert_id** - regresa el identificador generado automáticamente usado en la última consulta.

mysqli_result

Métodos:

- **close** - cierra el resultado.
- **data_seek** - mueve el puntero interno del resultado.
- **fetch_field** - obtiene la información de la columna en el resultado.
- **fetch_fields** - obtiene la información para todas las columnas del resultado.
- **fetch_field_direct** - obtiene información de la columna para la columna dada.
- **fetch_array** - recupera una fila como una matriz asociativa, una matriz numérica, o ambos.
- **fetch_assoc** - obtiene una fila como una matriz asociativa.
- **fetch_object** - obtiene una fila como un objeto.
- **fetch_row** - obtiene una fila como una matriz enumerada.
- **close** - libera la memoria ocupada por el resultado.
- **field_seek** - fija el apuntador del resultado a la posición especificada.

Propiedades:

- **current_field** - regresa la posición de puntero actual.
- **field_count** - regresa el número de campos en el resultado.
- **lengths** - regresa una matriz con la longitud de los campos.
- **num_rows** - regresa el número de filas en el resultado.

Los pasos para acceder desde PHP a una base de datos son los siguientes:

1. Conectar con el servidor de bases de datos.
2. Seleccionar una base de datos.
3. Enviar la instrucción SQL a la base de datos.
4. Obtener y procesar los resultados.
5. Cerrar la conexión con el servidor de bases de datos.

Los métodos concretos en PHP que realizan estas operaciones son:

1. Conectar con el servidor de bases de datos (Crear el objeto mysqli):
`$mysqli = new mysqli("localhost", "user", "password");`
2. Seleccionar una base de datos:
`$mysqli->select_db("dbname")`

3. Enviar la instrucción SQL a la base de datos:
`$consulta = $mysqli->query("sentencia sql")`
4. Obtener y procesar los resultados:
`$consulta->num_rows()` y `$consulta ->fetch_object()`
5. Cerrar la conexión con el servidor de bases de datos:
`$mysqli->close()`

Conectar con el servidor de bases de datos:

`mysqli_connect()`: Devuelve un objeto `mysqli` con la conexión en caso de éxito y `false` en caso contrario

Sintaxis: `$mysqli = new mysqli(servidor, user, pass);`

Ejemplo: `$mysqli = new mysqli("localhost", "root", "root")`
or die ("No se puede conectar con el servidor");

Seleccionar una base de datos:

`mysqli_select_db()`: Devuelve `true` en caso de éxito y `false` en caso contrario.

Sintaxis: `$mysqli->select_db(database);`

Ejemplo: `$mysqli->select_db ("lindavista")`
or die ("No se puede seleccionar la base de datos");

Enviar la instrucción SQL a la base de datos:

`mysqli_query()`: Devuelve un objeto **`mysqli_result`** o `true` (dependiendo de la instrucción) si la instrucción se ejecuta correctamente y `false` en caso contrario.

Sintaxis: `$consulta = $mysqli->query(instrucción);`

Ejemplo:

`$consulta = $mysqli->query("select * from noticias")`
or die ("Fallo en la consulta");

Obtener y procesar los resultados:

`mysqli_num_rows()`, `mysqli_fetch_object()`: En el caso de que la instrucción enviada produzca unos resultados, `mysqli_query()` devuelve las filas de la tabla

afectadas por la instrucción. La función `mysqli_num_rows()` devuelve el número de filas afectadas y para obtener las distintas filas del resultado se utiliza la función `mysqli_fetch_object()`, que obtiene una fila del resultado en un objeto, donde los atributos son los campos de la consulta.

Sintaxis: `$nfilas = $consulta ->num_rows($consulta);`
`$fila = $consulta->fetch_object($consulta);`

Obtención de las filas:

```
while ($fila = $consulta->fetch_object())  
{  
    procesar fila i-ésima de los resultados  
    echo "titulo: ".$file->titulo;  
    echo "fecha: ".$file->fecha;  
}
```

Cerrar la conexión con el servidor de bases de datos:

`mysqli_close()`

Sintaxis:

`$mysqli->close();`

Ejemplo:

`$mysqli->close();`

2.7. Soporte para programación Web con PHP y MySQL

2.7.1. Servidor WAMP.

WAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- *Windows, como sistema operativo;*
- *Apache, como servidor web;*
- *MySQL, como gestor de bases de datos;*
- *PHP (generalmente), Perl, o Python, como lenguajes de programación.*

El uso de un WAMP permite servir páginas html a internet, además de poder

gestionar datos en ellas, al mismo tiempo un WAMP, proporciona lenguajes de programación para desarrollar aplicaciones web.

- *LAMP es el sistema análogo que corre bajo ambiente Linux*
- *WAMP es el sistema análogo que corre bajo ambiente Windows*
- *MAMP es el sistema análogo que corre bajo ambiente Macintosh*

2.7.2. APPSERV

AppServ es un programa que aúna una serie de aplicaciones y utilidades, de manera que al instalarlo, no solo tendremos instalados todo esto, sino que, además estarán perfectamente configurados y listos para usarlos en cuestión de minutos, mientras que si tuviésemos que instalar por separado estos programas, deberíamos dedicarle mucho más tiempo a su correcta configuración.

Pero ¿qué aplicaciones nos instala este paquete?

- *Apache*
- *PHP*
- *MySQL*
- *phpMyAdmin*

Apache es un servidor HTTP de código abierto disponible para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras. Esta aplicación es necesaria para montar un host local en el cual se apoyarán el programa **SGMPROTEL (Sistema de Gestión y Mantenimiento de Proyectos de Telecomunicaciones)** que vamos a instalar.

2.7.3. PHP

PHP es un lenguaje de programación interpretado que se ejecuta del lado del servidor, es decir, al hacerle la petición de una página PHP al servidor, éste antes de enviársela al usuario, lo que hace es interpretarlo, traducirlo a html y, acto seguido enviarla al ordenador que ha realizado la petición.

2.7.4. MYSQL

MySQL es un sistema de gestión de bases de datos relacionales multiusuario que está desarrollada por la empresa Sun Microsystems. MySQL es un complemento perfecto para el desarrollo de sitios web dinámicos, como por ejemplo foros, ya que cada usuario podrá modificar el contenido de la página o simplemente ampliándolo.

2.7.5. PHPMYADMIN

phpMyAdmin es una utilidad adicional, y que, en absoluto es necesaria para el desarrollo de un sitio web dinámico que se sustente sobre una base de datos, pero facilita mucho a tarea, ya que nos permite administrar una base de datos sin necesidad de tener que recurrir a la escritura de líneas de comandos sobre la consola del equipo, ya que nos proporciona un entorno gráfico para ello, que además es bastante intuitivo.

Capítulo 3: Desarrollo de la solución propuesta

3.1. Arquitectura de la Solución

Este sistema está basado en un modelo de clases de objetos, el motor de WorkFlow es un motor simple diseñado, codificado y embebido en la Base de Datos.

El sistema está formado por:

- *Capa de Aplicación (desarrollada en php)*
- *Servidor de Aplicaciones Apache*
- *Capa de Base de Datos (Lógica del negocio)*

3.2. Modelado de la Base de Datos.

Para el modelado de la base de datos del presente proyecto se muestra en la figura 3.1. Dicho modelo permite el funcionamiento correcto del programa desarrollado para la operatividad del mismo. Adicional se debe incluir los parámetros y configuración que se muestra en la figura 3.2 y finalmente las plantillas del tipo de proyecto mostrado en la figura 3.3.

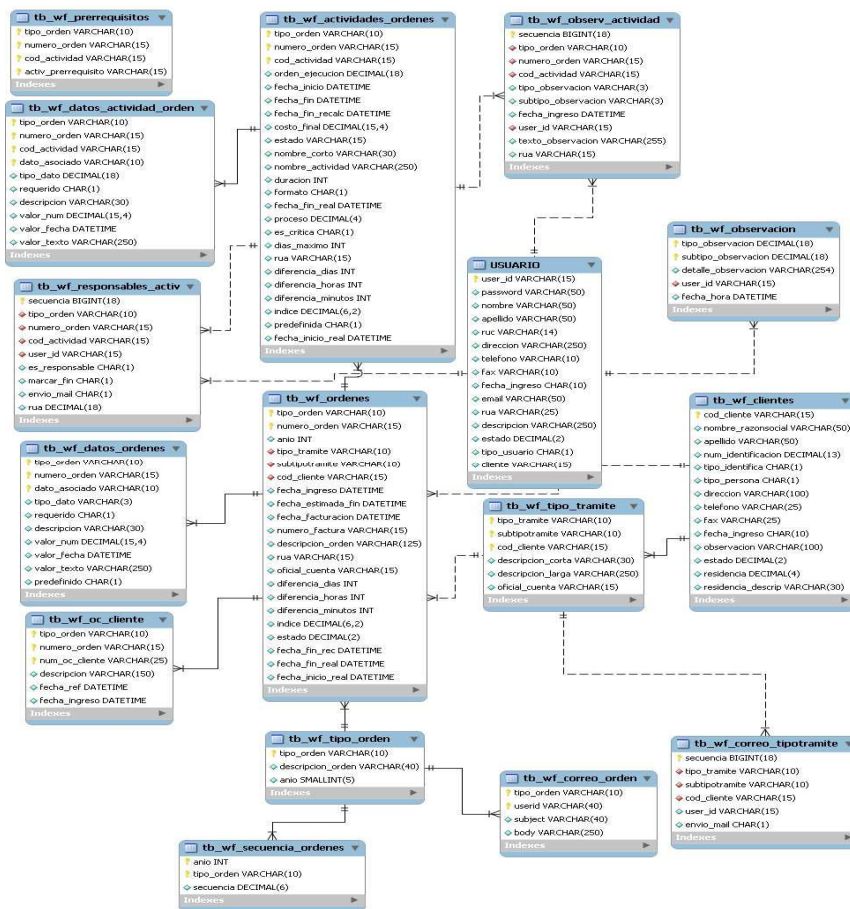


Figura 3. 1: Modelo de Órdenes o Proyectos

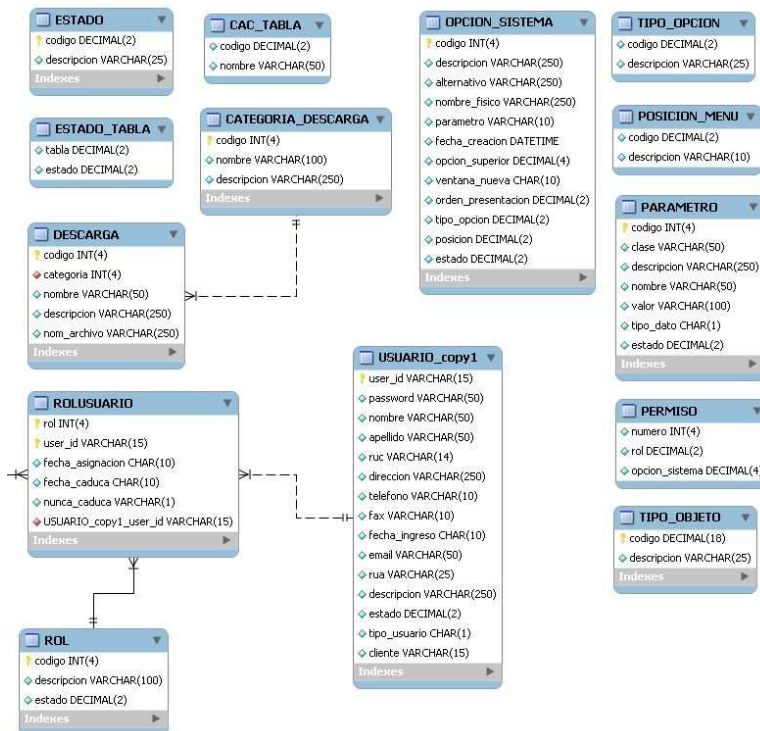


Figura 3. 2: Parámetros y Configuración

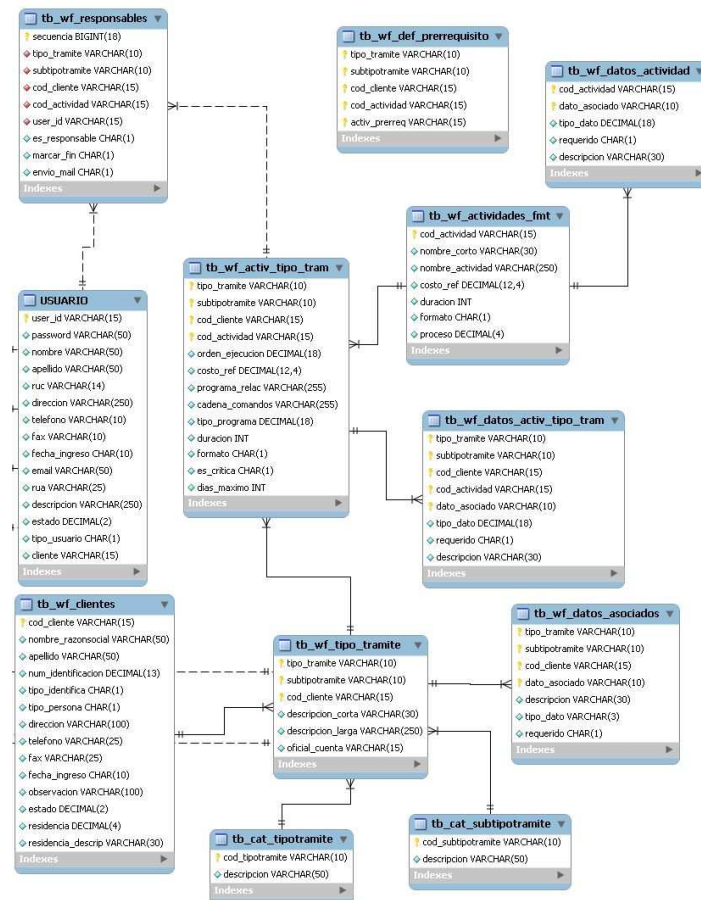


Figura 3. 3: Plantillas de Tipos de Proyectos

3.3. Diseño de Objetos.

A continuación se muestra el diseño orientado a objetos en PHP 5 del Sistema de Seguimiento de Proyectos de Construcción y Mantenimiento para Empresas de Telecomunicaciones a través de la Web:

Cls_activ_tipo_tram

```

class cls_activ_tipo_tram{
    var $tipo_tramite;
    var $cod_actividad;
    var $orden_ejecucion;
    var $costo_ref;
    var $programa_relac;
    var $cadena_comandos;

```

```

var $tipo_programa;
var $duracion;
var $formato;
var $es_critica;
function
cls_activ_tipo_tram($txt_tipo_tramite="", $txt_cod_actividad="", $txt_
orden_ejecucion=0,
$txt_costo_ref=0, $txt_programa_relac=0, $txt_cadena_comandos="
",
$txt_tipo_programa="", $txt_duracion=0, $txt_formato="", $txt_es_crit
ica=""){
    $this->tipo_tramite=$txt_tipo_tramite;
    $this->cod_actividad=$txt_cod_actividad;
    $this->orden_ejecucion=$txt_orden_ejecucion;
    $this->costo_ref=$txt_costo_ref;
    $this->programa_relac=$txt_programa_relac;
    $this->cadena_comandos=$txt_cadena_comandos;
    $this->tipo_programa=$txt_tipo_programa;
    $this->duracion=$txt_duracion;
    $this->formato=$txt_formato;
    $this->es_critica=$txt_es_critica;
}
function      setTipoTramite($tipo_tramite)      {$this-
>tipo_tramite=$tipo_tramite;}
function      setCodActividad($cod_actividad)    {$this-
>cod_actividad=$cod_actividad;}
function      setOrdenEjecucion($orden_ejecucion) {$this-
>orden_ejecucion=$orden_ejecucion;}
function      setCostoRef($costo_ref)            {$this-
>costo_ref=$costo_ref;}
function      setProgramaRelac($programa_relac)  {$this-

```

```

>programa_relac=$programa_relac;}
    function setCadenaComandos($cadena_comandos) {$this-
>cadena_comandos=$cadena_comandos;}
    function    setTipoPrograma($tipo_programa)    {$this-
>tipo_programa=$tipo_programa;}
    function    setDuracion($duracion)            {$this-
>duracion=$duracion;}
    function setFormato($formato) {$this->formato=$formato;}
    function    setEsCritica($es_critica)        {$this-
>es_critica=$es_critica;}
    function getTipoTramite() { return $this->tipo_tramite;}
    function getCodActividad(){ return $this->cod_actividad;}
    function    getOrdenEjecucion()    {    return    $this-
>orden_ejecucion;}
    function getCostoRef() { return $this->costo_ref;}
    function    getProgramaRelac()    {    return    $this-
>programa_relac;}
    function    getCadenaComandos()    {    return    $this-
>cadena_comandos;}
    function getTipoPrograma() { return $this->tipo_programa;}
    function getDuracion() { return $this->duracion;}
    function getFormato() { return $this->formato;}
    function getEsCritica() { return $this->es_critica;}

```

Cls_actividades

```

class cls_actividades{
    var $cod_actividad;
    var $nombre_corto;
    var $nombre_actividad;
    var $costo_ref;
    var $duracion;
    var $formato;

```

```

var $proceso;
function
cls_actividades($txt_cod_actividad="", $txt_nombre_corto="", $txt_nombre_actividad="",
$txt_costo_ref=0, $txt_duracion=0, $txt_formato="", $txt_proceso=0){
    $this->cod_actividad=$txt_cod_actividad;
    $this->nombre_corto=$txt_nombre_corto;
    $this->nombre_actividad=$txt_nombre_actividad;
    $this->costo_ref=$txt_costo_ref;
    $this->duracion=$txt_duracion;
    $this->formato=$txt_formato;
    $this->proceso=$txt_proceso;
}
function setCodActividad($cod_actividad) {$this->cod_actividad=$cod_actividad;}
function setNombreCorto($nombre_corto) {$this->nombre_corto=$nombre_corto;}
function setNombreActividad($nombre_actividad) {$this->nombre_actividad=$nombre_actividad;}
function setCostoRef($costo_ref) {$this->costo_ref=$costo_ref;}
function setDuracion($duracion) {$this->duracion=$duracion;}
function setFormato($formato) {$this->formato=$formato;}
function setProceso($proceso) {$this->proceso=$proceso;}
function getCodActividad(){ return $this->cod_actividad;}
function getNombreCorto(){ return $this->nombre_corto;}
function getNombreActividad(){ return $this->nombre_actividad;}
function getCosto() { return $this->costo_ref;}
function getDuracion() { return $this->duracion;}

```

```
function getFormato() { return $this->formato;}  
function getProceso() { return $this->proceso;}
```

