



UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

**TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIERO EN SISTEMAS COMPUTACIONALES**

TEMA:

**SISTEMA DE GESTIÓN DE PROCESOS DE
PRODUCCIÓN PARA LA EMPRESA DIKAPSA**

AUTOR: CARLOS ANDRÉS BURGOS VARGAS

DIRECTOR: ING. MAURICIO REA

Ibarra - Ecuador

2015



UNIVERSIDAD TÉCNICA DEL NORTE BIBLIOTECA UNIVERSITARIA

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

1. IDENTIFICACIÓN DE LA OBRA

La UNIVERSIDAD TÉCNICA DEL NORTE dentro del proyecto Repositorio Digital Institucional determina la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información.

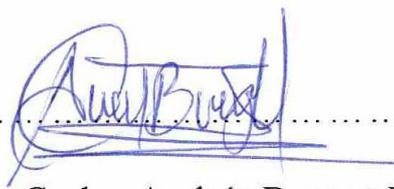
DATOS DE CONTACTO	
Cédula de Identidad	1003194618
Apellidos y Nombres	Burgos Vargas Carlos Andrés
Dirección	Sánchez y Cifuentes 17-53 y Teodoro Gómez.- Yacucalle
Email	caburgos@outlook.com
Teléfono Fijo	062954132
Teléfono Móvil	0981363431

DATOS DE LA OBRA	
Título	SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA
Autor	Burgos Vargas Carlos Andrés
Fecha	16 de diciembre de 2015
Programa	Pregrado
Título por el que se aspira	Ingeniero en Sistemas Computacionales
Director	Ing. Mauricio Rea

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, Carlos Andrés Burgos Vargas, con cédula de identidad Nro. 1003194618, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad de material y como apoyo a la educación, investigación y extensión, en concordancia con la ley de Educación Superior Artículo 144.

Ibarra a los siete días del mes de diciembre de 2015



Carlos Andrés Burgos Vargas

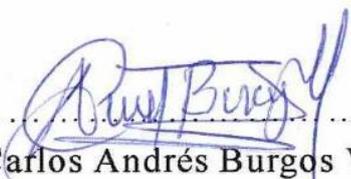
100319461-8

CONSTANCIA

El Autor (Carlos Burgos) manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es el titular de los derechos patrimoniales, por lo que asumo la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

En la ciudad de Ibarra, 16 de diciembre de 2015.

EL AUTOR (CARLOS BURGOS):


.....
Carlos Andrés Burgos Vargas
100319461-8

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

Yo, Carlos Andrés Burgos Vargas, con cédula de identidad Nro. 100319461-8, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor de la obra o trabajo de grado denominado: SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA, que ha sido desarrollado para optar por el título de: Ingeniero en Sistemas Computacionales en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En mi condición de autor me reservo los derechos morales de la obra antes citada.

En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Técnica del Norte.

Ibarra, 16 de diciembre de 2015


.....
Carlos Andrés Burgos Vargas
100319461-8

CERTIFICACIÓN

Certifico, que el presente trabajo de titulación "SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA" fue desarrollado en su totalidad por el señor Carlos Andrés Burgos Vargas bajo mi supervisión.

Ibarra, 16 de diciembre 2015

A handwritten signature in blue ink, appearing to read "Mauricio Rea", is written over a horizontal dotted line.

Ing. Mauricio Rea
DIRECTOR DE PROYECTO

CERTIFICACIÓN DE LA EMPRESA

Otavalo, 01 de Diciembre del 2015

Señores

UNIVERSIDAD TÉCNICA DEL NORTE

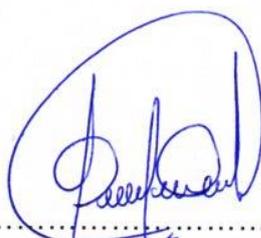
Presente.-

De mis consideraciones:

Siendo auspiciantes del proyecto de tesis del egresado CARLOS ANDRÉS BURGOS VARGAS con CI: 100319461-8 quien desarrolló su trabajo con el tema "SISTEMA DE GESTION DE PROCESOS DE PRODUCCION PARA LA EMPRESA DIKAPSA", me es grato informar que se han superado con satisfacción las pruebas técnicas y la revisión de cumplimiento de los requerimientos funcionales, por lo que se recibe el proyecto como culminado y realizado por parte del egresado CARLOS ANDRÉS BURGOS VARGAS. Una vez que hemos recibido la capacitación y documentación respectiva, nos comprometemos a continuar utilizando el antes mencionado aplicativo en beneficio de nuestra empresa.

El egresado CARLOS ANDRÉS BURGOS VARGAS puede hacer uso de este documento para los fines pertinentes en la Universidad Técnica del Norte.

Atentamente,



RUC: 1001403839001

Ing. Diego Oña

Gerente Comercial DIKAPSA

AGRADECIMIENTOS

Mi gratitud y agradecimiento infinito a Dios y mis padres por su apoyo incondicional en todo momento de mi vida, junto a mis hermanas.

A la Universidad Técnica del Norte, por haberme brindado la posibilidad de superarme como profesional y alcanzar mis metas propuestas.

A los docentes de la Carrera de Ingeniería en Sistemas Computacionales de la Facultad de Ingeniería en Ciencias Aplicadas por su guía en mi formación personal y académica.

Al Director de Proyecto Ing. Mauricio Rea, por su ayuda profesional y desinteresada en la elaboración de este proyecto de titulación.

A la empresa DIKAPSA y todo su personal por compartir su amistad y solidaridad, en especial al Ing. Diego Oña, Gerente Comercial de la misma, por brindarme su confianza y colaboración para desarrollar este trabajo; por su importante aporte y participación activa en el transcurso del mismo.

Carlos Andrés Burgos V.

DEDICATORIA

A mis padres por brindarme su apoyo incondicional durante toda mi vida. A mi Padre Carlos Hernán por enseñarme el valor de la palabra, por ser ejemplo de rectitud, honestidad y sacrificio incondicional. A mi Madre Nelly Irene por su ejemplo, consejos, cuidados, amor y desvelos.

A mis hermanas Verónica, Belén y a mis sobrinos, por darme ánimos para seguir luchando por mis sueños.

A mi novia JAVL, por brindarme su amor y apoyo en todo momento de alegría y tristeza, por tener toda esa constancia y anhelo de que llegara este momento.

A mis abuelitos por su ejemplo y todas sus oraciones.

A mis amigos y compañeros por todos los momentos compartidos.

Carlos Andrés Burgos V.

RESUMEN

El presente Trabajo Final consiste en desarrollar el Sistema de Gestión de Procesos de Producción para la Empresa DIKAPSA, la idea del mismo surge a raíz de la necesidad de automatizar los métodos tradicionales de producción.

Este documento está dividido en cinco capítulos, el primer capítulo describe la situación actual de la empresa DIKAPSA además de las necesidades por las que se desarrolla este sistema. El segundo capítulo indica todo los fundamentos teóricos que se debe conocer para desarrollar este sistema, las diferentes herramientas, arquitecturas y librerías a utilizar.

El tercer capítulo describe el levantamiento de procesos y los beneficios de utilizar procesos dentro de una empresa además de la norma de calidad ISO 9001 en lo que se refiere a gestión de calidad. En el cuarto capítulo se describe el desarrollo e implementación del sistema de procesos de producción desde su instalación hasta su interfaz de usuario.

El quinto capítulo contiene los diferentes impactos que produce la implementación de este sistema, además del análisis de costos. Finalmente, se expone las conclusiones y recomendaciones obtenidas durante la elaboración de este proyecto de titulación; además se anexa los manual técnico y de usuario.

ABSTRACT

This final work consist in the developed the “Sistema de Gestión de Procesos de Producción para la Empresa DIKAPSA”, that originated of from the need to automate traditional production methods.

This document is divided into five chapters, in the first chapter describes the current situation of the company DIKAPSA and also the needs of why develops this system. The second chapter discusses all the theoretical foundations that should be known to develop this system, different tools, architectures and libraries.

The third chapter describes the raising of process and the benefits of using a process within company, well as the standard of quality ISO 9001 in relation to quality management. In the fourth chapter describe the development and implementation of the system of production processes from installation until to user interface.

The fifth chapter contains the different impacts produced by the implementation of this system, in addition to the cost analysis. Finally, conclusions and recommendations obtained during the preparation of this graduation project is exposed; also the technical manual and user annexed.

ÍNDICE DE CONTENIDOS

1. IDENTIFICACIÓN DE LA OBRA	II
2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD.....	III
CONSTANCIA	IV
CERTIFICACIÓN	VI
CERTIFICACIÓN DE LA EMPRESA	VII
AGRADECIMIENTOS	VIII
DEDICATORIA	IX
RESUMEN.....	X
ABSTRACT	XI
ÍNDICE DE CONTENIDOS.....	XII
ÍNDICE DE FIGURAS	XX
ÍNDICE DE TABLAS.....	XXII
PRESENTACIÓN.....	XXIII
CAPÍTULO 1 SITUACIÓN ACTUAL	1
1.1. INTRODUCCIÓN.....	1
1.2. ANTECEDENTES	2
1.2.1. Misión.....	2
1.2.2. Visión.....	3
1.2.3. Organización de la Empresa.....	3
1.2.4. Productos y Servicios que ofrece DIKAPSA.....	4
1.2.4.1. Material Promocional.....	4
1.2.4.2. Material Institucional.....	4
1.2.4.3. Material Editorial.....	5
1.2.4.4. Gigantografías.....	5
1.2.4.5. Comprobantes (Libretines).....	6
1.3. SITUACIÓN ACTUAL DE LA EMPRESA DIKAPSA	6
1.4. PROBLEMA	6
1.5. OBJETIVOS DEL PROYECTO	7
1.5.1. Objetivo General.....	7
1.5.2. Objetivos Específicos.....	7
1.6. JUSTIFICACIÓN.....	8
1.7. ALCANCE	8
CAPÍTULO 2 MARCO TEÓRICO	10
2.1. METODOLOGÍA DE DESARROLLO ÁGIL SCRUM.....	11
2.1.1 Introducción.....	11
2.1.2 Características.....	11
2.1.3 Roles de Equipo o SCRUM TEAM.....	12
2.1.3.1 El Dueño del Producto o Product Owner.....	12
2.1.3.2 El Equipo de Desarrollo o Develop Team.....	13
2.1.3.3 El SCRUM Master	13
2.1.4 Ciclo de Vida del Proyecto.....	13

2.1.4.1	El Sprint	13
2.1.4.2	Reunión de Planificación del Sprint o Sprint Planning Meeting	14
2.1.4.3	Objetivo del Sprint o Sprint Goal	15
2.1.4.4	Reunión Diaria o Daily SCRUM	15
2.1.4.5	Revisión Del Sprint o Sprint Review Meeting	15
2.1.4.6	Retrospectiva del Sprint o Sprint Retrospective	15
2.1.5	<i>Artefactos de SCRUM</i>	16
2.1.5.1	Lista de Producto o Product Backlog	16
2.1.5.2	Lista de Pendientes del Sprint (Sprint Backlog)	17
2.1.5.3	Incrementos	17
2.1.6	<i>Ventajas</i>	18
2.1.7	<i>Desventajas</i>	18
2.2	Lenguaje de Programación Java	19
2.2.1	<i>Introducción</i>	19
2.2.2	<i>Componentes</i>	19
2.2.3	<i>Ediciones</i>	20
2.2.4	<i>Características</i>	20
2.2.5	<i>Requerimientos JDK</i>	22
2.2.6	<i>Java Virtual Machine (JVM)</i>	22
2.2.7	<i>Etapas para la ejecución de un programa</i>	23
2.2.7.1	Editor	23
2.2.7.2	Compilación	24
2.2.7.3	Interpretación	24
2.3	Entorno de Desarrollo Integrado IDE para Java (Eclipse Luna)	24
2.3.1	<i>Introducción</i>	24
2.3.2	<i>Definición de Eclipse Luna</i>	25
2.3.3	<i>Requisitos del Software</i>	25
2.3.4	<i>Ventajas</i>	26
2.3.5	<i>Características</i>	26
2.3.6	<i>Licencias</i>	27
2.3.7	<i>Plataforma de Cliente Enriquecido</i>	27
2.3.8	<i>Compilar</i>	28
2.3.9	<i>Ejecutar</i>	28
2.4	Contenedor Web de Aplicaciones Apache Tomcat	28
2.4.1	<i>Descripción</i>	28
2.4.2	<i>Características</i>	29
2.4.3	<i>Definición de contenedor</i>	29
2.5	Diseñador de Reportes	30
2.5.1	<i>Diseñador iReport</i>	30
2.5.1.1	Características iReport	30
2.5.1.2	Prerrequisitos	31
2.5.1.3	Requerimientos de instalación	31
2.5.2	<i>Jasper report</i>	31
2.5.2.1	Características	32
2.5.2.2	Funciones	32
2.5.2.3	Librerías JasperReport	33
2.5.3	<i>Jasper server</i>	33
2.5.3.1	Elementos	33
2.6	PostgreSQL 9.3	34
2.6.1	<i>Introducción a PostgreSQL</i>	34
2.6.2	<i>Características principales de PostgreSQL</i>	35

2.6.2.2.	Programación / Desarrollo	35
2.6.3.	<i>Arquitectura</i>	36
2.6.4.	<i>Ventajas de PostgreSQL</i>	37
2.6.5.	<i>Requerimientos para instalación</i>	38
2.7.	JDBC.....	38
2.7.1.	<i>Introducción</i>	38
2.7.2.	<i>Tipos</i>	39
2.7.3.	<i>Interfaces y clases</i>	41
2.8.	INTERFAZ DE PERSISTENCIA JAVA (JPA).....	42
2.8.1.	<i>Introducción</i>	42
2.8.2.	<i>Definición JPA Entity Manager</i>	43
2.8.3.	<i>Utilidades de JPA</i>	43
2.8.4.	<i>Requerimientos para la instalación</i>	44
2.8.5.	<i>Arquitectura</i>	45
2.8.6.	<i>Object Relation Mapping (ORM)</i>	46
2.8.6.1.	Funciones avanzadas	46
2.8.6.2.	Arquitectura ORM	47
2.8.7.	<i>Anotaciones</i>	48
2.8.8.	<i>Entidad</i>	49
2.8.9.	<i>Persistencia</i>	50
2.8.10.	<i>Identidad</i>	50
2.8.11.	<i>Transaccionalidad</i>	51
2.8.12.	<i>Granularidad</i>	51
2.8.13.	<i>Persistiendo una Entidad</i>	51
2.8.14.	<i>Buscar una entidad</i>	52
2.8.15.	<i>Eliminar la entidad</i>	52
2.8.16.	<i>Transacciones</i>	52
2.9.	JAVA PERSISTENCE QUERY LANGUAGE.....	53
2.9.1.	<i>Definiendo Consultas</i>	53
2.9.2.	<i>Consultas Dinámicas</i>	53
2.9.3.	<i>Tipos de Parámetros</i>	54
2.9.4.	<i>Interfaces JPA</i>	54
2.9.5.	<i>Estados</i>	55
2.9.6.	<i>Relaciones Múltiples de la Entidad</i>	56
2.9.7.	<i>Relaciones y Borrado en cascada</i>	57
2.9.7.1.	Tipos de Memoria cache	57
2.9.7.2.	Clases con claves primarias	57
2.10.	JAVA SERVER FACES (JSF).....	58
2.10.1.	<i>Introducción</i>	58
2.10.2.	<i>Características</i>	59
2.10.3.	<i>Ventajas</i>	59
2.10.4.	<i>Definición de aplicación JSF</i>	60
2.10.5.	<i>Los backbeans</i>	61
2.10.5.1.	Respondiendo a las acciones del usuario	62
2.10.5.2.	La navegación entre páginas	63
2.10.6.	<i>Ciclo de Vida Estándar JSF</i>	64
2.10.7.	<i>Facelets</i>	66
2.10.8.	<i>Beans</i>	67
2.10.8.1.	ManagedBean	67
2.10.8.2.	BackingBean	68

2.10.9.	Ámbito de los beans gestionados	68
2.10.9.1.	@RequestScoped	69
2.10.9.2.	@CustomScoped	69
2.10.9.3.	@ConversationScoped	69
2.10.9.4.	@ApplicationScope	70
2.10.9.5.	@SessionScope	70
2.10.9.6.	@ViewScope	70
2.10.9.7.	@RequestScope	71
2.10.10.	Manejo del Scope y los ManagedBeans	71
2.10.11.	Etiquetas JSF	71
2.10.12.	Pasos para el Desarrollo	73
2.10.12.1.	Arquitectura JSF	73
2.10.12.2.	Clases de ayuda del lado del servidor	73
2.10.13.	Modelo Vista Controlador en JSF	74
2.10.13.1.	Vista	74
2.10.13.2.	Modelo: beans gestionados	74
2.10.13.3.	Controlador	74
2.11.	FRAMEWORK RIA PARA JAVA SERVER FACES	75
2.11.1.	Definición	75
2.11.2.	Características	75
2.11.3.	Desarrollar RIA con estándares	76
2.11.4.	Implementación RIA para JSF	77
2.11.4.1.	Primefaces	77
2.12.	ATMÓSFERA SOCKET	78
2.12.1.	Introducción	78
2.12.2.	Componentes:	78
2.12.2.1.	Atmosphere-runtime:	78
2.12.2.2.	Atmosphere-jersey:	78
2.12.2.3.	Atmosphere gwt:	79
2.12.3.	Características:	79
2.12.4.	Modos de funcionamiento:	80
2.13.	SISTEMAS DE INFORMACIÓN DE PROCESOS DE PRODUCCIÓN	80
2.13.1.	Introducción a los procesos	80
2.13.2.	Tipos	80
2.13.3.	Fases de Análisis Del Proceso	81
CAPÍTULO 3 LEVANTAMIENTO DE PROCESOS		82
3.1.	DEFINICIÓN DE PROCESOS DE PRODUCCIÓN	82
3.1.1	Definición Gestión de Procesos	82
3.1.2	Objetivos de la Gestión por Procesos	83
3.1.3	Clasificación de procesos	83
3.1.3.1	Procesos estratégicos	84
3.1.3.2	Procesos del negocio o clave	84
3.1.3.3	Procesos de apoyo	84
3.1.4	Elementos de los procesos	84
3.1.5	Importancia de la gestión de procesos	85
3.1.6	Beneficios de la gestión por procesos	85
3.1.7	Motores de la gestión BPM	86
3.2.	ORGANIZACIÓN INTERNACIONAL DE NORMALIZACIÓN ISO 9001	87
3.2.1	Resumen	87
3.2.2	Sección 0.- Introducción	87
3.2.3	Sección 1.- Objeto Y Campo De Aplicación	88

3.2.4	<i>Sección 2.- Referencias Normativas</i>	88
3.2.5	<i>Sección 3.- Términos Y Definiciones</i>	88
3.2.6	<i>Sección 4.- Requisitos Del Sistema</i>	88
3.2.7	<i>Sección 5.- Responsabilidad De La Dirección</i>	89
3.2.8	<i>Sección 6.- Gestión de Recursos</i>	89
3.2.9	<i>Sección 7.- Realización Del Producto</i>	90
3.2.10	<i>Sección 8.- Medición Análisis Y Mejora</i>	91
3.3	MODELADO DE PROCESOS.....	91
3.3.1	<i>Diagrama de flujo</i>	92
3.3.1.1	Características de los diagramas de Flujo	93
3.3.1.2	Tipos de diagramas de Flujo	94
3.3.1.3	Símbolos en los Diagramas de Flujo	95
3.3.1.4	Ventajas de los Diagramas de Flujo	95
3.3.1.5	Recomendaciones para el uso y aplicación de diagramas	96
3.4	SISTEMA DE GESTIÓN DE PROCESOS DE LA EMPRESA DE DISEÑO E IMPRESIÓN DIKAPSA / MAPA DE PROCESOS	97
3.4.1	<i>Proceso general de la empresa</i>	97
3.4.1.1	Procesos Gerenciales	98
3.4.1.2	Procesos Operativos	98
3.4.1.3	Procesos de Soporte	99
3.4.2	<i>Gigantografías</i>	99
3.4.2.1	Departamento de ventas	100
3.4.2.2	Departamento de diseño	101
3.4.2.3	Departamento de producción	101
3.4.3	<i>Comprobantes (Libretines)</i>	102
3.4.3.1	Departamento de ventas	103
3.4.3.2	Departamento de diseño	103
3.4.3.3	Departamento de producción	104
CAPÍTULO 4 DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA		105
4.1.	DESCRIPCIÓN GENERAL DE LA METODOLOGÍA EN EL SISTEMA	106
4.1.1.	<i>Fundamentación</i>	106
4.2.	PERSONAS Y ROLES DEL PROYECTO	107
4.3.	ARTEFACTOS	107
4.4.	DOCUMENTOS.....	108
4.4.1.	<i>Pila de producto (Product Backlog)</i>	108
4.4.1.1.	Identificador (ID) de la Historia	108
4.4.1.2.	Enunciado de la Historia	108
4.4.1.3.	Origen	109
4.4.1.4.	Alias	109
4.4.1.5.	Estado	109
4.4.1.6.	Dimensión / Esfuerzo	109
4.4.1.7.	Iteración (Sprint)	110
4.4.1.8.	Prioridad	110
4.4.1.9.	Comentarios	110
4.4.2.	<i>Planificación de la Pila del sprint (Sprint Backlog)</i>	112
4.5.	IMPLEMENTACIÓN DE LA ARQUITECTURA DE LA SOLUCIÓN.....	115
4.5.1.	<i>Implementación de Diseño de Datos</i>	115
4.5.2.	<i>Arquitectura del Sistema</i>	117
4.5.2.1.	Modelo	119
4.5.2.2.	Controller	119
4.5.2.3.	Vista	119

4.6.	SPRINT	120
4.6.1.	<i>Sprint 1</i>	121
4.6.1.1.	Priorización	121
4.6.1.2.	Asignación del Sprint	122
4.6.1.3.	Prototipo interfaz gráfica	122
4.6.2.	<i>Sprint 2</i>	123
4.6.2.1.	Priorización	123
4.6.2.2.	Asignación del Sprint	124
4.6.2.3.	Prototipo interfaz gráfica	124
4.6.3.	<i>Sprint 3</i>	125
4.6.3.1.	Priorización	126
4.6.3.2.	Asignación del Sprint	126
4.6.3.3.	Prototipo interfaz gráfica	126
4.6.4.	<i>Sprint 4</i>	130
4.6.4.1.	Priorización	130
4.6.4.2.	Asignación del Sprint	131
4.6.4.3.	Prototipo interfaz gráfica	131
4.6.5.	<i>Sprint 5</i>	132
4.6.5.1.	Priorización	132
4.6.5.2.	Asignación del Sprint	133
4.6.5.3.	Prototipo interfaz gráfica	133
4.6.6.	<i>Sprint 6</i>	134
4.6.6.1.	Priorización	134
4.6.6.2.	Asignación del Sprint	135
4.6.6.3.	Prototipo interfaz gráfica	135
4.6.7.	<i>Sprint 7</i>	136
4.6.7.1.	Priorización	136
4.6.7.2.	Asignación del Sprint	136
4.6.8.	<i>Sprint 8</i>	136
4.6.8.1.	Priorización	137
4.6.8.2.	Asignación del Sprint	137
4.7.	INCREMENTO E IMPLEMENTACIÓN DE LOS SPRINTS	138
4.7.1.	<i>Entregable Creación, Modificación, Vista y Eliminación de Clientes</i>	138
4.7.1.1.	Modelo Físico	138
4.7.1.2.	Diagrama de Clases	139
4.7.1.3.	Interfaz	139
4.7.2.	<i>Entregable Toma de Orden de Trabajo Estandarizadas</i>	140
4.7.2.1.	Modelo Físico	141
4.7.2.2.	Diagrama de Clases	142
4.7.2.3.	Interfaz	142
4.7.3.	<i>Entregable Creación de Secuencias Procesos de una Línea de Producción</i>	143
4.7.3.1.	Modelo Físico	144
4.7.3.2.	Diagrama de Clases	144
4.7.3.3.	Interfaz	145
4.7.4.	<i>Entregable Ver Orden de Trabajo en Producción</i>	145
4.7.4.1.	Modelo Físico	146
4.7.4.2.	Diagrama de Clases	146
4.7.4.3.	Interfaz	147
4.7.5.	<i>Entregable Inicio de Proceso de una Orden de Trabajo</i>	147
4.7.5.1.	Modelo Físico	148
4.7.5.2.	Diagrama de Clases	148
4.7.5.3.	Interfaz	149
4.7.6.	<i>Entregable Gestión de Tareas de una Orden de Trabajo</i>	149

4.7.6.1.	Modelo Físico.....	150
4.7.6.2.	Diagrama de Clases.....	151
4.7.6.3.	Interfaz.....	152
4.7.7.	<i>Entregable Entregas Órdenes de Trabajos.....</i>	<i>152</i>
4.7.7.1.	Modelo Físico.....	153
4.7.7.2.	Diagrama de Clases.....	153
4.7.7.3.	Interfaz.....	154
4.8.	GRAFICAS DE REGISTRO Y SEGUIMIENTO DEL AVANCE	154
4.8.1.	<i>Ejecución avance Sprint 1.....</i>	<i>155</i>
4.8.1.1.	Gráfica de producto o Burn Up	155
4.8.1.2.	Gráfica de Avance o Burn Down	155
4.8.2.	<i>Ejecución avance Sprint 2.....</i>	<i>156</i>
4.8.2.1.	Gráfica de producto o Burn Up	156
4.8.2.2.	Gráfica de Avance o Burn Down	156
4.8.3.	<i>Ejecución avance Sprint 3.....</i>	<i>156</i>
4.8.3.1.	Gráfica de producto o Burn Up	156
4.8.3.2.	Gráfica de Avance o Burn Down	157
4.8.4.	<i>Ejecución avance Sprint 4.....</i>	<i>157</i>
4.8.4.1.	Gráfica de producto o Burn Up	157
4.8.4.2.	Gráfica de Avance o Burn Down	157
4.8.5.	<i>Ejecución avance Sprint 5.....</i>	<i>158</i>
4.8.5.1.	Gráfica de producto o Burn Up	158
4.8.5.2.	Gráfica de Avance o Burn Down	158
4.8.6.	<i>Ejecución avance Sprint 6.....</i>	<i>158</i>
4.8.6.1.	Gráfica de producto o Burn Up	158
4.8.6.2.	Gráfica de Avance o Burn Down	159
4.8.7.	<i>Ejecución avance Sprint 7.....</i>	<i>159</i>
4.8.7.1.	Gráfica de producto o Burn Up	159
4.8.7.2.	Gráfica de Avance o Burn Down	159
4.8.8.	<i>Ejecución avance Sprint 8.....</i>	<i>160</i>
4.8.8.1.	Gráfica de producto o Burn Up	160
4.8.8.2.	Gráfica de Avance o Burn Down	160
4.9.	PRUEBAS.....	161
4.9.1.	<i>Pruebas de Creación y Modificación de Clientes</i>	<i>161</i>
4.9.2.	<i>Pruebas de Toma de ODT Estandarizadas.....</i>	<i>161</i>
4.9.3.	<i>Pruebas de Creación de Secuencias Procesos de una Línea de Producción.....</i>	<i>162</i>
4.9.4.	<i>Pruebas de Ver Orden de Trabajo en Producción</i>	<i>162</i>
4.9.5.	<i>Pruebas de Inicio de Proceso de una Orden de Trabajo</i>	<i>163</i>
4.9.6.	<i>Pruebas de Gestión de Tareas de una Orden de Trabajo</i>	<i>163</i>
4.9.7.	<i>Entregas Órdenes de Trabajos</i>	<i>164</i>
4.9.8.	<i>Pruebas Ejecución de los Sprint del Sistema</i>	<i>165</i>
4.9.8.1.	LISTA DE CHEQUEO: Control de Calidad de Sprints desarrollados.....	165
4.9.9.	<i>Pruebas de Puesta en Producción e Integración con los Sprints</i>	<i>166</i>
CAPÍTULO 5 ANÁLISIS DE IMPACTOS		167
5.1.	IMPACTOS DEL PROYECTO	168
5.1.1.	<i>Impacto económico</i>	<i>168</i>
5.1.1.1.	Análisis impacto económico	169
5.1.2.	<i>Impacto ambiental.....</i>	<i>169</i>
5.1.2.1.	Análisis impacto ambiental	170
5.1.3.	<i>Impacto social y cultural</i>	<i>170</i>
5.1.3.1.	Análisis impacto social y cultural	170