Cls_clientes

```
class cls_clientes{  
    var $cod_cliente;  
    var $tipo_identifica;  
    var $num_identificacion;  
    var $nombre_razonsocial;  
    var $apellido;  
    var $direccion;  
    var $telefono;  
    var $fax;  
    var $tipo_persona;  
    var $fecha_ingreso;  
    var $observacion;  
    var $estado;  
    var $residencia;  
    var $residencia_descrip;  
    function  
cls_clientes($txt_cod_cliente="", $cbo_tipo_identifica="C", $txt_num  
_identificacion="",  
$txt_nombre_razonsocial="", $txt_apellido="", $txt_direccion="", $txt_  
telefono="", $txt_fax="",  
$cbo_tipo_persona="N", $txt_fecha_ingreso="", $txt_observacion="",  
$cbo_estado=1,  
$txt_residencia="", $txt_residencia_descrip=""){  
        $this->cod_cliente=$txt_cod_cliente;  
        $this->tipo_identifica=$cbo_tipo_identifica;  
        $this->num_identificacion=$txt_num_identificacion;  
        $this->nombre_razonsocial=$txt_nombre_razonsocial;  
        $this->apellido=$txt_apellido;
```

```

        $this->direccion=$txt_direccion;
        $this->telefono=$txt_telefono;
        $this->fax=$txt_fax;
        $this->tipo_persona=$cbo_tipo_persona;
        $this->fecha_ingreso=$txt_fecha_ingreso;
        $this->observacion=$txt_observacion;
        $this->estado=$cbo_estado;
        $this->residencia=$txt_residencia;
        $this->residencia_descrip=$txt_residencia_descrip;
    }
    function setCod_cliente($cod_cliente)    {$this->
cod_cliente=$cod_cliente;}
    function setNombre_razonsocial($nombre_razonsocial)
{$this->nombre_razonsocial=$nombre_razonsocial;}
    function setApellido($apellido) {$this->apellido=$apellido;}
    function setTipo_identifica($tipo_identifica)    {$this->
tipo_identifica=$tipo_identifica;}
    function setNum_identificacion($num_identificacion) {$this->
num_identificacion=$num_identificacion;}
    function setTipo_persona($tipo_persona)    {$this->
tipo_persona=$tipo_persona;}
    function setDireccion($direccion)    {$this->
direccion=$direccion;}
    function setTelefono($telefono) {$this->telefono=$telefono;}
    function setFax($fax) {$this->fax=$fax;}
    function setFecha_ingreso($fecha_ingreso)    {$this->
fecha_ingreso=$fecha_ingreso;}
    function setObservacion($observacion)    {$this->
observacion=$observacion;}
    function setEstado($estado) {$this->estado=$estado;}
    function setResidencia($residencia)    {$this->

```



```

>residencia=$residencia;}
    function setResidencia_descrip($residencia_descrip) {$this-
>residencia_descrip=$residencia_descrip;}
    function getCod_cliente(){return $this->cod_cliente;}
    function getTipo_identifica(){return $this->tipo_identifica;}
    function      getNum_identificacion(){return      $this-
>num_identificacion;}
    function      getNombre_razonsocial(){return      $this-
>nombre_razonsocial;}
    function getApellido(){return $this->apellido;}
    function getDireccion(){return $this->direccion;}
    function getTelefono(){return $this->telefono;}
    function getFax(){return $this->fax;}
    function getTipo_persona(){return $this->tipo_persona;}
    function getFecha_ingreso(){return $this->fecha_ingreso;}
    function getObservacion(){return $this->observacion;}
    function getEstado(){return $this->estado;}
    function      getResidencia()      {      return      $this-
>residencia=$residencia;}
    function      getResidencia_descrip()      {      return      $this-
>residencia_descrip=$residencia_descrip;}

```

Cls_ordenes

```

class cls_ordenes{
    var $tipo;
    var $numero;
    var $tipo_tramite;
    var $cod_cliente;
    var $fec_ingreso;
    var $fec_fin;
    var $fec_factura;
    var $numero_factura;

```

```

var $descripcion;
var $subtipo;
var $oficial;
var $anio;
var $numero_orden;//campo con formato "año-numero"
var $nom_cli;
var $desc_tram;
var $estado;
var $fec_ini_real;
var $Login;
function
cls_ordenes($txt_tipo="", $txt_numero="", $txt_tipo_tramite="",
            $txt_cod_cliente="",
            $txt_fec_ingreso="", $txt_fec_fin="", $txt_fec_factura="",
            $txt_numero_factura="", $txt_descripcion="", $txt_subtipo="", $txt_ofi
            cial="", $txt_anio="",
            $numero_orden="", $txt_nom_cli="", $txt_desc_tram="", $txt_estado=
            "", $txt_fec_ini_real="", $Login=""){
    $this->tipo=$txt_tipo;
    $this->numero=$txt_numero;
    $this->tipo_tramite=$txt_tipo_tramite;
    $this->cod_cliente=$txt_cod_cliente;
    $this->fec_ingreso=$txt_fec_ingreso;
    $this->fec_fin=$txt_fec_fin;
    $this->fec_factura=$txt_fec_factura;
    $this->numero_factura=$txt_numero_factura;
    $this->descripcion=$txt_descripcion;
    $this->subtipo=$txt_subtipo;
    $this->oficial=$txt_oficial;
    $this->anio=$txt_anio;
    $this->numero_orden=$numero_orden;

```

```

        $this->nom_cli=$txt_nom_cli;
        $this->desc_tram=$txt_desc_tram;
        $this->estado=$txt_estado;
        $this->fec_ini_real=$txt_fec_ini_real;
        $this->Login=$Login;
    }
    function settipo($tipo) { $this->tipo=$tipo; }
    function setnumero($numero) { $this->numero=$numero; }
    function      settipo_tramite($tipo_tramite)      {      $this-
>tipo_tramite=$tipo_tramite; }
    function      setcod_cliente($cod_cliente)      {      $this-
>cod_cliente=$cod_cliente; }
    function      setfec_ingreso($fec_ingreso)      {      $this-
>fec_ingreso=$fec_ingreso; }
    function setfec_fin($fec_fin) { $this->fec_fin=$fec_fin; }
    function      setfec_factura($fec_factura)      {      $this-
>fec_factura=$fec_factura; }
    function      setnumero_factura($numero_factura) { $this-
>numero_factura=$numero_factura; }
    function      setdescripcion($descripcion)      {      $this-
>descripcion=$descripcion; }
    function setsubtipo($subtipo) { $this->subtipo=$subtipo; }
    function setoficial($oficial) { $this->oficial=$oficial; }
    function setanio($anio) { $this->anio=$anio; }
    function      setnumero_orden($numero_orden)    {      $this-
>numero_orden=$numero_orden; }
    function setestado($estado) { $this->estado=$estado; }
    function      setfec_ini_real($fec_ini_real)    {      $this-
>fec_ini_real=$fec_ini_real; }
    function setLogin($Login) { $this->Login=$Login; }
    function gettipo() { return $this->tipo; }

```

```

function getnumero() { return $this->numero; }
function gettipo_tramite() { return $this->tipo_tramite; }
function getnumero_tramite() { return $this->numero_tramite;
}

function getcod_cliente() { return $this->cod_cliente; }
function getorden() { return $this->orden; }
function getfec_ingreso() { return $this->fec_ingreso; }
function getfec_fin() { return $this->fec_fin; }
function getfec_factura() { return $this->fec_factura; }
function getnumero_factura() { return $this->numero_factura;
}

function getdescripcion() { return $this->descripcion; }
function getsubtipo() { return $this->subtipo; }
function getoficial() { return $this->oficial; }
function getanio() { return $this->anio; }
function getnumero_orden() { return $this->numero_orden; }
function getestado() { return $this->estado; }
function getfec_ini_real() { return $this->fec_ini_real; }
function getLogin() { return $this->Login; }

```

Cls_tipo_tram

```

class cls_tipotramite{
    //Atributos de las Clase
    var $tipotramite;
    var $cliente;
    var $desc_corta;
    var $desc_larga;
    var $oficial_cuenta;
    var $subtipotramite;
    var $desc_tram;
    var $desc_sub;
    var $nom_cli;

```

```

function
cls_tipotramite($txt_tipotramite="", $txt_cliente="", $txt_desc_corta="
", $txt_desc_larga="", $txt_oficial_cuenta="",
$txt_subtipotramite="", $txt_tiponame="", $txt_subtiponame="", $txt_
nom_cliente=""){
    $this->tipotramite=$txt_tipotramite;
    $this->cliente=$txt_cliente;
    $this->desc_corta=$txt_desc_corta;
        $this->desc_larga=$txt_desc_larga;
        $this->oficial_cuenta=$txt_oficial_cuenta;
        $this->subtipotramite=$txt_subtipotramite;
        $this->desc_tram = $txt_tiponame;
        $this->desc_sub = $txt_subtiponame;
        $this->nom_cli = $txt_nom_cliente;
    }
    function        setTipoTramite($tipotramite)        {$this-
>tipotramite=$tipotramite;}
    function        setSubTipoTramite($subtipotramite)    {$this-
>subtipotramite=$subtipotramite;}
    function setCliente($cliente) {$this->cliente=$cliente;}
    function        setDesc_corta($desc_corta)        {$this-
>desc_corta=$desc_corta;}
    function        setDesc_larga($desc_larga)        {$this-
>desc_larga=$desc_larga;}
    function        setOficial_cuenta($oficial_cuenta)    {$this-
>oficial_cuenta=$oficial_cuenta;}
    function        setdesc_tram($desc_tram)            {$this-
>desc_tram=$desc_tram;}
    function        setdesc_sub($desc_sub)              {$this-
>desc_sub=$desc_sub;}
    function setOnom_cli($nom_cli) {$this->nom_cli=$nom_cli;}

```

```
function getTipoTramite(){return $this->tipotramite;}
function getSubTipoTramite(){return $this->subtipotramite;}
function getCliete(){return $this->cliente;}
function getDesc_corta(){return $this->desc_corta;}
function getDesc_larga(){return $this->desc_larga;}
function getOficial_cuenta(){return $this->oficial_cuenta;}
function getdesc_tram(){return $this->desc_tram;}
function getdesc_sub(){return $this->desc_sub;}
function getnom_cli(){return $this->nom_cli;}
```

3.4. Alcances y Restricciones del software

El sistema fue orientado originalmente para llevar el control de actividades operativas de una empresa, lo cual significa que la empresa que desee implantar el sistema debe contar con la definición clara de sus procesos. Sin embargo esto no lo limita a unos pocos tipos de proyectos. Es posible definir actividades por casi cualquier tipo de proyectos, como por ejemplo:

- *Proyectos de construcción*
- *Gestión de instituciones educativas*
- *Gestión de empresas de servicio*
- *Gestión de logística*
- *Gestión de tecnologías*

Esta característica obliga en cierta forma a documentar los procedimientos de la empresa, colaborando de esta manera con la organización. Además puede servir como herramienta para iniciar el camino a una certificación de calidad tipo ISO 9000. Para lograr estos objetivos se presenta una herramienta parametrizada y flexible, que permita hacer el seguimiento a proyectos de diversos tipos, denominados "Orden de Trabajo". Cada "Orden de Trabajo" se relaciona con un conjunto de actividades y usuarios que deben encargarse del seguimiento de éstas.

Cada actividad podrá dar flujo a otras actividades, de manera que se forman Flujos de Trabajo con tiempos de inicio y duración de cada actividad. La duración de las actividades puede medirse en Semanas, Días, Horas o Minutos. Es posible asignarle presupuesto de costos a un proyecto, y a medida que el proyecto avance, agregarle costos referenciales asignados al proyecto y costos referenciales asignadas a cada actividad. De esta manera se podrán comparar luego el costo presupuestado con los referenciales. Cada actividad que finaliza, mueve el tiempo de inicio de sus actividades sucesoras, de forma que el tiempo total de la "Orden de Trabajo", modifica su fecha final estimada. Estos tiempos se guardan diferenciándolos de los tiempos inicial y final originales, de forma que existen los tiempos recalculados inicial y final.

3.4.1. Restricciones

El sistema trata de cumplir con los requisitos básicos para la administración de proyectos, como definición de tareas, actividades, responsables y flujos, sin embargo dado que el sistema está orientado a servir como herramienta de apoyo para el seguimiento y control de las diversas actividades planificadas, inclusive le otorga la responsabilidad al cliente de marcar el cumplimiento de una o varias actividades. Así mismo se puede parametrizar para que el cliente reciba una notificación por correo electrónico del estado de avance del proyecto. Para resumir algunas de las características del sistema se definen en la tabla 3.1.

Tabla3. 1: Características del Sistema

Características	
Arquitectura	2 capas
Lógica del negocio almacenada en	BD - stored procedures
Interfaz de cliente	Basado en Web
Máximo número de actividades	Ilimitado
Lenguaje de programación	php + sql script
Menús parametrizados	SI

Acceso por contraseña	SI
Definición de actividades	SI
Definición de plantillas	SI
Plantillas personalizables por cliente	SI
Definición de flujos de actividades	SI
Seguimiento por actividades/usuario	SI
Registrar costos para actividades	SI
Marcar observaciones por actividad	SI
Email sobre estado de actividades	SI
Calcular índices de eficiencia	SI
Reportes de avances	SI
Gráficos de Gantt	NO
Máximo número de regiones	Una
Flujos de actividades condicionados	No
Asignación de documentos	No
Análisis de costos unitarios	No

El sistema mantiene la característica principal de las herramientas para administración de proyectos; “las actividades pueden agruparse para formar flujos de trabajo”, sin embargo solamente se permiten aquellos grafos que contengan una sola región. A continuación se explica mediante gráficos, algunos ejemplos de grafos de flujos de actividades que pueden definirse y otros que no pueden definirse en esta versión del programa.

3.4.2. Grafo sin ciclos con región única

Este es un ejemplo de un grafo de flujo con una sola región, que puede definirse en el sistema. Supongamos que un proyecto está definido con seis actividades enumeradas de la **A** hasta la **F**, y presentan el siguiente flujo (ver figura 3.4).

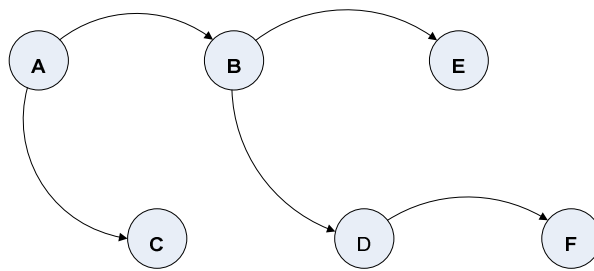


Figura 3. 4: Flujo de actividades con 1 sola región, grafo a cíclico

La figura 3.4 muestra el flujo de dependencias de las actividades, para registrarlo en el sistema, debemos llenar una matriz de adyacencia como la que se muestra a continuación:

	A	B	C	D	E	F
A						
B	X					
C	X					
D		X				
E		X				
F				X		

3.4.3. Grafos cíclicos

Los grafos de flujo de actividades que presentan ciclos, no son permitidos en esta versión, por lo tanto, al momento de guardar su definición de dependencias o flujo de actividades, el sistema le indicará cuando se haya encontrado un ciclo, un ejemplo que describe lo anterior se muestra en la figura 3.5.

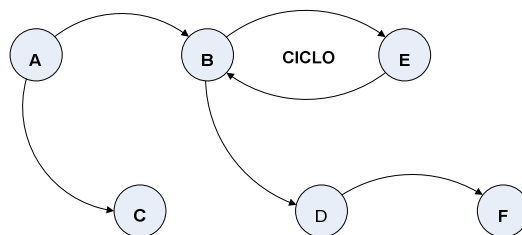


Figura 3. 5: Flujo de actividades cíclica, en este ejemplo el grafo posee 2 regiones

El flujo de dependencias (ver figura 3.5) de las actividades en un sistema con un ciclo, que forma dos regiones. Este tipo de grafo no se puede registrar en el sistema, sin embargo se muestra la matriz de adyacencia que le correspondería a

este grafo:

	A	B	C	D	E	F
A						
B	X				X	
C	X					
D		X				
E		X				
F				X		

3.4.4. Grafos multi-camino

Otro ejemplo de grafo de dependencias de actividades que no está permitido en esta versión del programa es aquel que define múltiples caminos hasta una actividad, desde un mismo origen, es decir que tiene varios predecesores, como se muestra en la figura 3.6.

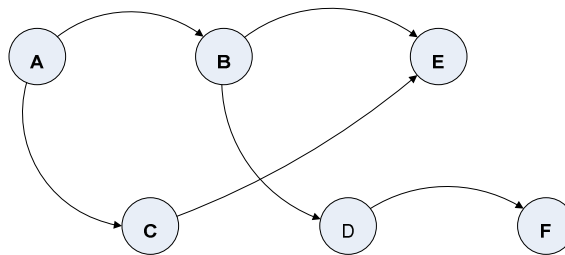


Figura 3. 6: Dos caminos diferentes desde el mismo origen y con el mismo destino

La figura 3.6 muestra el flujo de dependencias de las actividades, donde existen dos caminos diferentes para llegar desde la actividad A hacia la actividad E, este es otro ejemplo de grafo de flujo con dos regiones y no se puede registrar en el sistema, sin embargo se muestra la matriz de adyacencia que le correspondería a este grafo:

	A	B	C	D	E	F
A						
B	X					
C	X					
D		X				
E		X				
F				X		

Otro ejemplo de grafo de flujos multicamino se muestra en la figura 3.7.

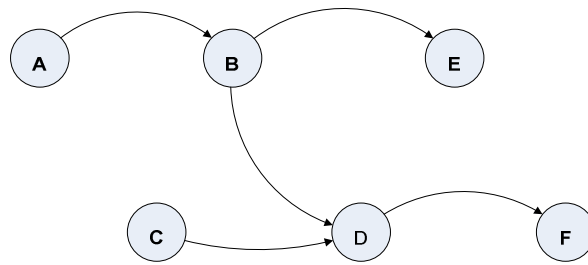


Figura 3. 7: Dos caminos diferentes hacia el mismo destino, desde orígenes distintos

La figura 3.7 se muestra el flujo de dependencias de las actividades, donde existen dos caminos diferentes para llegar a la actividad E, los cuales son: A, B, D, E y C, D, E, este ejemplo sin embargo representa un grafo de flujo con una sola región y no se puede registrar en el sistema, sin embargo se muestra la matriz de adyacencia que le correspondería a este grafo:

	A	B	C	D	E	F
A						
B	X					
C	X					
D		X				
E		X				
F				X		

Para permitir estos casos en una próxima versión, se deberá incluir en cada actividad el número de predecesores pendientes, donde cada vez que un prerrequisito finalice, se re calculan los tiempos y se disminuye este número en uno, cuando este valor llegue a cero, entonces la actividad sucesora podrá iniciar.

3.5. Seguridad y Control de acceso

La seguridad de la información es uno de los temas principales para cualquier sistema, sobretodo si estará basado en una plataforma web. Por ello se han diseñado tres niveles de seguridad y control de acceso:

3.5.1. Seguridad general de acceso a la base de datos

En la base de datos MySQL debe definirse y registrarse el usuario que tendrá acceso a la base de datos del sistema, y además se debe registrar la dirección IP del host desde el cual se accederá a la base. Este es un control propio del motor de MySQL y que puede ser fácilmente configurado desde cualquier administrador de la base de datos, como por ejemplo phpMyAdmin.

3.5.2. Seguridad de acceso al sistema

El acceso al sistema es el siguiente nivel de control, cada contraseña de usuario se registra en la base de datos, encriptándolas con MD5. Si desea mayor nivel de protección, puede reemplazarlo con algún algoritmo de cifrado como por ejemplo TwoFish. Una vez definido el usuario, es importante asignarle un rol dentro del sistema, si no lo hace, entonces simplemente no tendrá acceso a ninguna de las opciones del menú.