5.1.4.	<i>Impacto general</i>	171
5.1.4.1.	Análisis impacto general	171
5.2.	ANÁLISIS DE COSTOS	171
5.2.1	<i>Costo de mantenimiento</i>	172
5.2.2	<i>Análisis económico</i>	172
5.2.2.1	<i>Costo de inversión</i>	172
CONCLUSIONES		174
RECOMENDACIONES		176
GLOSARIO DE TÉRMINOS		177
LISTA DE REFERENCIAS		183
ANEXOS		187

ÍNDICE DE FIGURAS

Figura 1. 1 Estructura Orgánica de la Empresa. _____	3
Figura 2. 1 Diagrama de la arquitectura _____	12
Figura 2. 2 Ejemplo de proceso SCRUM / continuo. _____	14
Figura 2. 3 Incremento Iterativo / continuo _____	18
Figura 2. 4 Etapas para la ejecución de un programa Java. _____	23
Figura 2. 5 Diagrama de flujo Jasper report _____	34
Figura 2. 6 Componentes de un sistema PostgreSQL _____	36
Figura 2. 7 Componentes JDBC comunes _____	41
Figura 2. 8 Diagrama de funcionamiento. _____	44
Figura 2. 9 Nivel de clase arquitectura de JPA _____	45
Figura 2. 10 ORM arquitectura _____	47
Figura 2. 11 Ciclo de vida de JSF _____	65
Figura 2. 12 JSF _____	73
Figura 2. 13 Capas de programación para una página web _____	76
Figura 2. 14 Características atmosphere framework _____	79
Figura 2. 16 Fases de análisis de proceso _____	81
Figura 3. 1 Símbolos diagramas de flujo. _____	95
Figura 3. 2 Diagrama de flujo general. _____	97
Figura 3. 3 Portal para investigadores y profesionales _____	100
Figura 3. 4 Proceso de producción afiches _____	102
Figura 4. 1 Pila del producto _____	111
Figura 4. 2 Planificación de pila del Spring _____	114
Figura 4. 3 Modelo Entidad Relación del Sistema. _____	116
Figura 4. 4 Arquitectura del Sistema. _____	118
Figura 4. 5 Interfaz de Gestión de Módulos _____	123
Figura 4. 6 Interfaz de Gestión de Ventas _____	125
Figura 4. 7 Interfaz de Gestión de Catalogo Gestión de Procesos _____	128
Figura 4. 8 Interfaz de Gestión de Inicio de Orden de Producción _____	129
Figura 4. 9 Interfaz de Gestión de Historial Tareas Procesos ODT _____	130
Figura 4. 10 Interfaz de Control de Orden de Trabajo _____	132
Figura 4. 11 Interfaz de Gestión de Usuarios _____	134
Figura 4. 12 Interfaz de Gráfica de Reporte _____	135
Figura 4. 13 Modelo físico creación y modificación de clientes _____	138
Figura 4. 14 Diagrama de clases de creación y modificación de clientes _____	139
Figura 4. 15 Interfaz de creación y modificación del cliente _____	140
Figura 4. 16 Modelo físico creación nueva orden de trabajo _____	141
Figura 4. 17 Diagrama de clases de creación nueva orden de trabajo _____	142
Figura 4. 18 Interfaz de creación de una nueva orden de trabajo _____	143
Figura 4. 19 Modelo físico creación nueva secuencia de procesos _____	144
Figura 4. 20 Diagrama de clases de creación nueva secuencias de procesos _____	144
Figura 4. 21 Interfaz de creación de una nueva secuencia de procesos _____	145
Figura 4. 22 Modelo físico Ver Orden de Trabajo en Producción _____	146
Figura 4. 23 Diagrama de clases Ver Orden de Trabajo en Producción _____	146
Figura 4. 24 Interfaz Ver Orden de Trabajo en Producción _____	147
Figura 4. 25 Modelo físico gestión de inicio de una orden de producción _____	148
Figura 4. 26 Diagrama de clases Inicio de Procesos de Producción de una Orden de Trabajo _____	148
Figura 4. 27 Interfaz Inicio de Procesos de Producción de una Orden de Trabajo _____	149
Figura 4. 28 Modelo físico gestión tareas de una orden de producción _____	150

Figura 4. 29 Diagrama de clases gestión tareas de una orden de producción	151
Figura 4. 30 Interfaz gestión tareas de una orden de producción	152
Figura 4. 31 Modelo físico gestión entregas ordenes trabajos.....	153
Figura 4. 32 Diagrama de clases gestión entregas ordenes trabajos	153
Figura 4. 33 Interfaz gestión entregas ordenes trabajos	154
Figura 4. 34 Gráfico de tareas pendientes del Sprint 1	155
Figura 4. 35 Gráfico de Horas de Trabajo del Sprint 1	155
Figura 4. 36 Gráfico de tareas pendientes del Sprint 2	156
Figura 4. 37 Gráfico de Horas de Trabajo del Sprint 2	156
Figura 4. 38 Gráfico de tareas pendientes del Sprint 3	156
Figura 4. 39 Gráfico de Horas de Trabajo del Sprint 3	157
Figura 4. 40 Gráfico de tareas pendientes del Sprint 4	157
Figura 4. 41 Gráfico de Horas de Trabajo del Sprint 4	157
Figura 4. 42 Gráfico de tareas pendientes del Sprint 5	158
Figura 4. 43 Gráfico de Horas de Trabajo del Sprint 5	158
Figura 4. 44 Gráfico de tareas pendientes del Sprint 6	158
Figura 4. 45 Gráfico de Horas de Trabajo del Sprint 6	159
Figura 4. 46 Gráfico de tareas pendientes del Sprint 7	159
Figura 4. 47 Gráfico de Horas de Trabajo del Sprint 7	159
Figura 4. 48 Gráfico de tareas pendientes del Sprint 8	160
Figura 4. 49 Gráfico de Horas de Trabajo del Sprint 8	160
Figura 4. 50 Informe de ejecución de Pruebas	165

ÍNDICE DE TABLAS

Tabla 2. 1 Requerimientos Básicos PostgreSQL _____	38
Tabla 2. 2 Requisitos del sistema _____	44
Tabla 2. 3 Unidades mostradas en la arquitectura _____	45
Tabla 2. 4 Lista de anotaciones de ejemplo _____	49
Tabla 4. 1 Personas y roles _____	107
Tabla 4. 2 Iteraciones del Sprint _____	121
Tabla 4. 3 Asignación del sprint 1 _____	122
Tabla 4. 4 Asignación del sprint 2 _____	124
Tabla 4. 5 Asignación Sprint 3 _____	126
Tabla 4. 6 Asignación Sprint 4 _____	131
Tabla 4. 7 Asignación sprint 5 _____	133
Tabla 4. 8 Asignación del sprint 6 _____	135
Tabla 4. 9 Asignación del sprint 7 _____	136
Tabla 4. 10 Asignación del sprint 8 _____	137
Tabla 4. 11 Pruebas de creación y modificación de clientes _____	161
Tabla 4. 12 Pruebas de una nueva orden de trabajo _____	161
Tabla 4. 13 Pruebas de Secuencias Procesos de una Línea de Producción _____	162
Tabla 4. 14 Pruebas de Ver Orden de Trabajo en Producción _____	162
Tabla 4. 15 Pruebas de Inicio de Proceso de una Orden de Trabajo _____	163
Tabla 4. 16 Pruebas de Inicio de Proceso de una Orden de Trabajo _____	163
Tabla 4. 17 Pruebas de Entregas Órdenes de Trabajo. _____	164
Tabla 4. 18 Pruebas de puesta en producción e integración con los Sprints _____	166
Tabla 5. 1 Escala de Valores _____	167
Tabla 5. 2 Nivel de impacto económico _____	168
Tabla 5. 3 Nivel de impacto ambiental _____	169
Tabla 5. 4 Nivel de impacto social y cultural _____	170
Tabla 5. 5 Nivel de impacto general _____	171
Tabla 5. 6 Requerimiento hardware _____	173
Tabla 5. 7 Costo de elaboración _____	173
Tabla 5. 8 Costo de implementación _____	173

PRESENTACIÓN

La rápida evolución de la tecnología en estos últimos tiempos ha invadido todo el mundo, el aumento de requerimientos de los usuarios y la competencia de las empresas por cubrir estas demandas, el manejo de la información y la agilización de procesos se ha convertido en un factor esencial para el crecimiento de empresas; por ende la elección de buenas herramientas y la utilización de software para desarrollo hará que se presente un sistema que garantice el servicio de calidad a los clientes y así incrementen a futuro la posibilidad de asegurar una posición exitosa y brindar servicios de calidad a nivel nacional.

Un sistema informático está compuesto por hardware y software. En cuanto al hardware, se puede conseguir de manera fácil y bajo ciertas especificaciones dependiendo de las necesidades claramente definida. Y en cuanto a software, para este sistema se diseñará acorde a la metodología SCRUM que permitirá hacer el levantamiento de requerimientos de manera ordenada y utilizando lenguajes Open Source de desarrollo como Java, Framework JSF y PostgreSQL como motor de base de datos, Primefaces para agilizar las vistas, pantallas, formularios y JPA para la persistencia de datos.

La herramienta a presentar se construyó para solventar los procesos básicos de control de la empresa DIKPASA, ya que estos procesos se los realizaba de manera manual, provocando pérdidas importantes de información, retraso en las respuestas de atención al cliente, errores en entregas que pueden ocasionar pérdidas al cliente y a la empresa.

A lo largo de la presente tesis se irá analizando todos estos factores para poder realizar un diseño que se ajuste a las necesidades del cliente y los trabajadores de la empresa.

Siendo así, surge la necesidad de implementar un SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA en especial para brindar el servicio a la ciudadanía del cantón Otavalo, con el objeto de garantizar un servicio de impresión de calidad, seguridad y confianza en el cliente.

Capítulo 1

SITUACIÓN ACTUAL



En el presente capítulo se describe la situación actual en la que se encuentra la empresa DIKAPSA, la deficiencia del proceso y la necesidad de implementar un sistema de gestión de procesos de producción.

1.1. Introducción

En la actualidad las empresas tienden a mejorar constantemente los procesos de producción haciendo uso de software para tener mayor productividad. Esta mejora se logra mediante la sistematización en todos los modelos de procesos obteniendo una alta competitividad, eficiencia y agilidad en la demanda que requiere el mercado.

DIKAPSA maneja varios procesos de producción tales como: Comprobantes, Gigantografías, Offset Full Color, Offset 1 -2 Colores y Digital 550.

Como consecuencia a lo descrito anteriormente se desarrollará un software adecuado a los procesos existentes dentro de la empresa DIKAPSA.

Capítulo 1: Situación Actual

1.2. Antecedentes

DIKAPSA es una empresa gráfica dedicada a brindar soluciones impresas que satisfagan las necesidades de sus clientes, lleva 13 años de operaciones en los cuales se le ha sido posible crecer y tener una marca reconocida. Durante su trayectoria ha tenido varios cambios importantes, ha realizado sus operaciones en 5 establecimientos hasta poder adquirir sus instalaciones propias.

Hasta la actualidad a incorporado maquinaria poco a poco hasta llegar a tener una planta de producción completa en todas sus áreas, 5 computadores Apple, 5 computadores PC, 1 impresora full color (docucolor 12), 1 impresora de placas, 4 impresoras B/N, 1 duplicadora; en producción cuentan con 2 offset monocolor, 1 offset Heidelberg Bi-color, 1 plotter de impresión, 2 guillotinas, 4 máquinas pequeñas de acabados entre otros.

En cuanto a recursos humanos se refiere de igual manera se ha notado el crecimiento que ha tenido DIKAPSA empezando con 2 personas hasta llegar a un equipo de cerca de 18 personas en 4 áreas estratégicas.

Es una empresa que actualmente se dedica a brindar los servicios de impresiones y soluciones gráficas al norte del país, como son diseño e impresión de afiches promocionales, logotipos, carpetas, sobres, trípticos, agendas, calendarios, libros, revistas, catálogos, gigantografías, señalética, facturas, tarjetas de presentación y hojas membretadas, cubriendo una buena porción del mercado con productos impresos de gran calidad, generando valor en todos sus clientes a través de una atención personalizada y servicio de excelencia.

1.2.1. Misión.

Proveer soluciones con productos impresos de gran calidad, generando valor en todos nuestros clientes a través de una atención personalizada y servicio de excelencia.

Capítulo 1: Situación Actual

1.2.2. Visión.

Ser la empresa líder en el mercado gráfico de la provincia de Imbabura, captando empresas significativas, entregando productos de calidad y brindando un servicio de excelencia a través de desarrollo de nuestro equipo de trabajo y la adquisición de recursos tecnológicos de vanguardia.

1.2.3. Organización de la Empresa.

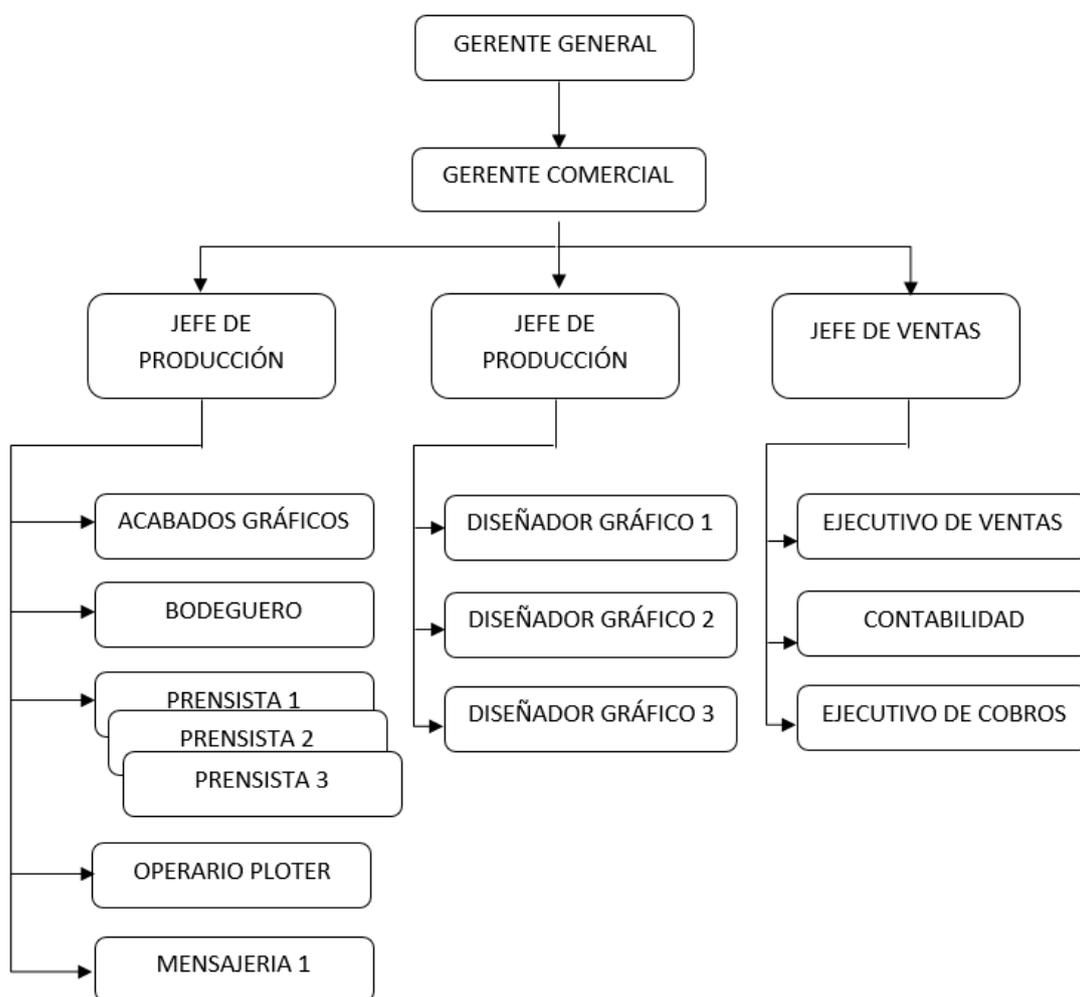


Figura 1. 1 Estructura Orgánica de la Empresa.

Fuente: DIKAPSA (2015) Recuperado de: <http://www.dikapsa.com/>

Capítulo 1: Situación Actual

1.2.4. Productos y Servicios que ofrece DIKAPSA.

1.2.4.1. *Material Promocional.*

En su línea promocional, elabora todo el material publicitario que su empresa requiere para generar gran impacto en el mercado y sobre todo en sus clientes, como son:

- Folletos
- Afiches
- Plegables (trípticos)
- Agendas pasta dura
- Calendarios
- Etiquetas
- Material POP

1.2.4.2. *Material Institucional.*

Cuenta con la tecnología necesaria para garantizarle productos de gran calidad, elaboradas con los mejores materiales y con acabados gráficos de excelencia que darán un distintivo único a su empresa, siempre con la calidad de DIKAPSA, como son:

- Carpetas
- Hojas membretadas
- Sobres corporativos
- Papelería Institucional
- Tarjetas de presentación
- Documentos Institucionales

Capítulo 1: Situación Actual

1.2.4.3. *Material Editorial.*

Elabora su material editorial cuidando cada detalle, lo que le permite llegar más lejos y ofrecer la mejor impresión de:

- Libros
- Revistas
- Catálogos
- Anuarios
- Periódicos

1.2.4.4. *Gigantografías.*

La imagen a gran escala, hoy en día es una estrategia eficaz al momento de realizar un manejo de marca efectivo, logra captar la atención del consumidor y genera una gran atracción de su imagen corporativa. Imprime en gran formato a todo color en calidad fotográfica para interior y exterior.

- Decoración de locales
- Micro perforados
- Rotulación
- Señalética
- Banner
- Lonas – Vallas
- Backlights
- Exhibidores

Capítulo 1: Situación Actual

1.2.4.5. Comprobantes (Libretines).

Son una empresa autorizada por el SRI, para de esta manera evitar sanciones y multas, cuenta con el servicio de impresiones de:

- Facturas
- Retenciones
- Guías de remisión
- Notas de venta
- Boletos

1.3. Situación Actual de la Empresa DIKAPSA

La empresa DIKAPSA, actualmente maneja todos sus procesos en las varias líneas de producción de forma manual, obteniendo como resultado deficiencias en algunas de las órdenes y el desconocimiento de la etapa del estado de manufactura en el que se encuentran, por lo que impide determinar con precisión que órdenes se encuentran pendientes por entregar y en algunos casos existen trabajos que son entregados con retrasos causando malestar e insatisfacción en sus clientes.

1.4. Problema

Actualmente el registro del proceso de producción en la empresa gráfica DIKAPSA de la ciudad de Otavalo, se lo realiza de forma manual, teniendo como principales inconvenientes las pérdidas y retrasos en las órdenes en la etapa de producción.

El alto índice de órdenes de producción provoca confusiones y el desconocimiento del estado en el que se encuentra dicha orden; es otro de los inconvenientes, ya que surgen retrasos en las entregas de los trabajos.

Capítulo 1: Situación Actual

Una gran desventaja de tomar los pedidos de forma manual, es no poder generar reportes estadísticos, que sirvan como referencia al Gerente para conocer el rendimiento de ventas generadas y datos generales de producción de la empresa.

Durante el proceso manual que se lleva actualmente, no se puede monitorear el estado de las órdenes pendientes o finalizadas y mucho menos tener un seguimiento de la ubicación exacta de la etapa de producción en la que se encuentra.

La empresa DIKAPSA no cuenta con ningún sistema, que le permita solucionar los inconvenientes mencionados anteriormente en la varias líneas de producción, por lo que se ha visto la necesidad de crear un sistema que cumpla estos requerimientos, evitando que los procesos se sigan llevando manualmente y así evitar el consumo excesivo de un recurso natural como es el papel.

1.5. Objetivos del Proyecto

1.5.1. Objetivo General.

Implementar un sistema de gestión de procesos de producción para la empresa DIKAPSA, aplicando una metodología de desarrollo de software.

1.5.2. Objetivos Específicos.

- Determinar el alcance de la situación actual de la empresa DIKAPSA y presentar una alternativa de solución para el problema planteado.
- Construir el marco teórico para el desarrollo del proyecto usando la metodología SCRUM, para construir el sistema de procesos de producción y definir su arquitectura.
- Realizar el levantamiento del proceso de producción que se maneja actualmente en la empresa DIKAPSA, únicamente de dos de las líneas de producción.

Capítulo 1: Situación Actual

- Implementar el sistema de gestión de procesos en la empresa DIKAPSA y su puesta en producción.
- Realizar un análisis de impacto para conocer el valor aproximado del proyecto final y sus beneficios.

1.6. Justificación

En la actualidad se requiere de un sistema que automatice los procesos de órdenes de trabajo como etapa preliminar se diseñará únicamente para dos líneas de producción como son las gigantografías y libretines, por su alta demanda en ventas para obtener un rendimiento eficiente en las líneas de manufactura.

La empresa de diseño e impresión DIKAPSA es una empresa privada que en los últimos años ha ido creciendo notablemente debido a la demanda de dichos productos brindando más fuentes de trabajo y entregando trabajos de calidad sin demoras, es por ello que se requiere de manera inmediata la implementación del sistema propuesto en este proyecto.

El presente proyecto tiene también la finalidad de poner en práctica los diferentes conocimientos adquiridos durante el transcurso de la formación académica en la Carrera de Ingeniería en Sistemas Computacionales, así mismo como previo requisito para la obtención del título de ingeniería.

1.7. Alcance

La empresa DIKAPSA contará con procesos automatizados que nos permitirán la toma de órdenes de producción de manera eficaz y estandarizada incluyendo informes detallados el registro de la hora de entrada y salida de la orden en cada fase de producción que son: Ventas, Diseño, Impresión, Acabados y Entrega.

Capítulo 1: Situación Actual

Para la implementación de este proyecto se realizará el levantamiento de dos procesos de las líneas de producción con mayor demanda en el empresa que son: gigantografías y Comprobantes “libretines”.

Haciendo el uso de la metodología ágil de desarrollo de software SCRUM, se construirá el sistema de procesos de producción en la empresa DIKAPSA y la documentación necesaria para una futura escalabilidad del sistema.

Definir la Arquitectura del Sistema de Gestión de Procesos de Producción para la empresa DIKAPSA. Adicionalmente se implementará un servidor de aplicaciones para el debido funcionamiento del sistema.

Para el desarrollo del sistema se hará uso de los frameworks de JAVA; la base de datos se implementará bajo la plataforma de software libre PostgreSQL, posteriormente se procederá a la fase de implantación, pruebas y finalmente puesta en producción en las instalaciones de la empresa DIKAPSA.

El análisis de impacto nos permitirá conocer el costo aproximado de la implementación del proyecto presentado y su rentabilidad económica, ambiental y general, además de las respectivas conclusiones y recomendaciones pertinentes del proyecto.

Capítulo 2

MARCO TEÓRICO



El capítulo 2 contempla las bases de conocimientos previos para la posterior implementación del proyecto. El aplicativo se basa en el lenguaje de programación Java, el entorno de desarrollo integrado Eclipse, API de persistencia JPA, motor de base de datos PostgreSQL y como herramienta para la generación de reportes de usuario final se utilizará iReports. Además se manejará la metodología de desarrollo SCRUM para mantener software de calidad.

El sistema se implementará y desarrollará bajo las herramientas totalmente libres y de código abierto. De esta forma el sistema estará desarrollado en el Lenguaje de Programación Java EE, y su entorno de desarrollo Eclipse Luna, el mapeo objeto relacional Java Persistence API, para el motor de la base de datos PostgreSQL. Las Aplicaciones de Internet Enriquecidas RIA, para la visualización y generación de reportes se usará Jasper Report, como guía y metodología de desarrollo de software del sistema se utilizará SCRUM ya que es una metodología de desarrollo ágil y además ofrece un conjunto de técnicas y nos permite generar software de calidad.

Capítulo 2: Marco Teórico

2.1. Metodología de Desarrollo Ágil SCRUM

2.1.1 Introducción

Las metodologías y estándares utilizados en el desarrollo de software nos proporciona una guía para conocer la línea que se debe recorrer desde sus inicios, e incluso antes de empezar la implementación, mediante la cual nos certifica la calidad del producto final, así como también el cumplimiento en la entrega del mismo en el tiempo establecido.

La palabra SCRUM procede del vocabulario del rugby y significa melé; es decir en la que los compañeros del equipo se amontonan y empujan todos en la misma dirección, por todas sus funcionalidades ha sido utilizado por varias empresas de renombre como: Microsoft, Yahoo, Google, Electronic Arts, Studio, Philips, Siemens, Nokia, BBC, Lexis. (SCRUM, 2015)

2.1.2 Características

SCRUM es un marco de trabajo ágil que se caracteriza por (SCRUM, 2015):

- Metodología de trabajo ágil.
- Diseñada para acortar el ciclo de desarrollo.
- Consigue una mejor aproximación entre las funcionalidades del software y los requerimientos del cliente evitando el papeleo innecesario.
- Manejo más eficiente de los requerimientos cambiantes en un proyecto.
- Mejorar la comunicación entre el cliente y el equipo desarrollador.
- Aumento de la productividad.
- Desarrollo incremental e iterativo producción frecuente de prototipos para evaluación del cliente.
- Nos dice “Qué” hacer sino “Cómo” hay que hacer las cosas.

Capítulo 2: Marco Teórico

2.1.3 Roles de Equipo o SCRUM TEAM

En cualquier proyecto como en los proyectos ágiles, es importante el establecimiento de equipos, donde se establecen roles y responsabilidades para cada integrante. Es así que SCRUM clasifica a todas las personas que intervienen o tienen interés en el desarrollo del proyecto en: dueño del producto, equipo y gestor de SCRUM (Deemer, 2012).