Cada rol posee la capacidad de asociarse a una o varias opciones del menú principal del sistema (ver figura 3.8), y el sistema permite definir el número de roles que desee, sin embargo se recomienda un número no mayor a cuatro roles, entre los cuales se encuentran:

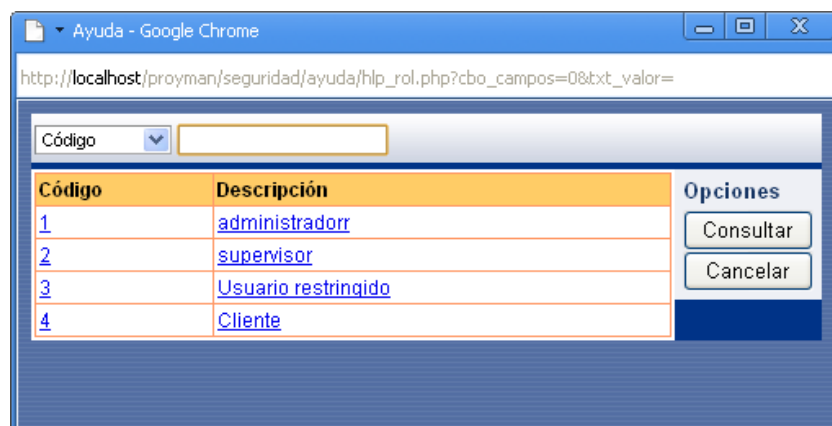


Figura 3. 8: Tipos de roles predefinidos

Fuente: El autor

Rol Administrador del sistema.- tiene la capacidad de definir a otros usuarios, modificar el orden de aparición del menú, asignar permisos nuevos, definir nuevos

roles, etc., y no debe interferir en el seguimiento de los proyectos.

Rol de Supervisor.- tiene la capacidad de observar el avance de los proyectos y marcar la finalización de algunas actividades que están bajo su responsabilidad.

Rol de usuario restringido.- este rol se definió para aquellos usuarios que únicamente pueden consultar ciertas opciones del sistema, pueden ser empleados de la misma compañía que cumplen un papel de control sobre ciertas actividades dentro del proyecto.

Rol de Cliente.- se utiliza para identificar el conjunto de opciones del sistema que estarán disponibles para los clientes que hagan uso de la aplicación.

3.5.3. Seguridad de acceso a la información

El acceso a la información está relacionado con el tipo de usuario definido en el sistema y con el rol que le fue asignado, esto permite dividir este acceso en dos. Por un lado el control de roles le da acceso al menú con las opciones definidas, y por otro lado el tipo de usuario determina las atribuciones que puede tener en la manipulación de la información, por ello se definen los siguientes tipos de usuarios:

Tipos de Usuarios	Descripción
C	Cliente
F	Gerente Proyecto
A	Administrador
O	Supervisor
T	Otros

Los usuarios del tipo F, C y O (Gerente Proyecto, Cliente y Supervisor), tienen las siguientes propiedades dentro del sistema:

- Pueden consultar todos los proyectos, excepto el tipo Cliente, quien solamente podrá consultar sus proyectos y podrán ver todas las actividades
- Pueden ver datos y observaciones si son responsables

- Pueden Marcar Fin de la actividad en aquellas en las que esté autorizado
- Puede enviar email a los destinatarios predefinidos de una actividad, si está marcado como el responsable de dicha actividad

Los demás usuarios (tipo A y T), tienen las siguientes propiedades:

- Podrán ver las actividades en las cuales el usuario sea responsable
- Puede ingresar nuevos datos a las actividades
- Puede marcar fin si esta autorizado
- Puede ingresar observaciones
- Puede enviar email a los destinatarios predefinidos de una actividad, si está marcado como el responsable de dicha actividad

3.6. Parametrización General

El sistema contiene una serie de parámetros que le permiten operar y brindar servicios a los usuarios, entre éstos se encuentran:

- Códigos de tablas definidas en el sistema.- es el código que se utiliza para identificar el listado de tipos de datos asociados a las actividades.
- Pié de página utilizado en los reportes.- texto que aparece en el pie de página de todos los reportes.
- Intervalo de reordenamiento de las actividades.- cuando se añaden nuevas actividades, es posible reordenarlas automáticamente, el sistema les dará un nuevo número de secuencia en intervalos iguales.
- Parámetros para correos electrónicos.- se definen los datos utilizados en la cláusula desde (From) y asunto (Subject). Tanto para cuando se inicia una orden de trabajo, cuando se hace un envío explícito y para cuando una actividad ha finalizado
 - From
 - Subject
- Código de la tabla para definir los procesos programables que se ejecutan con las actividades. Aunque esto es una funcionalidad que no está vigente, está considerada para añadirse en la siguiente versión del sistema

Catálogo de Tablas

Los campos con * son obligatorios

Información General

Descripción: * Código:

Tipo Objeto:

Opción Superior: ...

Valor:

Figura 3. 9: Parametrización por catálogo de tablas
Fuente: El autor

3.6.1. Administración de Feriados y Jornada laboral

La administración de las actividades de un proyecto no sería posible de gestionar efectivamente en sus tiempos, si no se consideran los días feriados y los días laborables, así como la jornada laboral. Lamentablemente en esta versión la jornada laboral se define en forma general, es decir que afecta a todas las actividades que no han finalizado y las que no han iniciado aún, en todos los proyectos definidos dentro del sistema. En la siguiente versión se prevé incluir la jornada laboral como parte de la información de cada proyecto.

Los fines de semana han sido considerados dentro de la estructura para días feriados, sin embargo esta característica se prevé mejorarla para la siguiente versión, al igual que también se previene mejorar el control de feriados para que sean heredables cada año. La definición de la jornada laboral, es utilizada para definir el inicio y fin de cada actividad asociada al proyecto, lo cual también afecta la fecha final estimada del proyecto. Modificar la jornada laboral, afectará al cálculo de las fechas de inicio de las actividades pendientes, las actividades finalizadas no serán afectadas.

3.6.2. Definición de Categorías de proyectos

El sistema permite llevar un registro de las diferentes categorías de proyectos (vr figura 3.10) que se pueden manejar, para ello se definen categorías y subcategorías de proyectos, que luego son utilizados para incluir las plantillas de actividades con

todos los detalles que éstas requieren.

Catálogo de Tipo de Proyectos

Los campos con * son obligatorios

Información General

Tipo de Proyecto: 110 ? *

Descripción: Proceso de graduación *

Nuevo Cancelar Modificar Eliminar

Figura 3. 10: Definición de Tipos de Proyectos

Fuente: El autor

3.7. Plantillas

Las plantillas son conjuntos de Proyectos, definidos por categorías y subcategorías, las cuales pueden contener un conjunto de actividades con su propio flujo de proceso. Las plantillas pueden ser definidas para:

- Uso general, una plantilla de uso general es aquella que no está asociada a un cliente en particular.
- Asociadas a un cliente en particular, es posible definir una plantilla de una categoría/subcategoría dada, para asociarla a un cliente en particular, esto se utiliza cuando el cliente tiene una manera un poco diferente de registrar sus procesos en un proyecto determinado.

La definición de las plantillas de proyectos se realiza definiendo la categoría y subcategoría del proyecto, luego se determina si va a estar asociada con un usuario o es de uso general, como se muestra en la figura 3.11.

Tipos de Proyectos

Los campos con * son obligatorios

Información General

Tipo de Proyecto: 110
Proceso de graduación

Subtipo Proyecto: 10
Postgrado

Cliente: 10
Espol
 Seleccione un Cliente Seleccione TODOS los Clientes

Supervisor : edgar *

Referencia: Imp. Consumo Puerto CONTEN

Descripción: Importación a Consumo Puerto Guayaquil Contenedores

© 2004 Desarrollado bajo licencia GPL. Todos los derechos reservados [Cont-Science]

Figura 3. 11: Definición de plantillas de proyectos
Fuente: El autor

Es posible que necesite notificar a uno o varios usuarios, sobre el estado de un proyecto, para ello es necesario asociar cuentas de usuarios definidos en el sistema con cada plantilla de proyecto (ver figura 3.12). Esta asignación se realiza presionando el botón **Asignar Correos**.



Figura 3. 12: Definición de usuarios que recibirán avisos por email de proyectos
Fuente: El autor

Como se puede apreciar en la figura 3.12 el listado de usuarios registrados en el sistema, si se desea que los usuarios reciban notificaciones por correo del estado de avances del proyecto, entonces se debe marcar la casilla etiquetada "Enviar Email". Las notificaciones que recibe un usuario se relacionan con el estado de avance de las actividades asociadas al proyecto, cada proyecto tiene su propia definición de usuarios o notificaciones por correo electrónico (email).

3.7.1. Definición de etapas de un proyecto

Desde un punto de vista muy general, podría considerarse que todo proyecto tiene al menos tres etapas bien identificadas, que son: Planeación, Ejecución y Entrega o puesta en marcha.

Etapas de Planeación(Desmond, 2004).- durante esta etapa, el equipo de trabajo se debe concentrar en determinar el alcance del proyecto, satisfacer las necesidades y restricciones, estimar los costes, número de recursos, tiempos de ejecución. Durante esta etapa se descompone el proyecto en actividades que luego serán sincronizadas y ejecutadas de acuerdo con un orden específico con los recursos

que se requieran para ello.

Etapa de Ejecución (Heerkens, 2007).- determinada por las características propias de cada tipo de proyecto, y consiste en la realización misma de las actividades definidas durante la planeación, supone además la gestión adecuada de los recursos asignados para llevar el proyecto hacia su etapa final y el monitoreo de los avances previstos según los tiempos asignados a cada tarea durante la planeación.

Etapa de Entrega o puesta en marcha.- (Cleland & Ireland, 2006) todo proyecto se inicia con el objetivo de culminarse en un plazo de tiempo determinado y con los recursos definidos para ello, entregando finalmente el proyecto al cliente. Sin embargo, en la práctica, pueden presentarse complicaciones que alarguen innecesariamente la duración del proyecto, elevando los costos y provocando retrasos imprevistos.

La figura 3.13 muestra una ampliación de las etapas. El sistema que proponemos, permitirá administrar y hacer un seguimiento del proyecto en las etapas de Ejecución, Control y Cierre.



Figura 3. 13: Etapas generales de la administración de Proyectos
Fuente: El autor

Las etapas de Inicio y Planeación obedecen más a un análisis particular que no está considerado dentro de la solución propuesta en esta tesis, debido a los factores que explicamos en el Capítulo 2. Sin embargo el resultado que se obtiene durante la etapa de Planeación sirve como dato de entrada al sistema para poder encargarse

de las etapas siguientes. (Knutson & Bitz, 1991)

3.7.2. Definición de Actividades Generales de proyectos

Es posible contar con una definición general de actividades(Desmond, 2004) que pueden utilizarse en varios proyectos. Aquellas que son de uso frecuente, o que se repiten en diversos proyectos y que forman parte de las tareas o actividades generales que se deben realizar sin importar el proyecto que se lleve a cabo. El registro de las actividades generales, se realiza mediante la pantalla que se muestra en la figura 3.14.

Los campos con * son obligatorios

Información General

Código Actividad: ASIORDTRA

Referencia: Asignación de Orden de trabajo

Descripción: Asignación de Orden de trabajo

Costo Referencial: 0.0000

Duración: 10 Día

Proceso: Seleccione

Nuevo Cancelar Modificar Eliminar Datos >>

Figura 3. 14: Definición de Actividades Generales

Fuente: El autor

3.7.3. Registro de plantillas por categoría de proyectos

El sistema, se basa en definir plantillas de proyectos. (Knutson & Bitz, 1991) Las actividades se definen durante la etapa de Planeación del proyecto, estas actividades son ingresadas al sistema, dentro de la categoría de proyecto que le corresponda, lo cual permite de alguna manera, poder definir plantillas generales de actividades.

Para definir una plantilla(Desmond, 2004) en el sistema, se debe empezar por la documentación de los procesos, lo cual se traduce en la definición de las actividades que se deben llevar a cabo para concluir un proyecto determinado. Para la figura 3.15, se muestra una plantilla de actividades para un cliente específico, sin embargo, es posible definir plantillas de actividades para proyectos que corresponden a una categoría específica, de esta manera podemos utilizar esta plantilla con cualquier cliente que requiera iniciar un proyecto. (Knutson & Bitz, 1991)

Registro de Actividades por Tipo de Proyecto

Los campos con * son obligatorios

Información General

Proyecto : 110 10 Imp. Conusmo Puerto Conten
 Cliente : 10 Espol
 Actividad : *

Especificaciones Actividad

Orden Ejecución : *
 Costo Ref :
 Duración : * Día *
 Es Crítica ? : Día(s)

Aceptar
Eliminar

Guardar Consultar Flujo PDF Cerrar

Lista de Actividades

Actividad	Descripción	Ord.Ejec.	Duración	Es Crítica	Responsable	Datos Act.
ASIORDTRA	Asignación de Orden de trabajo	10	30 Minuto	SI - 10 día(s)	Usuario	Datos
REVFISDOC	Revisión física documentos	20	30 Minuto	No	Usuario	Datos
ENVDOCCAE	Envío documentación CAE	30	4 Hora	No	Usuario	Datos
INGDATSIS	Ingreso de datos al sistema	40	30 Minuto	No	Usuario	Datos
ENVRECCDAU	Envío y recepción DAU	50	1 Hora	SI - 1 día(s)	Usuario	Datos
OBTCEERINS	Obt. cert. inspección	60	1 Día	No	Usuario	Datos
LIQDEF	Liquidación definitiva	70	2 Hora	No	Usuario	Datos
PLATRA	Planificación de transporte	80	120 Minuto	No	Usuario	Datos

Figura 3. 15: Plantilla de Actividades para un tipo de Proyecto
 Fuente: El autor

3.8. Registro de proyectos

La planificación inicia con el registro de los proyectos (ver figura 3.16), para registrar un proyecto que va a empezar a monitorearse, es importante definir una Orden de Trabajo.

Registro de Ordenes

Los campos con * son obligatorios

Información General

Año - Tipo - Num. de Orden: 2008 - 1 - 4
 Cliente : 10 Espol
 Tipo de Proyecto : 110 10 Imp. Conusmo Puerto Conten
 Supervisor : edgar
 Descripción : Ejemplo de prueba
 Estado : **PENDIENTE**

Ordenes de Clientes

Insertar Línea Eliminar Línea

Elim.	Numero de OC	Descripción	Fecha
<input type="checkbox"/>			

Nuevo Cancelar Modificar Eliminar
 PDF Asociar Datos Actividades

© 2004 Desarrollado bajo licencia GPL. Todos los derechos reservados [Cambiar]

Figura 3. 16: Registro de orden de trabajo para un proyecto

Fuente: El autor

Una Orden de Trabajo se crea cuando se va a iniciar un proyecto de una categoría específica. Al crear una Orden de Trabajo, se crean todas las actividades definidas en la plantilla del tipo de proyecto seleccionado, colocando la estimación de tiempos para cada actividad, y calculando la fecha final estimada para el proyecto.(Sherif, 2006)

3.8.1. Asignación de tiempos

Una vez iniciado un proyecto, los tiempos(Sherif, 2006)se asignan al proyecto dependiendo de la plantilla seleccionada. Cada plantilla incluye la definición de las actividades, determinadas según la documentación de los procesos de la empresa, así como también los tiempos y flujos de ejecución de cada una de las actividades (ver figura 3.17). No es posible modificar las actividades heredadas de una plantilla, pues éstas obedecen a las definiciones de los procedimientos de la empresa, y por lo tanto no deben ser alteradas.

Los campos con * son obligatorios

Información General

Proyecto : 110 10 Imp. Conusmo Puerto Conten

Cliente : 10 Espol

Actividad : *

Especificaciones Actividad

Orden Ejecución : *

Costo Ref :

Duración : * Día *

Es Crítica ? : Día(s)

Aceptar Eliminar

Guardar Consultar Flujo PDF Cerrar

Lista de Actividades

Actividad	Descripción	Ord.Ejec.	Duración	Es Crítica	Responsable	Datos Act.
ASIORDTRA	Asignación de Orden de trabajo	10	30 Minuto	Si - 10 día(s)	Usuario	Datos
REVFISDOC	Revisión física documentos	20	30 Minuto	No	Usuario	Datos
ENVDOCCAE	Envío documentación CAE	30	4 Hora	No	Usuario	Datos
INGDATSIS	Ingreso de datos al sistema	40	30 Minuto	No	Usuario	Datos
ENVRECDAU	Envío y recepción DAU	50	1 Hora	Si - 1 día(s)	Usuario	Datos
OBTCERINS	Obt. cert. inspección	60	1 Día	No	Usuario	Datos
LIQDEF	Liquidación definitiva	70	2 Hora	No	Usuario	Datos
PLATRA	Planificación de transporte	80	120 Minuto	No	Usuario	Datos

Figura 3. 17: Asignación de tiempos para las actividades de un tipo de proyecto

Fuente: El autor

Los tiempos definen la duración de las actividades(Desmond, 2004), y ésta se

determina en días, horas o minutos, adicionalmente puede determinarse si una actividad se convierte en crítica, cuando ha sobrepasado un límite de tiempo definido, por ejemplo: Una actividad tiene un tiempo de ejecución asignado de 1 hora, y se ha determinado que se vuelve crítica cuando ha pasado 1 día sin completarse.

3.8.2. Responsables de las actividades

A cada actividad definida en una plantilla, se le puede asignar uno o varios responsables(Sherif, 2006), sin embargo únicamente uno de ellos es quien debe marcar la finalización de dicha actividad.Es posible asociar uno o más usuarios a una actividad, entre las tareas de los usuarios se encuentran:

- Ser responsables(Desmond, 2004) de la actividad
- Marcar la finalización(Desmond, 2004) de la actividad
- Enviar notificaciones por email a los usuarios definidos en las notificaciones por email del proyecto.



Figura 3. 18: Registro de usuarios asignados a una actividad

Fuente: El autor

3.8.3. Elaboración de redes

Las redes definen el comportamiento de las actividades dentro del proyecto. Una red es un grafo de flujo de actividades, donde se define el orden de ejecución y por consiguiente el seguimiento(Desmond, 2004) de cada una de ellas. La definición de las redes(Desmond, 2004)se revisó en la sección 3.1.3 “Grafo acíclico con una sola región”, en la figura 3.19 se muestra otro ejemplo de un grafo acíclico.

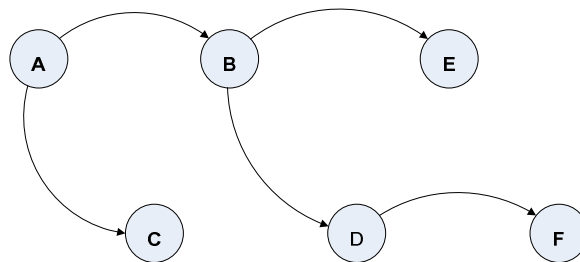


Figura 3. 19: Flujo de actividades con 1 sola región, grafo acíclico

Fuente: El autor

Esta definición se logra llenando una matriz de adyacencia(Desmond, 2004), la que se muestra en la figura 3.20.

Flujo de Actividades del Tipo de Proyecto

Los campos con * son obligatorios

Información General

Proyecto: 110 10
Imp. Consumo Puerto Cor

Cliente: 10
Espol

Actividades del Trámite

	Predecesores									
	1	2	3	4	5	6	7	8	9	10
1) Asignación de Orden de trabajo	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2) Revisión física documentos	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3) Envío documentación CAE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4) Ingreso de datos al sistema	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5) Envío y recepción DAU	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6) Obt. cert. inspección	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7) Liquidación definitiva	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8) Planificación de transporte	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9) Transportación de contenedor	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10) Org. documentos finalizados	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	1	2	3	4	5	6	7	8	9	10

Figura 3. 20: Matriz de adyacencia, para flujo de actividades

Fuente: El autor

3.9. Seguimiento de proyectos

Uno de los principales informes que presenta la aplicación es el informe de seguimiento a los proyectos. El seguimiento muestra un resumen de las actividades terminadas y permite marcar la finalización de las actividades pendientes, recordemos que para que un usuario pueda marcar el fin de una actividad, es necesario que haya sido definido dentro de la plantilla del tipo de proyecto. (Desmond, 2004)

3.10. Cálculo de costos del proyecto

El cálculo de los costos del proyecto se realiza de manera muy sencilla, simplemente se suman los costos de cada una de las actividades asociadas al proyecto. Cada actividad se define con un costo referencial, el cual no se modifica durante la ejecución del proyecto, sin embargo es posible asignar costos en forma de Datos para cada actividad, estos datos se suman para calcular el costo del proyecto. (Heerkens, 2007)

3.11. Índices de eficiencia

Los índices de eficiencia se calculan para cada una de las actividades del proyecto y se lo realiza computando la diferencia entre el tiempo total estimado y el tiempo total real de culminación de una actividad. (Cleland & Ireland, 2006) El resultado es un valor que puede ser:

- *Igual a uno, la actividad terminó en el tiempo determinado.*
- *Menor a uno, la actividad terminó antes del tiempo determinado.*
- *Mayor a uno, la actividad terminó atrasada.*

Los índices de eficiencia aparecen en un informe, listando por cada actividad el índice calculado. El cálculo del índice de eficiencia puede modificarse alterando el procedimiento almacenado "sp_calcula_indice".

3.12. Gráficos y Reportes

El sistema presenta tres reportes principales, los datos de cada uno pueden ser utilizados para ser exportados a Excel, y a partir de ellos generar informes gráficos aprovechando las herramientas de gráficos de Excel, según la necesidad del usuario. Los reportes que se presentan en esta versión discriminan el tipo de usuario, de tal manera que se cumple lo siguiente:

- Los usuarios de tipo Clientes, solo podrán consultar sus proyectos,
- Los usuarios Supervisores, podrán visualizar todos los proyectos que se les haya asignado.
- Los administradores, podrán visualizar todos los proyectos de todos los supervisores y clientes.

3.12.1. Informe de Actividades pendientes

La figura 3.21 muestra la lista del informe de las actividades que aún no han culminado, a la fecha de corte indicada, pudiendo filtrarse los resultados por Cliente, Supervisor o Tipo de Proyecto.

Los campos con * son obligatorios

Informe de Actividades Pendientes

Información General

Fecha : 2008-10-17 *

Cliente : * TODOS

Supervisor : * TODOS

Tipo Proyecto : * TODOS

Contenido : Resumido Completo

Resultado ordenado por : Cliente, Tipo Proyecto, Orden Supervisor, Cliente, Tipo Proyecto, Orden

PDF XLS

Figura 3. 21: Datos para generar informe de actividades pendientes

Fuente: El autor

De acuerdo a la figura 3.21 podemos ver si el tipo del contenido es:

- ✓ *Resumido*.- Muestra todas las Actividades No finalizadas que no tengan predecesores y todas las actividades que tengan al menos un predecesor finalizado.
- ✓ *Completo*.- Muestra todas las actividades finalizadas y no finalizadas.

Los resultados del informe pueden mostrarse ordenados y agrupados por dos criterios:

- a) Cliente, Tipo de Proyecto y Número de Orden,
- b) Supervisor, Cliente Tipo de Proyecto y Número de Orden.

El informe mostrado por la figura 3.22 nos indica los costos referenciales de cada actividad y el tiempo de atraso en días – horas: minutos: segundos.

INFORME DE ACTIVIDADES PENDIENTES

Fecha de corte de actividades : 2008-10-17
 Supervisor : Todos los Supervisores
 Cliente: Todos los clientes
 Proyecto: Todos los tipos de Proyecto
 Descargar solo resumen: SI
 Ordenado por : Cliente, Tipo de Proyecto, Orden

Tipo de Proyecto	Orden	Descripción Orden	Fec.Planificada	Fec.Recalculada	Referencia	Estado	Costo
Cód.Activ.	Descripción Actividad				Dias-Horas?		
Cliente: Espol							
Supervisor: ddd ddd							
110-10	2005-1-0034	Prueba de inicio de ordenes al crear	2005-09-17 19:30:19	2005-09-17 19:30:19			
COGAPO		Coordinación de aforo	2005-09-17 20:19:30	2005-09-17 20:19:30	29 - 7:13:20	Atrasado	
INFAPD	1900-01-01	Informe de aforo	00:00:00	1900-01-01 00:00:00	278 - 7:13:20	Atrasado	
REVFISDOC		Revisión física documentos	2005-09-20 09:10:00	2005-09-20 09:10:00	27 - 6:33:20	Atrasado	0.00
110-10	2005-2-0022	none	2005-09-30 19:09:24	2005-09-30 19:09:24			
ASFOROTRA		Asignación de Orden de trabajo	2005-10-01 09:00:00	2005-10-01 09:00:00	16 - 6:43:20	Atrasado	21.00
110-10	2005-2-0023	none	2005-09-30 19:17:08	2005-09-30 19:17:08			
ASFOROTRA		Asignación de Orden de trabajo	2005-10-01 09:00:00	2005-10-01 09:00:00	16 - 6:43:20	Atrasado	21.00
110-10	2005-2-0027	qwqw	2005-09-30 19:49:08	2005-09-30 19:49:08			
REVFISDOC		Revisión física documentos	2005-10-01 09:30:00	2005-10-01 14:49:00	16 - 6:13:20	Atrasado	0.00
TOTALES DE SUPERVISOR: ddd ddd							Costo: 42
Supervisor: Elizalde Jannett							
110-10-10	2005-1-0035	none	2005-09-30 11:42:19	2005-09-30 11:42:19	00-987		
COMDAUFIS		Comprobación de DAU	2005-09-30 14:52:00	2005-09-30 14:52:00	17 - 0:51:20	Atrasado	0.00
COGAPO		Coordinación de aforo	2005-10-01 14:52:00	2005-10-01 14:52:00	16 - 0:51:20	Atrasado	0.00
ELAGUIREM		Elaboración de guía remisión	2005-10-02 12:52:00	2005-10-02 12:52:00	15 - 2:51:20	Atrasado	0.00
ENVDOCAE		Envío documentación CAE	2005-10-02 13:22:00	2005-10-02 13:22:00	15 - 2:12:20	Atrasado	0.00
INFAPD		Informe de aforo	2005-10-03 08:52:00	2005-10-03 08:52:00	14 - 6:51:20	Atrasado	0.00
LIQDEF		Liquidación definitiva	2005-10-03 09:52:00	2005-10-03 09:52:00	14 - 5:51:20	Atrasado	0.00
OPTCARSAL		obtención carta salida	2005-10-03 12:52:00	2005-10-03 12:52:00	14 - 2:51:20	Atrasado	0.00
OPTCERINS		obt. cert. inspección	2005-10-05 12:52:00	2005-10-05 12:52:00	12 - 2:51:20	Atrasado	0.00
OPTTABALM		obt. tarjeta almacén temp.	2005-11-04 12:52:00	2005-11-04 12:52:00	17 - 21:9:40	Atrasado	0.00
PLATZA		Planificación de transporte	2005-11-05 13:17:00	2005-11-05 13:17:00	19 - 21:33:40	Atrasado	0.00
PREDOCAE		Pres.declaración CAE	2005-11-05 13:32:00	2005-11-05 13:32:00	19 - 21:48:40	Atrasado	0.00
PREDOCAE		Preparación de documentos CAE	2005-11-05 14:02:00	2005-11-05 14:02:00	19 - 22:10:40	Atrasado	0.00
REVFISDOC		Revisión física documentos	2005-12-05 14:02:00	2005-12-05 14:02:00	49 - 22:10:40	Atrasado	0.00
110-10-10	2005-1-0036	Prueba nueva del flujo	2005-09-30 12:14:09	2005-09-30 12:14:09			
COMDAUFIS		Comprobación de DAU	2005-09-30 15:14:00	2005-09-30 15:14:00	17 - 0:19:20	Atrasado	0.00
COGAPO		Coordinación de aforo	2005-10-01 12:14:00	2005-10-01 12:14:00	16 - 3:29:20	Atrasado	0.00

Figura 3. 22: Ejemplo de informe de actividades pendientes
 Fuente: El autor

3.12.2. Informe de Actividades Finalizadas

Para el presente informe muestra la lista (ver figura 3.23) de las actividades que han finalizado dentro de la fecha de corte indicadas, pudiendo filtrarse los resultados por Cliente, Supervisor o Tipo de Proyecto. Los usuarios identificados como Otros o Terceros, no podrán emitir este informe.

Informe de Actividades Finalizadas

Los campos con * son obligatorios

Información General

Rango de Fechas : 2003-10-01 * 2008-10-17 *

Cliente: * TODOS

Supervisor : * TODOS

Tipo Proyecto : * TODOS

Resultado ordenado por : Proceso, Actividad
 Cliente, Actividad
 Supervisor, Actividad

PDF XLS

Figura 3. 23: Datos para generar informe de actividades finalizadas
Fuente: El autor

Este informe permite analizar los resulta por tres criterios:

- i. Proceso y Actividad
- ii. Cliente y Actividad
- iii. Supervisor, Actividad

El índice de las actividades mostrado en este informe corresponde al promedio de los índices obtenido por la actividad en cada uno de los proyectos en los que aparece.

Logo de su empresa aquí
 Área de Seguimiento y control

Emisión el: 2008-10-17 20:59:01 por: ani
 Reporte de: wpg_rpt_inf_act_final.php

INFORME DE ACTIVIDADES FINALIZADAS

Actividades finalizadas entre: 2003-10-01 - 2008-10-17 Cliente: Todos los clientes
 Oficial: Todos los oficiales de cuenta Proyecto: Todos los tipos de Proyecto
 Ordenado por: Proceso, Actividad

Cód.Act	Descripción de Actividad	Núm.Ordenes	Exclusiones	%Excl.	Índice	Costo
Proceso :						
act4	acrcuatro	1	0	0	0	1
ASIORDTRA	Asignación de Orden de trabajo	3	0	0	1.66	0
ASIORDTRA	Asignación de orden de trabajo	11	0	0	70.71	21
COMDAUFIS	Comprobación de DAV	1	0	0	37	0
COOAP0	Coordinación de aforo	1	0	0	1.96	0
ELAGUIREM	Elaboración de guía remisión	1	0	0	1.96	0
ENVDOCCAR	Envío documentación CAE	3	0	0	1.71	0
ENVDOCCAR	Envío y recepción DAV	3	0	0	15.13	0
INFAP0	Informe de aforo	1	0	0	31	0
INGDATSIS	Ingreso de datos al sistema	3	0	0	1.71	0
LIQDEF	Liquidación definitiva	3	0	0	106.48	0
OBTCARSAI	Obtención carta salida	1	0	0	1.96	0
OSTCERINS	Obt. cert. inspección	3	0	0	5.46	0
OSTTARALM	Obt. tarja almacén temp.	1	0	0	1.96	0
OSTVTORNO	Obtención de visto bueno	1	0	0	1.96	0
ORSDOCPIN	Org. documentos finalizados	2	0	0	0.72	0
PLATRA	Planificación de transporte	3	0	0	0.48	0
PREDECCAR	Fres.declaración CAE	1	0	0	1.96	0
PREDOCCAR	Preparación de documentos CAE	2	0	0	1.93	0
REVFISDOC	Revisión física documentos	4	0	0	1.28	0
TRACON	Transportación de contenedor	2	0	0	0.72	0
Totales de Proceso :		51	0	0%	13.7	22
Totales Generales		51	0	0%	13.7	22

Figura 3. 24: Ejemplo de informe de actividades finalizadas

Fuente: El autor

Este informe también permite determinar las Actividades que se consideran “Exclusiones”, entendiéndose como exclusión aquella actividad cuyo índice obtenido en un proyecto, está fuera del promedio más menos una desviación estándar. El número de “Exclusiones” determina en cuántos proyectos la actividad se considera Excluida del promedio. Este dato nos permite determinar si los tiempos planificados para las actividades han sido calculados exitosamente, o si tienen que ser revisados.

3.12.3. Informe de Órdenes finalizadas

Este informe muestra una lista de los proyectos completados dentro de un rango de fechas determinado, pudiendo filtrarse los resultados por Cliente, Supervisor o Tipo de Proyecto.

Informe de Órdenes Finalizadas

Los campos con * son obligatorios

Información General

Rango de Fechas :

Cliente: TODOS

Supervisor : TODOS

Tipo Proyecto : TODOS

Resultado ordenado por : Supervisor, Cliente Cliente, Tipo Proyecto

Figura 3. 25: Datos para la generación de informe de órdenes finalizadas

Fuente: El autor

Este informe permite analizar los resultados por dos criterios:

- *Supervisor y Cliente,*
- *Cliente y Tipo de Proyecto*

En ambos casos se muestran tres fechas importantes: la fecha de inicio real, la fecha planificada de finalización del proyecto y la fecha real de fin.

Logo de la empresa aquí
 Área I.D.
 Departamento y control

Entido el: 2008-10-17 21:51:16 por: ani
 Reporte de: wpa_rpt_inf_orden_final.php

ANALISIS DE ORDENES FINALIZADAS

Ordens finalizadas entre: 2003-10-01 - 2008-10-17
 Supervisor: Todos los Supervisores
 Ordenado por: Supervisor, Cliente
 Cliente: Todos los clientes
 Proyecto: todos los tipos de Proyecto

Orden	Referencia	Descripción Orden	Fec. Orden	Fec. Planif.	Fec. Real	Índice	Excl.
SUPERVISOR : Ortiz Edgar							
CLIENTE : 10 Espol							
2005-1-0037		FINALIZACION	30-9-2005 14:57	3-11-2005 9:22	20-10-2005 15:47	0	NO
Totales de CLIENTE : 10 Espol			==> Número de Proyectos : 1	Excluidos : 0	0%	Índice : 0	
Totales de SUPERVISOR : Ortiz Edgar			==> Número de Proyectos : 1	Excluidos : 0	0%	Índice : 0	
SUPERVISOR : Pelaez Jose Luis							
CLIENTE : 10 Espol							
2005-2-0025		PPP	30-9-2005 19:25	3-10-2005 9:50	20-10-2005 15:42	9	NO
2005-2-0026		54545	30-9-2005 19:38	3-10-2005 9:50	6-10-2005 14:28	0	NO
Totales de CLIENTE : 10 Espol			==> Número de Proyectos : 2	Excluidos : 0	0%	Índice : 4.5	
Totales de SUPERVISOR : Pelaez Jose Luis			==> Número de Proyectos : 2	Excluidos : 0	0%	Índice : 4.5	
TOTALES GENERALES			==> TOTAL DE PROYECTOS : 3	EXCLUIDOS : 0	0%	INDICE : 3	

Figura 3. 26: Ejemplo del informe de órdenes finalizadas
 Fuente: El autor

Los valores de índice y exclusión, consideran los mismos criterios de los reportes anteriores.

3.12.4. Comparativo de costos en Órdenes Finalizadas

Este informe muestra una lista de los proyectos completados dentro de un rango de fechas determinado, pudiendo filtrarse los resultados por Cliente, Supervisor o Tipo de Proyecto, pero muestra un comparativo de los costos de proyecto.

Comparativo de Ordenes Finalizadas por Costos

Los campos con * son obligatorios

Información General

Rango de Fechas : 2003-10-01 * 2008-10-17 *

Cliente: TODOS

Supervisor : TODOS

Tipo Proyecto : TODOS

Resultado ordenado por : Supervisor, Cliente Cliente, Tipo Proyecto

Figura 3. 27: Datos para la generación de informe de comparativo de costos.
 Fuente: El autor

Este informe permite analizar los resultados por dos criterios:

- *Supervisor y Cliente,*
- *Cliente y Tipo de Proyecto*

En ambos casos se muestra el costo presupuestado y final (referencial) de cada proyecto, con lo cual se obtiene un porcentaje comparativo de costos que puede interpretarse según lo siguiente:

- *Un valor positivo indica que el proyecto tuvo un presupuesto de costo mayor al referencial final,*
- *Un valor negativo indica que el proyecto tuvo un presupuesto de costo inferior al referencial final,*
- *Un valor de cero o muy cercano a cero podría indicar que el presupuesto se calculó con bastante acierto.*

Emitted on: 2008-10-17 22:16:57 por: ani
Reporte de: appg_ord_final_costo.php

Orden	Referencia	Descripción Orden	Fec. Orden	Fec. Real	Índice	Costo Ref.	Costo Presup.	Porc.
ORDENES FINALIZADAS ENTRE: 2003-10-01 - 2008-10-17 CLIENTE: Todos los clientes								
SUPERVISOR: Todos los Supervisores PROYECTO: Todos los tipos de Proyecto								
Ordenado por: Supervisor, Cliente								
SUPERVISOR : Ortiz Edgar								
CLIENTE : 10 Espol								
2005-1-0037		FINALIZACION	30-9-2005 14:57	20-10-2005 15:47	0	1234567.89	1234007.09	-0.05%
Totales de CLIENTE : 10 Espol			==> Número de Proyectos : 1	Índice : 0		Costo : 1234567.89	Presup : 1234007.09	(-0.05%)
Totales de SUPERVISOR : Ortiz Edgar			==> Número de Proyectos : 1	Índice : 0		Costo : 1234567.89	Presup : 1234007.09	(-0.05%)
SUPERVISOR : Palaez Jose Luis								
CLIENTE : 10 Espol								
2005-2-0025		PPP	30-9-2005 19:25	20-10-2005 15:42	9	1234567.89	4321017.59	71.43%
2005-2-0026		54343	30-9-2005 19:28	4-10-2005 14:28	0	1234567.89	1234761.19	0.02%
Totales de CLIENTE : 10 Espol			==> Número de Proyectos : 2	Índice : 4.5		Costo : 2469135.78	Presup : 5555782.78	(55.56%)
Totales de SUPERVISOR : Palaez Jose Luis			==> Número de Proyectos : 2	Índice : 4.5		Costo : 2469135.78	Presup : 5555782.78	(55.56%)
TOTALES GENERALES			==> TOTAL DE PROYECTOS : 3	INDICE : 3		Costo : 3703703.67	Presup : 6789789.87	(45.45%)

Figura 3. 28: Resumen de costos.

Fuente: El autor

3.12.5. Reporte de Orden de Trabajo

Este reporte aparece cuando un proyecto Inicia, si el proyecto u Orden de Trabajo no ha sido iniciado, es posible consultarlo y luego iniciarlo, entonces este reporte aparece.