Figura 2. 1 Diagrama de la arquitectura

Fuente: Deemer, P. (2012). Información Básica de SCRUM. Recuperado de: http://www.scrumprimer.org/primers/es_scrumprimer20.pdf

2.1.3.1 El Dueño del Producto o Product Owner.

Es la persona encargada de gestionar las necesidades que serán satisfechas por el proyecto como son: recolectar las necesidades del usuario, gestionar y ordenar las necesidades de los usuarios, aceptar el producto software al finalizar cada sección y maximizar el entorno del proyecto. (Deemer, 2012)

Capítulo 2: Marco Teórico

2.1.3.2 El Equipo de Desarrollo o Develop Team.

Está formado por los desarrolladores que convierten las necesidades del dueño del producto en un conjunto de nuevas funcionalidades, modificaciones o incrementos del producto software final; el equipo tiene características especiales que se detallan a continuación:

- **Autogestionado:** el mismo equipo supervisa su trabajo y no existe un jefe de proyecto.
- **Multifuncional:** no existe especialistas en cada área, todos los integrantes del equipo pueden encargarse de tareas de programación, pruebas, despliegues.
- **No distribuidos:** el medio físico debe ser el mismo ya que facilita la comunicación de los integrantes.
- **Tamaño Óptimo:** sin tomar en cuenta al Product owner y SCRUM master, un equipo de desarrollo SCRUM debe estar compuesto por al menos tres personas y máximo nueve; ya que la productividad del trabajo decae si son menos de tres y si son más de nueve es difícil la autogestión.

2.1.3.3 El SCRUM Master

Es el responsable de asegurar que el equipo siga las prácticas de SCRUM; sus funciones son: Liderar el equipo SCRUM, ayudar a la autogestión del equipo, gestionar e intentar resolver los impedimentos con los que el equipo se encuentre (Deemer, 2012).

2.1.4 Ciclo de Vida del Proyecto

2.1.4.1 El Sprint

Una de las bases de los proyectos ágiles es el desarrollo mediante las iteraciones incrementales por lo que SCRUM a cada iteración se la llama Sprint. SCRUM recomienda

Capítulo 2: Marco Teórico

que las iteraciones deben ser cortas entre 1 y 4 semanas además como resultado debe entregar un software potencialmente entregable y operativo.

El equipo de desarrollo selecciona las historias de usuarios que se van a desarrollar en el Spring conformando la pila del Spring o Spring Backlog, una de las implementaciones que más lleva a cabo es crear un desglose de tareas normalmente representadas en una tabla, donde se describe cómo se va a implementar.

Al realizar las tareas se dividen en horas donde ninguna tarea debe durar más de 16 horas al ser realizada por un integrante, además la lista de tareas se mantendrá inmóvil durante toda la iteración.

Es muy complicado realizar una división de tareas a partir de las historias de usuarios elegidas para el Spring. Por lo que se puede relacionar el Product Backlog, el Spring Backlog y el Spring como se indica en la siguiente figura (Palacio, 2015).

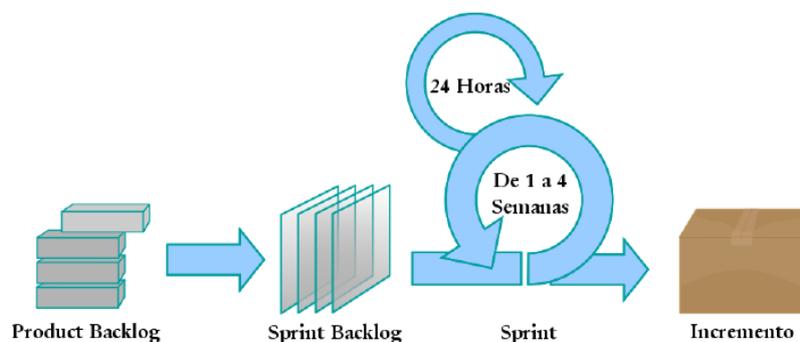


Figura 2. 2 Ejemplo de proceso SCRUM / continuo.

Fuente: Palacio, J. (2015) Gestión técnica de proyectos con SCRUM.

2.1.4.2 Reunión de Planificación del Sprint o Sprint Planning Meeting

Se lo lleva a cabo al principio de cada Sprint, participan todos los roles, el product owner presenta el conjunto de historias de usuario en el product Backlog y el equipo de

Capítulo 2: Marco Teórico

desarrollo selecciona las historias de usuario sobre las que se trabajará. No debe ser mayor de 8 horas.

2.1.4.3 Objetivo del Sprint o Sprint Goal.

Es una meta establecida para el Sprint que puede ser alcanzada mediante la implementación de la Lista de Producto, brindando una guía al Equipo de Desarrollo de por qué se está construyendo el incremento.

2.1.4.4 Reunión Diaria o Daily SCRUM

Tiene una duración de máximo 15 minutos, sus participantes son el equipo de desarrollo y el SCRUM master; donde cada miembro presenta lo que se realizó el día anterior y lo que se va a hacer el día de hoy.

2.1.4.5 Revisión Del Sprint o Sprint Review Meeting.

Se lo realiza al final del Sprint, participan el equipo de desarrollo, el SCRUM master y el product owner; se indica que ha podido completarse y que no, el product owner verifica el incremento del producto y actualización con nuevas historias de usuario. Tiempo máximo de 4 horas.

2.1.4.6 Retrospectiva del Sprint o Sprint Retrospective.

Se realiza al final del Sprint sirve para que los integrantes del equipo SCRUM y el SCRUM Master den sus recomendaciones sobre el Sprint que acaba de terminar. No debe durar más de 4 horas y se utiliza para mejorar el proceso y se trabaja en dos columnas, una con aspectos positivos y negativos del Sprint.

Capítulo 2: Marco Teórico

2.1.5 Artefactos de SCRUM

Representan trabajo o valor en diversas formas que son útiles para proporcionar transparencia y oportunidades, los artefactos definidos por SCRUM están diseñados para maximizar la transparencia de la información clave, para asegurar que todos tengan el mismo entendimiento.

2.1.5.1 *Lista de Producto o Product Backlog*

También llamado pila de producto, es un elemento fundamental de SCRUM que permite tener una única visión durante todo el proyecto; consiste en un listado de historias del usuario que se incorporarán al producto de software a partir de incrementos sucesivos. El responsable de ordenar el product backlog es el product owner.

Un product Backlog cuenta con cuatro cualidades que se detallan a continuación:

- **Detallado adecuadamente:** dependiendo de la prioridad se le destina el grado de detalle; tomando en cuenta que las historias de usuario de mayor prioridad se describen con más detalle.
- **Estimado:** las estimaciones se expresan en días ideales o en términos abstractos, tomando en cuenta que el tamaño de los elementos del product Backlog ayuda a darle prioridad y a planificar los siguientes pasos.
- **Emergente:** los nuevos elementos se descubren y se agregan a la lista teniendo en cuenta los comentarios de los clientes y usuarios, así mismo otros elementos podrán ser modificados o eliminados.
- **Priorizado:** los elementos más importantes y de mayor prioridad aparecen en la parte superior de la lista. Puede o no ser priorizado todos los elementos en un primer momento.

Capítulo 2: Marco Teórico

Los diagramas, reglas, resúmenes y prototipos no sustituyen al product Backlog sino que explican su contenido.

2.1.5.2 *Lista de Pendientes del Sprint (Sprint Backlog)*

Es el conjunto de elementos de la lista de producto seleccionados para el Spring, más un plan para entregar el incremento de producto y conseguir el objetivo. Las listas de pendientes es una predicción hecha por el equipo de desarrollo acerca de una nueva funcionalidad y del trabajo necesario para entregar esa funcionalidad en un incremento “Terminado” (Palacio, 2015).

2.1.5.3 *Incrementos*

Es la suma de todos los elementos de la lista de productos completados durante un sprint y el valor de todos los Sprints anteriores. El incremento debe estar en condiciones de utilizarse o sea que cumpla la definición de “Terminado”.

- **Incremento iterativo:** Este tipo de incrementos está basado en iteraciones de tiempos prefijados para el desarrollo “timeboxing”.
- **Incremento continuo:** Este tipo de incrementos que usa técnicas que le hacen lograr un flujo continuo en las funcionalidades del producto, estos tipos de incrementos permiten que se entregue de forma continua al cliente las partes del producto.

Capítulo 2: Marco Teórico

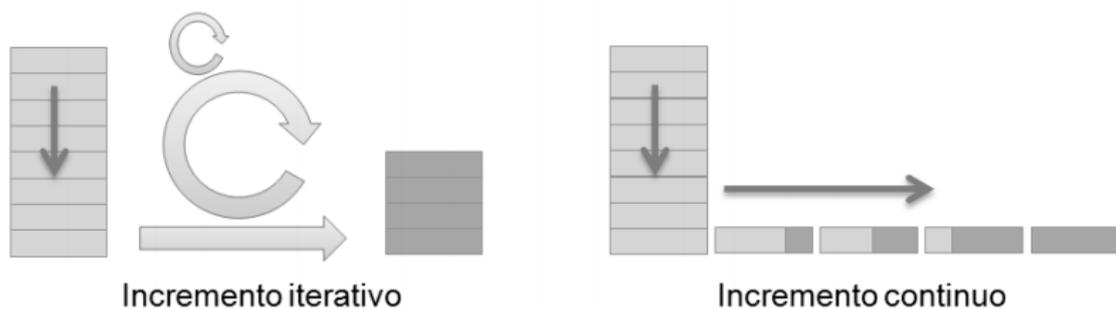


Figura 2. 3 Incremento Iterativo / continuo

Fuente: Palacio, J. (2015) Artículo Gestión técnica de proyectos con SCRUM

2.1.6 Ventajas

- Se obtiene software lo más rápido posible y este cumple con los requerimientos más importantes.
- Se trabaja en iteraciones cortas, de alto enfoque y total transparencia.
- Se acepta que el cambio es una constante universal y se adapta el desarrollo para integrar los cambios que son importantes.
- Se incentiva la creatividad de los desarrolladores haciendo que el equipo sea auto administrado.
- Se mantiene la efectividad del equipo habilitando y protegiendo un entorno libre de interrupciones e interferencias.
- Permite producir software de una forma consistente, sostenida y competitiva.

2.1.7 Desventajas

- Requiere delegar responsabilidades al equipo, incluso permite fallar si es necesario.
- Es una metodología que difiere del resto, y esto causa cierta resistencia en su aplicación para algunas personas.

Capítulo 2: Marco Teórico

2.2 Lenguaje de Programación JAVA

2.2.1 Introducción

Ángel Esteban (2010) afirma que:

Java es un lenguaje de programación orientado a objetos de propósito general, nace a partir de los años 1990 cuando un equipo de la compañía SUN MICROSYSTEMS investigaba, en sus principios se enfocaba más para pequeños dispositivos electrónicos, éstos dispositivos móviles cambiaban y se innovaban constantemente con chips más potentes y había que adaptar a sus características en la plataforma que debían funcionar para explotar su potencial; de esta forma surge la necesidad de crear un nuevo lenguaje de programación que permita desarrollar programas independientes del tipo de plataforma como son: calculadoras, relojes, equipos de música; la primera versión de este nuevo lenguaje se denominó Oak (roble), pero más tarde Sun descubrió que este nombre estaba ya registrado y lo tuvieron que cambiar, el nuevo nombre fue Java (una de las versiones sobre el significado del nombre es que Java es un término popularmente empleado en California para designar café de buena calidad. (p. 10).

2.2.2 Componentes

La plataforma Java está conformada por tres importantes componentes básicos que son:

- **El Lenguaje:** Un lenguaje de propósito general que emplea el modelo de programación orientado a objetos.

Capítulo 2: Marco Teórico

- **La Máquina Virtual:** Todos los programas Java se compilan a un código intermedio interpretado por un máquina virtual Java (JVM), lo que permite su portabilidad.
- **Bibliotecas:** El conjunto de Interfaz de programación de aplicaciones API's que proporcionan herramientas para el desarrollo.

2.2.3 Ediciones

Java dispone de 3 ediciones diferentes enunciadas a continuación:

- **Java ME:** Es una Edición Micro creada para ejecutar aplicaciones en dispositivos móviles.
- **Java SE:** Para aplicaciones en general como son de Escritorio.
- **Java EE:** Para aplicaciones corporativas o empresariales.

2.2.4 Características

El lenguaje java nos ofrece varias características por las cuales los programadores que desarrollan software se inclinan por este lenguaje:

- **Simple:** La funcionalidad que ofrece es alta, Java está desarrollado y basado en los lenguajes más conocidos como son C, C++. Por tal forma Java se diseñó para ser parecido a C++ y así permitir a los desarrolladores un fácil y rápido aprendizaje.
- **Orientado a Objetos:** Java en la mayoría trabaja con los datos como objetos y con interfaces a dichos objetos. También incluye tres características propias de la orientación a objetos como son: Herencia, Encapsulación y Polimorfismo.
- **Distribuido:** Java permite realizar interconexiones TCP/IP, de esta forma permite abrir puertos y establecer conexiones con ordenadores remotos; que ayudan a

Capítulo 2: Marco Teórico

interactuar con protocolos HTTP y FTP, lo que facilita a los programadores crear sus aplicaciones distribuidas.

- **Robusto:** Su rigidez le hace altamente fiable en comparación a otros lenguajes como C, este realiza un análisis en tiempo real durante la compilación como en la ejecución lo que le permite la detección de problemas y errores lo antes posible.
- **Seguro:** En el lenguaje Java la seguridad es una característica importante y que este trae implementado algunos escudos de seguridad en el lenguaje y el sistema de ejecución que actúan en tiempo real.
- **Arquitectura Neutral:** Java es indiferente a la arquitectura es decir es compatible con varios entornos de red. Es totalmente amigable con los sistemas operativos tales como con: Unix, Windows y Mac.
- **Portable:** Por el hecho que java es indiferente a cualquier arquitectura de sistema operativo hace que su portabilidad sea más fácil de forma que sus programas van a ser los mismos en cualquier plataforma lo que le convierte en eficiente gracias a su máquina virtual.
- **Interpretado y Compilado:** Un programa en lenguaje Java puede ser compilado e interpretado en tiempo real, de forma que cuando en desarrollado el código fuente este es transformado a una especie de código de máquina.
- **Multihebra o Multihilos:** Java permite tener varias actividades simultáneas en un programa, es decir cumple con muchas funciones al mismo tiempo, son básicamente pequeños procesos independientes de un gran proceso y cada hilo que el programa tenga se ejecutarán en tiempo real muchas funciones al mismo tiempo.
- **Dinámico:** Por ser orientado a objetos este hace uso de todos sus recursos posibles de dicha tecnología, lo que le permite trabajar de manera dinámica con sus clases.

Capítulo 2: Marco Teórico

- **Alto Rendimiento:** Los programadores lo prefieren por su alto rendimiento en el momento correr los programas y también por ahorrarse muchas líneas de código.

2.2.5 Requerimientos JDK

Por sus siglas en inglés Java Development Kit o Kit de desarrollo de Java, es un software que provee herramientas de desarrollo para la creación de programas en java. A continuación se detalla algunas de las utilidades que se pueden encontrar en el JDK:

- **Javac:** Es el compilador de Java. Se encarga de convertir el código fuente escrito en Java a byte - code.
- **Java:** Es el intérprete de Java. Ejecuta el byte - code a partir de los archivos .class.
- **Javadoc:** Se utiliza para crear documentación en formato HTML a partir del código fuente Java y los comentarios que contiene.
- **Javap:** Es un desensamblador de Java.

Para poder ejecutar cualquier aplicación Java en cualquier sistema operativo es necesario tener instalado el JRE o Java RunTime Environment - Entorno de desarrollo de Java. El JRE se compone de herramientas necesarias como la máquina virtual de java (java.exe) y el conjunto de librerías estándar de Java. El JDK incluye a JRE.

2.2.6 Java Virtual Machine (JVM)

La JVM es la que le proporciona la independencia de las plataformas a los programas compilados en el lenguaje Java. Ya que este genera un mismo fichero en byte-code para todos los programas que fueran compilados ya sea en distintas plataformas, de esta forma el

Capítulo 2: Marco Teórico

compilador de la JVM de Java se encarga de traducirlo el fichero generado a código ejecutable.

Es el entorno en el que se ejecutan los programas Java, su misión principal es la de garantizar la portabilidad de las aplicaciones Java. Se le llama "máquina virtual" porque, sin importar el tipo de máquina en la cual se esté ejecutando el programa, crea una máquina simulada que proporciona la plataforma correcta para ejecutar estas aplicaciones.

2.2.7 Etapas para la ejecución de un programa

Los siguientes definen los pasos para crear y ejecutar un programa Java:

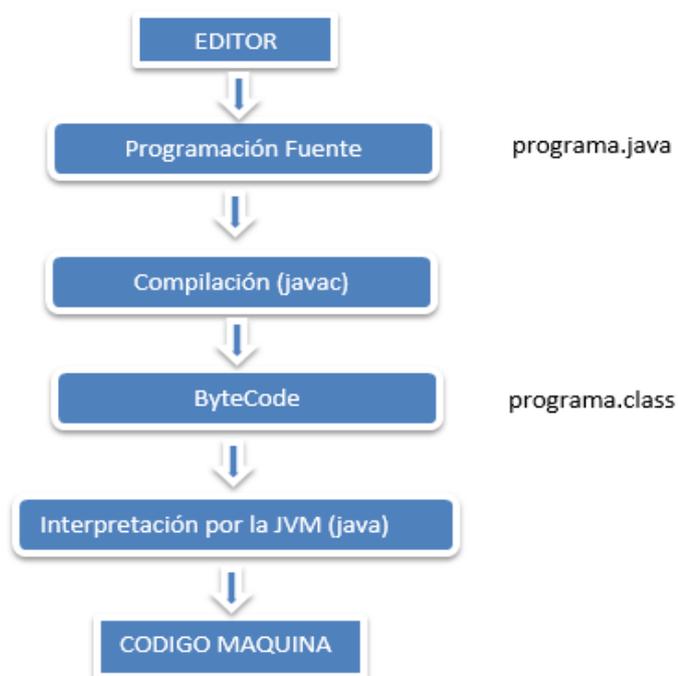


Figura 2. 4 Etapas para la ejecución de un programa Java.

Fuente: González, A. (2015) Introducción a Java

2.2.7.1 Editor

Consiste en crear un archivo de código fuente que contiene texto escrito en el lenguaje de programación Java. Este proceso se lo realiza utilizando cualquier IDE de Java

Capítulo 2: Marco Teórico

como por ejemplo eclipse y NetBeans. Al finalizar la edición, se guarda el nombre del archivo con la extensión .java.

2.2.7.2 *Compilación*

Con la compilación se convierte el código fuente a un código intermedio que la máquina virtual de Java puede entender, este conjunto de instrucciones recibe el nombre de bytecode y se guardan en un archivo cuya extensión es .class. Estas instrucciones son independientes de la plataforma en que se trabaje. Javac es el compilador de Java y se encarga de convertir el código fuente escrito en Java a bytecode.

2.2.7.3 *Interpretación*

La máquina virtual de Java es la encargada de interpretar el bytecode dando lugar a la ejecución del programa. El intérprete ejecuta cada una de las instrucciones bytecode en un ordenador específico ya sea Windows, Macintosh y más, en realidad no va a ejecutarse en ninguno de los sistemas operativos, sino en su propia máquina virtual. Solamente es necesario, por tanto, compilar una vez el programa, pero se interpreta cada vez que se ejecuta en un ordenador.

Cada intérprete Java es una implementación de la Máquina Virtual Java (JVM). Los bytecode posibilitan el objetivo de escribir el programa una vez y que se pueda correr en cualquier plataforma que disponga de una implementación de la JVM.

2.3. Entorno de Desarrollo Integrado IDE para JAVA (ECLIPSE LUNA)

2.3.1. Introducción

Eclipse fue desarrollado originalmente por IBM, ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro; Eclipse es una plataforma que

Capítulo 2: Marco Teórico

consiste en un Entorno de Desarrollo Integrado (IDE), es decir es un programa que se encuentra compuesto por herramientas útiles para un desarrollador de software.

Este programa trae consigo un editor de código, un depurador de errores, un compilador y un intérprete. Eclipse ha ganado popularidad por su versatilidad ya que se le puede añadir extensiones y así lograr una extensa funcionalidad, también tiene soporte hacia muchos lenguajes de programación C / C ++, Python, Perl, Ruby; pero con el que se ha destacado es precisamente es con el lenguaje Java.

2.3.2. Definición de Eclipse Luna

Eclipse es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama Aplicaciones de Cliente Enriquecido, opuesto a las aplicaciones Cliente-liviano basadas en navegadores.

2.3.3. Requisitos del Software

Para ejecutar el IDE Eclipse Java 7 JRE / JDK es necesario para la mayoría de las descargas de paquetes de Luna basado en Eclipse 4.4.

El IDE de Eclipse contiene su compilador Java personalizada tanto, un JRE es suficiente para la mayoría de las tareas con Eclipse. El JDK versión de Java sólo es necesario si se compila el código fuente de Java en la línea de comandos y de escenarios de desarrollo avanzadas, por ejemplo, si utiliza construye automático o si desarrolla aplicaciones web Java.

Capítulo 2: Marco Teórico

2.3.4. Ventajas

- El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos o plugins para proporcionar toda su funcionalidad al frente de la Plataforma de Cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
- Eclipse provee al programador con Frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de Software, Aplicaciones web, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional.
- El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código.

2.3.5. Características

Entre las principales características de Eclipse se encuentran:

- Herramienta de código abierto.
- Salvo el núcleo de la aplicación, todas las funcionalidades de Eclipse están desarrolladas como plugins.
- La funcionalidad del paquete básico puede ampliarse mediante la descarga de otros plugins, y además es posible desarrollar plugins personalizados.
- Proporciona una plataforma RCP para el desarrollo de aplicaciones de propósito general.

Capítulo 2: Marco Teórico

- Permite el desarrollo de aplicaciones para el servidor y servicios web.

2.3.6. Licencias

Plataforma Eclipse y otros plugins de Eclipse Foundation es liberado bajo la Licencia Pública Eclipse (EPL). EPL asegura que Eclipse es libre de descargar e instalar. También permite a Eclipse para ser modificado y distribuido.

2.3.7. Plataforma de Cliente Enriquecido

La base de eclipse es la plataforma de cliente enriquecido, está constituida por los siguientes componentes:

- Dispone de un editor de texto con resaltado de sintaxis donde se puede ver el contenido del fichero en el que se está trabajando.
- Contiene una lista de tareas y otros módulos similares
- La compilación es en tiempo real.
- Tiene pruebas unitarias con JUnit.
- Integración con Ant, asistentes (wizards) para creación de proyectos, clases y tests
- Es posible añadir un sistema de control de versiones a través de Subversion y a la vez lograr una integración mediante Hibernate.
- Los widget de eclipse están implementados por una herramienta de widget para Java llamada SWT, a diferencia de la mayoría de las aplicaciones Java, que usan las opciones estándar Abstract Window s Toolkit AWT o Swing.
- La interfaz de usuario de eclipse también tiene una capa GUI intermedia llamada Jface, la cual simplifica la construcción de aplicaciones basadas en SWT.

Capítulo 2: Marco Teórico

2.3.8. Compilar

Una de las características más curiosas del IDE eclipse es el modo en que se compilan de los proyectos. No existe en eclipse ningún botón que permita compilar individualmente un fichero en concreto. La compilación es una tarea que se lanza automáticamente al guardar los cambios realizados en el código. Por esta razón es prácticamente innecesarios controlar manualmente la compilación de los proyectos.

2.3.9. Ejecutar

Una vez compilado correctamente, ejecutar el proyecto es la parte más sencilla si el proyecto está correctamente programado. Prácticamente todas las opciones de ejecución se pueden manejar desde el botón “RUN” de la barra de herramientas principal.

El menú de ejecución tiene dos partes. La entrada “RUN AS” permite ejecutar directamente la clase que se está mostrando en la ventana del Editor activo, utilizando la configuración de ejecución por defecto y la entrada “RUN” permitirá definir nuevas configuraciones de ejecución.

2.4. Contenedor Web de Aplicaciones Apache TOMCAT

2.4.1. Descripción

Tomcat es un contenedor web basado en el lenguaje Java que actúa como motor de servlets y JSPs o Java Server Pages y puede ser utilizado como un producto independiente, o bien combinado con el Apache HTTP Server.

Capítulo 2: Marco Teórico

2.4.2. Características

- Tomcat es una aplicación de software de código abierto de las tecnologías Java Servlet y JavaServer Pages. Las diferentes versiones de Apache Tomcat están disponibles para diferentes versiones del Servlet y JSP especificaciones.
- Si la nueva versión de Java introduce nuevas características del lenguaje entonces el compilador por defecto JSP no puede apoyarlos inmediatamente. Cambio del compilador JSP para Java se puede habilitar estas nuevas características del lenguaje a utilizar en JSP.
- Tomcat es mantenido y desarrollado por miembros de la Apache Software Foundation y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software License.
- Tomcat es un servidor web con soporte de servlets y JSPs. Tomcat no es un servidor de aplicaciones, como JBoss o JOnAS. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets.
- Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones.

2.4.3. Definición de contenedor

Es la interfaz entre el componente y la plataforma sobre la que se ejecuta y que le facilita los servicios que éste necesita para su funcionamiento. Antes de que cualquier componente Web, EJB o de cliente pueda ser ejecutado, debe ser empaquetado en su correspondiente módulo JEE y desplegado en su contenedor.

Capítulo 2: Marco Teórico

2.5. Diseñador de Reportes

El Diseñador de Reportes permite crear proyectos, conexiones a bases de datos, definir grupos de usuarios y usuarios y asegurar el sistema de reportes. Permite la creación de reportes, hojas de datos, gráficas y tablas dinámicas. Puede distribuir sus reportes a través de archivos físicos de reportes los cuales pueden ser abiertos por los usuarios finales.

2.5.1. Diseñador iReport

Es el código abierto diseñador de informes libre para JasperReports y JasperReports Server. Crea diseños muy sofisticados que contienen gráficos, imágenes, subinformes, tablas de contingencia y mucho más. Acceda a sus datos a través de JDBC, TableModels, JavaBeans, fuentes XML, Hibernate, CSV y personalizados. Luego de publicar sus informes en PDF, RTF, XML, XLS, CSV, HTML, XHTML, texto, DOCX, u OpenOffice.

Aquí se utiliza la herramienta iReport para generar la plantilla JRXML. Este archivo es un archivo xml puro para describir la disposición y el lugar de los titulares. Entonces tenemos que compilar en un archivo de jaspe. Su formato binario. Este proceso de compilación que puede hacer de 2 maneras.

- Usando iReport para compilar el archivo jrxml en jaspe
- Usando la API JasperReport que podemos hacer a través del programa.

2.5.1.1. Características ireport

La lista siguiente describe algunas de las características importantes de IReport:

- 100% escrito en Java además OpenSource y gratuito.
- Maneja el 98% de las etiquetas de JasperReport.
- Permite diseñar con sus propias herramientas: rectángulos, líneas, elipses, campos de los textfields, cartas, subreportes.

Capítulo 2: Marco Teórico

- Soporta internacionalización nativamente.
- Recopilador y exportador integrados.
- Soporta JDBC, JavaBeans como orígenes de datos
- Incluye Wizard's o también llamados asistentes para crear automáticamente informes.
- Tiene asistentes para generar los subreportes y plantillas.
- Facilidad de instalación.

2.5.1.2. Prerrequisitos

Necesitará disponer de lo siguiente: Apache Ant, IReport y Jasperreport-0.x.x

2.5.1.3. Requerimientos de instalación

- Sun JDK 1.4 (SDK) o superior.
- Acrobat 5.0 no es requerido, pero es fuertemente recomendado.
- Si se desea conectar con una base de datos, se debe proporcionar el Driver JDBC correspondiente.
- Usar la versión IReport-0.5.1 o superior.

2.5.2. Jasper report

Es una herramienta de creación de informes que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML (Ramakrishnan, 2015).

Su principal propósito es crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. Se usa comúnmente con IReport, un front-end gráfico de

Capítulo 2: Marco Teórico

código abierto para la edición de informes. Está bajo GNU Lesser General Public License, por lo que es Software libre.

2.5.2.1. Características

Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de este tipo, incluyendo J2EE o aplicaciones web, para generar contenido dinámico.

2.5.2.2. Funciones

Es una biblioteca que puede ser embebida en cualquier aplicación Java. Sus funciones incluyen:

- **Scriptlets**, que pueden acompañar a la definición del informe, y pueden ser invocados en cualquier momento para realizar un procesamiento adicional. El scriptlet se basa en Java, y tiene muchos hooks o ganchos que se pueden invocar antes o después de las etapas de la generación de informes, como el Informe, Página, Columna o Grupo.
- **Sub-informes**: para usuarios con requisitos más sofisticados de gestión, los informes diseñados para JasperReports pueden ser fácilmente importados a JasperServer que es el servidor de informes interactivos.

JasperReports es una biblioteca que puede ser embebida en cualquier aplicación Java además para exportar a extensiones docx, xlsx, odt, ods, pptx se debe utilizar JRDocxExporter, JRXlsxExporter, JROdtExporter, JROdsExporter, JRPptxExporter.

Capítulo 2: Marco Teórico

2.5.2.3. *Librerías JasperReport*

- jasperreports-1.0.1.jar
- commons-digester.jar
- commons-beanutils.jar
- commons-collections.jar
- commons-logging.jar
- itext-1.02b.jar
- poi-2.0-final-20040126.jar

2.5.3. **Jasper server**

Es una aplicación web desarrollada en Java que permite generar reportes y acceder a ellos a través del explorador.

Provee bastante flexibilidad y es muy fácil de usar. Se pueden crear los reportes usando iReport, y cuando esté listo el .jrxml, se sube al JasperServer, se crean los parámetros y se configuran las fuentes de datos: jdbc, jdbc-odbc, en el JasperServer. También se pueden exportar los reportes en varios formatos distintos como pdf y xls.

2.5.3.1. *Elementos*

Arquitectura de servidor flexible, repositorio centralizado, informes Ad hoc y dashboards.

Capítulo 2: Marco Teórico

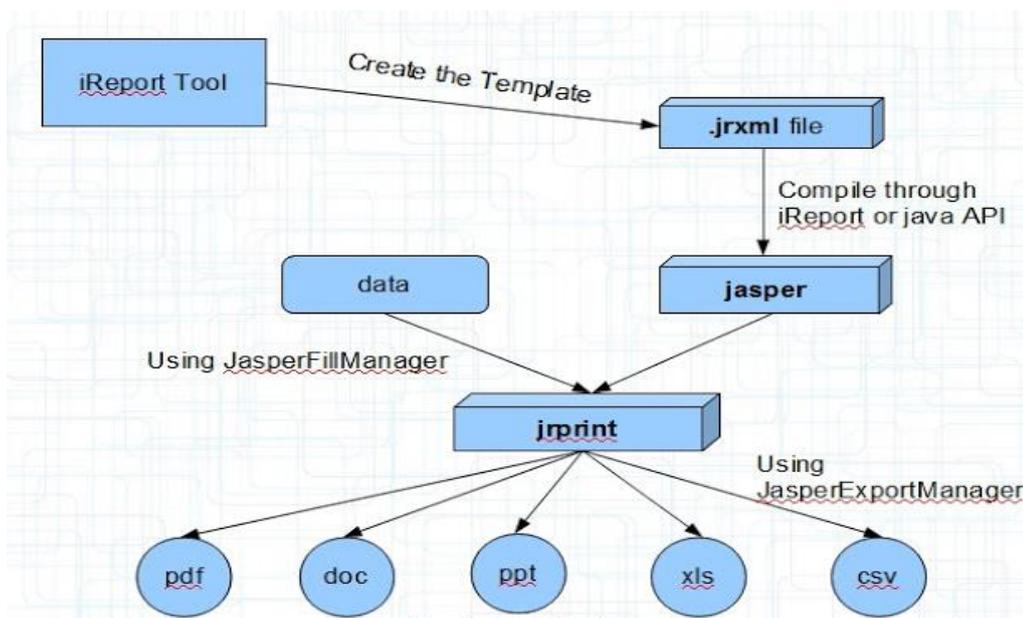


Figura 2. 5 Diagrama de flujo Jasper report

Fuente: Ramakrishnan, E. (2015) Recuperado de <http://www.ramkitech.com/2011/11/jsf-jpa-jasperreports-ireport.html>

2.6. PostgreSQL 9.3

2.6.1. Introducción a PostgreSQL

PostgreSQL es un lenguaje de administración de bases de datos, destacando sus partes, tablas, campos que al ser un excelente lenguaje de soporte para consultas con varios elementos recopilados al nivel web sirve para manejo y total fiabilidad al hacerse de inventarios de negocios. Se han creado Programas facilitando el uso tanto del PostgreSQL, SQL y Oracle a los más inexpertos pero con una eficiencia del 100%. (PostgreSQL-es, 2015).

Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

Capítulo 2: Marco Teórico

2.6.2. Características principales de PostgreSQL

A continuación se enumeran las más importantes características de este gestor de bases de datos:

- Implementa un motor que asegura la integridad de los datos.
- Está diseñado para soportar grandes volúmenes de datos.
- Es soportado por varios sistemas operativos como Windows y Linux, entre otros.
- Brinda la posibilidad de realizar transacciones, sub-selecciones, triggers, vistas y bloqueos sofisticados.
- Ofrece integridad referencial de claves externas.
- Implementación del estándar SQL92/SQL99.
- Soporta el uso de índices, reglas y vistas.
- Es una base de datos 100% ACID.
- Copias de seguridad en caliente (Online/hot backups)
- Múltiples métodos de autenticación
- Acceso encriptado via SSL
- Licencia BSD
- Disponible para Linux y UNIX y Windows 32/64bit.

2.6.2.2. Programación / Desarrollo

- Funciones/procedimientos almacenados (stored procedures) en numerosos lenguajes de programación, entre otros PL/pgSQL
- Bloques anónimos de código de procedimientos

Capítulo 2: Marco Teórico

- Numerosos tipos de datos y posibilidad de definir nuevos tipos. Además de los tipos estándares en cualquier base de datos, tenemos disponibles, entre otros, tipos geométricos, de direcciones de red, de cadenas binarias, UUID, XML, matrices.
- Soporta el almacenamiento de objetos binarios grandes: gráficos, videos, sonido.
- APIs para programar en C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, PHP, Lisp, Scheme, Qt y muchos otros.

2.6.3. Arquitectura

A continuación hay un gráfico que ilustra de manera general los componentes más importantes en un sistema PostgreSQL.

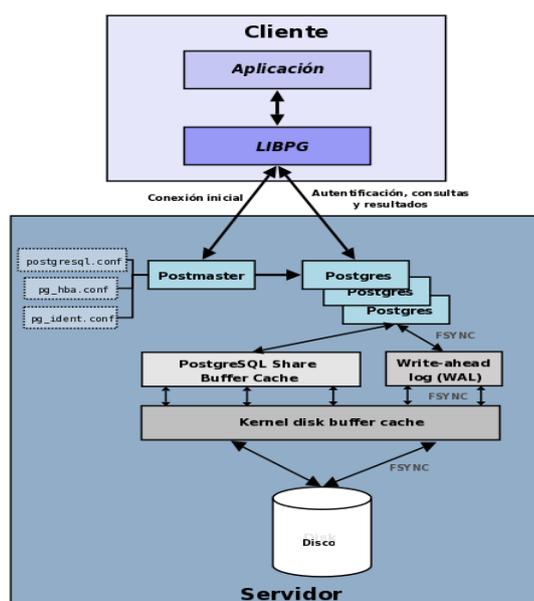


Figura 2. 6 Componentes de un sistema PostgreSQL

Fuente: PostgreSQL-es. (2015) Recuperado de http://www.postgreSQL.org.es/sobre_postgreSQL

- **Aplicación cliente:** Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir via TCP/IP o sockets locales.
- **Demonio postmaster:** Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es

Capítulo 2: Marco Teórico

el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes

- **Ficheros de configuración:** Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg_hba.conf y pg_ident.conf
- **Procesos hijos postgresQL:** Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes
- **PostgreSQL share buffer cache:** Memoria compartida usada por PostgreSQL para almacenar datos en caché.
- **Write-Ahead Log (WAL):** Componente del sistema encargado de asegurar la integridad de los datos.
- **Kernel disk buffer cache:** Caché de disco del sistema operativo.
- **Disco:** Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

2.6.4. Ventajas de PostgreSQL

- PostgreSQL es un motor de base de datos de software libre que proporciona un manejador de conexión nativa JDBC.
- PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD; su código fuente está disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

Capítulo 2: Marco Teórico

- Las características técnicas de esta base de datos la hacen una de las más potentes y robustas del mercado. Su desarrollo comenzó hace más de 15 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo.

2.6.5. Requerimientos para instalación

Tabla 2. 1

Requerimientos Básicos PostgreSQL

Límite	Valor
Máximo tamaño base de dato	Ilimitado (Depende de tu sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

Nota Recuperado de <http://es.slideshare.net/carlosgruiz.arahat/requerimientos-de-postgres-final-12880923>

2.7. JDBC

2.7.1. Introducción

Java Database Connectivity, más conocida por sus siglas JDBC, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice. (DevelopersBOOK.com, 2015)

JDBC permite que existan múltiples implementaciones y ser usado por la misma aplicación. La API proporciona un mecanismo para cargar dinámicamente los paquetes Java correctos y registrarlos con el Administrador de controladores JDBC. El Administrador de

Capítulo 2: Marco Teórico

controladores se utiliza como una fábrica de conexiones para la creación de conexiones JDBC.

Las instrucciones de actualización pueden ser tales como SQL de CREAR, INSERT, UPDATE y DELETE, o pueden ser instrucciones de consulta como SELECT. Además, los procedimientos almacenados pueden ser invocados a través de una conexión JDBC. JDBC representa declaraciones utilizando una de las siguientes clases:

- **Statement** - la declaración se envía al servidor de base de datos cada vez.
- **PreparedStatement** - la declaración se almacena en caché y luego la ruta de ejecución es pre-determinado en el servidor de base de datos que le permite ser ejecutado varias veces de una manera eficiente.
- **CallableStatement** - utilizado para la ejecución de los procedimientos almacenados en la base de datos.

Actualizar declaraciones como INSERT, UPDATE y DELETE devolver un recuento de actualización que indica cuántas filas fueron afectadas en la base de datos. Estas declaraciones no devuelven cualquier otra información.

Controladores JDBC están del lado del cliente adaptadores (instaladas en la máquina cliente, no en el servidor) que convierten las peticiones de los programas Java a un protocolo que el DBMS puede entender.

2.7.2. Tipos

Hay conductores comerciales y gratuitos disponibles para la mayoría de los servidores de bases de datos relacionales. Estos controladores caen en uno de los siguientes tipos:

Capítulo 2: Marco Teórico

- **Tipo 1:** que llama al código nativo del controlador ODBC localmente disponibles.
- **Tipo 2:** que llama proveedor de base de datos biblioteca nativa en un lado del cliente. Este código se habla con la base de datos sobre la red.
- **Tipo 3:** usa el controlador pure-java para el middleware del servidor que posteriormente habla con la base de datos.
- **Tipo 4:** el controlador pure-java que utiliza el protocolo nativo de base de datos.

Fundamentalmente, JDBC es una especificación que proporciona un completo conjunto de interfaces que permite el acceso portátil a una base de datos subyacente. Java se puede utilizar para escribir diferentes tipos de ejecutables, tales como:

- Aplicaciones Java
- Los applets de Java
- Los servlets Java
- Java ServerPages (JSP)
- Java Server Faces (JSF)
- Enterprise JavaBeans (EJB)

A continuación se presenta el diagrama de arquitectura, que muestra la ubicación del gestor de controladores con respecto a los controladores JDBC y la aplicación Java

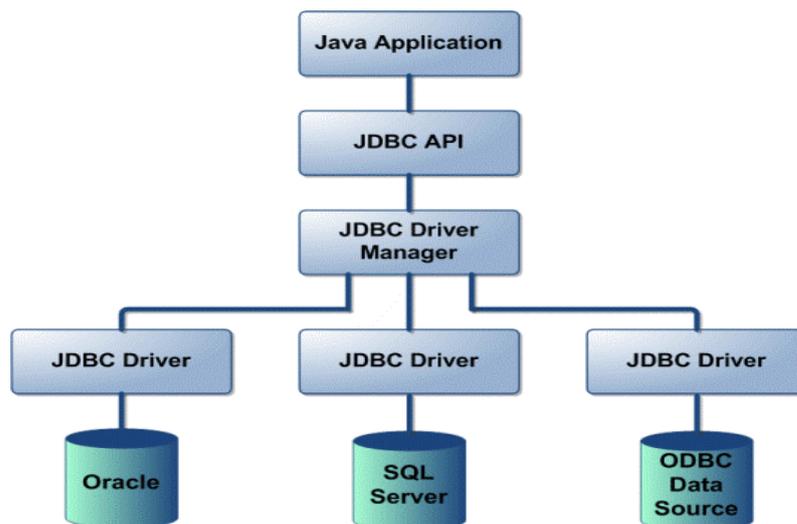


Figura 2. 7 Componentes JDBC comunes

Fuente: DevelopersBOOK.COM. (2015) Recuperado de <http://www.developersbook.com/jdbc/interview-questions/jdbc-interview-questions-faqs.php>

2.7.3. Interfaces y clases

El API JDBC proporciona las siguientes interfaces y clases:

- **DriverManager:** Esta clase maneja una lista de controladores de bases de datos. Coincide con solicitudes de conexión de la aplicación Java con el conductor de base de datos adecuada utilizando el protocolo de comunicación sub. El primer piloto que reconoce un cierto sub protocolo bajo JDBC se utiliza para establecer una conexión de base de datos.
- **Conductor:** Esta interfaz se encarga de las comunicaciones con el servidor de base de datos. Va a interactuar directamente con los objetos del controlador muy raramente. En su lugar, utiliza objetos DriverManager, que gestiona los objetos de este tipo.
- **Conexión:** Esta interfaz con todos los métodos para ponerse en contacto con una base de datos. El objeto de conexión representa contexto de comunicación, es decir, toda la comunicación con la base de datos es a través de único objeto de conexión.

Capítulo 2: Marco Teórico

- **Declaración:** Utiliza objetos creados a partir de esta interfaz para enviar las sentencias SQL para la base de datos. Algunas interfaces derivadas aceptan parámetros además de la ejecución de los procedimientos almacenados.
- **ResultSet:** Estos objetos contienen datos recuperados de una base de datos después de ejecutar una consulta SQL usando objetos Statement. Actúa como un repetidor para permitir que usted se mueva a través de sus datos.
- **SQLException:** Esta clase se encarga de todos los errores que se producen en una aplicación de base de datos.

2.8. Interfaz de Persistencia Java (JPA)

2.8.1. Introducción

Un marco de trabajo o framework brinda ayuda en las distintas áreas del proceso de desarrollo de una aplicación. Los frameworks están diseñados para permitir el desarrollo fácil y rápido de las aplicaciones. Gracias a esto, los programadores ya no tendrán que preocuparse por manejar programación de bajo nivel, como por ejemplo: concurrencia masiva, manejo de transacciones, seguridad, entre otros. (Anónimo, 2015)

También llamado por sus siglas Java Persistence API (JPA), este proporciona un modelo de persistencia basado en POJO's para mapear bases de datos relacionales en Java. La persistencia de Java fue desarrollada por expertos de EJB 3.0 como parte de JSR 220, aunque su uso no se limita a los componentes software EJB. Se puede utilizar en aplicaciones web y aplicaciones clientes. (Tutorials Point, 2015)

Capítulo 2: Marco Teórico

2.8.2. Definición JPA Entity Manager

El API JPA provee el Application-managed entity manager para las situaciones en donde no requerimos los servicios ofrecidos por un EJB3 Container, pero queremos utilizar el modelo de persistencia de JPA. En estos casos las aplicaciones son denominadas aplicaciones standalone. Estas aplicaciones que corren fuera de un EJB3 container usaran el resource-local transaction provisto por el entity manager, lo que permite que la aplicación será la encargada de manejar el ciclo de vida del entity manager.

El API JPA provee el Container-managed entity manager para las situaciones en donde sí se requiera de los servicios de un EJB3 container como por ejemplo JBoss o WebLogic application server.

Siempre que una transacción sea iniciada, un nuevo contexto de persistencia o persistence context es creado. Esto es así tanto para el Application-managed entity manager como también para el Container-managed entity manager.

2.8.3. Utilidades de JPA

Para reducir la carga de escribir códigos relacionales para gestión de objetos, un programador sigue el "Proveedor" marco JPA, que permite la fácil interacción con instancia de la base de datos. Aquí el marco necesario se realiza a través de JPA.

Capítulo 2: Marco Teórico

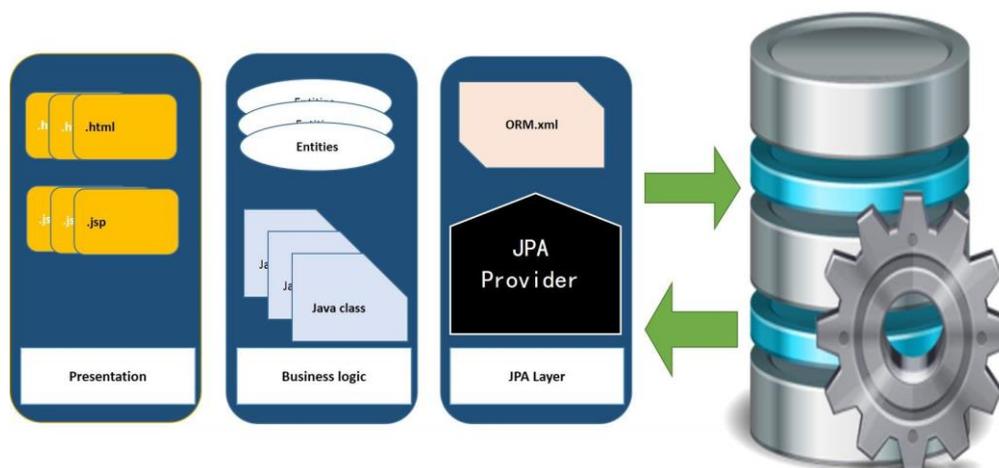


Figura 2. 8 Diagrama de funcionamiento.

Fuente: Tutorials Point. (2015) Recuperado de http://www.tutorialspoint.com/es/jpa/jpa_introduction.htm

JPA utiliza muchas interfaces y tipos de anotaciones definidas en el paquete de `javax.persistence` disponible con la versión 5 de *Java EE*. JPA utiliza las clases de entidad que se asignan a las tablas de la base de datos. Estas clases de entidad se definen en las anotaciones de la JPA.

2.8.4. Requerimientos para la instalación

Conocimiento de programación Java con JDK 1.6 o posterior es un requisito previo para comprender este tutorial. Por otra parte, nosotros asumimos que los lectores están familiarizados con los conceptos de JDBC en Java.

Tabla 2. 2

Requisitos del sistema

JDK	Java SE 2 JDK 1.5 or above
Memoria	1 GB RAM (se recomienda)
Espacio en disco	No hay requisito mínimo
Versión del sistema operativo	Windows XP or above, Linux

Nota. Recuperado de <https://docs.oracle.com/javase/tutorial/jdbc/>

Capítulo 2: Marco Teórico

2.8.5. Arquitectura

La siguiente imagen muestra el nivel de clase arquitectura de JPA. Muestra las clases principales y las interfaces de JPA.

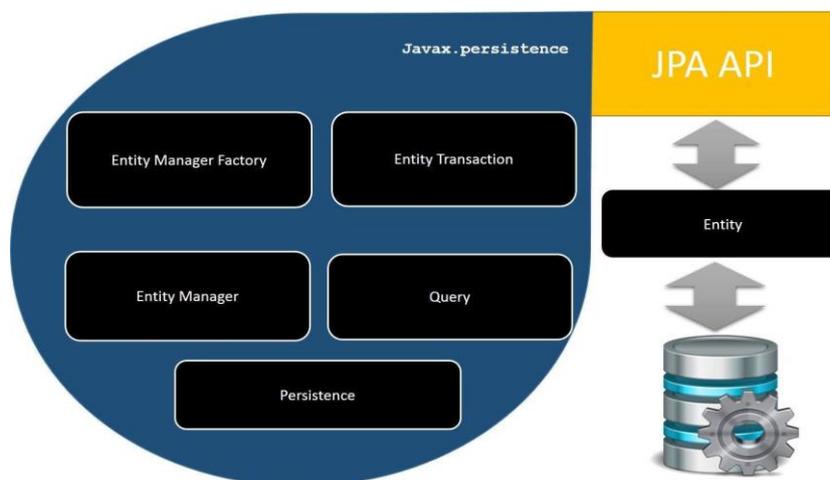


Figura 2. 9 Nivel de clase arquitectura de JPA

Fuente: Oracle. (2015) Recuperado de <http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html>

Tabla 2. 3

Unidades mostradas en la arquitectura

Unidades	Descripción
EntityManagerFactory	Esta es una clase de fábrica de EntityManager. Crea y gestiona múltiples instancias EntityManager.
EntityManager.	Es una interfaz, que gestiona la persistencia de objetos. Funciona como instancia de consulta.
Entidad	Las entidades son los objetos de persistencia, tiendan como registros en la base de datos.
EntityTransaction	Tiene una relación de uno a uno con EntityManager. Para cada método EntityManager, se mantienen las operaciones de EntityTransaction clase.
Persistencia	Esta clase contiene métodos estáticos para obtener EntityManagerFactory.
Consulta	Esta interfaz es implementada por cada proveedor JPA relacional para obtener objetos que cumplan los criterios.

Nota. Recuperado de <http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html>

Capítulo 2: Marco Teórico

Por encima de las clases y las interfaces se utilizan para almacenar entidades en una base de datos como un registro. Ayudan a los programadores por reducir sus esfuerzos por escribir los códigos para almacenar datos en una base de datos, de modo que puedan concentrarse en actividades más importantes, tales como escribir los códigos para la cartografía de las clases con tablas de la base de datos.

2.8.6. Object Relation Mapping (ORM)

Object Relation Mapping (ORM) informa brevemente sobre lo que es ORM y cómo funciona. ORM es una capacidad de programación para convertir los datos de tipo de objeto de tipo relacional y viceversa.

La principal característica de ORM es de cartografía o enlazar un objeto a sus datos en la base de datos. Si bien la cartografía, tenemos que tener en cuenta los datos, el tipo de datos, y sus relaciones con la entidad o entidades en cualquier otra tabla.

2.8.6.1. Funciones avanzadas

- **Persistencia idiomática:** le permite escribir la persistencia clases utilizando las clases orientadas a objetos.
- **Alto rendimiento:** tiene muchas técnicas de bloqueo y esperanzador.
- **Fiable:** es muy estable y utilizado por muchos programadores profesionales.

Capítulo 2: Marco Teórico

2.8.6.2. Arquitectura ORM

El ORM arquitectura es similar al siguiente.

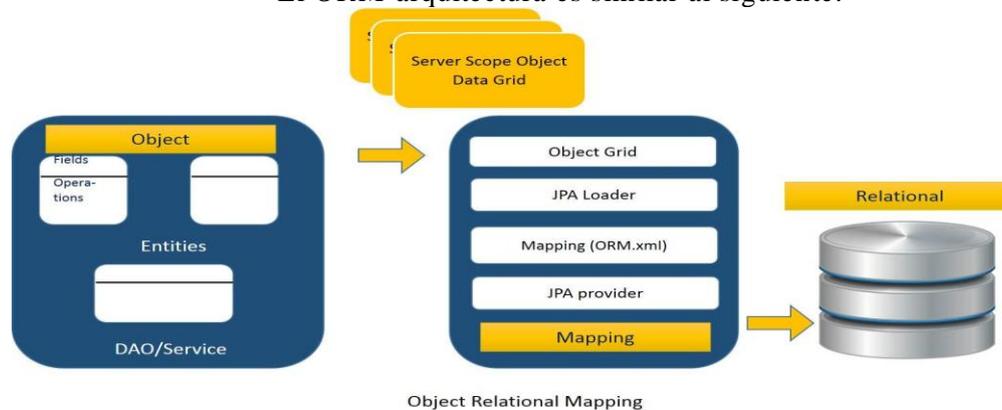


Figura 2. 10 ORM arquitectura

Fuente: Metodologías del Sistema. (2015) Recuperado de <http://metodologiasdesistemas.blogspot.com/2007/10/que-es-un-orm-object-relational-mapping.html>

La arquitectura se explica cómo datos del objeto se almacenan en bases de datos relacionales en tres fases.

- **Fase 1:** La primera fase, denominada fase de datos del objeto, contiene las clases POJO, interfaces y clases. Es el principal componente de empresa de capa, que tiene lógica de negocios operaciones y atributos. POJO empleado clase contiene atributos como ID, nombre, salario, y designación. También contiene método setter y getter de esos atributos. Empleado DAO/clases de servicio contienen métodos de servicio tales como crear empleado, encontrar los empleados y eliminar empleado.
- **Fase 2:** La segunda fase, denominada fase de mapeo o persistencia JPA proveedor, contiene, archivo de mapas (ORM.xml), JPA Cargadora y rejilla de objeto.
 - **JPA Provider:** es el producto de proveedor que contiene el JPA sabor (javax.persistence).
 - **Archivo de asignación:** El archivo de asignación (ORM.xml) contiene configuración de la asignación entre los datos de un POJO clase y los datos en una base de datos relacional.

Capítulo 2: Marco Teórico

- **JPA Cargador:** La APP funciona como una memoria caché. Puede cargar los datos relacionales. Funciona como una copia de base de datos para interactuar con las clases de servicio de datos POJO.
- **Rejilla de Objeto:** es una ubicación temporal que puede almacenar una copia de los datos relacionales, como una memoria caché. Todas las consultas en la base de datos se efectuará, primero en los datos del objeto grid. Sólo después de que se ha comprometido, que afecta a la base de datos principal.
- **Fase 3:** La tercera fase es la fase de datos relacionales. Contiene los datos relacionales que lógicamente está conectado al componente comercial. Solo cuando el componente comercial se compromete los datos, que se almacenan en la base de datos físicamente. Hasta entonces, los datos modificados se almacenan en una memoria caché como un formato de cuadrícula. El proceso de obtención de los datos es idéntica a la de almacenar los datos.

El mecanismo de la interacción mediante programación por encima de tres fases se denomina asignación objeto-relacional.

2.8.7. Anotaciones

Por lo general se utilizan archivos xml para configurar los componentes específicos, o asignación de dos diferentes especificaciones de los componentes. Al escribir un archivo de asignación xml, necesitamos comparar los atributos de clase POJO con las etiquetas de la entidad en el archivo mapping.xml.

Las anotaciones se utilizan para las clases, propiedades y métodos. Las anotaciones comienzan con "@" el símbolo. Las anotaciones son declaradas antes de una clase, propiedad o método. Todas las anotaciones de JPA se definen en el paquete javax.persistence.