Google Chrome
 http://localhost:8080/proyman/seguridad/webpages/wpg_rpt_orden_email.php?tipo_orden=1&hidden_num_orden=2008-1-0019&user=ani&doc=wpg_rpt_orden_email.php

Logo de su empresa aquí
 Emitido el: 2008-10-20 01:07:22 por: ani
 Reporte de: wpg_rpt_orden_email.php

Orden de Trabajo

Cliente: 10 - Espol
 Año - Tipo - # de Orden: 2008-1-0019
 Proyecto: 110 - 10 - Imp. Consumo Puerto Conton
 Descripción de la Orden: Este es un ejemplo de ingreso de datos
 Supervisor: edgar
 Fecha: 2008-10-20 00:05
 Presupuesto: 0

Datos de la Orden:
 1 fecha n 2
 3 crew

Usuarios de Correos:
 999999999 999999999999 gag@yahoo.com.es
 usuarioNuevo usuarioNuevo usuarioNuevo usuarioNuevo

PLANIFICACIÓN DE LA ORDEN

Actividad	Duración	Inicio	Final	Es Crítica?
Asignación de Orden de trabajo	30 Minuto	2008-10-20 08:30	2008-10-20 09:00	Si-10 día(s)
DATOS: none	Texto	No requerido		
USUARIOS: anyone	Texto	Requerido		
USUARIOS: ani	Responsable,Marca Fin,Recibe Mail			
jl	Recibe Mail			
usuarioNuevo	Responsable			
Revisión física documentos	30 Minuto	2008-10-20 09:00	2008-10-20 09:30	No
DATOS: dato xtra	Texto	No requerido		
USUARIOS: ani	Responsable,Marca Fin			
Envío documentación CAR	4 Hora	2008-10-20 09:30	2008-10-20 13:30	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Ingreso de datos al sistema	30 Minuto	2008-10-20 13:30	2008-10-20 14:00	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Envío y recepción DAU	1 Hora	2008-10-20 14:00	2008-10-20 15:00	Si-1 día(s)
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Obt. cert. inspección	1 Día	2008-10-20 15:00	2008-10-21 15:00	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Liquidación definitiva	2 Hora	2008-10-21 15:00	2008-10-21 17:00	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Planificación de transporte	120 Minuto	2008-10-21 17:00	2008-10-22 09:00	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Transportación de contenedor	40 Minuto	2008-10-22 09:00	2008-10-22 09:40	No
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			
Org. documentos finalizados	10 Minuto	2008-10-22 09:40	2008-10-22 09:50	Si-0 día(s)
USUARIOS: edgar	Responsable,Marca Fin,Recibe Mail			

Figura 3. 29: Reporte de ordenes de trabajos.
 Fuente: El autor

3.12.6. Reporte de Seguimiento de Órdenes

El seguimiento de las Órdenes permite hacer un análisis del estado de cada proyecto, donde se puede apreciar la última actividad ejecutada y la siguiente actividad por ejecutarse.

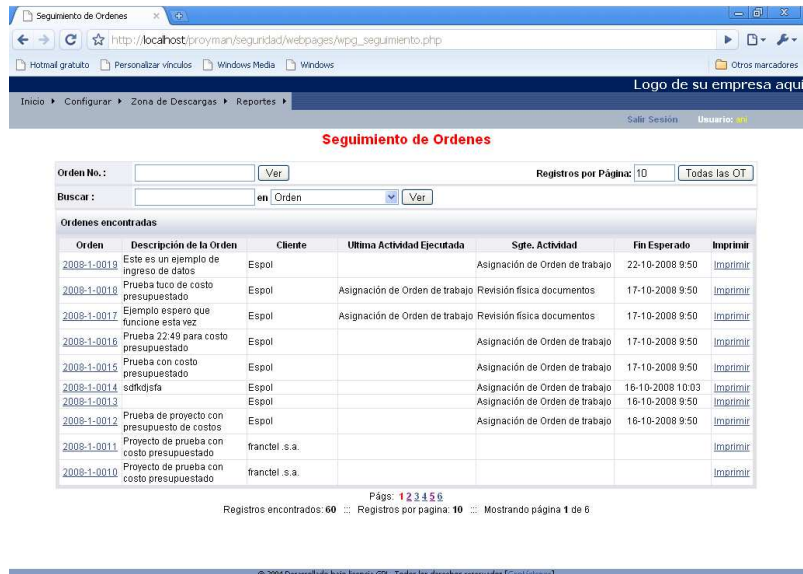


Figura 3. 30: Seguimiento de órdenes.

Fuente: El autor

Al hacer clic sobre el número de una orden, aparece el seguimiento de la Orden, según se puede apreciar en la siguiente imagen.

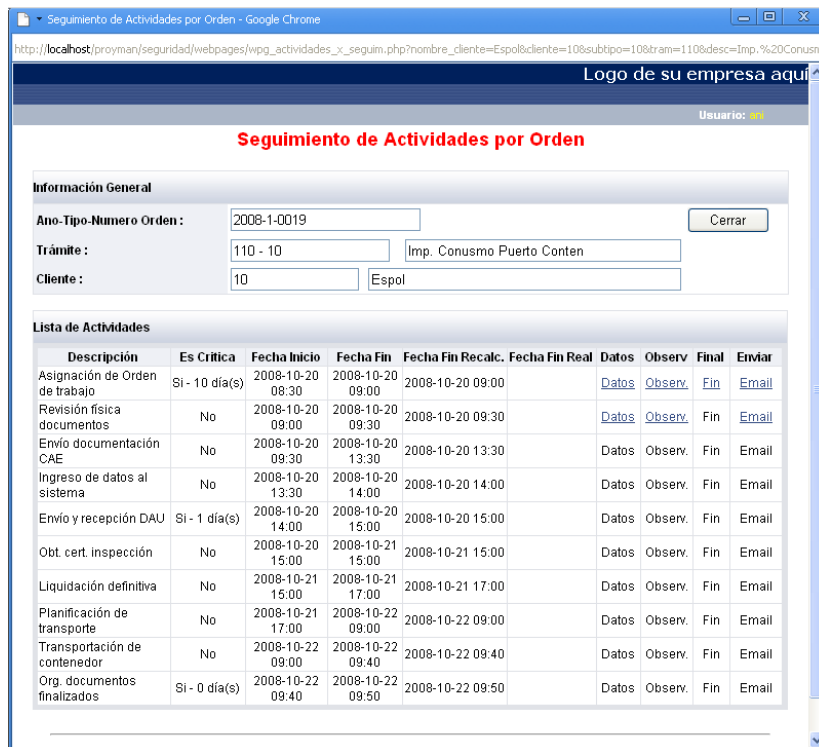


Figura 3. 31: Reporte de seguimiento de actividades por órdenes.

Fuente: El autor

Las opciones de Fin para una actividad se activan si el usuario está definido como

responsable con permiso para finalizar una actividad. La impresión del reporte de seguimiento, se lo realiza desde el enlace “Imprimir”, mostrando el siguiente informe:

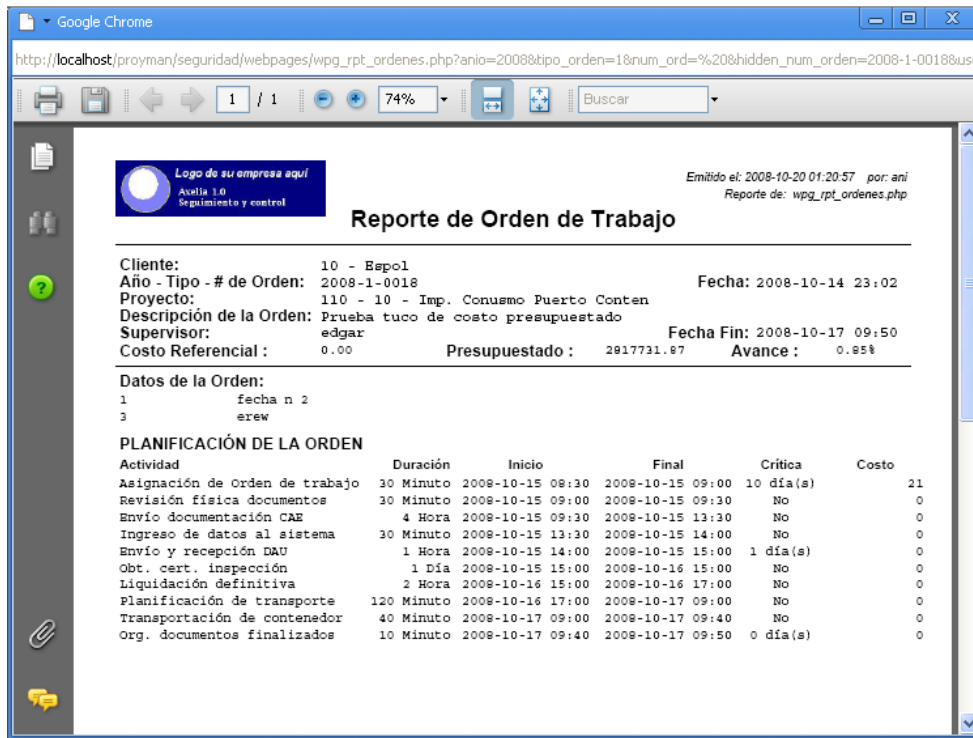


Figura 3. 32: Reporte de seguimiento de ordenes de trabajos.

Fuente: El autor

El porcentaje de Avance mostrado en este reporte indica que se han completado el 0.95% del tiempo planificado en las actividades.

Capítulo 4: Conclusiones y Recomendaciones

4.1. Conclusiones

- ✓ La flexibilidad del sistema permite adaptarlo a diferentes tipos de empresas, y un grupo importante de ellas son las empresas estatales, y empresas que están implantando certificaciones de calidad en el área operativa/administrativa.
- ✓ Es posible adaptarlo para que permita el Manejo de estadísticas de indicadores de gestión, que permitan a los directores y/o auditores llevar un registro sobre la evolución de estas actividades y planificar su revisión o mejorar su planificación.
- ✓ Un aspecto interesante que no fue considerado, es aprovechar los avisos automáticos de mensajes SMS (celular), notificando la ocurrencia de eventos.

4.2. Recomendaciones

- ✓ Implementar el presente sistema en una empresa, pero previamente debe realizar una revisión de los procesos que desean monitorearse, de manera que puedan ser definidos en el sistema y aprovechar su uso para mejorar la gestión de estos procesos.
- ✓ Al ser un software de código abierto, puede mejorarse poniéndolo disponible para la comunidad, además puede aprovecharse la herramienta para explotar un posible negocio de servicios, impulsando el uso del mismo a través de algún modelo de marketing por internet.

ANEXO A: GUÍA DE INSTALACIÓN

Este sistema está basado en la plataforma Apache MySQL y PHP, sin embargo por sus características, es posible instalarlo para que funcione sobre Windows o Linux, según sea la preferencia del usuario. Para la instalación de la base de datos, refiérase al manual de Mysql y luego debe crear la base de datos con el nombre **SGMPROTELProject**.

I.1. Instalación en Windows

Para la instalación del sistema sobre el Sistema Operativo Windows, se recomienda utilizar la plataforma integrada AppServ, que puede descargarla gratuitamente en la siguiente dirección <http://www.appservnetwork.com>. Finalmente siga los pasos de la instalación y luego copie la carpeta del contenido del proyecto dentro de la ruta C:\AppServ\www y la aplicación estará lista para funcionar. Es importante que configure la habilitación de las variables globales de PHP, para ello cambie la configuración del archivo php.ini y coloque **register_globals = On**.

I.2. Instalación en Linux

Para la instalación del sistema sobre Linux, siga los pasos de instalación para LAMP, Linux, Apache, MySQL y PHP, que puede obtenerlos gratuitamente en la web. Finalmente siga los pasos de la instalación y luego copie la carpeta del contenido del proyecto dentro de la ruta raíz de Apache y la aplicación estará lista para funcionar. Es importante que configure la habilitación de las variables globales de PHP, para ello cambie la configuración del archivo php.ini y coloque **register_globals = On**.

I.3. Configuración Inicial del Software

El sistema viene preparado con la configuración inicial que se requiere, simplemente ejecutando el script llamado carga_inicial.sql, pero primero debe asegurarse de haber creado la base de datos.

ANEXO II MANUAL DE USUARIO

Este documento supone que Usted tiene experiencia en el uso de un computador y de un navegador Web. En este documento encontrará algunos símbolos, tales como:

<Botón>

Los botones están encerrados entre los símbolos <>.

Enlace

También conocido como link, usualmente utilizado para:

- a. ejecutar una instrucción especial con respecto de los datos que acompaña, como anular, borrar, imprimir, etc.
- b. Saltar a otra página de la aplicación.
- c. Descargar un archivo.
- d. Cerrar la sesión de Usuario.



Icono de búsqueda, utilizado para buscar valores referentes al campo que acompañan.

Casilleros


Los nombres de los casilleros de datos o etiquetas aparecerán en negritas.


II.1. Plataforma de ejecución

El sistema ha sido desarrollado bajo ambiente Web, lo que significa que para utilizarlo, necesitará tener instalado en su computador un Navegador Web, entre los recomendados, están: Internet Explorer 6.0 o superior, Mozilla Firefox y Google Chrome

Algunas propiedades de los navegadores han sido bloqueadas por cuestiones de seguridad, por tanto, no siempre podrá hacer uso de la tecla Backspace para regresar a la última página visitada.

II.2. La plataforma de usuario incluye

1. **La barra de menú.**- contiene las opciones del sistema que el Administrador ha activado para cada usuario, no podrá hacer otra cosa que no sea lo definido en el menú.
2. **La barra de información.**-contiene tres partes:
 - a. Información del tipo de cambio: la cual contiene:
 - i. El tipo de cambio del euro vigente al momento del ingreso y
 - ii. La fecha en que se registró ese tipo de cambio.
 - b. Información del usuario: la cual contiene:
 - i. El usuario que ha iniciado sesión y
 - ii. Un enlace para terminar el programa, identificado como Cierre de Sesión.
 - c. Información de la tienda, que identifica la tienda en la que solicitó el ingreso al sistema.
3. **Área de trabajo, ó área de datos.**- donde se realiza toda la interacción con el sistema, ingreso de datos, cotizaciones, políticas de descuentos, reportes, etc. Existen 3 tipos de pantallas:
 - a. Pantallas de Ingreso/Mantenimiento de datos, sirven para registrar o modificar datos, siempre encontrará algunos casilleros acompañados de un asterisco (*) de color **rojo**, lo cual indica que este casillero no puede estar vacío.
 - b. Pantallas de Transacciones, permiten seleccionar opciones para búsqueda de datos y luego poder realizar alguna operación con ellos.
 - c. Pantallas de Reportes, contienen filtros de búsqueda, tales como fechas, rangos de valores, valores específicos, etc., con la finalidad de mostrar un conjunto de resultados que cumplan con los criterios de búsqueda solicitados.
 - d. Usualmente dentro del área de datos, existen algunos que tienen un icono de búsqueda de datos, representado por éste símbolo: , el cual al ser presionado, muestra una pantalla azul con varias opciones de búsqueda, puede utilizar el casillero de la parte superior para efectuar una búsqueda introduciendo unas pocas letras del elemento solicitado.

Otra forma de buscar es: presione el mismo botón , también puede utilizar el botón consultar para traer todos los elementos y luego buscar el que necesita en el listado obtenido, por ejemplo, fíjese en la siguiente figura.

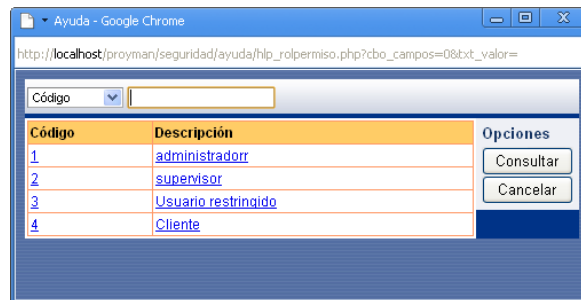


Figura II. 1: Ejemplo de pantalla de ayuda y/o consulta

II.3. Cómo iniciar el programa

Debido a que la aplicación está instalada sobre un ambiente Web, es necesario contar con una dirección Web (URL) para poder acceder a la aplicación. Para iniciar la aplicación debe ingresar la dirección del servidor Web que contiene la aplicación, que al momento de emitir este manual es: <http://localhost/proyman/index.php>. Luego podrá ver la pantalla de acceso, como se muestra en la siguiente ilustración:

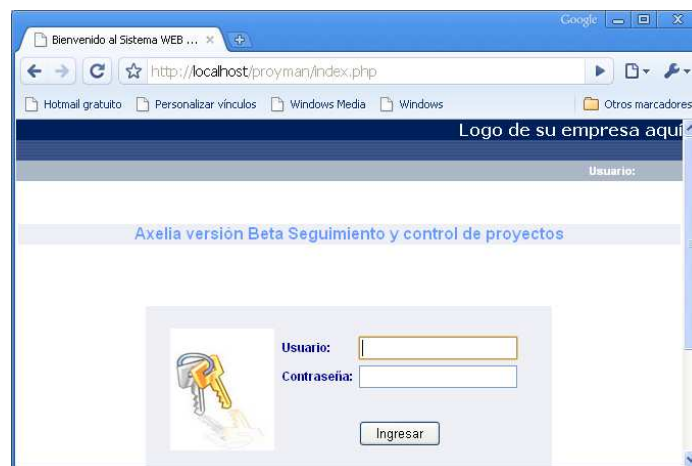


Figura II. 2: Pantalla de inicio de sesión

Escriba el usuario y contraseña que le asignó el administrador del sistema, si desea ingresar como administrador puede utilizar admin, admin, para definir los demás usuarios del sistema.

II.4. Terminar de utilizar el sistema

Para terminar una sesión (el uso del sistema), simplemente presione el enlace identificado como Cierre de Sesión, que se encuentra en la barra de información, junto al código del Usuario, en la parte superior derecha de la pantalla de la aplicación.

II.5. Seguridad

Esta opción debe estar disponible para los administradores del sistema. A través de las opciones incluidas en este menú se puede asignar o quitar acceso a las demás opciones del menú, es por ello que debe ser administrada con mucho cuidado. Revisaremos cada una de las opciones incluidas aquí:

II.6. Catálogo de tablas

Esta opción forma parte medular de la configuración del sistema, en el catálogo de tablas se encuentran distribuidos los parámetros principales que se utilizan en el sistema, que están clasificadas en.

- *Tipos de Datos*
- *Tipos de Observación*
- *Tipos de Programas*
- *Tipos de Proyectos*
- *Parámetros de correo electrónico*
- *Procesos para Actividades*

II.7. Usuarios del sistema

Esta opción le permitirá ingresar nuevos usuarios al sistema o modificar los datos de los usuarios ya registrados.