Capítulo 2: Marco Teórico

Tabla 2. 4

Lista de anotaciones de ejemplo

ANOTACIÓN	DESCRIPCIÓN
@ Entidad	Declara la clase como una entidad o una tabla.
@ Tabla	Declara nombre de la tabla.
@ Basic	Especifica no campos de restricción explícita.
@ Embedded	Especifica las propiedades de la clase o de una entidad cuyo valor es una instancia de una clase se puede incrustar.
@ Id	Especifica la propiedad, el uso de la identidad (la clave principal de una tabla de la clase.
@ Columna	Especifica el atributo de columna para la propiedad persistence.
@ TableGenerator	Especifica el generador de valor para la propiedad especificada en la anotación @GeneratedValue. Crea una tabla de generación de valor.
@ JoinColumn	Especifica la entidad asociación o entidad colección. Esto se utiliza en muchos-a-uno y uno-a-muchas asociaciones.
@ ColumnResult	Hace referencia al nombre de una columna de la consulta SQL que utiliza cláusula select.
@ ManyToOne	Define una relación de many-to-one entre el unir tablas.
@ OneToOne	Define una relación one-to-one entre los unir tablas.
@ NamedQueries	Especifica la lista de consultas con nombre.

Nota. Elaborado por el Autor Carlos Burgos

2.8.8. Entidad

La entidad no es una cosa nueva en la gestión de datos. De hecho, las entidades han estado alrededor de muchos lenguajes de programación y, ciertamente, más de Java. Fueron introducidos por primera vez por Peter Chen en su artículo seminal sobre el modelado entidad-relación. Describió entidades como las cosas que tienen atributos y relaciones. La expectativa era que los atributos y las relaciones se persistían en una base de datos relacional.

Una entidad es esencialmente un sustantivo, o una agrupación de estado asociados juntos como una sola unidad. El estado de persistencia de una entidad se representa a través

Capítulo 2: Marco Teórico

de campos persistentes o propiedades persistentes. Estos campos o propiedades usan anotaciones para el mapeo de estos objetos en el modelo de base de datos.

El estado persistente de una entidad puede ser accesible a través de variables de instancia a la entidad o bien a través de las propiedades de estilo de JavaBean.

2.8.9. Persistencia

También llamado almacenamiento permanente, es una de las necesidades básicas de cualquier sistema de información de cualquier tipo, por lo que persistir objetos Java en una base de datos relacional implica serializar un árbol de objetos Java en una base de datos de estructura tabular y viceversa, además es fundamental la necesidad de mapear objetos Java para optimizar velocidad y eficiencia de la base de datos.

La unidad de persistencia se define como un conjunto de todas las clases que son gestionadas por la instancia del Entity Manager en una aplicación.

La primera y más básica característica de las entidades es que son persistentes. En general, esto sólo significa que pueden hacerse persistentes. Más concretamente, significa que su estado se puede almacenar en una base de datos y se puede acceder en otro momento, tal vez mucho después del final del proceso que lo creó.

2.8.10. Identidad

Al igual que cualquier otro objeto Java, una entidad tiene una identidad de objeto, pero cuando existe en la base de datos también posee una identidad persistente. Esta identidad persistente, o un identificador, es la clave que identifica de forma única una instancia de la entidad y la distingue de todas las otras instancias de la misma entidad.

Capítulo 2: Marco Teórico

2.8.11. Transaccionalidad

Las entidades son lo que podríamos llamar cuasi-transaccionales. Aunque se pueden crear, actualizar y eliminar en cualquier contexto, estas operaciones se realizan normalmente en el contexto de una transacción debido a que una transacción es necesaria para que los cambios que sean guardados en la base de datos. Los cambios son realizados en la base de datos ya sean éxito o fracaso, por lo que la visión persistente de una entidad de hecho debe ser transaccional.

2.8.12. Granularidad

Las entidades que están destinadas a ser objetos de grano fino tienen un conjunto de estado global, el cual es normalmente almacenado en un solo lugar, como una fila de una tabla, y suelen tener relaciones con otras entidades. En el sentido más general, son objetos de dominio de negocio que tienen un significado específico en la aplicación que tiene acceso a ellos.

Si bien es cierto que las entidades pueden ser definidas de manera exagerada, a ser tan fina como almacenamiento de una sola cadena o de grano grueso suficiente para contener un valor de 500 columnas de datos. Las entidades de la JPA estaban destinadas a ser definitivamente el extremo más pequeño del espectro de la granularidad. Idealmente, las entidades deben ser diseñadas y definidas como objetos bastante ligeros, de un tamaño comparable al de la media del objeto de Java.

2.8.13. Persistiendo una Entidad

La persistencia de una entidad es la operación de tomar una entidad transitoria, o una que todavía no tiene ninguna representación persistente en la base de datos, y almacenar su

Capítulo 2: Marco Teórico

estado para que pueda ser recuperada más tarde. Esto es realmente la base de la persistencia de creación de estado, que puede durar más que el proceso que lo creó.

2.8.14. Buscar una entidad

Una vez que una entidad está en la base de datos, lo siguiente que uno normalmente quiere hacer es encontrarla de nuevo.

Estamos pasando como parámetros la clase de la entidad que se busca y la clave de identificación o primaria que identifica a la entidad en particular. Ésta es toda la información necesaria para que el EntityManager pueda encontrar la instancia en la base de datos, y cuando la llamada termina, el empleado que se devuelve sea gestionado como una entidad, lo que significa que existirá en el contexto de persistencia actual asociado con el entidad gerente.

2.8.15. Eliminar la entidad

La eliminación de una entidad de la base de datos no es tan común como usted podría pensar. Muchas aplicaciones nunca eliminan objetos, o si lo hacen sólo los datos que ya no son válidos. Para eliminar la entidad, la buscamos e invocamos un comando para eliminarla. Esto normalmente genera un problema porque la aplicación que acusó la llamada está siendo manejada como parte del proceso para determinar que ese era el objeto que quería eliminarse.

2.8.16. Transacciones

La situación típica cuando se ejecuta en el entorno del contenedor Java EE es que se utiliza el estándar Java Transaction API (JTA) para el manejo de transacciones. El modelo de transacción, cuando se ejecuta en el contenedor, asume que la aplicación asegura que un

Capítulo 2: Marco Teórico

contexto transaccional estará presente cuando éste sea necesario. Si una transacción no está presente, entonces o bien la operación de modificación lanza una excepción o el cambio nunca será persistido en la base de datos.

2.9. Java Persistence Query Language

2.9.1. Definiendo Consultas

JPA posee las interfaces Query y TypedQuery para configurar y ejecutar consultas. La interfaz Query se utiliza para los casos en los que los resultados se esperan que sean de tipo Objeto. La interfaz TypedQuery extiende a Query, por lo que una consulta con tipo puede ser tratada como si fuera una consulta sin tipo, pero no al revés. Se obtiene una implementación de la interfaz apropiada para una consulta dada a través de uno de los métodos factoría de la interfaz del EntityManager. La elección de un método depende del tipo de la consulta.

2.9.2. Consultas Dinámicas

Una consulta se define dinámicamente pasando la cadena de la consulta JPQL y el tipo de resultado esperado al método createQuery () de la interfaz EntityManager. El tipo de resultado puede ser omitido para obtener consultas sin tipo. No existen restricciones a la hora de definir consultas. La habilidad de construir una cadena en tiempo de ejecución y utilizarla para una definición de la consulta es útil, especialmente para aplicaciones donde el usuario puede especificar criterios complejos y la forma exacta de la consulta no puede ser conocida con anticipación.

Una cuestión a considerar con cadenas de consultas dinámicas, sin embargo, es el costo de la traducción de JPQL en SQL para su ejecución. Un motor de consulta típica tendrá

Capítulo 2: Marco Teórico

que analizar la cadena de JPQL en un árbol de sintaxis, obtener los metadatos de mapeo objeto-relacional para cada entidad en cada expresión, y luego generar el SQL equivalente. Para las aplicaciones de muchas consultas, el costo de procesamiento de la consulta dinámica puede ser un problema importante de rendimiento.

2.9.3. Tipos de Parámetros

JPA es compatible tanto con nombre y parámetros de posición de las consultas JPQL. La métodos de la factoría de consultas del EntityManager devuelven una implementación de la interfaz de consultas (Query interface). Los valores de los parámetros se establecen con el método setParameter () de la interfaz de consultas. Hay tres variantes de este método para ambos, parámetros con nombre y los parámetros de posición.

- El primer argumento es siempre el nombre de parámetro o un número.
- El segundo argumento es el objeto que se ha vinculado al parámetro con el nombre.
- Date y Calendar también requieren un tercer argumento de que especifica si el tipo pasado a JDBC, si es una java.sql.Date, java.sql.Time o java.sql.TimeStamp.

2.9.4. Interfaces JPA

JPA se compone de los siguientes 6 tipos de interfaces:

- **Persistence:** Contiene métodos estáticos de ayuda para obtener una instancia de Entity Manager Factory de una forma independiente al vendedor de la implementación de JPA. Una clase de inicialización que va proporcionar un método estático para la creación de una Entity Manager Factory.

Capítulo 2: Marco Teórico

- **EntityManagerFactory**: La clase `javax.persistence.Entity.Manager.Factory` nos ayuda a crear objetos de `EntityManager` utilizando el patrón de diseño del Factory. Este objeto en tiempo de ejecución representa una unidad de persistencia particular.
- **EntityManagerFactory**: La clase `javax.persistence.Entity` es una anotación Java que se coloca a nivel de clases Java serializables y que cada objeto de una de estas clases anotadas representa un registro de una base de datos.
- **EntityManager**: Es la interfaz principal de JPA utilizada para la persistencia de las aplicaciones. Cada Entity Manager puede realizar operaciones CRUD (Create, Read, Update, Delete) sobre un conjunto de objetos persistentes. Es un objeto único, no compartido que representa una unidad de trabajo particular para el acceso a datos. Proporciona métodos para gestionar el ciclo de vida de las instancias entidad y para crear instancias Query.
- **Query**: La interface `javax.persistence.Query` está implementada por cada vendedor de JPA para encontrar objetos persistentes manejando cierto criterio de búsqueda. JPA estandariza el soporte para consultas utilizando Java Persistence Query Language (JPQL) y Structured Query Language (SQL). Podemos obtener una instancia de Query desde una instancia de un Entity Manager.
- **EntityTransaction**: Cada instancia de Entity Manager tiene una relación de uno a uno con una instancia de `javax.persistence.EntityTransaction`, permite operaciones sobre datos persistentes de manera que agrupados formen una unidad de trabajo transaccional. Maneja el concepto de todos o ninguno para mantener la integridad de los datos.

2.9.5. Estados

La persistencia puede tener 4 estados diferentes:

Capítulo 2: Marco Teórico

- **Transient:** Un objeto recién creado que no ha sido enlazado con el gestor de persistencia.
- **Persistent:** Un objeto enlazado con la sesión y todos los cambios serán persistentes.
- **Detached:** Un objeto persistente que sigue en memoria después de que termina la sesión sigue existiendo en Java y en la BDD.
- **Removed:** Un objeto marcado para ser eliminado de la BBDD: existe en Java y se borrará de la BDD al terminar la sesión.

2.9.6. Relaciones Múltiples de la Entidad

Hay cuatro tipos de relaciones: uno a uno, uno a muchos, muchos a uno, y muchos a muchos.

- **Uno a uno:** Cada entidad se relaciona con una sola instancia de otra entidad. Las relaciones uno a uno utilizan anotaciones de la persistencia de Java "OneToOne".
- **Uno a muchos:** Una entidad, puede estar relacionada con varias instancias de otras entidades. Las relaciones uno a muchos utilizan anotaciones de la persistencia de Java "OneToMany" en los campos o propiedades persistentes.
- **Muchos a uno:** Múltiples instancias de una entidad pueden estar relacionadas con una sola instancia de otra entidad. Esta multiplicidad es lo contrario a la relación uno a muchos. Las relaciones muchos a uno utilizan anotaciones de la persistencia de Java "ManyToOne" en los campos o propiedades persistentes.
- **Muchos a muchos:** En este caso varias instancias de una entidad pueden relacionarse con múltiples instancias de otras entidades. Este tipo de relación utiliza anotaciones de la persistencia de Java "ManyToMany" en los campos o propiedades persistentes.

Capítulo 2: Marco Teórico

2.9.7. Relaciones y Borrado en cascada

Existen entidades que utilizan relaciones con dependencias de relaciones de otra entidad. Por ejemplo, una línea es parte de una orden, y si la orden es eliminada, entonces la línea también debe eliminarse. Esto se llama borrado en cascada. Las relaciones de borrado en cascada se especifican utilizando `cascade=REMOVE`, elemento que viene en la especificación de las relaciones `@OneToOne` y `@OneToMany`.

2.9.7.1. Tipos de Memoria cache

- **La Cache de Nivel 1**

La caché de nivel 1 está siempre disponible y es la zona de memoria que utiliza JPA para guardar los datos que vamos recuperando de base de datos y vamos modificando. La única forma de vaciar esta caché es con un `clear ()` o si se consume el ciclo de vida del contexto de JPA.

- **La Cache de Nivel 2**

A caché de nivel 2 se habilita en la configuración, y es la memoria que JPA pone a disposición de la aplicación para guardar objetos de uso frecuente y conseguir una mejora en el rendimiento. Los objetos que se pueden almacenar en caché son Entidades (Datos) y Querys (Consultas).

2.9.7.2. Clases con claves primarias

Una clase con clave primaria debe cumplir los siguientes requerimientos:

- El modificador de control de acceso de la clase debe ser público

Capítulo 2: Marco Teórico

- Las propiedades de la clave primaria deben ser públicas o protegidas si se utiliza el acceso a la base de la propiedad.
- La clase debe tener un constructor público por defecto.
- La clase debe implementar los métodos hashCode() y equals(Object other)
- La clase debe ser serializable.

Una clave primaria debe representarse y mapearse por campos múltiples o propiedades de la clase de la entidad, o debe representarse y mapearse como una clase embebida. Si la clave primaria está compuesta por varios campos o propiedades, los nombres y tipos de campos de la clave primaria o propiedades en la clave primaria debe coincidir con las de la entidad.

2.10. Java Server Faces (JSF)

La tecnología JSF es un framework para aplicaciones Java basadas en Web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. Además usa JSP como la tecnología que permite hacer el despliegue de las páginas.

2.10.1. Introducción

JavaServer Faces o JSF es un framework de aplicaciones web basado en Java para desarrollar interfaces de usuario basadas en web.

JSF conecta widgets de interfaz de usuario con fuentes de datos y controladores de eventos de servidor. JSF permite la reutilización y la extensión de los componentes de interfaz de usuario estándar existentes.

Para el desarrollo de aplicaciones de negocio se utiliza frecuentemente el patrón de diseño MVC Modelo Vista Controlador (Model View Controller) que además es sencillo de

Capítulo 2: Marco Teórico

implementar en las aplicaciones web. En este patrón el modelo es modificable por las funciones de negocio.

Estas funciones son solicitadas por el usuario mediante el uso de un conjunto de vistas de la aplicación que solicitan dichas funciones de negocio a través de un controlador, que es el módulo que recibe las peticiones de las vistas y las procesa. Se suele clasificar en dos tipos a las aplicaciones basadas en MVC:

- **Tipo 1.** Las vistas conocen la acción que se va a invocar en su petición. Normalmente la función esta cableada dentro de la vista
- **Tipo 2.** El controlador introduce un conjunto de reglas que mapean a las peticiones con las funciones, controlando además el flujo de navegación por la aplicación.

2.10.2. Características

- Contiene un conjunto de APIs para representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Un conjunto por defecto de componentes para la interfaz de usuario.
- Dos librerías de etiquetas personalizadas para Java Server Pages que permiten expresar una interfaz Java Server Faces dentro de una página JSP.
- Administración de estados.

2.10.3. Ventajas

- Debido a su alta mantenibilidad pues a pesar de estar sobre servlets, su desarrollo no implica directamente la creación de los servlets sino directamente las páginas que fácilmente se pueden volver dinámicas.

Capítulo 2: Marco Teórico

- Los componentes web se enlazan con fuentes de datos que permiten la construcción dinámica de la presentación y estos a su vez con capas de negocio de la aplicación, de esta manera la separación es clara y la estrategia de desarrollo se convierte en una propuesta eficiente para la construcción de aplicaciones web Java.
- JSF nos permite desarrollar rápidamente aplicaciones de negocio dinámicas en las que toda la lógica de negocio se implementa en Java, o es llamada desde Java, creando páginas para las vistas muy sencillas.
- El código JSF con el que creamos las vistas etiquetas jsp es muy parecido al HTML estándar.
- JSF se integra dentro de la página JSP y se encarga de la recogida y generación de los valores de los elementos de la página.
- JSF permite introducir JavaScript en la página, para acelerar la respuesta de la interfaz en el cliente.
- JSF es extensible, por lo que se pueden desarrollar nuevos componentes a medida, También se puede modificar el comportamiento del framework mediante APIs que controlan su funcionamiento.

2.10.4. Definición de aplicación JSF

En su mayoría, las aplicaciones JavaServer Faces son como cualquier otra aplicación web Java. Se ejecutan en un contenedor de servlets de Java y, típicamente, contienen:

Componentes JavaBeans llamados objetos del modelo en tecnología Java Server Faces conteniendo datos y funcionalidades específicas de la aplicación.

- Oyentes de Eventos.

Capítulo 2: Marco Teórico

- Páginas, principalmente páginas JSP.
- Clases de utilidad del lado del servidor, como beans para acceder a las bases de datos.

Además de estos ítems, una aplicación JavaServer Faces también tiene:

- Una librería de etiquetas personalizadas para dibujar componentes UI en una página.
- Una librería de etiquetas personalizadas para representar manejadores de eventos, validadores y otras acciones.
- Componentes UI representados como objetos con estado en el servidor.

Toda aplicación Java Server Faces debe incluir una librería de etiquetas personalizadas que define las etiquetas que representan componentes UI, así como una librería de etiquetas para controlar otras acciones importantes, como validadores y manejadores de eventos.

Y la librería principal o core hace fácil registrar eventos, validadores y otras acciones de los componentes.

2.10.5. Los backbeans

A las clases Java que se asocian a los formularios JSF se les denomina backend beans ya que son los beans (clases Java) que están detrás del formulario. Estos beans se referencian en el fichero de configuración de JSF en el apartado de managed beans, ya que son beans gestionados por el controlador JSF. Este se encarga de su construcción y destrucción automáticas cuando es necesario.

Capítulo 2: Marco Teórico

En su versión más sencilla, cada página JSF está formada por una página JSP que contiene un formulario (HTML FORM) y un backbean.

El controlador JSF registra en el servidor de aplicaciones un tipo especial de petición, típicamente *.jsf, que estará asociado a estas páginas.

Finalmente el servidor devuelve al usuario una página creada a partir de una página JSP que incluye normalmente etiquetas JSF, cuyos valores se extraerán del backbean asociado, ahora ya actualizados.

2.10.5.1. Respondiendo a las acciones del usuario

Una vez que el usuario ve la página web que se ha construido con JSF, está listo para comenzar a interactuar con ella. El método más sencillo de interacción es el uso de formularios web. Un formulario web simple consta de:

- Etiquetas que muestran información
- Campos editables
- El botón de envío del formulario

El controlador JSF dispone de un conjunto de etiquetas que permiten definir formularios JSF. Las más sencillas son:

- **Form:** Esta etiqueta sustituye al form de HTML, añadiendo la funcionalidad JSF al formulario
- **OutputText:** Sirve para imprimir valores en la página
- **InputText:** Sirve para crear campos editables en los que introducir los datos
- **CommandButton:** Crea botones que envían el formulario

Capítulo 2: Marco Teórico

Cuando la página JSF contiene elementos que incluyen acciones se ejecuta una fase más en el procesamiento de la petición al servidor. Si en nuestro formulario hay botones u otros elementos que tienen una propiedad action, si se pulsa sobre el elemento cuando la petición sea procesada por el servidor se ejecutará la lógica de la acción asociada a este elemento. Este es el mecanismo JSF habitual para ejecutar a lógica de la aplicación. Esto se hace ejecutando los métodos del backbean asociado a la página

2.10.5.2. La navegación entre páginas

Cuando se ejecuta una petición que incluye una acción se ejecuta el mecanismo de navegación de JSF. Tras la ejecución de la acción, el controlador determina cómo se debe mostrar al usuario el resultado de la petición. Hay varias posibilidades:

- Finalizar la petición mostrando la página jsf que originó la petición, que es la opción por defecto.
- Mostrando otra página jsf diferente.

Enviando al usuario una petición de redirección, por lo que el navegador del usuario se dirigirá automáticamente a otra página cuando reciba la respuesta a su petición.

Las reglas de navegación definen:

- La página de origen. Indica el jsf que originó la petición.
- La etiqueta de destino. Es la cadena que identifica el destino. Esta cadena es devuelta por el método del backbean que procesa la acción.
- La página de destino para cada etiqueta. Normalmente es el jsf el que procesará la petición de salida, utilizando los datos que hay en la request y en la sesión.

Capítulo 2: Marco Teórico

- Si es un envío directo interno o una redirección externa. En el primer caso la respuesta se generará en la misma petición mediante una redirección interna a otro jsp o servlet. En el segundo caso se enviará como respuesta al navegador una instrucción de redirección para que el navegador realice una nueva petición de otra página.

Además las direcciones de origen admiten el * para que una misma regla sirva para múltiples páginas. También se pueden poner reglas por defecto que se aplican a todas las peticiones: converter, events y listeners, mensajes información que se muestra al usuario y navegación

2.10.6. Ciclo de Vida Estándar JSF

Cuando se carga la aplicación web en el servidor se inicializa el framework JSF. Se lee el fichero de configuración faces-config.xml y se crean los beans gestionados definidos con el ámbito application, realizando las sentencias de inicialización necesarias. Después el motor de JSF está listo para recibir peticiones y para lanzar el ciclo de vida de JSF con cada una.

Lo que en JSF se denomina ciclo de vida no es más que una secuencia de fases por las que pasa una petición JSF desde que se recibe en el servidor hasta que se genera la página HTML resultante.

Las fases del ciclo de vida son las siguientes:

- Restaurar la vista (restore view). En este paso se obtiene el árbol de componentes correspondiente a la vista JSF de la petición. Si se ha generado antes se recupera, y

Capítulo 2: Marco Teórico

si es la primera vez que el usuario visita la página, se genera a partir de la descripción JSF.

- Aplicar los valores de la petición (apply request values). Una vez obtenido el árbol de componentes, se procesan todos los valores asociados a los mismos. Se convierten todos los datos de la petición a tipos de datos Java y, para aquellos que tienen la propiedad immediate a cierta, se validan, adelantándose a la siguiente fase.
- Procesar las validaciones (process validations). Se validan todos los datos. Si existe algún error, se encola un mensaje de error y se termina el ciclo de vida, saltando al último paso (renderizar respuesta).
- Actualizar los valores del modelo (update model values). Cuando se llega a esta fase, todos los valores se han procesado y se han validado. Se actualizan entonces las propiedades de los beans gestionados asociados a los componentes.
- Invocar a la aplicación (invoke application). Cuando se llega a esta fase, todas las propiedades de los beans asociados a componentes de entrada (input) se han actualizado. Se llama en este momento a la acción seleccionada por el usuario.

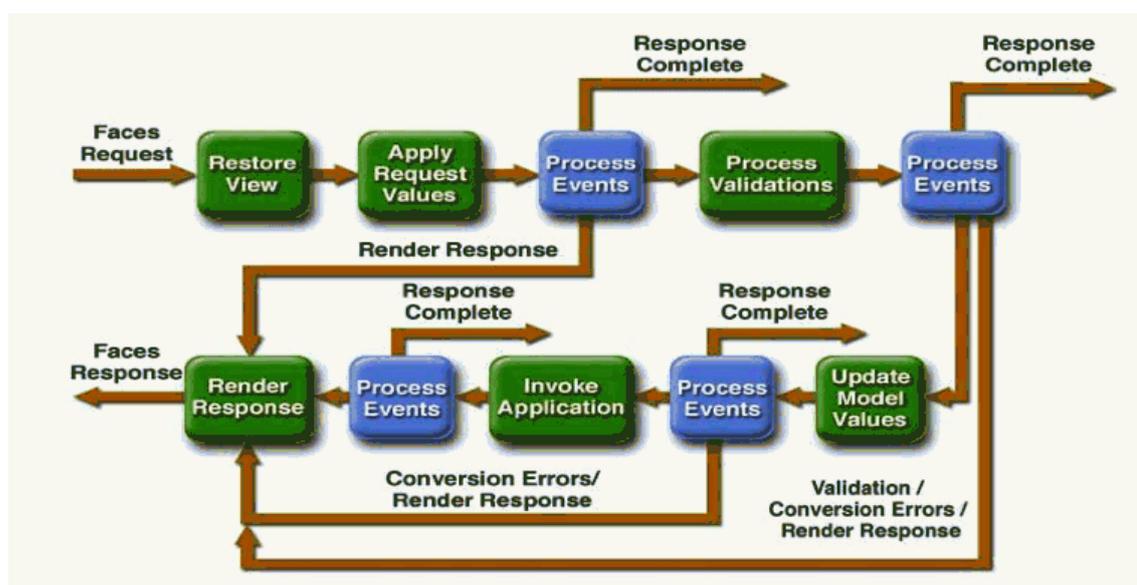


Figura 2. 11 Ciclo de vida de JSF

Fuente: Universidad de Alicante. (2015) Recuperado de <http://www.jtech.ua.es/j2ee/publico/jsf-2012-13/sesion03-apuntes.html>

Capítulo 2: Marco Teórico

- Renderizar la respuesta (render response). Durante esta fase, el JSF pide servidor de contenedor/aplicación para representar la página si la aplicación está utilizando páginas JSP. En cualquier caso, los componentes se hacen a sí mismos como el servidor de contenedor / Aplicación JSP atraviesa las etiquetas en la página.
- Después de que se dictó el contenido de la vista, se guarda el estado de respuesta para que las solicitudes posteriores pueden acceder a ella y que está disponible para la fase de restaurar la vista.
- Al final de cada una de las fases, se comprueba si hay algún evento que debe ser procesado en esa fase en concreto y se llama a su manejador. También se llaman a los manejadores de los eventos que deben ser procesados en cualquier fase. Los manejadores, a su vez, pueden saltar a la última fase del ciclo para renderizar el árbol de componentes llamando al método `renderResponse ()` del `FacesContext`. También pueden renderizar el componente asociado al evento y llamar al método `responseComplete ()` del `FacesContext` para terminar el ciclo de vida.
- JSF emite un evento `PhaseListener` al comienzo y al final de cada fase del ciclo de vida de la petición. Para capturar el evento, debemos definir una clase que implemente la interfaz `PhaseListener` y sus métodos `beforePhase` y `afterPhase`.

2.10.7. Facelets

Facelets es un framework basado en el servidor que permite definir la estructura general de las páginas mediante plantillas. Se trata de un framework similar a Tiles, Velocity, Tapestry o Sitemesh.

Facelets se adapta perfectamente al enfoque de JSF y se incorpora a la especificación en la última revisión 2.1. Anteriormente, las páginas JSF se definían utilizando etiquetas

Capítulo 2: Marco Teórico

específicas de JSP, lo que generaba cierta confusión porque se trata de enfoques alternativos para un mismo problema.

La sustitución de JSP por Facelets como lenguaje básico para definir la disposición de las páginas permite separar perfectamente las responsabilidades de cada parte del framework.