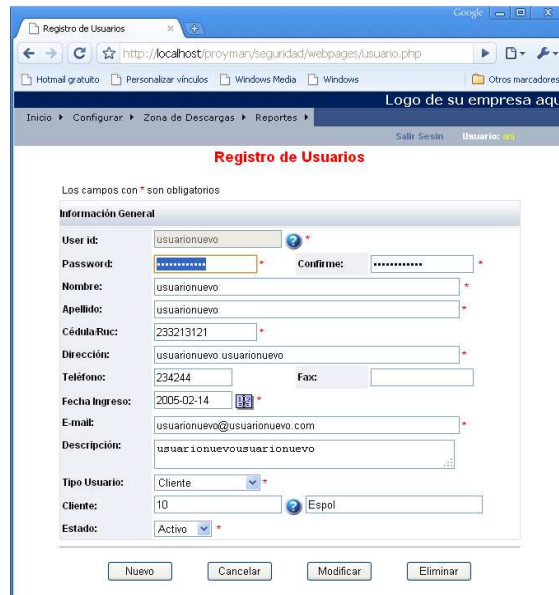


Figura II. 3: Definición de Usuarios del sistema

II.8. Roles

El sistema de asignación de permisos a los usuarios está basado en Roles, un Rol puede ser asignado a un grupo de opciones de menú, luego se asigna un Rol a uno o más usuarios para que éstos hereden todas las opciones de menú asignadas al Rol.



Figura II. 4: Definición de Roles

II.9. Asignación de Roles a usuarios

Esta opción le permite asignar un Rol a algún usuario, de manera que el usuario adquiera todos los permisos de acceso que han sido definidos para el Rol.



Figura II. 5: Roles de Usuario

II.10. Permisos de Roles

Permite asignar opciones del menú a un Rol específico, de esa manera cualquier usuario al que se le asigne el Rol, podrá tener acceso a dichas opciones del menú.

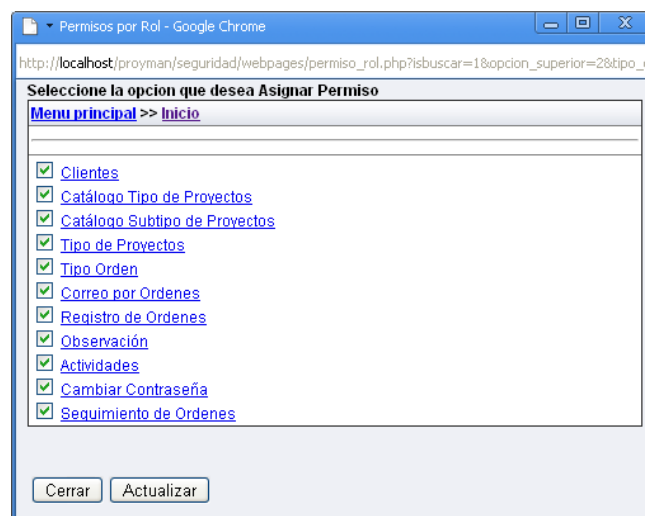


Figura II. 6: Ejemplo de selección de opciones del menú para un Rol

II.11. Menú del sistema

El menú es posible configurarlo y modificarlo para que muestre las descripciones que al usuario le agraden, también es posible cambiar la ubicación de las opciones en el menú desplegable y reorganizarlos para que aparezcan en una secuencia diferente, para hacerlo, debemos seleccionar el menú que deseamos editar y cambiar sus datos, tal como se muestra en la siguiente pantalla.

Figura II. 7: Edición de opciones del menú de la aplicación

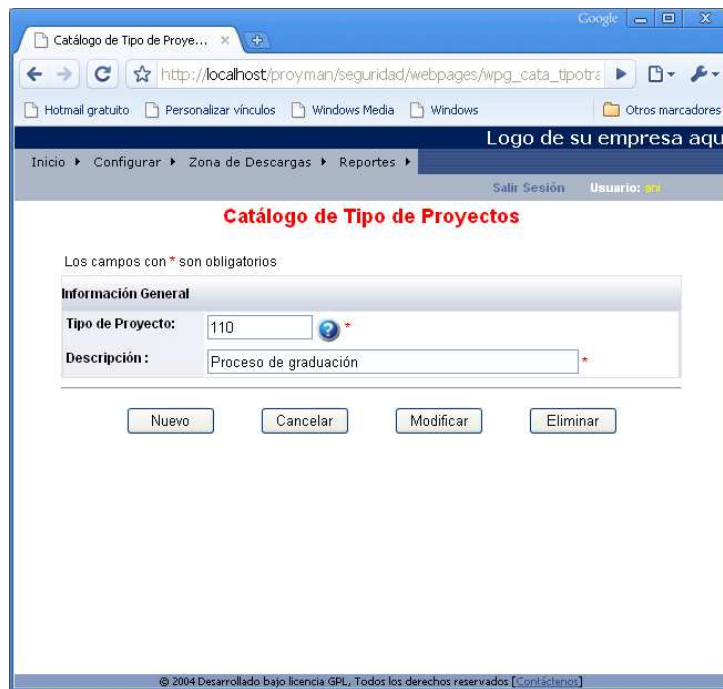
II.12. Registro de Clientes

Los clientes se ingresan mediante esta pantalla, debe completar la información y guardar el registro para que los cambios tengan efecto.

Figura II. 8: Registro de Clientes

II.13. Catálogo de tipo de Proyectos

Los tipos de proyectos definidos en el sistema se configuran con esta opción, tal como se muestra en la pantalla a continuación. Únicamente se requiere definir un código y una descripción para registrar los tipos de Proyectos que desea administrar en el sistema. Puede añadir cualquier tipo de proyecto que desee, se recomienda contar con la documentación de procesos respectiva para garantizar que el sistema se configure y funcione como se espera.



The screenshot shows a web browser window with the title 'Catálogo de Tipo de Proye...'. The address bar shows 'http://localhost/proyman/seguridad/webpages/wpg_cata_tipotr...'. The page content includes a navigation menu with 'Inicio', 'Configurar', 'Zona de Descargas', and 'Reportes'. A user session is active, showing 'Salir Sesión' and 'Usuario: an'. The main heading is 'Catálogo de Tipo de Proyectos'. Below it, a note states 'Los campos con * son obligatorios'. The 'Información General' section contains two fields: 'Tipo de Proyecto:' with the value '110' and a help icon, and 'Descripción:' with the value 'Proceso de graduación'. At the bottom, there are four buttons: 'Nuevo', 'Cancelar', 'Modificar', and 'Eliminar'. A footer at the bottom of the page reads '© 2004 Desarrollado bajo licencia GPL. Todos los derechos reservados [Contactanos]'.

Figura II. 9: Catálogo de tipo de Proyectos

II.14. Subtipo de Proyectos

Los proyectos se clasifican en tipos y subtipos, la creación de los subtipos es independiente de los tipos de proyectos, sin embargo se relacionan cuando se crea una plantilla de proyecto. Para la creación de Subtipo de Proyectos, ingrese los datos en la siguiente pantalla:

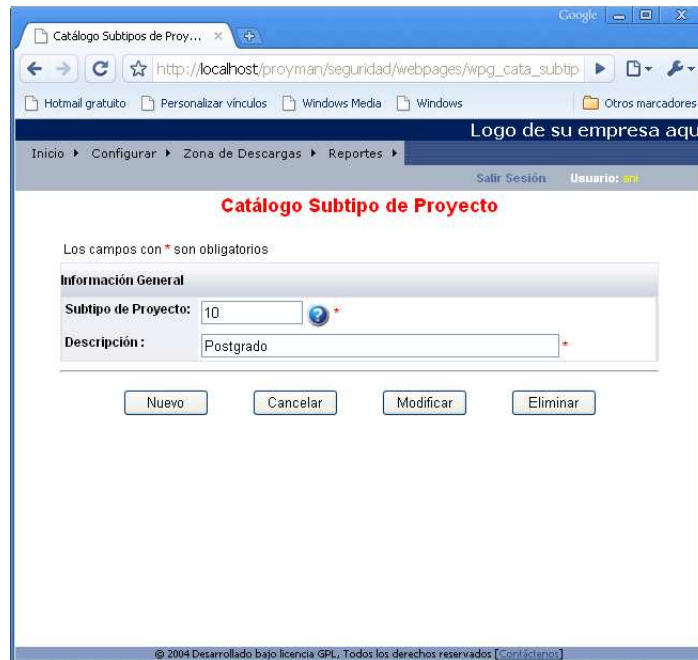


Figura II. 10: Catálogo de Subtipo de Proyectos

II.15. Plantillas

Las plantillas, le permiten predefinir los tipos y subtipos de proyectos, con sus actividades, prerequisites, responsables y datos, así como la secuencia de ejecución de las actividades y el tiempo asignado a cada una de ellas, para registrarlos, debe llenar la siguiente pantalla:

Tipos de Proyectos

Los campos con * son obligatorios

Información General

Tipo de Proyecto: 110

Subtipo Proyecto: 10

Cliente: 10

Supervisor : edgar

Referencia: Imp. Consumo Puerto Conten

Descripción: Importación a Consumo Puerto Guayaquil Contenedores

Guardar Cancelar Modificar Eliminar

PDF Plantilla Datos Importar

© 2004 Desarrollado bajo licencia GPL. Todos los derechos reservados [Contactarnos]

Figura II. 11: Tipos de proyectos

II.16. Definición cuentas correo

Es posible asociar usuarios de correo a una plantilla, para ello presione le botón <Asignar Correos> y marque los usuarios que desea asociar, tal como se muestra en la siguiente ilustración.



Figura II. 12: Selección de usuarios en la plantilla de proyectos

II.17. Definición de las actividades en una plantilla de proyecto

Las actividades se definen presionando el botón <Plantilla>, en esta pantalla podrá ingresar las actividades predefinidas en el sistema y asignarle datos, usuarios y definir el orden de ejecución de ellas. Para completar esta información, debe llenar una plantilla como se muestra en la siguiente ilustración.

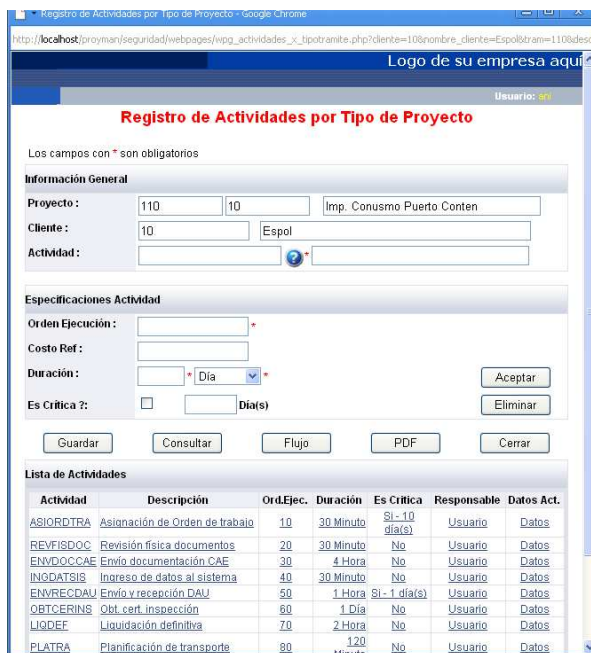


Figura II. 13: Actividades del tipo de proyecto

Presione el botón <PDF> para obtener un informe de la plantilla, donde se muestra el resumen de las actividades, con sus datos, predecesores y tiempos de ejecución asignados, así como también se muestra el orden de ejecución de las actividades. El botón <Flujo>, le permite definir la matriz de predecesores del sistema.

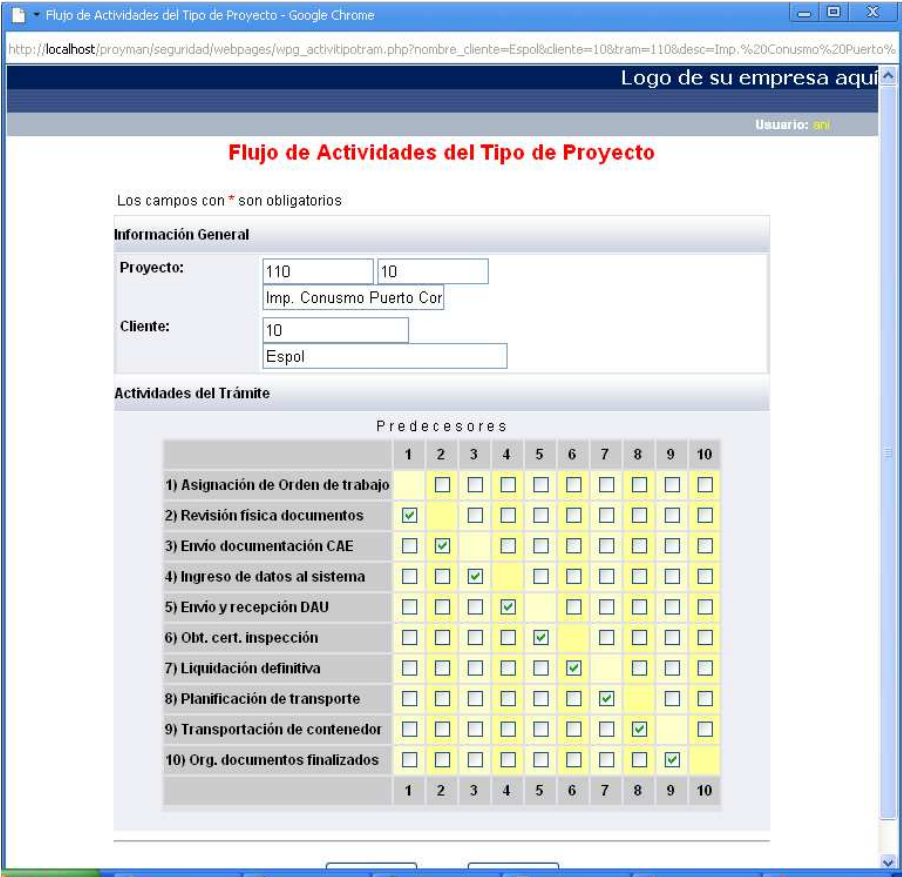


Figura II. 14: Definición del Flujo de actividades para la plantilla

II.18. Registro de Usuarios por actividad

En la plantilla, las actividades pueden tener asociados usuarios, para ello, presione el enlace Usuarios en la columna "Responsables", de la lista de actividades definidas para la plantilla, la siguiente ilustración muestra un ejemplo de ello:

Registro de Usuarios por Actividad - Google Chrome

http://localhost/proyman/seguridad/webpages/wpg_usuarioresponsable.php?nombre_cliente=Espol&cliente=10&subtipo=

Logo de su empresa aquí

Usuario: **adm**

Registro de Usuarios por Actividad

Los campos con * son obligatorios

Información General

Proyecto : 110 10 Imp. Conusmo Puerto Conten

Cliente : 10 Espol

Actividad : ASIORDTRA Asignación de Orden de trabajo

Asignación de Usuarios

Línea	Usuarios	Responsable	Marcar Fin	Enviar Email
1	Ana María Vivar	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	Edgar Ortiz	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	gggggggggg gggggggggggg	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	Jannett Elizaldes	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	j l pr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	jose f pelaez	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	jose pelaez	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
8	josel josel	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	joseuser joseuser	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	userjose userjose userjose	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	usuarionuevo usuarionuevo	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Guardar Cerrar

Figura II. 15: Usuarios definidos por actividad

II.19. Copiar plantilla

Una funcionalidad muy importante es la de Copiar Plantilla. Ocurre que puede tener plantillas que pertenezcan a tipos de proyectos diferentes, sin embargo la mayoría de las actividades definidas en una plantilla son muy parecidas a la nueva, por lo que es más sencillo hacer una copia y modificarla, que volver a ingresar la plantilla completa. Esto se logra mediante el botón <Importar>, en la pantalla de plantillas.

Copiar Plantilla de Tipo de Proyecto - Google Chrome

http://localhost/proyman/seguridad/webpages/wpg_importar_tram.php?cliente_dest=10&tram_dest=110&subtipo_dest=10

Copiar Plantilla de Tipo de Proyecto

Los campos con * son obligatorios

Datos de Origen a Copiar

Tipo de Proyecto: ?*

Imp a consumo Negocorp

Subtipo Proyecto: ?*

Postgrado

Cliente: ?*

Espol

Lista de Actividades

Actividad	Referencia	Orden Ejec.	Costo Ref.	Duración
ASIOROTRA	Asignación de Orden de trabajo	10	0.0000	10 Minuto
OBTVOBNO	Obtención de visto bueno	20	0.0000	1 Día
OBTTCERINS	Obt. cert. inspección	30	0.0000	2 Día
REVFISDOC	Revisión física documentos	40	0.0000	30 Día
INGDATSIS	Ingreso de datos al sistema	50	0.0000	30 Minuto
ENVRECCDAU	Envío y recepción DAU	60	0.0000	30 Minuto
PREDOCCAE	Preparación de documentos CAE	70	0.0000	30 Minuto
ENVOCCAE	Envío documentación CAE	80	0.0000	30 Minuto
OBTTCARSAL	Obtención carta salida	90	0.0000	3 Hora
ELAGUIREM	Elaboración de guía remisión	110	0.0000	8 Hora
PREDECCAE	Pres.declaración CAE	120	0.0000	15 Minuto
OBTTCARALM	Obt. tarjeta almacén temp.	130	0.0000	30 Día
COMDAUFIS	Comprobación de DAU	140	0.0000	3 Hora
COOAF0	Coordinación de aforo	150	0.0000	1 Día
INF0AFO	Informe de aforo	160	0.0000	5 Hora
LIQDEF	Liquidación definitiva	170	0.0000	30 Minuto
PLATRA	Planificación de transporte	190	0.0000	25 Minuto

Figura II. 16: Pantalla para importar plantillas

II.20. Registro de Órdenes

Cuando tiene que iniciar un proyecto, se debe iniciar una orden, las órdenes sirven para asignar responsables y tareas en un tipo de proyecto determinado. Las órdenes heredan las definiciones (plantillas) del tipo de proyecto seleccionado, incluyendo los responsables de cada actividad, los datos definidos para las actividades y el flujo de ejecución de las actividades. Pero además es importante completar el costo Presupuestado, para poder hacer una comparación final con el costo referencial del proyecto. La siguiente ilustración muestra como luce una Orden Iniciada para un tipo de proyecto dado.

Registro de Ordenes

Los campos con * son obligatorios

Información General

Año - Tipo - Num. de Orden: 2008 - 1 - 17

Cliente: 10
Espol

Tipo de Proyecto: 110 - 10
Imp. Conusmo Puerto Conten

Supervisor: edgar

Descripción: Ejemplo espero que funcione esta vez

Costo Presupuestado: 12345678.99

Costo Referencial: 0.00

Estado: **INICIADO**

Documentos de Referencia

Insertar Línea Eliminar Línea

Elim.	Número de Referencia	Descripción	Fecha
<input type="checkbox"/>			

Nuevo Cancelar Modificar Eliminar

PDF Asociar Datos Actividades

© 2004 Desarrollado bajo licencia GPL. Todos los derechos reservados [Confidencial]

Figura II. 17: Registro / Inicio de Flujo de Trabajo de un proyecto

II.21. Asociar Órdenes de Clientes

Los Clientes pueden tener documentos que desean asociar a un proyecto, esta versión contempla la posibilidad de guardar una referencia que permita relacionar uno o más proyectos, para ello se debe completar la siguiente pantalla.