Entre las características de Facelets destacan:

- Definición de plantillas
- Composición de componentes
- Etiquetas para definir funciones y lógica
- Desarrollo de páginas amistoso para el diseñador
- Posibilidad de crear librerías de componentes

Facelets nos permite usar un mecanismo de plantillas para encapsular los componentes comunes en una aplicación. Así también podremos modificar el aspecto de nuestra página aplicando cambios sobre la plantilla, y no individualmente sobre las páginas.

2.10.8. Beans

2.10.8.1. *ManagedBean*

A las clases java que se asocian a los formularios JSF se les denomina backend beans ya que son los beans (clases java) que están detrás del formulario. Estos beans se referencian en el fichero de configuración de JSF en el apartado de managed beans, ya que son beans gestionados por el controlador JSF este se encarga de su construcción y destrucción automáticas cuando es necesario.

Capítulo 2: Marco Teórico

Marca el bean para ser un managed bean con el nombre específico en el atributo nombre. En el caso de no especificar el nombre en la anotación, el nombre del managed bean será por defecto el nombre de la clase.

2.10.8.2. *BackingBean*

Un Java Server Faces típica aplicación incluye uno o más beans de respaldo, cada uno de los cuales es un Java Server Faces frijol que se asocia con los componentes de interfaz de usuario se utilizan en una página en particular logró. Frijoles administrados son componentes JavaBeans que se pueden configurar mediante la instalación bean gestionado, que se describe en Configuración de Beans. Esta sección presenta los conceptos básicos sobre la creación, configuración y uso de los granos que respaldan en una aplicación.

Y en realidad es la misma cosa pero el concepto de "backing bean" o bean de respaldo significa que la función del bean es almacenar el estado de las páginas JSP. En cuanto a "Managed Bean", proviene de la declaración en el faces-config, en la cual se le indica a JSF que administre ese Bean, es decir se encargue de instanciarlo, hacerlo disponible en las JSP y también de removerlo cuando su scope haya salido de contexto.

2.10.9. *Ámbito de los beans gestionados*

El ámbito de los beans determina su ciclo de vida: cuándo se crean y destruyen instancias del bean y cuándo se asocian a las páginas JSF. Es muy importante definir correctamente el ámbito de un bean, porque en el momento de su creación se realiza su inicialización. JSF llama al método constructor de la clase, donde habremos colocado el código para inicializar sus valores, creando una instancia de la clase que pasará a ser gestionada por el framework.

Capítulo 2: Marco Teórico

En JSF se definen los siguientes ámbitos para los beans: petición, sesión, vista y aplicación. El ámbito de vista es un elemento nuevo de JSF 2.0. Los otros ámbitos son similares a los definidos con los JavaBeans de JSP.

El modelo JSF se define mediante beans idénticos a los que se utilizan en JSP. Un bean es una clase con un conjunto de atributos y métodos *getters* y *setters* que devuelven y actualizan sus valores. Las propiedades del bean se pueden leer y escribir desde las páginas JSF utilizando el lenguaje de expresiones EL.

2.10.9.1. @RequestScoped

Petición: Se define con el valor request en la propiedad managed-bean-scope del faces-config.xml o con la anotación @RequestScoped en la clase. El bean se asocia a una petición HTTP. Cada nueva petición crea un nuevo bean y lo asocia con la página.

2.10.9.2. @CustomScoped

Custom: Un ámbito al fin y al cabo no es más que un mapa que enlaza nombres y objetos. Lo que distingue un ámbito de otro es el tiempo de vida de ese mapa. Los tiempos de vida de los ámbitos estándar de JSF: sesión, aplicación, vista y petición, son gestionados por la implementación de JSF. En JSF 2.0 podemos crear ámbitos personalizados, que son mapas cuyo ciclo de vida gestionamos nosotros.

2.10.9.3. @ConversationScoped

Conversación: provee de persistencia de datos hasta que se llega a un objetivo específico, sin necesidad de mantenerlo durante toda la sesión. Está ligado a una ventana o

Capítulo 2: Marco Teórico

pestaña concreta del navegador. Así, una sesión puede mantener varias conversaciones en distintas páginas. Es una característica propia de CDI, no de JSF.

2.10.9.4. @ApplicationScope

Aplicación: Se define con el valor `application` y con la anotación `@ApplicationScoped`. Los beans con este ámbito viven asociados a la aplicación. Definen *singletons* que se crean e inicializa sólo una vez, al comienzo de la aplicación. Se suelen utilizar para guardar características comunes compartidas y utilizadas por el resto de beans de la aplicación.

2.10.9.5. @SessionScope

Sesión: Se define con el valor `session` en el `faces-config.xml` o con la anotación `@SessionScoped` en la clase. Las sesiones se definen internamente con el API de Servlets. Una sesión está asociada con una visita desde un navegador. Cuando se visita la página por primera vez se inicia la sesión. Cualquier página que se abra dentro del mismo navegador comparte la sesión. La sesión mantiene el estado de los elementos de nuestra aplicación a lo largo de las distintas peticiones.

2.10.9.6. @ViewScope

Vista (JSF 2.0): Se define con el valor `view` en el `faces-config.xml` o con la anotación `@ViewScoped` en la clase. Un bean en este ámbito persistirá mientras se repinte la misma página (vista = página JSF), al navegar a otra página, el bean sale del ámbito. Es bastante útil para aplicaciones que usen Ajax en parte de sus páginas.

Capítulo 2: Marco Teórico

2.10.9.7. *@RequestScope*

Establece el ámbito en el que se sitúa el bean. Si no se establece, por defecto el ámbito será un requestscope. Los posibles ámbitos son @NoneScoped, @RequestScoped, @ViewScoped, @SessionScoped, @ApplicationScoped, y @CustomScope.

2.10.10. Manejo del Scope y los ManagedBeans

Se denomina scope a la disponibilidad o contexto de un objeto y a su período de vida en una aplicación web.

Para tener disponibles valores en otras páginas o incluso en la misma página lo lógico sería guardar los valores. Antes de crear una propiedad para que guarde un valor, se debe determinar el scope apropiado para ese valor, ya que muchos usuarios podrían acceder a la aplicación al mismo tiempo, así que se necesitaría utilizar el scope más corto posible para hacer un mejor uso de los recursos del servidor. La siguiente figura muestra la duración de cada tipo de scope.

2.10.11. Etiquetas JSF

JSF dispone de un conjunto básico de etiquetas que permiten crear fácilmente componentes dinámicos en las páginas web. Estas etiquetas son:

- **CommandButton:** Un botón al que podemos asociar una acción.
- **CommandLink:** Un enlace hipertexto al que podemos asociar una acción.
- **DataTable:** Crea una tabla de datos dinámica con los elementos de una propiedad de tipo Array o Map del bean.
- **Form:** Define el formulario JSF en la página JSP.

Capítulo 2: Marco Teórico

- **GraphicImage:** Muestra una imagen jpg o similar.
- **InputHidden:** Incluye un campo oculto del formulario.
- **InputSecret:** Incluye un campo editable de tipo contraseña, no muestra lo que se escribe.
- **InputText:** Incluye un campo de texto normal.
- **InputTextarea:** Incluye un campo de texto multilínea.
- **Message:** Imprime un mensaje de error en la página en el caso que se ha producido alguno.
- **Messages:** Imprime varios mensajes de error en la página, si se han producido.
- **OutputFormat:** Muestra texto parametrizado. Utiliza la clase de formateo MessageFormat.
- **OutputLabel:** Muestra un texto fijo.
- **OutputLink:** Crea un enlace hipertexto.
- **OutputText:** Es la etiqueta más común utilizada para mostrar texto plano, y no genera ningún elemento HTML extra.
- **PanelGrid:** Crea una tabla con los componentes incluidos en el panelGrid.
- **PanelGroup:** Agrupa varios componentes para que cierto componente los trate como un único componente.
- **SelectBooleanCheckbox:** Crea una casilla con dos estados: activado y desactivado.
- **SelectManyCheckbox:** Crea un conjunto de casillas activables.
- **SelectManyListbox:** Crea una lista que permite seleccionar múltiples elementos.
- **SelectManyMenu:** Crea una lista desplegable de selección múltiple.
- **SelectOneListbox:** Crea una lista en la que se puede seleccionar un único elemento.
- **SelectOneMenu:** Crea una lista desplegable de selección.

Capítulo 2: Marco Teórico

- **SelectOneRadio:** Crea una lista de botones, redondos normalmente, excluyentes.

2.10.12. Pasos para el Desarrollo

2.10.12.1. Arquitectura JSF

Una aplicación JSF es similar a cualquier otra aplicación web basada en la tecnología de Java; que se ejecuta en un contenedor de servlets Java y contiene. Componentes JavaBeans como modelos que contienen funcionalidad y los datos específicos de la aplicación.

Una biblioteca de etiquetas personalizadas para la representación de los controladores de eventos y validadores.

Una biblioteca de etiquetas personalizadas para la prestación de los componentes de interfaz de usuario. Componentes de interfaz de usuario representadas como objetos con estado sobre el servidor

2.10.12.2. Clases de ayuda del lado del servidor.

Validadores, controladores de eventos y controladores de navegación. Archivo de recursos de configuración de aplicaciones para la configuración de recursos de la aplicación

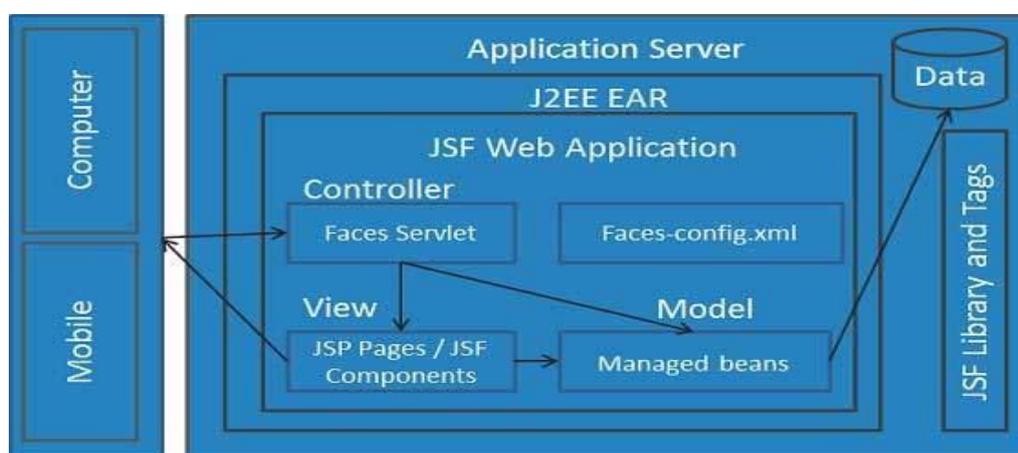


Figura 2. 12 JSF

Fuente: JSF (2015) Recuperado de <https://alexandrev.wordpress.com/2009/08/05/struts-2-0-vs-jsf/>

Capítulo 2: Marco Teórico

JSF proporciona varios mecanismos para la prestación de un componente individual. Es hasta el diseñador de páginas web para recoger la representación deseada, y el desarrollador de aplicaciones no necesita saber qué mecanismo se utiliza para representar un componente JSF interfaz de usuario

2.10.13. Modelo Vista Controlador en JSF

JSF utiliza el framework MVC (Modelo-Vista-Controlador) para gestionar las aplicaciones web. Vamos a verlo con cierto detalle.

2.10.13.1. Vista

La forma más común de definir la vista en JSF 2.0 es utilizando ficheros XHTML con etiquetas especiales que definen componentes JSF. Al igual que en JSP, estos componentes se convierten al final en código HTML, que se pasa al navegador para que lo muestre al usuario. El navegador es el responsable de gestionar la interacción del usuario.

2.10.13.2. Modelo: beans gestionados

El modelo JSF se define mediante beans idénticos a los que se utilizan en JSP. Un bean es una clase con un conjunto de atributos y métodos getters y setters que devuelven y actualizan sus valores. Las propiedades del bean se pueden leer y escribir desde las páginas JSF utilizando el lenguaje de expresiones EL.

2.10.13.3. Controlador

Debemos diferenciar dos tipos de acciones, las acciones del componente y las acciones de la aplicación. En el primer tipo de acciones es el propio componente el que contiene el código HTML o JavaScript que le permite reaccionar a la interacción del usuario. En este caso no hay ninguna petición al controlador de la aplicación para obtener datos o modificar algún elemento, sino que toda la interacción la maneja el propio componente.

Capítulo 2: Marco Teórico

RichFaces y JSF 2.0 es posible utilizar eventos JavaScript para configurar este comportamiento.

Las acciones de la aplicación son las que determinan las funcionalidades de negocio de la aplicación. Se trata de código que queremos que se ejecute en el servidor cuando el usuario pulsa un determinado botón o pincha en un determinado enlace. Este código realizará llamadas a la capa de negocio de la aplicación y determinará la siguiente vista a mostrar o modificará la vista actual.

2.11. Framework RIA para java server faces

2.11.1. Definición

Una RIA por sus siglas en inglés Rich Internet Application es un nuevo tipo de aplicación Web cuyo objetivo es incrementar y mejorar las opciones y capacidades de las aplicaciones Web tradicionales (Universitat Pompeu Fabra, 2015).

Aparecen para subsanar ciertas limitaciones en la capa de presentación de los actuales navegadores Web y del lenguaje HTML.

Permiten, entre otras cosas, mejorar la experiencia del usuario al utilizar la aplicación, la ejecución de contenido multimedia y la carga de aplicaciones online/offline. Existen distintas tecnologías con las cuales construir aplicaciones RIA.

2.11.2. Características

- Mejora de la experiencia visual gracias a la aportación de nuevos componentes más avanzados mediante la utilización de audio, video y gráficos.
- La mayoría de tecnologías RIA se basan en el lenguaje de programación XML, tanto para las interfaces gráficas como para el intercambio de datos.

Capítulo 2: Marco Teórico

- Conectividad. La mayoría de RIAs permiten ejecutar las aplicaciones online/offline.
- Requiere la utilización de un cliente rico en el lado del usuario, que será el motor de la tecnología RIA.
- Los datos pueden ser obtenidos desde el servidor sin necesidad de actualizar toda la pantalla.
- Facilidad de despliegue, administración y desarrollo

2.11.3. Desarrollar RIA con estándares

Desde el punto de vista del desarrollador, implementar una RIA mediante estándares no representa cambios muy importantes en lo que se refiere al proceso de desarrollo.

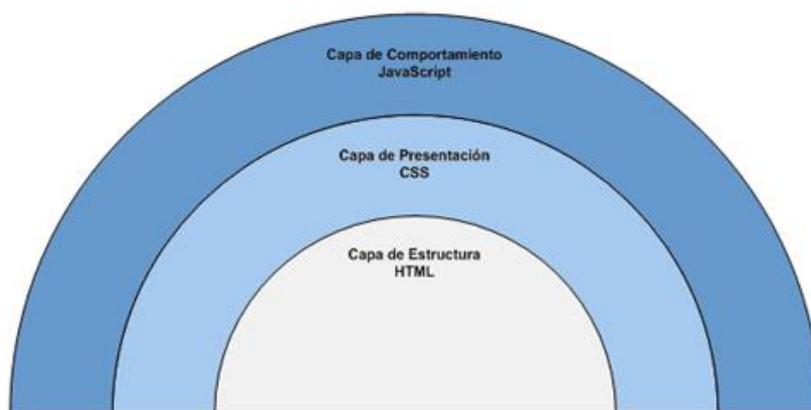


Figura 2. 13 Capas de programación para una página web

Fuente: Universitat Pompeu Fabra (2015) Recuperado de <https://www.upf.edu/hipertextnet/numero-9/ria-accesibilidad-web.html>

Como se puede apreciar en la figura anterior, y de forma muy resumida, en cualquier página web se pueden distinguir hasta 3 capas de desarrollo (Voces et al., 2009):

- **La capa de estructura y contenido.** Donde se definen los diferentes bloques que componen la página: cabecera, contenido, pie, navegación y el contenido que se presenta.

Capítulo 2: Marco Teórico

- **La capa de presentación.** Es la capa donde se diseña la apariencia visual y la distribución de los bloques estructurales y de los contenidos.
- **La capa de comportamiento.** Es la capa donde se programa cómo debe reaccionar la página frente a las acciones del usuario. Esta capa interactúa directamente con el DOM (*Document Object Model*) del agente de usuario, lo que permite conocer todos los detalles de interacción o movimiento de ratón, clic, presión de tecla; acceder a todos los contenidos que se presentan en la página y modificarlos si es preciso.

2.11.4. Implementación RIA para JSF

2.11.4.1. Primefaces

Es una librería de componentes para JavaServer Faces (JSF) de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web., está bajo la licencia de Apache License V2. Una de las ventajas de utilizar Prime faces, es que permite la integración con otros componentes como por ejemplo RichFaces.

Algunas de las características que nos ofrece Prime Faces son:

- Un interesante conjunto de componentes (editor HTML, autocompletado, gráfica)
- Fácil de instalar y sin dependencias, ni configuraciones, además de ser muy liviano
- Soporte para Ajax, basándose en el estándar JSF 2.0 Ajax API.
- Muy liviana, solo un archivo jar y no se requiere configuración.
- Soporte para interfaces de usuario sobre dispositivos móviles, nos provee de un kit para este menester.
- Múltiples temas de apariencia, listos para usar.
- Soporta Ajax Push via websockets

Capítulo 2: Marco Teórico

- Mobile UI kit que permite la creación de aplicaciones web para dispositivos móviles, especiales para Iphones, Palm, Android y teléfonos móviles Nokia.
- Soporta Skinning y posee más de 35 temas reconstruidos. Soporta el editor visual de temas.

2.12. **Atmósfera socket**

2.12.1. **Introducción**

Atmosphere es un framework java o javascript para la creación de aplicaciones Web asincrónicas portables usando Groovy, Scala y Java. El objetivo del framework es permitir a los desarrolladores escribir una aplicación y dejar que éste descubra el mejor canal de comunicación entre el cliente y el servidor, de forma transparente. (Gitub, 2015).

2.12.2. **Componentes:**

La atmósfera soporta de forma nativa tres componentes:

2.12.2.1. *Atmosphere-runtime:*

Es el módulo principal de la atmósfera. Este módulo presenta dos API simples para la construcción de aplicaciones: AtmosphereHandler y Meteor.

- **AtmosphereHandler.-** es una interfaz sencilla para implementaciones.
- **API Meteor.-** es una clase que puede recuperar o implementar aplicaciones basadas en Servlets.

2.12.2.2. *Atmosphere-jersey:*

Es una extensión al framework Jersey REST. Este módulo expone un nuevo conjunto de anotaciones, mostrando así la funcionalidad de tiempo de ejecución de la Atmósfera.

Capítulo 2: Marco Teórico

2.12.2.3. Atmosphere gwt:

Es una extensión del framework GWT.

2.12.3. Características:

Algunas de las más importantes características indicadas en la figura siguiente:

- Atmosphere es soportado por los principales navegadores y servidores
- Se integración con los más populares frameworks java y javascript.
- Soporta diferentes tecnologías para implementar comet, como WebSockets, Server Side Events (SSE), Long-Polling, HTTP Streaming y JSONP.
- Como servidor atmosphere soporta Tomcat, Jboss, Jetty, Resin, GlassFish, Undertow, WebSphere, WebLogic.
- Tiene extensiones para los frameworks más usados PrimeFaces, Wicket, Vaadin, Grails, richfaces, Spring.
- Ofrece los clustering plugins que permiten ofrecer clustering a través de JMS.

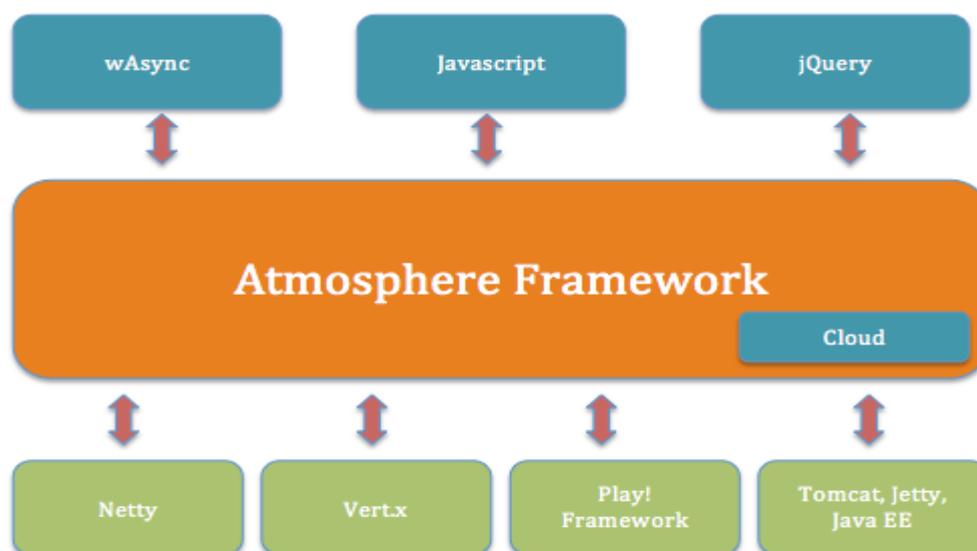


Figura 2. 14 Características atmosphere framework

Fuente: Gitub. (2015) Recuperado de <https://github.com/Atmosphere/atmosphere/wiki/Atmosphere-PlugIns-and-Extensions>

Capítulo 2: Marco Teórico

2.12.4. Modos de funcionamiento:

Atmosphere puede funcionar en dos modos: **embedido** o **standalones** (JWebSocket).

- **Embebido.-** permite correr Atmosphere en un contenedor.
- **Standalones o Nettosphere.-** es un contenedor de Atmosphere sobre el framework Netty muy útil para test de integración.

2.13. Sistemas de información de procesos de producción

Las empresas de manufactura enfrentan grandes retos en la actualidad del mundo innovador, donde la competencia se incrementa y requiere de la exigencia en la calidad, una mejor calidad en el servicio, más variedad de productos y sobre todo más reglamentos para el cuidado del medio ambiente; todo esto exige una mayor rapidez, flexibilidad y efectividad en el desempeño de las operaciones productivas.

Es evidente que para optimizar el uso de recursos se requiere que día a día se exijan mejoras continuas en los procesos productivos para ello como solución se desarrollan sistemas informáticos.

2.13.1. Introducción a los procesos

Es una secuencia o conjunto estructurado de actividades diseñado para producir algo específico para un mercado o un cliente, todas las actividades de un proceso deben tener un objetivo en común relacionado con la satisfacción del cliente. La gestión por procesos sirve para desplegar internamente las necesidades de los clientes externos. (Pérez, 2010)

2.13.2. Tipos

ISO propone tres tipos de procesos:

Capítulo 2: Marco Teórico

- **Procesos claves:** realizan el producto o servicio y desarrollan la estrategia de la empresa.
- **Subprocesos:** proporcionan recursos o entradas para los procesos claves.
- **Procesos de apoyo:** necesarios para el funcionamiento de la empresa.

2.13.3. Fases de Análisis Del Proceso

Para mejorar un proceso es la utilización de la misma metodología, que debe ser lo más disciplinada posible. Se aconseja utilizar el modelo siguiente, que resume las 5 principales actividades que se deben llevar a cabo para mejorar un proceso y que son las siguientes:

- **Identificar.-** los procesos clave de la empresa o del área que se pretende mejorar.
- **Seleccionar.-** los procesos por los cuales se van a empezar.
- **Analizar.-** evaluar el proceso seleccionado con el fin de conocerlo lo mejor posible antes de mejorarlo.
- **Mejorar.-** el proceso evaluado.
- **Actuar.-** paso final.

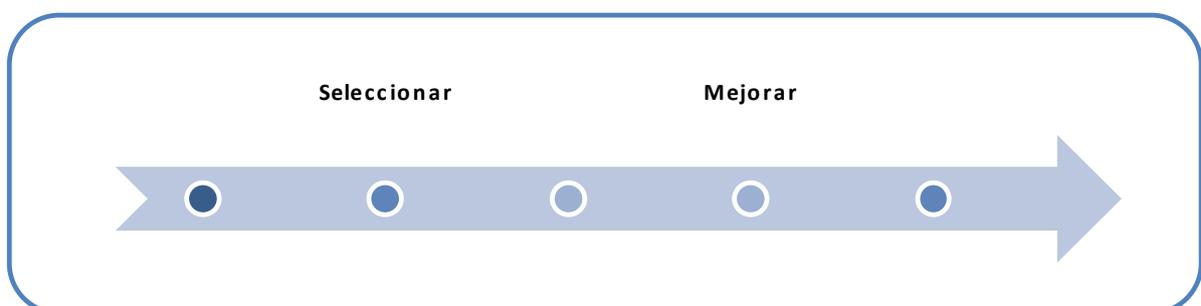
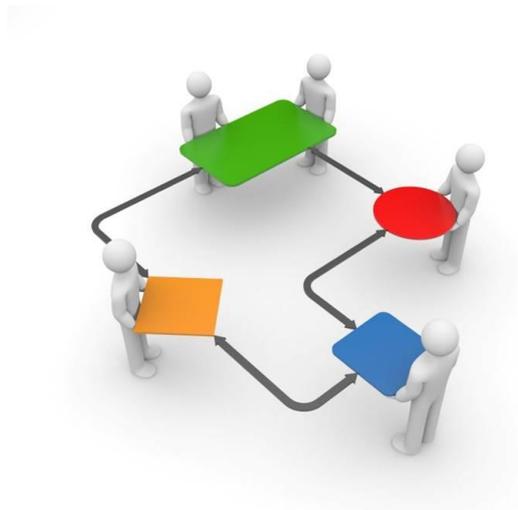


Figura 2. 15 Fases de análisis de proceso

Fuente: Moreno, A. (2015) Recuperado de <http://elies.rediris.es/elies9/3-1-1.htm>

Capítulo 3

LEVANTAMIENTO DE PROCESOS



En el capítulo 3 se va a tratar sobre la gestión por procesos donde se dice que los sistemas son un conjunto de procesos interrelacionados con un propósito final que es asegurar que los procesos de una organización se desarrollan de forma coordinada, mejorando la efectividad y satisfacción de todas las partes.

3.1. Definición de procesos de producción

3.1.1 Definición Gestión de Procesos

Business Process Management o también llamado Gestión o administración por procesos de negocio, con sus siglas BPM, es una metodología cuyo objetivo es mejorar el desempeño de la Organización a través de la gestión de los procesos de negocio, que se deben diseñar, modelar, organizar, documentar y optimizar de forma continua.

La gestión de procesos es una forma sistémica de identificar, comprender y aumentar el valor agregado de los procesos de la empresa para cumplir con la estrategia del negocio y elevar el nivel de satisfacción de los clientes.

Capítulo 3: Levantamiento de Procesos

De acuerdo con Voehl (2007):

El propósito de la Administración de procesos, es asegurarse de que todos los procesos claves trabajen en armonía para maximizar la efectividad organizacional. La meta es alcanzar una ventaja competitiva a través de una mayor satisfacción del cliente. Las herramientas y técnicas principales usadas en estos procesos son: diagrama de flujo, tormenta de ideas, votación, diagrama de Pareto, y gráficas.