Asociar Ordenes por Cliente - Google Chrome

http://localhost/proyman/seguridad/webpages/wpg_asociar_ordenes.php?cliente=10&nombre_cliente=Espol&tipo_ord...

Logo de su empresa aquí

Usuario: ●●

Asociar Ordenes por Referencia del Cliente

Los campos con * son obligatorios

Información General

Año-Tipo- Numero Orden : 2008-1-0008

Cliente : 10
Espol

Búsqueda de Ordenes

Número de referencia : OC-traz01 *

Solo las OT sin asociar Que sean del mismo tipo
 Todas las OT disponibles

Ordenes encontradas

Orden	Descripción	Apertura	Asociar
2008-1-0009	ejemplo2w	2008-10-06 22:35	<input checked="" type="checkbox"/>

Figura II. 18: Asociar Proyectos utilizando números de referencia

ANEXO III GLOSARIO DE TÉRMINOS

AESOFT: Asociación Ecuatoriana de Software

EXCLUSION: Aquella actividad cuyo índice obtenido en un proyecto, está fuera del promedio más menos una desviación estándar.

GPL: Generic Public License, Texto de la Licencia Pública General en <http://www.gnu.org/licenses/gpl.txt>

LAMP: Linux Apache MySQL Php

Open Source: Código Abierto

ANEXO IV DICCIONARIO DE DATOS

CAC_TABLA								
Columna	Tipo Dato	de	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	AutoInc
Codigo	NUMERIC(2)						Código de la tabla	
Nombre	VARCHAR(50)						Nombre de la tabla	

CATEGORIA_DESCARGA								
Columna	Tipo Dato	de	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	AutoInc
codigo	INTEGER(4)		PK	NN			Código de objeto	
nombre	VARCHAR(100)						Nombre de objeto	
descripcion	VARCHAR(250)						Descripción de objeto para descarga	
Nombre de Índice			Tipo de Índice			Columnas		
PRIMARY			PRIMARY			Codigo		

DESCARGA								
Columna	Tipo Dato	de	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	AutoInc
codigo	INTEGER(4)		PK	NN			Código de archivo de descarga	
categoria	INTEGER(4)						Categoría del archivo de descarga	
nombre	VARCHAR(50)						Nombre del archivo	

descripcion	VARCHAR(250)		NN			Descripción del contenido del archivo	
nom_archivo	VARCHAR(250)					Nombre Físico del archivo en disco	
Nombre de Índice		Tipo de Índice			Columnas		
PRIMARY		PRIMARY			codigo		

dias_feriados							
Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	AutoInc
codigo	BIGINT(18)	PK	NN			Código de registro del día feriado	
dia_feriado	DATETIME		NN			Fecha del día feriado	
pais	CHAR(10)		NN			Pais donde aplica el día feriado	
estado	CHAR(10)		NN			Estado o provincia donde aplica el feriado	
ciudad	CHAR(10)		NN			Ciudad donde aplica el feriado	
Nombre de Índice		Tipo de Índice			Columnas		
PRIMARY		PRIMARY			Codigo		

ESTADO							
Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	AutoInc
codigo	NUMERIC(2)	PK	NN			Código de estado de los registros	
descripcion	VARCHAR(25)					Descripción del estado	

Nombre de Índice	Tipo de Índice	Columnas
PRIMARY	PRIMARY	codigo

ESTADO_TABLA							
Columna	Tipo de Dato	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	AutoInc
tabla	NUMERIC(2)					Código del Estado de las Tablas	
estado	NUMERIC(2)					Descripción General del Estado de las tablas	

jornada_laboral							
Configuración de la Jornada Laboral (JL) para el sistema, solo se puede definir un registro en todo el sistema							
Columna	Tipo de Dato	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	BIGINT(18)	PK	NN			Código del registro	
hora_ini	TINYINT(3)		NN			Hora inicial de la JL	
minuto_ini	TINYINT(3)		NN			Minutos iniciales de la JL	
hora_fin	TINYINT(3)		NN			Hora final de la JL	
minuto_fin	TINYINT(3)		NN			Minutos finales de la JL	
Nombre de Índice	Tipo de Índice		Columnas				
PRIMARY	PRIMARY		codigo				

OPCION_SISTEMA							
Opciones del menú del sistema							
Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
codigo	INTEGER(4)	PK	NN			Código del registro	
descripcion	VARCHAR(250)					Descripción del menú	
alternativo	VARCHAR(250)		NN			Nombre alternativo	
nombre_fisico	VARCHAR(250)		NN			Nobre fisico del archivo asociado al menú	
parametro	VARCHAR(10)		NN			Parámetro que recibe el archivo o programa	
fecha_creacion	DATETIME				(getdate())	Fecha de creación de la opción	
opcion_superior	NUMERIC(4)		NN			Código de menú superior	
ventana_nueva	CHAR(10)					Bandera para indicar si la opción se carga en una ventana nueva	
orden_presentacion	NUMERIC(2)					Orden de presentación del menú	
tipo_opcion	NUMERIC(2)		NN			Tipo de opción (menú, submenú)	
posicion	NUMERIC(2)					Posición de ubicación del menú, default top	

estado	NUMERIC(2)					Código estado registro	del del	
Nombre de Índice		Tipo de Índice			Columnas			
PRIMARY		PRIMARY			codigo			

PARÁMETRO								
Parámetros Generales del Sisitema								
Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc	
codigo	INTEGER(4)	PK	NN			Código registro	del	
clase	VARCHAR(50)					Clase de parámetro	de	
descripcion	VARCHAR(250)					Descripción del parámetro	del	
nombre	VARCHAR(50)					Nombre del parámetro	del	
valor	VARCHAR(100)					Valor asignado		
tipo_dato	CHAR(1)					Tipo de valor/dato (texto, número)		
estado	NUMERIC(2)					Código estado registro	del del	
Nombre de Índice		Tipo de Índice			Columnas			
PRIMARY		PRIMARY			codigo			

PERMISO								
Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc	
numero	INTEGER(Numero del	del	

	4)					registro	
rol	NUMERIC(2)					Código del Rol	
opcion_sistema	NUMERIC(4)					Opción del menú asignada al Rol	

POSICION_MENU

Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
codigo	NUMERIC(2)					Código del item de menú	
descripcion	VARCHAR(10)					Descripción del Item del Menú	

ROL

Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
codigo	INTEGER(4)	PK	NN			Código del Rol	
descripcion	VARCHAR(100)					Descripción del Rol	
estado	NUMERIC(2)					Estado (Activo, Inactivo, Eliminado)	

Nombre de Índice	Tipo de Índice	Columnas
PRIMARY	PRIMARY	codigo

ROL_USUARIO

Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
rol	INTEGER(4)	PK	NN			Código de rol	

user_id	VARCHAR(15)	PK	NN			Usuario	
fecha_asignacion	CHAR(10)					Fecha de asignación del rol	
fecha_caduca	CHAR(10)		NN			Fecha de caducidad	
nunca_caduca	VARCHAR(1)		NN			Bandera para indicar que la asignación no caduca	
Nombre de Índice		Tipo de Índice		Columnas			
PRIMARY		PRIMARY		rol user_id			

tb_cat_subtipotramite

Columna	Tipo de Dato	Clave de Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_subtipo tramite	VARCHAR(10)	PK	NN			Código de Categoría de Subtipo de Proyecto	
descripcion	VARCHAR(50)		NN			Descripción de Subcategoría	
Nombre de Índice		Tipo de Índice		Columnas			
PRIMARY		PRIMARY		cod_subtipotramite			

tb_cat_tipotramite

Columna	Tipo de Dato	Clave de Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_tipotramite	VARCHAR(10)	PK	NN			Código de la categoría del tipo de Proyecto	
descripcion	VARCHAR		NN			Descripción de	

(50)	Categoría	
Nombre de Índice	Tipo de Índice	Columnas
PRIMARY	PRIMARY	cod_tipotramite

tb_catalogo_tablas

Catálogo de tablas del sistema, es posible configurar varias tablas con sus elementos en el sistema.

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	BIGINT(18)	PK	NN			Código del registro	
descripcion	VARCHAR(50)					Descripción de la tabla	
tipo_objeto	NUMERIC(9)		NN			Tipo de objeto (registro, tabla)	
pertenece_a	NUMERIC(18)		NN			Código de la tabla a la que pertenece el registro	
valor	VARCHAR(255)		NN			Valor asignado al registro de la tabla	
Nombre de Índice	Tipo de Índice		Columnas				
PRIMARY	PRIMARY		codigo				

tb_wf_activ_tipo_tram

Definición de Actividades por Tipo de Proyecto

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_tramite	VARCHAR(10)	PK	NN			Tipo de Proyecto	
subtipotramite	VARCHAR(10)	PK	NN			Subtipo de Proyecto	
cod_cliente	VARCHAR(15)	PK	NN			Cliente	

cod_actividad	VARCHAR(15)	PK	NN	Actividad
orden_ejecucion	NUMERIC(18)		NN	Orden de ejecución de la Actividad
costo_ref	DECIMAL(12, 4)		NN	Costo referencial
programa_relac	VARCHAR(255)		NN	Programa asociado con la actividad, para compatibilidad con sistemas externos
cadena_comandos	VARCHAR(255)		NN	Cadena de comandos a ejecutar, para compatibilidad con sistemas externos
tipo_programa	NUMERIC(18)		NN	Tipo de programa a ejecutar, para compatibilidad con programas externos
duracion	INTEGER		NN (0)	Duración de la Actividad
formato	CHAR(1)		NN	Código de duración de la actividad (días, horas, minutos, semanas)
es_critica	CHAR(1)		NN	Bandera para indicar si la actividad es crítica
dias_maximo	INTEGER		NN (0)	Tiempo máximo en días que deben transcurrir para

		declarar como crítica la actividad
Nombre de Índice	Tipo de Índice	Columnas
PRIMARY	PRIMARY	tipo_tramite subtipotramite cod_cliente cod_actividad

tb_wf_actividades_fmt

Plantilla de Actividades Generales

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_actividad	VARCHAR(15)	PK	NN			Código de Actividad	
nombre_corto	VARCHAR(30)		NN			Nombre corto de referencia	
nombre_actividad	VARCHAR(250)		NN			Descripción larga de la Actividad	
costo_ref	DECIMAL(12, 4)		NN			Costo referencial	
duracion	INTEGER		NN			Duración estimada de la Actividad	
formato	CHAR(1)		NN			Formato de duración (días, semanas, horas, minutos)	
proceso	NUMERIC(4)		NN			Nombre de proceso externo, para compatibilidad con otros sistemas	
Nombre de Índice	Tipo de Índice	Columnas					
PRIMARY	PRIMARY	cod_actividad					

tb_wf_actividades_ordenes								
Columna	Tipo Dato	de	Clave Primaria	Nul	Band	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)		PK	NN			Código del tipo de Orden	
numero_orden	VARCHAR(15)		PK	NN			Número de la Orden	
cod_actividad	VARCHAR(15)		PK	NN			Código de la Actividad	
orden_ejecucion	NUMERIC(18)			NN			Orden de ejecución de la Actividad	
fecha_inicio	DATETIME			NN			Fecha de inicio de la Actividad	
fecha_fin	DATETIME			NN			Fecha de culminación de la Actividad	
fecha_fin_recalc	DATETIME			NN			Fecha de Inicio Recalculada	
costo_final	DECIMAL(15, 4)			NN		(0)	Costo Final de la Actividad	
estado	VARCHAR(15)			NN			Estado de la Actividad	
nombre_corto	VARCHAR(30)			NN			Nombre corto de la Actividad	
nombre_actividad	VARCHAR(250)			NN			Descripción de la Actividad	
duracion	INTEGER			NN			Tiempo de Duración de la Actividad, para calcular la fecha final	
formato	CHAR(1)			NN			Formato de Tiempo de Duración (Días, Horas, Minutos)	
fecha_fin_r	DATETIME			NN			Fecha Final	

Real	E			Real de la Actividad
proceso	NUMERIC (4)	NN		Código del Proceso asociado de la Actividad
es_critica	CHAR(1)	NN		Marca para indicar si la Actividad es crítica (S,N)
dias_maximo	INTEGER	NN		Días que deben transcurrir para que la actividad sea crítica, solo si el campo es_critica = S
rua	VARCHAR (15)	NN		Código del Registro Unico de Auditoría
diferencia_dias	INTEGER	NN		Días de diferencia entre fecha final real y estimada
diferencia_horas	INTEGER	NN		Horas de diferencia entre fecha final real y estimada
diferencia_minutos	INTEGER	NN		Minutos de diferencia entre fecha final real y estimada
indice	NUMERIC (6, 2)	NN	(0)	Número de Índice, utilizado para ordenar las Actividades
predefinida	CHAR(1)	NN		Marca para determinar si la actividad viene predefinida de la plantilla

fecha_inicio_real	DATE TIME	NN	Fecha Real de Inicio de la Actividad
Nombre de Índice	Tipo de Índice	Columnas	
PRIMARY	PRIMARY	tipo_orden numero_orden cod_actividad	

tb_wf_clientes							
Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_cliente	VARCHAR R(15)	PK	NN			Código del Cliente	
nombre_razonsocial	VARCHAR R(50)				('')	Nombre o Razón Social	
apellido	VARCHAR R(50)		NN		(null)	Apellidos, en caso de Persona Natural	
num_identificacion	DECIMAL (13)				(0)	Número de Identificación	
tipo_identifica	CHAR(1)				('0')	Tipo de Identificación (Cédula, Pasaporte, Ruc)	
tipo_persona	CHAR(1)				('0')	Indicador de Persona Natural, Persona Jurídica	
direccion	VARCHAR R(100)				('')	Dirección domicilio del Cliente	
telefono	VARCHAR R(25)		NN		(null)	Número de teléfono del Cliente	
fax	VARCHAR R(25)		NN		(null)	Número de fax del Cliente	

fecha_ingreso	CHAR(10)			('0000-00-00')	Fecha de Ingreso del Cliente al Sistema
observacion	VARCHAR(100)	NN		(null)	Observaciones o Comentario
estado	DECIMAL(2)			(0)	Código del Estado
residencia	NUMERIC(4)	NN			Código de Referencia para Zonificar la residencia
residencia_descrip	VARCHAR(30)	NN			Descripción adicional de la Residencia (actualmente bloqueado)
Nombre de Índice		Tipo de Índice		Columnas	
PRIMARY		PRIMARY		cod_cliente	

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
secuencia	BIGINT(18)		PK	NN			Número secuencial de los elementos en la cola	
numeroorden	VARCHAR(15)						Número de la Orden de Servicio	
cod_actividad	VARCHAR(15)						Código de la Actividad incluido en la Orden	
procesado	CHAR(1)					('N')	Marca para indicar si el registro fue	

		procesado en la cola
Nombre de Índice	Tipo de Índice	Columnas
PRIMARY	PRIMARY	secuencial

tb_wf_correo_orden

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)	PK	NN			Tipo de Orden de servicio	
userid	VARCHAR(40)	PK	NN			Usuario que recibe email por el tipo de orden	
subject	VARCHAR(40)		NN			Asunto de email	
body	VARCHAR(250)		NN			Cuerpo del mensaje del email	
Nombre de Índice	Tipo de Índice	Columnas					
PRIMARY	PRIMARY	tipo_orden userid					

tb_wf_correo_tipotramite

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
secuencia	BIGINT(18)	PK	NN			Número secuencial del registro	
tipo_tramite	VARCHAR(10)		NN			Tipo de proyecto	
subtipotramite	VARCHAR(10)		NN			Subtipo de proyecto	
cod_cliente	VARCHAR(15)		NN			Código del Cliente	

user_id	VARCHAR (15)	NN	Usuario
envio_mail	CHAR(1)	NN	Bandera para indicar si envía email
Nombre de Índice	Tipo de Índice	Columnas	
PRIMARY	PRIMARY	secuencia	

tb_wf_datos_activ_tipo_tram

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_tramite	VARCHAR (10)	PK	NN			Tipo de Proyecto	
subtipotramite	VARCHAR (10)	PK	NN			Subtipo de Proyecto	
cod_cliente	VARCHAR (15)	PK	NN			Código del Cliente	
cod_actividad	VARCHAR (15)	PK	NN			Código de la Actividad	
dato_asociado	VARCHAR (10)	PK	NN			Dato asociado a la Actividad	
tipo_dato	NUMERIC(18)		NN			Tipo de dato (texto, costo, fecha)	
requerido	CHAR(1)		NN			Bandera para determinar obligatoriedad	
descripcion	VARCHAR (30)		NN			Descripción del Dato	
Nombre de Índice	Tipo de Índice	Columnas					
PRIMARY	PRIMARY	tipo_tramite subtipotramite cod_cliente cod_actividad dato_asociado					

tb_wf_datos_actividad

Datos de la actividad general

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_actividad	VARCHAR(15)		PK	NN			Código general de Actividad	
dato_asociado	VARCHAR(10)		PK	NN			Dato asociado a la Actividad	
tipo_dato	NUMERIC(18)						Tipo de dato (texto, costo, fecha)	
requerido	CHAR(1)			NN			Bandera para determinar si el dato es obligatorio	
descripcion	VARCHAR(30)			NN			Descripción del Dato	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		cod_actividad dato_asociado				

tb_wf_datos_actividad_orden

Datos de la actividad cuando se inicia una Orden de Trabajo (proyecto)

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)		PK	NN			Tipo de Orden de trabajo	
numero_orden	VARCHAR(15)		PK	NN			Número de la Orden de trabajo	
cod_actividad	VARCHAR(15)		PK	NN			Código de la Actividad	
dato_asociado	VARCHAR(10)		PK	NN			Dato asociado a la Actividad	
tipo_dato	NUMERIC(18)						Tipo de Dato (texto, costo,	

requerido	CHAR(1)	NN	fecha)
			Bandera para determinar si el dato es obligatorio
descripcion	VARCHAR(30)	NN	Descripción del dato
valor_num	DECIMAL(15, 4)	NN	Valor si es numérico
valor_fecha	DATETIME	NN	Valor si es fecha
valor_texto	VARCHAR(250)	NN	Valor si es texto
Nombre de Índice	Tipo de Índice	Columnas	
PRIMARY	PRIMARY	tipo_orden numero_orden cod_actividad dato_asociado	

tb_wf_datos_asociados

Datos definidos en la plantilla de Proyectos

Columna	Tipo Dato	Clave de Primaria	Nul	Band	Valor inicial	Comentario	Auto Inc
tipo_tramite	VARCHAR(10)	PK	NN			Tipo de Proyecto	
subtipotramite	VARCHAR(10)	PK	NN			Subtipo de Proyecto	
cod_cliente	VARCHAR(15)	PK	NN			Cliente predefinido	
dato_asociado	VARCHAR(10)	PK	NN			Dato asociado	
descripcion	VARCHAR(30)		NN			Descripción del dato	
tipo_dato	VARCHAR(3)					Tipo de dato (texto, fecha, costo)	

requerido	CHAR(1)	NN	Bandera para determinar si el dato es requerido
Nombre de Índice	Tipo de Índice	Columnas	
PRIMARY	PRIMARY	tipo_tramite subtipotramite cod_cliente dato_asociado	

tb_wf_datos_ordenes

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)	PK	NN			Tipo de Orden	
numero_orden	VARCHAR(15)	PK	NN			Número de Orden	
dato_asociado	VARCHAR(10)	PK	NN			Código del Dato asociado a la Orden	
tipo_dato	VARCHAR(3)					Tipo de Dato (numérico, texto, fecha, costo)	
requerido	CHAR(1)		NN			Marca para indicar que este dato es requerido en cualquier pantalla de ingreso	
descripcion	VARCHAR(30)		NN			Descripción general del tipo de dato	
valor_num	DECIMAL(15, 4)		NN			Se llena si el dato es numérico	
valor_fecha	DATETIME		NN			Se llena si el	

a	E							dato es fecha
valor_texto	VARCHAR (250)		NN					Se llena si el dato es texto
predefinido	CHAR(1)		NN					Bandera para determinar si el dato es predefinido o no
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		tipo_orden numero_orden dato_asociado				

tb_wf_def_prerrequisito								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_tramite	VARCHAR (10)		PK	NN			Tipo de Proyecto	
subtipotramite	VARCHAR (10)		PK	NN			Subtipo de Proyecto	
cod_cliente	VARCHAR (15)		PK	NN			Cliente	
cod_actividad	VARCHAR (15)		PK	NN			Actividad	
activ_prerrequisito	VARCHAR (15)		PK	NN			Prerrequisito de la Actividad	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		tipo_tramite subtipotramite cod_cliente cod_actividad activ_prerrequisito				

tb_wf_err_def_flujo								
Permite almacenar los errores encontrados al definir un flujo de actividades								
Columna	Tipo	de	Clave	Nul	Band	Valor	Comentario	Auto