Si estas son implementadas eficazmente, se obtienen excelentes resultados. (p.31)

3.1.2 Objetivos de la Gestión por Procesos

Los objetivos generales que persiguen una reingeniería y gestión de procesos son:

- Mayores beneficios económicos debidos tantos a la reducción de costos asociados al proceso como al incremento de rendimiento de los procesos.
- Mayor satisfacción del cliente debido a la reducción del plazo de servicio y mejora de la calidad del producto/servicio.
- Mayor satisfacción del personal debido a una mejor definición de procesos y tareas
- Mayor conocimiento y control de los procesos
- Conseguir un mejor flujo de información y materiales
- Disminución de los tiempos de proceso del producto o servicio

3.1.3 Clasificación de procesos

Existen 3 tipos de procesos: estratégicos, del negocio y de apoyo.

Capítulo 3: Levantamiento de Procesos

3.1.3.1 *Procesos estratégicos.*

Son aquellos relacionados con la estrategia de la organización, considera la forma como se establece la visión, misión, valores, directrices funcionales, objetivos corporativos, departamentales y personales y el programa de acción entre otros componentes.

3.1.3.2 *Procesos del negocio o clave*

Atienden directamente la misión del negocio y satisfacen necesidades concretas de los clientes. También son aquellos que añaden valor al cliente o inciden directamente en su satisfacción o insatisfacción. Componen la cadena del valor de la organización. También pueden considerarse procesos clave aquellos que, aunque no añadan valor al cliente, consuman muchos recursos.

Los procesos clave intervienen en la misión, pero no necesariamente en la visión de la organización.

3.1.3.3 *Procesos de apoyo*

Son servicios internos necesarios para realizar los procesos del negocio. También se les llama procesos secundarios. En este tipo se encuadran los procesos necesarios para el control y la mejora del sistema de gestión, que no puedan considerarse estratégicos ni clave.

Normalmente estos procesos están muy relacionados con requisitos de las normas que establecen modelos de gestión.

3.1.4 Elementos de los procesos.

Cualquier proceso de trabajo, no importando si sea pequeño o grande, complicado o sencillo, involucra tres componentes principales:

- **Entradas:** Recursos del ambiente externo, incluyendo productos o salidas de otros subsistemas.

Capítulo 3: Levantamiento de Procesos

- **Procesos de transformación:** Las actividades de trabajo que transforman las entradas, agregando valor a ellas y haciendo de las entradas, las salidas del subsistema.
- **Salidas:** Los productos y servicios generados por el subsistema, usados por otro sistema en el ambiente externo.

3.1.5 Importancia de la gestión de procesos

Siendo la gestión de procesos una opción cercana y que puede aportar tantos beneficios, se pueden indicar algunas causas como:

- Inexistencia de departamentos dentro de las empresas destinadas a su organización.
- La visión de la empresa como si fuera un organismo: con un único personaje que piensa, el resto de los miembros sólo ejecuta.
- Descripción complicada de procesos y procedimientos como manuales extensos y redacción difícil.
- Que en las empresas no existen manuales ni diagramas con los procedimientos de la empresa obsoletos y desordenados.
- El desconocimiento de las técnicas de gestión de procesos es también una causa.
- Un mito antiguo que dice que a mayor productividad incrementará el desempleo.

3.1.6 Beneficios de la gestión por procesos

La Gestión por Procesos es la forma de gestionar toda la organización basándose en los Procesos. Entendiendo estos como una secuencia de actividades orientadas a generar un valor añadido sobre una ENTRADA para conseguir un resultado, y una SALIDA que a su vez satisfaga los requerimientos del Cliente, lo que produce numerosos beneficios para la empresa, como:

Capítulo 3: Levantamiento de Procesos

- Implantación de un sistema que permita operar con procesos definidos y documentados, que posibilite identificar, documentar, medir, controlar y mejorar continuamente los procesos que desarrolla una empresa o institución.
- Tener procesos controlados que permitan la obtención de productos con calidad.
- Atender las necesidades y requerimientos de los clientes
- Estructurar la organización de acuerdo a los procesos que maneja.
- Comprometer a todos los miembros de la organización de manera que aporten directamente a la obtención de los logros planificados de la empresa o institución.
- Generar valor agregado en todas las fases del proceso.
- Trabajar y hacer productos con calidad

3.1.7 Motores de la gestión BPM

Existen diversos motores que mueven la Gestión de Procesos de Negocio (BPM), la misma que se basa en los siguientes motores:

- Definir la planificación del negocio
- Establecer el programa institucional de calidad
- Observar el cumplimiento de leyes y reglamentos que normen la actividad de la empresa
- Crear y mantener una cadena de valor
- Crear y/o documentar procesos
- Entender que se está haciendo bien o mal a través de la comprensión de los procesos
- Medir y monitorear los procesos
- Retroalimentar políticas y estrategias
- Efectuar mejoras periódicas y/o requeridas de los procesos
- Automatización de procesos

Capítulo 3: Levantamiento de Procesos

3.2. Organización Internacional de Normalización ISO 9001

3.2.1 Resumen

La norma ha detallado 9 secciones, de la 4 a la 8 contienen los requisitos para la implementación del sistema de gestión de calidad y de la 0 a la 3 no contienen requisitos, sólo identifican el ámbito, las definiciones y los términos para la norma.

3.2.2 Sección 0.- Introducción

ISO 9001 considera el Sistema de Gestión de la Calidad la última versión en el año 2008 ha sido preparada por el Comité Técnico ISO/TC 176, Gestión y Aseguramiento de la calidad.

Las Normas ISO 9001 e ISO 9004 son normas de sistema de gestión de la calidad que se han disertado para complementarse entre sí, pero también pueden utilizarse de manera independiente.

La Norma ISO 9001 especifica los requisitos para un sistema de gestión de la calidad que pueden utilizarse para su aplicación interna por las organizaciones, para certificación o con fines contractuales.

La Norma ISO 9004 proporciona un enfoque más amplio sobre la gestión de la calidad que la Norma ISO 9001; trata las necesidades y las expectativas de todas las partes interesadas y su satisfacción.

Además explica que la prioridad de la Norma ISO 9001 es dar cumplimiento a los requisitos del cliente, en su implementación debe obedecer a una estrategia y que la norma es complementaria a los requisitos del servicio, su objetivo es evaluar la capacidad de la empresa para cumplir los requisitos del cliente, los legales y reglamentarios aplicables al producto y propios de la organización.

Capítulo 3: Levantamiento de Procesos

3.2.3 Sección 1.- Objeto Y Campo De Aplicación

El objeto de la norma consiste en que las empresas expongan la capacidad para proporcionar productos que cumplan con requisitos del cliente, legales y reglamentarios aplicables e incremente la satisfacción de clientes a través de una continua mejora.

En la parte aplicativa se explica que es genérica, que puede ser adaptada a cualquier tipo de organización sin importar tipo, tamaño y producto.

3.2.4 Sección 2.- Referencias Normativas

Utiliza como referencia la ISO 9000 donde indica la terminología y especificaciones de un sistema de gestión de calidad.

3.2.5 Sección 3.- Términos Y Definiciones

Hace referencia a la norma ISO 9000, que contiene el vocabulario adecuado aplicable al sistema de gestión de calidad como por ejemplo se explica que cuando se utiliza el término producto este significa también servicio.

3.2.6 Sección 4.- Requisitos Del Sistema

Indica los requisitos inevitables para establecer, documentar, implementar y mantener un sistema de gestión de calidad, dando cumplimiento la organización debe:

- Identificar los procesos y su interacción, esto lo hace a través de mapa y caracterizaciones de procesos.
- Establecer y documentar los procedimientos e instructivos necesarios para cada proceso.
- Asegurar los recursos para el desarrollo de las actividades planeadas.
- Establecer indicadores de gestión a los procesos.
- Establecer política, objetivos y manual de calidad.

Capítulo 3: Levantamiento de Procesos

- Asegurar el control de la documentación y los registros. Procedimiento para control de documentos y uno para control de registros. Procedimientos Obligatorios.

3.2.7 Sección 5.- Responsabilidad De La Dirección

Demuestra el compromiso que se debe tener obligatoriamente y lo debe realizar así:

- Comunicando la importancia de cumplir con los requisitos del cliente, los legales y los aplicables.
- Estableciendo y dando a conocer la política y los objetivos de calidad.
- Realizando revisiones periódicas al sistema de gestión de calidad establecido cada 6 meses.
- Nombrando un representante de la dirección con responsabilidades y autoridad suficiente para desarrollar cualquier actividad con el objetivo de mejorar el sistema de gestión de calidad.
- Asegurando la disponibilidad de recursos para el desarrollo de todas las actividades
- Definiendo y difundiendo manuales de funciones con las responsabilidades y la autoridad de los diferentes cargos.
- Asegurando una eficaz comunicación interna como ejemplo se tiene: Los comités, el chat, correos, celulares, fijos con extensiones en cada puesto de trabajo, memorandos, reuniones generales y más.

3.2.8 Sección 6.- Gestión de Recursos

Hace referencia a los siguientes recursos: el recurso humano, la infraestructura y el ambiente de trabajo.

- **Recurso humano:** el personal debe ser competente para realizar el trabajo, refiriéndose a competencia como la educación, formación, habilidad y experiencia.

Capítulo 3: Levantamiento de Procesos

Se debe definir la competencia en los manuales de funciones y se deben tener registros que lo evidencien.

- **Infraestructura:** debe contar con edificios, espacios de trabajo y equipos apropiados para el desarrollo de las labores como elevadores, computadores, programas contables, oficinas y teléfonos.
- **Ambiente de trabajo:** las condiciones bajo las cuales se realiza el trabajo deben ser las adecuadas para garantizar la conformidad con los requisitos del servicio como iluminación condiciones climáticas y más.

3.2.9 Sección 7.- Realización Del Producto

En esta sección se debe dar cumplimiento a los requisitos del cliente. Siendo la única sección donde se permiten exclusiones.

Y se debe cumplir con:

- Planificar y desarrollar los procesos necesarios para la prestación del servicio.
- Establecer procedimientos e instructivos para el desarrollo de las actividades.
- Demostrar que tanto los procesos de realización como el servicio resultante cumplan con los requisitos.
- Revisar los requisitos del servicio antes de que la organización se comprometa a proporcionarlo a un cliente ejemplo cotizaciones o aceptación de contratos.
- Especificar los requisitos determinados por el cliente, los legales y los reglamentarios aplicables al servicio.
- Implementar mecanismos de comunicación eficientes con el cliente para asegurar la información sobre el servicio, las consultas, la atención de pedidos.

Capítulo 3: Levantamiento de Procesos

- El servicio se debe prestar bajo condiciones controladas, se debe disponer de información sobre las características del servicio a prestar, como programaciones u órdenes de servicio.
- Cuando sea apropiado, se debe garantizar la trazabilidad del servicio a través de toda su realización. Cuando la trazabilidad sea un requisito se debe dejar registro de la identificación única del producto como un Registro Obligatorio.
- Se debe cuidar los bienes que son de propiedad del cliente, incluye la propiedad intelectual y los datos personales.

3.2.10 Sección 8.- Medición Análisis Y Mejora

En esta sección se recomienda hacer Medición de satisfacción de clientes, donde se pueden utilizar métodos como: encuestas donde se va a saber la opinión de usuarios.

Se deben realizar auditorías e informes de procedimientos de actividades donde se establezca indicadores de gestión a los procesos. Se debe dejar registro del servicio no conforme donde se indiquen las personas responsables.

Mejora continua de la eficacia del sistema usando como base la política y los objetivos de calidad, los resultados de auditorías internas y externas. Tomar acciones correctivas para eliminar las causas de las no conformidades para prevenir que vuelvan a ocurrir.

3.3 Modelado de procesos

Normalmente los sistemas que son conjuntos de procesos y subprocesos integrados en una organización son difíciles de comprender, amplios, complejos y confusos; con múltiples puntos de contacto entre sí y con un buen número de áreas funcionales,

Capítulo 3: Levantamiento de Procesos

departamentos y puestos implicados. Un modelo puede dar la oportunidad de organizar y documentar la información sobre un sistema.

Cuando un proceso es modelado, con ayuda de una representación gráfica o diagrama de proceso, pueden apreciarse con facilidad las interrelaciones existentes entre distintas actividades, analizar cada actividad, definir los puntos de contacto con otros procesos, así como identificar los subprocessos comprendidos. Al mismo tiempo, los problemas existentes pueden ponerse de manifiesto claramente dando la oportunidad al inicio de acciones de mejora.

Un adecuado modelo debe permitir:

- Mejorar el diseño de sistemas
- Facilitar la integración de nuevos sistemas o la mejora de los existentes.
- Servir de documentación de referencia para la comprensión de los sistemas
- Facilitar la comunicación entre las personas que intervienen en el diseño y funcionamiento de los sistemas
- La elaboración de un modelo que ofrezca estas ventajas requiere un método de representación específico, coherente, ágil, sencillo y flexible.
- El lenguaje convencional sea hablado o escrito presenta ciertas limitaciones e inconvenientes en la representación de sistemas.

3.3.1 Diagrama de flujo

Consiste en representar gráficamente hechos, situaciones, movimientos o relaciones de todo tipo, por medio de símbolos

Es un diagrama que expresa gráficamente las distintas operaciones que componen un procedimiento o parte de este, estableciendo su secuencia cronológica. Según su formato o

Capítulo 3: Levantamiento de Procesos

propósito, puede contener información adicional sobre el método de ejecución de las operaciones, el itinerario de las personas, las formas, la distancia recorrida el tiempo empleado.

Son también diagramas que emplean símbolos gráficos para representar los pasos o etapas de un proceso, útil para determinar cómo funciona realmente el proceso para producir un resultado. También permiten describir la secuencia de los distintos pasos o etapas y su interacción.

Es una actividad que agrega valor, pues el proceso que representa está ahora disponible para ser analizado, no sólo por quienes lo llevan a cabo, sino también por todas las partes interesadas que aportarán nuevas ideas para cambiarlo y mejorarlo.

3.3.1.1 Características de los diagramas de Flujo

- Sintética: La representación que se haga de un sistema o un proceso deberá quedar resumido en pocas hojas, de preferencia en una sola.
- Los diagramas extensivos dificultan su comprensión y asimilación, por tanto dejan de ser prácticos.
- De forma visible a un sistema o un proceso: Los diagramas nos permiten observar todos los pasos de un sistema o proceso sin necesidad de leer notas extensas. Un diagrama es comparable, en cierta forma, con una fotografía aérea que contiene los rasgos principales de una región, y que a su vez permite observar estos rasgos o detalles principales.
- Permitir al analista asegurarse que ha desarrollado todos los aspectos del procedimiento.
- Es un medio para establecer un enlace con el personal que eventualmente operará el nuevo procedimiento.

Capítulo 3: Levantamiento de Procesos

3.3.1.2 *Tipos de diagramas de Flujo*

Existen tres tipos de Flujo gramas o Diagramas de Flujo

- **Diagrama de flujo vertical:** También denominado gráfico de análisis del proceso. Es un gráfico en donde existen columnas verticales y líneas horizontales. En la columnas verticales están los símbolos o convencionales de operación, transporte, control, espera y archivo, los funcionarios involucrados en la rutina, el espacio recorrido para la ejecución y le tiempo invertido. El diagrama de flujo vertical destaca la secuencia de la rutina y es extremadamente útil para armar una rutina o procedimiento para ayudar en la capacitación del personal y para racionalizar el trabajo.
- **Diagrama de flujo horizontal:** Es diferente al anterior, al revés de la secuencia que se traslada verticalmente, esta lo hace de manera horizontal; este utiliza los mismos símbolos y convenciones que el vertical. El Diagrama de flujo horizontal destaca a las personas u organismos que participan en una determinada rutina o procedimiento.
- **Diagrama de flujo de bloques:** Es un diagrama de flujo que representa la rutina a través de una secuencia de bloques, cada cual con su significado y encadenados entre sí. Utiliza una simbología mucho más rica y variada que los diagramas anteriores, y no se restringe a líneas y columnas preestablecidas en el gráfico.
- Los analistas de sistemas utilizan mucho este diagrama para representar los sistemas, es decir, para indicar entradas, operaciones, conexiones, decisiones, archivado, que constituyen el flujo o la secuencia de las actividades de los sistemas

Capítulo 3: Levantamiento de Procesos

3.3.1.3 Símbolos en los Diagramas de Flujo

Los símbolos tienen significados específicos y se conectan por medio de flechas que indican el flujo entre los distintos pasos o etapas.

Los símbolos más comunes son:

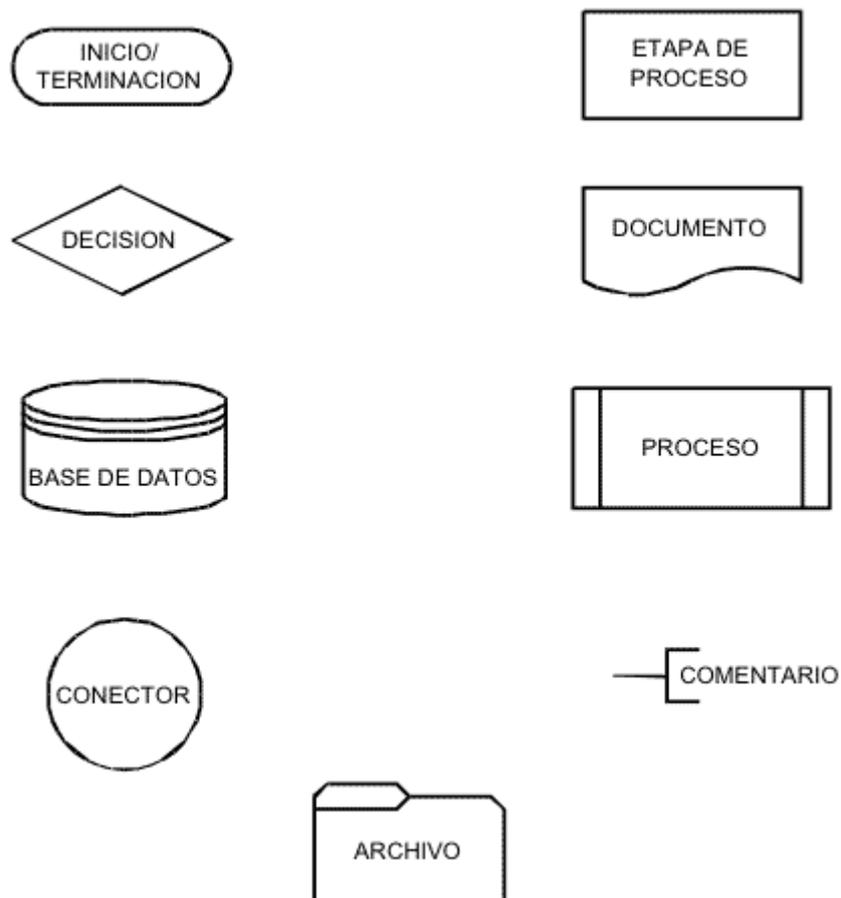


Figura 3. 1 Símbolos diagramas de flujo.

Fuente: Portal para investigadores y profesionales. (2015) Recuperado de <http://documents.mx/documents/portal-para-investigadores-y-profesionales.html>

3.3.1.4 Ventajas de los Diagramas de Flujo

- Permiten identificar las oportunidades de cambio en el proceso y desarrollar estimados de costos de la mala calidad.

Capítulo 3: Levantamiento de Procesos

- Favorecen la comprensión del proceso a través de mostrarlo como un dibujo. El cerebro humano reconoce fácilmente los dibujos. Un buen diagrama de flujo reemplaza varias páginas de texto
- Permiten identificar los problemas y las oportunidades de mejora del proceso. Se identifican los pasos redundantes, los flujos de los reproceso los conflictos de autoridad, las responsabilidades, los cuellos de botella, y los puntos de decisión.
- Involucra al personal de la organización en la resolución de problemas para reducir resistencia a cambios futuros.

3.3.1.5 Recomendaciones para el uso y aplicación de diagramas

- La redacción del contenido del símbolo de operación debe ser realizada con frases breves y sencillas
- El símbolo de conector puede ser alfabético o numérico, pero debe coincidir en los conectores de entrada y salida, cuando existen una gran cantidad de conectores, es conveniente adicionar un color al símbolo.
- Debe realizarse de forma limpia y ordenada.
- Especificar el objetivo del flujograma o diagrama de flujo.
- Pasar de lo conocido a lo desconocido. Es decir, identificar los subsistemas que, según se sabe, debe aparecer y desarrollar los que se relacionan con ellos.
- Cada página debe tener un encabezado que identifique claramente el proyecto, la gráfica, la flecha, El Autor (Carlos Burgos) y el número de páginas.
- Cuando las líneas de flujo son numerosas en diagramas complejos, utilizar conectores para reducir su número.
- Establecer el nivel en que van a construirse los flujos gramas o diagramas de flujo.

Capítulo 3: Levantamiento de Procesos

- Reunir los flujos de entrada, de modo que las líneas de flujo que aparecen entrando en un símbolo sean las menos posibles, lo mismo se harán con las líneas que salen.
- Escribir en los símbolos, usando el menor número posible de palabras.
- El símbolo de documento debe contener el nombre original de la forma que se utilice.

3.4 Sistema de Gestión de procesos de la empresa de Diseño e impresión DIKAPSA / Mapa de procesos

3.4.1 Proceso general de la empresa

Luego de haber recopilado información y datos se dedujo el mapa de procesos de manera general en la empresa DIKAPSA como se indica en la siguiente figura:



Figura 3. 2 Diagrama de flujo general.

Fuente: DIKAPSA. (2015). Recuperado de <http://www.dikapsa.com/>

Capítulo 3: Levantamiento de Procesos

A continuación se detalla cada uno de los procesos de manera general

3.4.1.1 *Procesos Gerenciales*

- **Planificación:** DIKAPSA se encuentra constantemente tomando decisiones anticipadas para proveer un buen futuro y realizarlas con éxito, de esta forma le ayuda a determinar: ¿Qué quiere conseguirse?, ¿Cómo se va a lograr? y ¿Cuáles son los recursos necesarios para ello?
- **Marketing:** Se refiere a la sección encargada de impulsar las ventas a nivel regional.
- **Diseño de proyectos y productos:** En esta área se encuentra la parte fundamental de la empresa con el fin para el que fue creada como es Diseñar.
- **Administración:** Donde se contabiliza y suministra de productos además de control interno.

3.4.1.2 *Procesos Operativos*

- **Cliente:** Es la parte más importante de la empresa a quien se debe dar la mayor atención rápida y eficaz.
- **Visita al cliente:** cuando sea necesaria la visita respecto a dudas en el proceso de producción o de instalación.
- **Atención al cliente:** Es la manera directa en la que entra en contacto el cliente con la empresa, cubre las necesidades de información que tiene el cliente.
- **Análisis de la necesidad del cliente:** Se reúnen todas las necesidades y requerimientos del cliente para poder brindar una conformidad y satisfacción.
- **Ventas:** Su comunicación se lo realiza entre el departamento de ventas y el cliente.
- **Diseño:** Se encarga se proporcionarle un modelo único y que sea de agrado del cliente para su previa elaboración.

Capítulo 3: Levantamiento de Procesos

- **Producción:** Cuando ya se indica de manera visual cualquier proyecto a entregar.
- **Facturación Entrega:** Un proceso obligatorio como retribución al trabajo realizado y entrega al usuario final o cliente.

3.4.1.3 Procesos de Soporte

- **Contabilidad:** Se encarga de la parte financiera de la empresa.
- **RRHH:** En esta sección se administra los contratos y control de deberes y derechos de los empleados.
- **Compras:** Trata de los insumos necesarios para que lleve a cabo las funciones la empresa.

Resumen: Como empresa de diseño e impresión DIKAPSA tiene varias líneas de producción y mediante recolección de información de estado inicial se llegó como conclusión final el modelo indicado en la figura anterior de los procesos en general enfatizando el orden y generando responsabilidades específicas al personal.

3.4.2 Gigantografías

Para el análisis de procesos se procedió a graficar el mapa de procesos del diseño e impresión de gigantografías que es uno de los productos a sistematizar.

Para la producción de una gigantografía se debe seguir el siguiente proceso indicado en la figura anterior, se detalla cada uno de los ítems del proceso:

Capítulo 3: Levantamiento de Procesos

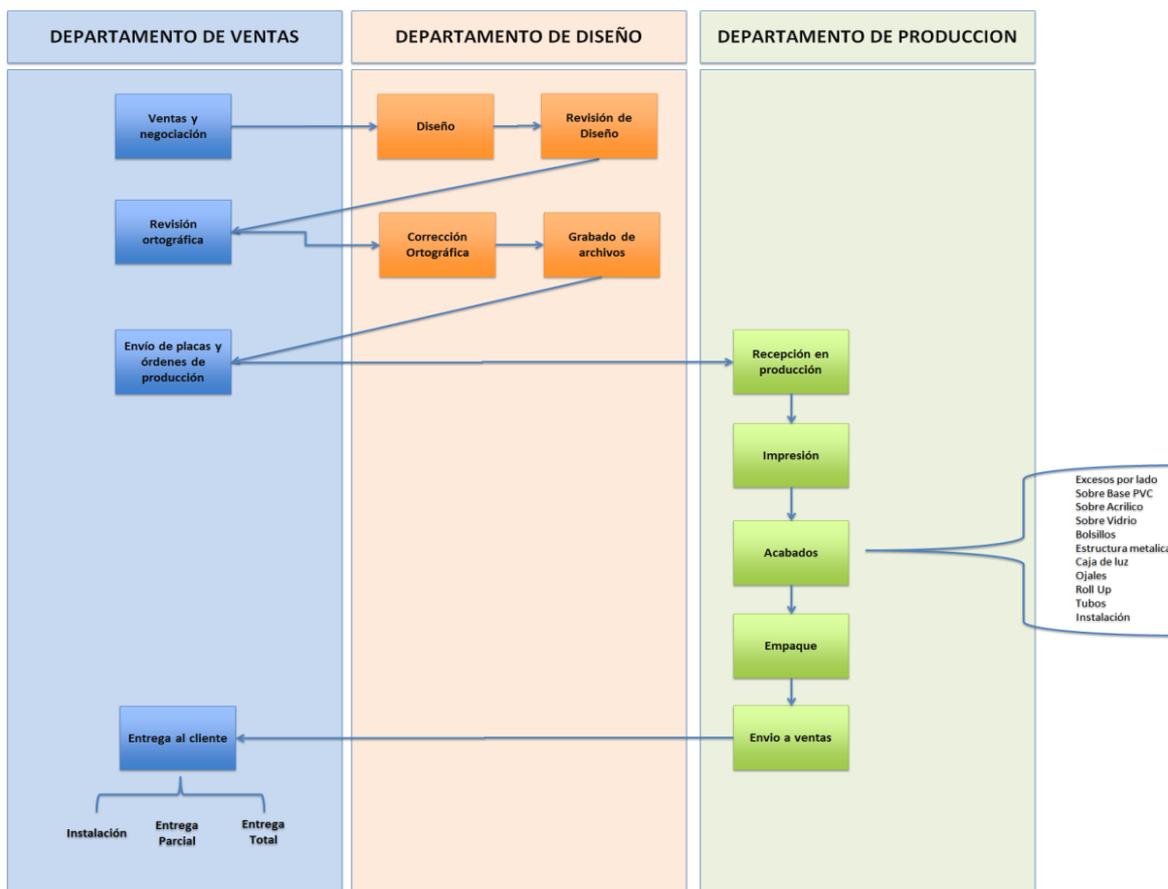


Figura 3. 3 Portal para investigadores y profesionales

Fuente: DIKAPSA. (2015). Recuperado de <http://www.dikapsa.com/>

3.4.2.1 Departamento de ventas

Ventas y negociación: Es el primero encuentro en el que el cliente entra en contacto con la empresa, específicamente con el departamento de ventas donde el cliente fija detalles de los productos, se realiza el proceso de venta y se definen medidas, materiales, resolución de impresión, precios, anticipos y plazos de entrega.