	Dato	Prima o ria	o era	inicial	Inc
cod_reg	BIGINT(18)	PK	NN		Código de registro
tipo_tramite	VARCHAR(10)				Tipo de Proyecto
subtipotramite	VARCHAR(10)				Subtipo de Proyecto
cod_cliente	VARCHAR(15)				Cliente
cod_actividad	VARCHAR(10)				Actividad
activ_prereq	VARCHAR(10)				Prerrequisito de la Actividad
fecha_proceso	DATETIME				Fecha de procesamiento
cadena_error	VARCHAR(255)		NN		Mensaje de error encontrado
Nombre de Índice		Tipo de Índice		Columnas	
PRIMARY		PRIMARY		cod_reg	

tb_wf_err_flujo_orden

Para almacenar los errores encontrados en la Orden de Trabajo

Columna	Tipo Dato	de Clave Prima ria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
cod_reg	BIGINT(18)	PK	NN			Código de registro	
tipo_orden	VARCHAR(10)					Tipo de Orden de Trabajo	
numero_orden	VARCHAR(10)					Número de Orden de Trabajo	
tipo_tramite	VARCHAR(10)					Tipo de Proyecto	
subtipotramite	VARCHAR(10)					Subtipo de Proyecto	

cod_cliente	VARCHAR (15)		Cliente
cod_actividad	VARCHAR (10)		Actividad
activ_prereq	VARCHAR (10)		Prerrequisito
fecha_proceso	DATETIME		Fecha de procesamiento
cadena_error	VARCHAR (255)	NN	Mensaje de error encontrado
Nombre de Índice		Tipo de Índice	Columnas
PRIMARY		PRIMARY	cod_reg

TB_WF_FORMATO_TIEMPO

Formato de tiempos medidos para las actividades, están definidos : días, horas, minutos, semanas

Columna	Tipo Dato	de Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
codigo	CHAR(1)	PK	NN			Código definido	
descripcion	VARCHAR (30)		NN			Descripción del formato de tiempo	
Nombre de Índice		Tipo de Índice	Columnas				
PRIMARY		PRIMARY	codigo				

tb_wf_observ_actividad

Observaciones grabadas en la Orden de Trabajo

Columna	Tipo Dato	de Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
secuencia	BIGINT(18)	PK	NN			Secuencia del registro	
tipo_orden	VARCHAR (10)					Tipo de Orden	
numero_ord	VARCHA					Número de	

en	R(15)			Orden
cod_actividad	VARCHAR R(15)			Actividad de la Orden
tipo_observacion	VARCHAR R(3)			Tipo de observación
subtipo_observacion	VARCHAR R(3)			Subtipo de observación
fecha_ingreso	DATETIME	NN	(getdate())	Fecha de registro
user_id	VARCHAR R(15)	NN		Usuario que registró la observación
texto_observacion	VARCHAR R(255)	NN		Texto de la observación
rua	VARCHAR R(15)	NN		Registro Único de Auditoría
Nombre de Índice		Tipo de Índice		Columnas
PRIMARY		PRIMARY		secuencia

tb_wf_observacion

Plantilla de tipos de observaciones

Columna	Tipo de Dato	Clave Primaria	Nulo	Bandera	Valor inicial	Comentario	Auto Inc
tipo_observacion	NUMERIC(18)	PK	NN			Tipo de observación	
subtipo_observacion	NUMERIC(18)	PK	NN			Subtipo de observación	
detalle_observacion	VARCHAR R(254)		NN			Detalle de la observación	
user_id	VARCHAR R(15)		NN			Usuario	
fecha_hora	DATETIME		NN		(getdate())	Fecha de registro	
Nombre de Índice		Tipo de Índice		Columnas			
PRIMARY		PRIMARY		tipo_observacion subtipo_observacion			

tb_wf_oc_cliente								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)		PK	NN			Tipo de Orden de trabajo	
numero_orden	VARCHAR(15)		PK	NN			Número de Orden de trabajo	
num_oc_cliente	VARCHAR(25)		PK	NN			Número de Orden del Cliente	
descripcion	VARCHAR(150)						Descripción de la Orden del Cliente	
fecha_ref	DATE TIME			NN			Fecha de referencia de la orden de cliente	
fecha_ingreso	DATE TIME			NN		(getdate())	Fecha de ingreso del registro	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		tipo_orden numero_orden num_oc_cliente				

tb_wf_ordenes								
Registro de la Órdenes de Trabajo, cuando se inicia un proyecto se inicia una Orden de Trabajo.								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)		PK	NN			Tipo de Orden	
numero_orden	VARCHAR(15)		PK	NN			Número de Orden de	

				trabajo
anio	INTEGER	NN		Año de procesamiento de la Orden
tipo_tramite	VARCHAR(10)			Tipo de Proyecto asociado a la Orden
subtipotramite	VARCHAR(10)	NN		Subtipo de Proyecto
cod_cliente	VARCHAR(15)			Cliente
fecha_ingreso	DATETIME	NN	(getdate())	Fecha de registro de la Orden
fecha_estimada_fin	DATETIME	NN		Fecha estimada de finalización de la Orden
fecha_facturacion	DATETIME	NN		Fecha de facturación de la Orden, para integrarlo con un sistema externo
numero_factura	VARCHAR(15)	NN		Número de Factura del sistema externo de facturación
descripcion_orden	VARCHAR(125)	NN		Descripción de la Orden de Trabajo
rua	VARCHAR(15)	NN	(getdate())	Registro Único de Auditoría
oficial_cuenta	VARCHAR(15)	NN		Usuario Oficial de la Cuenta
diferencia_dias	INTEGER	NN		Días de diferencia en la estimación de fechas
diferencia_horas	INTEGER	NN		Horas de

horas				diferencia en la estimación de fechas
diferencia_minutos	INTEGER	NN		Minutos de diferencia en la estimación de fechas
indice	NUMERIC(6, 2)		(0)	Indice calculado de eficiencia promedio
estado	NUMERIC(2)		(1)	Estado de ejecución de la Orden
fecha_fin_recal	DATETIME	NN		Fecha Final recalculada de la Orden
fecha_fin_real	DATETIME	NN		Fecha Final Real de la Orden
fecha_inicio_real	DATETIME	NN		Fecha de Inicio Real de la Orden
Nombre de Índice		Tipo de Índice		Columnas
PRIMARY		PRIMARY		tipo_orden numero_orden

tb_wf_ordenes_asociadas

Guarda las relaciones entre las Órdenes del Cliente y las Órdenes de Trabajo de la Empresa, cuando se inicia un proyecto

Columna	Tipo de Dato	Clave de Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR(10)	PK	NN			Tipo de Orden	
numero_orden	VARCHAR(15)	PK	NN			Número de Orden	
tipo_orden_asoc	VARCHAR(10)	PK	NN			Tipo de Orden Asociada	

numero_orden_asoc	VARCHAR R(15)	PK	NN	Número de Orden Asociada
numero_oc	VARCHAR R(25)	PK	NN	Número de Orden del Cliente asociada
fecha_registro	DATETIME			(getdate(Fecha de registro))
Nombre de Índice		Tipo de Índice		Columnas
PRIMARY		PRIMARY		tipo_orden numero_orden tipo_orden_asoc numero_orden_asoc numero_oc

tb_wf_prerrequisitos

Definición de los prerrequisitos de las actividades dentro de una Orden iniciada

Columna	Tipo Dato	Clave de Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR R(10)	PK	NN			Tipo de Orden de trabajo	
numero_orden	VARCHAR R(15)	PK	NN			Número de la Orden de trabajo	
cod_actividad	VARCHAR R(15)	PK	NN			Actividad	
activ_prerrequisito	VARCHAR R(15)	PK	NN			Prerrequisito de la actividad	
Nombre de Índice		Tipo de Índice		Columnas			
PRIMARY		PRIMARY		tipo_orden numero_orden cod_actividad activ_prerrequisito			

tb_wf_responsables

Asignación de responsables para las actividades en una plantilla de tipo de proyecto.

Columna	Tipo Dato	de	Clave Primaria	Nul	Band	Valor inicial	Comentario	Auto Inc
secuencia	BIGINT(18)		PK	NN			Secuencia de registro	
tipo_tramite	VARCHAR(10)			NN			Tipo de Proyecto	
subtipotramite	VARCHAR(10)			NN			Subtipo de Proyecto	
cod_cliente	VARCHAR(15)			NN			Cliente	
cod_actividad	VARCHAR(15)			NN			Actividad del proyecto	
user_id	VARCHAR(15)			NN			Usuario asignado a la actividad	
es_responsable	CHAR(1)			NN			Marca para indicar si el usuario es responsable	
marcar_fin	CHAR(1)			NN			Marca para indicar si el usuario puede marcar el fin de la actividad	
envio_mail	CHAR(1)			NN			Marca para indicar si el usuario puede enviar email a todos los involucrados	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		secuencia				

tb_wf_responsables_activ

Responsables por Actividad dentro de una Orden

Columna	Tipo Dato	Clave de Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
secuencia	BIGINT(18)	PK	NN			Secuencia del registro	
tipo_orden	VARCHAR(10)					Tipo de Orden de Trabajo	
numero_orden	VARCHAR(15)					Número de la Orden de Trabajo	
cod_actividad	VARCHAR(15)					Código de la Actividad	
user_id	VARCHAR(15)					Usuario asignado a esta actividad	
es_responsable	CHAR(1)		NN			Marca para determinar si el usuario es responsable	
marcar_fin	CHAR(1)		NN			Marca para determinar si el usuario puede finalizar la actividad	
envio_mail	CHAR(1)		NN			Marca para determinar si el usuario puede enviar notificaciones por email a otros usuarios de la Orden de Trabajo	
rua	NUMERIC(18)		NN			Registro Único de Auditoría	
Nombre de Índice		Tipo de Índice		Columnas			
PRIMARY		PRIMARY		secuencia			

tb_wf_secuencia_ordenes								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
anio	INTEGER		PK	NN			Año de registro de la secuencia	
tipo_orden	VARCHAR(10)		PK	NN			Tipo de Orden	
secuencia	NUMERIC(6)					(1)	Número de secuencia siguiente	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		anio tipo_orden				

TB_WF_TIPO_ESTADO								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	DECIMAL(2)		PK	NN			Código de estado	
descripcion	VARCHAR(30)			NN			Descripción del estado para uso de Órdenes y proyectos	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		codigo				

TB_WF_TIPO_IDENTIFICA								
Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	CHAR(1)		PK	NN			Tipo de identificación	

descripcion	VARCHAR (30)	NN	Descripción de tipo de identificación
Nombre de Índice		Tipo de Índice	Columnas
PRIMARY		PRIMARY	codigo

tb_wf_tipo_orden

Define el año que se utilizará para asignar las secuencias a las Órdenes de Trabajo, la secuencia se reinicia cada cambio de año.

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_orden	VARCHAR R(10)	PK	NN			Tipo de Orden de Trabajo	
descripcion_orden	VARCHAR (40)		NN			Descripción de la Orden de Trabajo	
anio	SMALLINT (5)		NN			Año definido para las secuencias	
Nombre de Índice		Tipo de Índice	Columnas				
PRIMARY		PRIMARY	tipo_orden				

TB_WF_TIPO_PERSONA

Para definición de Personas Naturales, Jurídicas, Sociedades, etc.

Columna	Tipo Dato	de Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	CHAR(1)	PK	NN			Código del Tipo de Persona	
descripcion	VARCHAR (30)		NN			Descripción del tipo de persona	
Nombre de Índice		Tipo de Índice	Columnas				
PRIMARY		PRIMARY	codigo				

tb_wf_tipo_tramite

Plantilla de Tipos de Proyectos

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
tipo_tramite	VARCHAR(10)		PK	NN			Tipo de Proyecto	
subtipotramite	VARCHAR(10)		PK	NN			Subtipo de Proyecto	
cod_cliente	VARCHAR(15)		PK	NN			Cliente	
descripcion_corta	VARCHAR(30)			NN			Descripción corta del Tipo de Proyecto	
descripcion_larga	VARCHAR(250)			NN			Descripción larga del Tipo de Proyecto	
oficial_cuenta	VARCHAR(15)			NN			Oficial Usuario asignado al tipo de Proyecto	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		tipo_tramite subtipotramite cod_cliente				

TB_WF_USUARIOS_X_CLIENTE

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
user_id	VARCHAR(15)		PK	NN			Código de usuario del sistema	
cod_cliente	VARCHAR(15)						Código del cliente	
Nombre de Índice		Tipo de Índice		Columnas				
PRIMARY		PRIMARY		user_id				

TIPO_OBJETO

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	NUMERIC(18)		PK	NN			Código del tipo de objeto	
descripcion	VARCHAR(25)			NN			Descripción del objeto	
Nombre de Índice			Tipo de Índice			Columnas		
PRIMARY			PRIMARY			codigo		

TIPO_OPCION

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
codigo	NUMERIC(2)						Código del tipo de opción del sistema	
descripcion	VARCHAR(25)						Descripción del tipo de Opción	

tmp_seg_ordenes

Para definición y construcción del seguimiento de las Órdenes de trabajo

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
login	VARCHAR(15)		PK	NN			Código de Usuario	
numero_orden	VARCHAR(15)		PK	NN			Número de Orden en seguimiento	
descripcion_orden	VARCHAR(125)			NN			Descripción de la Orden	
nombre_razonsocial	VARCHAR(50)						Nombre o Razón Social	
ult_act	VARCHAR						Última Actividad	

	R(50)			ejecutada
sigu_act	VARCHAR R(50)			Siguiente Actividad por ejecutar
tipo_tramite	VARCHAR R(10)			Tipo de Proyecto
subtipotramite	VARCHAR R(10)	NN		Subtipo de Proyecto
descripcion_corta	VARCHAR R(30)	NN		Descripción corta de la Actividad
cod_cliente	VARCHAR R(15)			Cliente
tipo_orden	VARCHAR R(10)			Tipo de Orden de trabajo
fecha_estimada_fin	DATETIME	NN		Fecha estimada de finalización
anio	INTEGER	NN		Año de proceso
Nombre de Índice		Tipo de Índice		Columnas
PRIMARY		PRIMARY		login numero_orden

USUARIO

Definición de Usuarios

Columna	Tipo Dato	de	Clave Primaria	Nul o	Band era	Valor inicial	Comentario	Auto Inc
user_id	VARCHAR (15)		PK	NN			Código de Usuario	
password	VARCHAR (50)						Contraseña	
nombre	VARCHAR (50)						Nombre del Usuario	
apellido	VARCHAR (50)						Apellido	

ruc	VARCHAR (14)		Registro Único de Contribuyente
direccion	VARCHAR (250)		Dirección
telefono	VARCHAR (10)	NN	Teléfono
fax	VARCHAR (10)	NN	Fax
fecha_ingreso	CHAR(10)		Fecha de registro
email	VARCHAR (50)	NN	Correo electrónico
rua	VARCHAR (25)	NN	Registro Único de Auditoría
descripcion	VARCHAR (250)		Descripción general
estado	NUMERIC(2)		Código del estado del Usuario
tipo_usuario	CHAR(1)		Tipo de Usuario (Administrador, Gerente, etc)
cliente	VARCHAR (15)	NN	Cliente asociado al usuario
Nombre de Índice	Tipo de Índice	Columnas	
PRIMARY	PRIMARY	user_id	

Bibliografía

- Cleland, D., & Ireland, L. (2006). *Project Management: Strategic Design and Implementarion*. United States: McGraw-Hill Professional.
- Cobo, Á., Gómez, P., Pérez, D., & Rocha, R. (2005). *PHP y MySQL, Tecnologías para el desarrollo de aplicaciones web*. España: Díaz de Santos.
- Desmond, C. L. (2004). *Project Management for Telecommunications Managers*. *Kluwer Academic Publishers*, 17, 41.
- Gilmore, W. (2008). *Beginning PHP and MySQL from Novice to Professional*. Apress.
- Gosselin, D., Kokoska, D., & Easterbrooks, R. (2010). *PHP Programming with MYSQL*. Boston: Cengage.
- Heerkens, G. (2007). *Project Management: 24 Steps to Help You Master Any Project*. USA: McGraw-Hill Professional.
- Knutson, J., & Bitz, I. (1991). *Project Management – How to Plan and Manage Successful Projects*. New York: AMACOM.
- Lavin, P. (2006). *Object-Oriented PHP: concepts techniques and code*. San Francisco, USA: No Starch Press, Inc.
- Sherif, M. H. (2006). *Managing Projects in Telecommunication Services*. Canada : John Wiley & Sons.
- Williams, H., & Lane, D. (2004). *Web database applications with PHP and MySQL*. Sebastopol, California: O'Reilly Media, Inc.

Project Management for Telecommunications Managers
Celia L. Desmond – Kluwer Academic Publishers (2004)

Managing Projects in Telecommunication Services
Jhon Wiley & Sons – Wiley Interscience (Oct 6, 2006)

Project Management – 24 steps to help you master any project
Gary R. Heerkens – McGraw-Hill (June 12, 2007)

Project Management – Strategic Design and Implementarion Fifth edition
David I. Cleland – Lewis R. Ireland (Aug 21, 2006)

Project Management – How to Plan and Manage Successful Projects
Joan Knutson and Ira Bitz (June 15, 1991)

Manual de PHP 15/10/2005 (<http://www.php.net/docs.php>)

Building a Database-Driven Web Site Using PHP and MySQL
Kevin Yank (Jan 26, 2009)

PHP Programming with MYSQL
Don Gosselin, Diana Kokoska, Robert Easterbrooks(Jan 12, 2010)

Object-Oriented PHP:concepts techniques and code
Peter Lavin (June 12, 2006)

Web database applications with PHP and MySQL
Hugh E. Williams, David John Lane (2004)

Create dynamic sites with PHP & MySQL
DeveloperWorks, ibm.com/developerWork