Revisión ortográfica: Se revisa si hubiere faltas ortográficas o errores de diseño previo a su envío.

Envío de placas y órdenes de producción: Se envía la orden de producción con todos los documentos necesarios y se registra la salida a la planta de producción.

Capítulo 3: Levantamiento de Procesos

Entrega al cliente:

- **Instalación:** Se realiza la instalación pero antes debe haber un documento que conste todo lo que se va a instalar.
- **Entrega parcial:** Cuando el producto no se entrega en su totalidad se realiza la entrega de los productos que fueron elaborados y entregados en ciertas cantidades.
- **Entrega total:** Cuando el producto fabricado es en su totalidad, se realiza la entrega total del bien graficado.

3.4.2.2 *Departamento de diseño*

Diseño: Tomando en cuenta las requerimientos se elabora de diseño de acuerdo a las especificaciones tomadas en ventas.

Revisión de Diseño: El cliente revisa el diseño realizado y procede a realizar correcciones en caso de que haya, y finalmente se aprueba el diseño.

Corrección: El diseñador corrige el diseño de acuerdo a las indicaciones de ventas.

Grabado de archivos: El diseñador graba los archivos necesarios y envía a la planta de producción a través de la red.

3.4.2.3 *Departamento de producción*

Recepción de la orden en producción: Se registra el ingreso de la orden de producción a la planta.

Impresión: La impresión se lo realiza conforme a los requerimientos adquiridos en Ventas, ya sea material, tamaños, resolución y más.

Acabados: Sobre Base PVC, Sobre Acrílico, Bolsillos, Estructura Metálica, Caja de Luz, Ojales, Roll Up, Tubos, Instalación.

Capítulo 3: Levantamiento de Procesos

Empaques: Finalizada la producción se realiza el empaque del producto.

Envío a ventas: Se realiza el envío desde el Departamento de Producción al Departamento de Ventas.

Resumen: El siguiente proceso describe su ciclo de vida previo a la elaboración del producto en este caso como es las gigantografías uno de los varios productos que maneja DIKAPSA que se tratará en el transcurso del presente proyecto.

3.4.3 Comprobantes (Libretines)

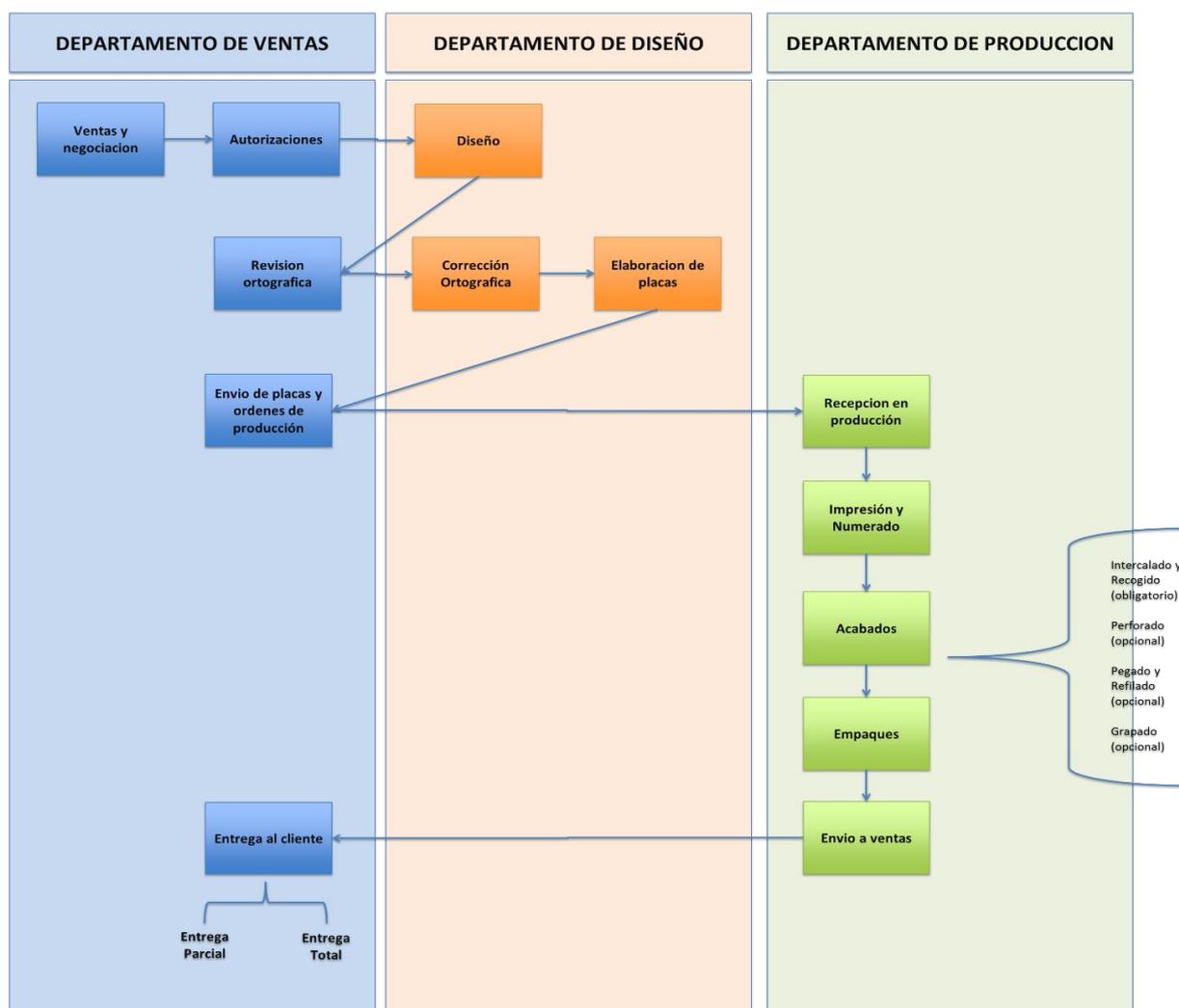


Figura 3. 4 Proceso de producción afiches

Fuente: DIKAPSA. (2015). Recuperado de <http://www.dikapsa.com/>

Capítulo 3: Levantamiento de Procesos

3.4.3.1 Departamento de ventas

Ventas y negociación: Es el primero encuentro en el que el cliente entra en contacto con la empresa, específicamente con el departamento de ventas donde el cliente fija detalles de los productos, se realiza el proceso de venta y se definen medidas, materiales, resolución de impresión, precios, anticipos y plazos de entrega.

Autorizaciones: La persona que toma la orden debe sacar autorizaciones del SRI.

Revisión ortográfica: Se procede con la revisión si hubiere faltas ortográficas o errores de diseño.

Envío de placas y órdenes de producción: Se envía las placas con todos los documentos necesarios a producción y se registra la salida de la orden de producción.

Entrega al cliente:

- **Instalación:** Se realiza la instalación pero antes debe haber un documento que conste todo lo que se va a instalar.
- **Entrega parcial:** Cuando el producto no se entrega en su totalidad se realiza la entrega de los productos que fueron elaborados y entregados en ciertas cantidades.
- **Entrega total:** Cuando el producto fabricado es en su totalidad, se realiza la entrega total del bien graficado.

3.4.3.2 Departamento de diseño

Diseño: Elaboración de diseño de acuerdo a las especificaciones tomadas en el departamento de ventas.

Capítulo 3: Levantamiento de Procesos

Corrección ortográfica: El diseñador corrige el diseño en el caso que hubiere fallas ortográficas de acuerdo a las indicaciones del departamento de ventas.

Elaboración de placas: El diseñador imprime las placas en el material poliéster.

3.4.3.3 *Departamento de producción*

Recepción de producción: se registra el ingreso de la orden de producción a la planta de producción.

Impresión y numerado: La impresión y numerado se lo realiza conforme a los requerimientos adquiridos en Ventas, ya sea material, tamaños o resolución.

Acabados: el vendedor deberá seleccionar que acabados debe tener el producto final.

- **Intercalado y Recogido.** - Se lo realiza obligatoriamente ya que este proceso necesariamente lo requiere.
- **Perforado.**- este acabado es opcional y se deberá consultar al cliente si lo requiere.
- **Pegado y Refilado.**- todos los comprobantes deben pasar por este proceso.
- **Grapado.**- este acabado es opcional y el vendedor deberá seleccionarlo en el caso que el cliente lo requiere.

Empaques: Finalizada la producción se realiza el empaque del producto.

Envío a ventas: Producción envía el producto terminado a ventas ya sea de forma parcial o total.

Resumen: Este proceso describe su ciclo de vida previo a la elaboración del producto en este caso como es de los comprobantes uno de los varios productos que produce la empresa de diseño DIKAPSA y que se tratará en el transcurso del presente proyecto.

Capítulo 4

DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA



En este capítulo se describe los resultados obtenidos durante el desarrollo del SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA aplicando la metodología de trabajo SCRUM en la empresa DIKAPSA.

Con la utilización de la metodología SCRUM se logra un proceso de desarrollo efectivo, ágil, iterativo e incremental que respeta las etapas tradicionales de un proyecto (Planificación, Análisis, Diseño, Construcción y Pruebas, e Implementación). Con las que facilitan su administración y control.

Incluye junto con la descripción de este ciclo de vida iterativo e incremental para el proyecto, los artefactos o documentos con los que se gestionan las tareas de adquisición y suministro: requisitos, monitorización y seguimiento del avance, así como las responsabilidades y compromisos de los participantes en el proyecto.

Capítulo 4: Desarrollo e Implementación del Sistema

4. DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA

4.1. Descripción General de la Metodología en el Sistema

4.1.1. Fundamentación

Las principales razones del uso de un ciclo de desarrollo iterativo e incremental de tipo SCRUM para la ejecución de este proyecto son:

Sistema modular. Las características del SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA permiten desarrollar una base funcional mínima y sobre ella ir incrementando las funcionalidades o modificando el comportamiento o apariencia de las ya implementadas.

Entregas frecuentes y continuas al cliente de los módulos terminados, de forma que puede disponer de una funcionalidad básica en un tiempo mínimo y a partir de ahí un incremento y mejora continua del sistema.

Previsible inestabilidad de requisitos.

- Es posible que el sistema incorpore más funcionalidades de las inicialmente identificadas.
- Es posible que durante la ejecución del proyecto se altere el orden en el que se desean recibir los módulos o historias de usuario terminadas.
- Para el cliente resulta difícil precisar cuál será la dimensión completa del sistema, y su crecimiento puede continuarse en el tiempo suspenderse o detenerse.

Antes de iniciar cualquier iteración SCRUM recomienda definir algunos aspectos importantes como son los objetivos para la realización del sistema, requerimientos y tareas que nos servirán para cubrir cada una de las iteraciones

Capítulo 4: Desarrollo e Implementación del Sistema

4.2. Personas y roles del proyecto

En esta metodología existen tres tipos de roles que conforman lo que se conoce como el equipo de SCRUM, En la que cada miembro del equipo es simplemente un miembro del equipo y su participación será coordinada y en equipo: Dueño del Producto (Product Owner), Equipo Técnico/Desarrollo (Team Group), y el SCRUM Master.

Tabla 4. 1
Personas y roles

Persona	Contacto	Rol
Ing. Mauricio Rea	0986099536	Tutor / SCRUM Manager
Diego Oña	0998677846	Gestor de Producto / Product Owner
Carlos Burgos	0981363431	Equipo Técnico / Team Group

Nota. Elaborado por el Autor Carlos Burgos.

4.3. Artefactos

SCRUM nos propone herramientas o “artefactos” para mantener organizados nuestros proyectos, Estos artefactos nos ayudan a planificar y revisar cada uno de los Sprints conforme vamos desarrollando, aportando una estructura organizada en el momento de efectuar cada uno de las tareas.

Para la planificación de los Sprints, la metodología de SCRUM nos brinda los artefactos que ayudaran a la realización de las tareas programadas, el primer artefacto que tenemos que crear para la planificación del desarrollo del sistema es el Producto BackLog, este artefacto nos permite conocer las tareas que tenemos que realizar y categorizar por prioridades, por lo cual se presenta el siguiente orden de los artefactos a utilizar.

Documentos: Pila de producto o Product Backlog y Pila de sprint o Sprint Backlog.

Capítulo 4: Desarrollo e Implementación del Sistema

Sprint

Incremento: Gráficas para registro y seguimiento del avance.

- Gráfica de producto o Burn Up
- Gráfica de avance o Burn Down.

Comunicación y reporting directo.

- Reunión de inicio de sprint
- Reunión técnica
- Reunión de cierre de sprint y entrega del incremento

4.4. Documentos

4.4.1. Pila de producto (Product Backlog)

Dentro de la primera reunión de retrospectiva Nro/Código que recomienda SCRUM en su marco de trabajo, se ha procedido a la elaboración de la lista de requerimientos definidos por el Producto Owner Diego Oña, se ha considerado establecer un formato de la lista en la que incluya al menos la siguiente información para cada columna de la tabla.

4.4.1.1. Identificador (ID) de la Historia

Código que identifica a la historia de forma unívoca, una vez asignado, no debe ser re-usado en otra historia, ni siquiera si la historia es descartada. El código identifica la historia en otros documentos, como por ejemplo la plantilla de historias de usuario.

4.4.1.2. Enunciado de la Historia

Nombre de la historia, el cual debe ser el mismo que se utiliza en otros

Capítulo 4: Desarrollo e Implementación del Sistema

documentos. Se puede utilizar el formato siguiente; Como un [Rol], Necesito [Descripción de la Funcionalidad], con la finalidad de [Razón o Resultado]

4.4.1.3. Origen

Área a la que va ser destinado dicha funcionalidad del requerimiento.

4.4.1.4. Alias

Título de la historia alternativo a la descripción, que servirá para identificar más fácilmente la historia sin tener que repetir todo su enunciado. Se puede utilizar por ejemplo el nombre de la funcionalidad o requerimiento que se pretende desarrollar.

4.4.1.5. Estado

Identifica los posibles estados de la historia durante su ciclo de vida:

- **Vacío:** La historia fue identificada pero aún no ha sido asignada a una iteración.
- **Planificada:** La historia fue asignada a una iteración y aún no ha comenzado su ejecución. Puede tener este estado incluyendo en la iteración donde está planificado ejecutarla pero que aún no ha comenzado.
- **En Proceso:** La historia fue seleccionada por el equipo y está en proceso de desarrollo en ejecución.
- **Hecho / Donde:** La historia fue desarrollada. Es importante clarificar la definición de “Hecho” con el equipo de trabajo. “Hecho” no sólo incluye el desarrollo sino la integración y pruebas integrales del Software. Una historia hecha puede presentarse al dueño de producto para sus pruebas de aceptación.
- **Descartada:** Se determinó que la historia ya no es relevante, su contenido se incluyó en otro grupo de historias o fue cancelada.

4.4.1.6. Dimensión / Esfuerzo

Medida del esfuerzo (tamaño) que implica desarrollar la historia, existen distintos

Capítulo 4: Desarrollo e Implementación del Sistema

métodos para medirlo, un ejemplo es los “puntos de historia” una medida de complejidad no necesariamente relacionado con jornadas o días. Otra forma de medirlo es con días o jornadas ideales. 1SP = 1 día = 8 horas.

4.4.1.7. Iteración (Sprint)

Iteración o Sprint al que se asigna la historia. Esta asignación puede cambiar en cada iteración donde se haga la revisión de la pila de producto (ProductBacklogReview), según las prioridades indicadas por el dueño de producto. Por medio de este campo se puede crear un “Plan de Salidas a Productivo”.

4.4.1.8. Prioridad

Siguiendo el marco de trabajo ágil y SCRUM, se le deben asignar prioridades a las historias, según las instrucciones del dueño de producto (ProductOwner). De esta forma pueden ordenarse. Las historias de mayor prioridad deben ser las que agregan más valor al negocio, y deben ser originadas en sus necesidades.

4.4.1.9. Comentarios

Comentarios o detalles relacionados que expliquen la historia. Para definiciones de mayor longitud deben usarse documentos externos, por ejemplo la plantilla de historias de usuario.

Capítulo 4: Desarrollo e Implementación del Sistema

Identificador (ID) de la Historia	Enunciado de la Historia	Alias	Origen/Módulo	Estado	Dimensión / Esfuerzo	Iteración (Sprint)	Prioridad	Comentarios
REQ-DIKAPSA-01	Como usuario de ventas, necesito que permita registrar a los clientes, con la finalidad tener una base de información.	Módulo Comercial.	Comercial	Hecho	30SP	SIGPROPED-SP1	100	
REQ-DIKAPSA-02	Como usuario de ventas, necesito que permita tomar órdenes de producción estandarizadas, con la finalidad de que sea el proceso rápido, eficiente y estandarizado.	Toma de ODT Estandarizadas	Ventas	Hecho	30SP	SIGPROPED-SP2	95	
REQ-DIKAPSA-03	Como un usuario de ventas, necesito que imprima de manera secuencial cada orden de producción, con la finalidad de conocer a detalle la información relevante de la misma.	Impresión Secuencias ODT	Ventas	Hecho	30SP	SIGPROPED-SP2	90	
REQ-DIKAPSA-04	Como un usuario de ventar, necesito que permita poner precios de manera manual con proyección a precios automáticos, con la finalidad de facilitar al usuario la eficiencia en lo toma de órdenes de trabajo.	Precios manuales automáticos	Ventas	Hecho	30SP	SIGPROPED-SP2	90	
REQ-DIKAPSA-05	Como un usuario de ventas, necesito que genere un anticipo del total de la orden, con la finalidad de que garantice la elaboración de dicha orden de producción.	Anticipo total de la Orden	Ventas	Hecho	30SP	SIGPROPED-SP2	90	
REQ-DIKAPSA-06	Como gerente comercial, necesito que señale la fecha y hora de tomada una orden de producción, con la finalidad de saber en qué momento se receptaron dichas órdenes.	Hora tomada una orden de producción	Producción	Hecho	30SP	SIGPROPED-SP2	90	
REQ-DIKAPSA-07	Como jefe de producción, necesito que permita guardar Productos con cada una de líneas de producción y acabados, además debe permitir ingresar los procesos de producción, con la finalidad de tener una base formal y que sirva para hacer un seguimiento.	Módulos de gestión de Producción.	Producción	Hecho	30SP	SIGPROPED-SP3	87	
REQ-DIKAPSA-08	Como un operador del sistema, necesito que genere un código numérico, con la finalidad de registrar en las diferentes etapas del proceso de elaboración de la orden de producción.	Código supervisor en etapas de ODT	Producción	Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-09	Como gerente comercial, necesito que el código generado sirva para indicar la hora de entrada y la hora de salida de cada etapa del proceso, con la finalidad de conocer los tiempos de duración en cada etapa.	Código supervisor de tiempos de ODT en proceso	Comercial	Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-10	Como un usuario público, necesito que se indique en un monitor la fecha de entrega y la etapa del proceso en la que se encuentra, con la finalidad de conocer a ciencia cierta cómo avanza el proceso de elaboración de dicho producto	Monitor de tiempos entregas ODT	Publico	Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-11	Como un usuario público, necesito que de alertas de cualquier tipo cuando la orden se encuentre en las etapas iniciales del proceso y este cerca la fecha de entrega, con la finalidad de dar prioridades en las producciones y evitar retrasos.	Alertas en etapas Iniciales de proceso	Publico	Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-12	Como un usuario de producción, necesito que la orden de producción tomada se la pueda ver al instante en la planta de producción, con la finalidad de saber que materiales se debe preparar para la elaboración de dicho orden de producción.	Ver ODT en área producción	Producción	Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-13	Como un usuario de ventas/diseño, necesito se pueda enviar archivos desde las oficinas de DIKPASA a producción, con la finalidad de facilitar la perdida de información adjunto a la orden de producción.	Enviar archivos		Hecho	30SP	SIGPROPED-SP04	80	
REQ-DIKAPSA-14	Como gerente comercial, necesito que se pueda controlar a los usuarios por sus roles y auditorias en cambios importantes, con la finalidad de tener separados por sus funciones y cargos.	Reporte ODT tomadas	Comercial	Hecho	30SP	SIGPROPED-SP05	75	
REQ-DIKAPSA-15	Como gerente comercial, necesito que genere reportes de órdenes tomadas, con la finalidad de evaluar las ventas.	Reporte ODT tomadas	Comercial	Hecho	30SP	SIGPROPED-SP-06	65	
REQ-DIKAPSA-16	Como gerente comercial, necesito genere un reporte por productos, rotación y cantidad, con la finalidad de conocer que productos y líneas de producción son más apetecibles para el cliente y su continua mejora.	Reporte por producto cantidad	comercial	Hecho	30SP	SIGPROPED-SP-06	65	
REQ-DIKAPSA-17	Como gerente comercial, necesito genere reportes de tiempos en cada etapa del proceso de producción de la orden de trabajo, con la finalidad de conocer, equilibrar y ajustar dichos tiempos con el personal.	Reporte de tiempos en etapas	Comercial	Hecho	30SP	SIGPROPED-SP-06	65	
REQ-DIKAPSA-18	Como gerente comercial, necesito que genere un reporte de todas las órdenes de trabajos entregadas tarde, con la finalidad de poder determinar las causas de esa demora y corregir.	Reporte ODT entregadas tarde	comercial	Hecho	30SP	SIGPROPED-SP-06	65	
REQ-DIKAPSA-19	Como gerente comercial, necesito que genere un reporte de compras por cliente y por categoría de productos, con la finalidad de poder distinguir a nuestros posible potenciales clientes.	Reporte de compras por cliente y categoría	comercial	Hecho	30SP	SIGPROPED-SP-06	65	
REQ-DIKAPSA-20	Como gerente comercial, necesito un reporte por vendedores, con la finalidad de saber cada una de las ventas realizadas por los usuarios de ventas.	Reportes por vendedores	comercial	Hecho	30SP	SIGPROPED-SP-06	65	

Figura 4. 1 Pila del producto

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.4.2. Planificación de la Pila del sprint (Sprint Backlog)

Una vez definida la lista de requerimientos del cliente, se procede a crear la planificación de cada uno de los Sprints en una tabla donde nos permitirá conocer cada una de las iteraciones, en la que definiremos las tareas a desarrollarse de acuerdo al nivel de prioridad establecida a cada uno de los requerimientos planteados en la Pila de Producto por el Product Owner (Dueño del Producto).

Esta pila del sprint también es una herramienta que provee de soporte para la comunicación directa con el equipo de desarrollo, donde el estado va ir cambiando mientras avanza la iteración. Para su elaboración y planificación se ha considerado necesario reunir con la siguiente información:

- **ID:** Código de identificación de la tarea o sprint a desarrollarse
- **Nombre de Tarea:** Identificación de la tarea para el grupo de desarrolladores
- **Sprint:** Sprint al que conforma dentro de la planificación para los entregables programados
- **Prioridad:** Prioridad que tiene el Sprint dentro del Grupo para ser implementado y entregado.
- **Trabajo Previsto/ Estimación Inicial:** Tiempo en días previsto de la duración del sprint en su desarrollo.
- **Trabajo Actual:** número de días que presenta de desarrollo en el sprint escogido por el equipo de desarrollo.
- **Porcentaje completado:** Porcentaje en números del avance del sprint determinado por equipo de desarrollo.

Capítulo 4: Desarrollo e Implementación del Sistema

- **Trabajo:** número de días reales
- **Duración/Estimación Ajustada:** Re ajuste al tiempo estimado en días para la elaboración del sprint.
- **Comienzo:** fecha de inicio del sprint
- **Fin:** fecha de finalización del sprint
- **Recurso:** Hipervínculo directo a la lista de requerimientos presentado por Product Owner (Dueño del Producto) en la Pila de Producto (Product Backlog)

Capítulo 4: Desarrollo e Implementación del Sistema

Id	Nombre de tarea	Sprint	Prioridad	Trabajo Previsto/ Estimación Inicial	Trabajo Actual	% completado	Trabajo	Duración/Estimación Ajustada	Comienzo	Fin	Recurso
	Arquitectura de la Solución - SIGPROPED		Alta	305 días	0 días	0%	0 días	303 días?	dom 16/02/14	mié 15/04/15	
TR1	Requerimientos		Alta	1 día	0 días	0%	0 días	1 día	dom 16/02/14	dom 16/02/14	
TR2	Levantamiento de procesos y diagramación		Media	18 días	0 días	0%	0 días	18 días	mar 18/02/14	jue 13/03/14	
TR3	Plataforma Tecnológica		Media	7 días	0 días	0%	0 días	14 días	vie 14/03/14	mié 02/04/14	
TR4	Diseño de Datos		Alta	34 días	0 días	0%	0 días	34 días	jue 03/04/14	mar 20/05/14	
SP1	Primer Sprint (Desarrollo de Módulos Comercial)		Alta	34 días?	0 días	0%	0 días	34 días?	mié 21/05/14	lun 07/07/14	
	Desarrollo de Catálogo Clientes	SP1	Alta	5 días	0 días	0%	0 días	5 días	mié 21/05/14	mar 27/05/14	REQ-DIKAPSA-01
	Desarrollo de Catálogo Ordenes de trabajos - Detalles ODT - Detalles ODT Acabados	SP1	Alta	20 días	0 días	0%	0 días	20 días	mié 28/05/14	mar 24/06/14	REQ-DIKAPSA-01
	Desarrollo de Catálogo Entregas ODT - Tipos Entregas	SP1	Media	9 días	0 días	0%	0 días	9 días	mié 25/06/14	lun 07/07/14	REQ-DIKAPSA-01
SP2	Segundo Sprint (Desarrollo Requerimientos Comercial)		Alta	34 días?	0 días	0%	0 días	34 días?	mar 08/07/14	vie 22/08/14	
	Toma de ODT Estandarizadas	SP2	Alta	25 días	0 días	0%	0 días	25 días	mar 08/07/14	lun 11/08/14	REQ-DIKAPSA-02
	Impresión Secuencias ODT	SP2	Media	2 días	0 días	0%	0 días	2 días	mar 12/08/14	mié 13/08/14	REQ-DIKAPSA-03
	Precios manuales automáticos	SP2	Alta	3 días	0 días	0%	0 días	3 días	jue 14/08/14	lun 18/08/14	REQ-DIKAPSA-04
	Anticipo total de la Orden	SP2	Alta	2 días	0 días	0%	0 días	2 días	mar 19/08/14	mié 20/08/14	REQ-DIKAPSA-05
	Hora tomada una orden de producción	SP2	Alta	2 días	0 días	0%	0 días	2 días	jue 21/08/14	vie 22/08/14	REQ-DIKAPSA-06
SP3	Tercer Sprint (Desarrollo Módulos EGP)		Alta	34 días	0 días	0%	0 días	34 días	sáb 23/08/14	mié 08/10/14	
	Desarrollo Catálogo Líneas de Producción - Productos	SP3	Alta	5 días	0 días	0%	0 días	5 días	sáb 23/08/14	jue 28/08/14	REQ-DIKAPSA-07
	Desarrollo Catálogo Tipos Acabados - Acabados	SP3	Alta	5 días	0 días	0%	0 días	5 días	vie 29/08/14	jue 04/09/14	REQ-DIKAPSA-07
	Desarrollo Catálogo Departamentos - Etapas - Procesos - Secuencias Procesos	SP3	Alta	7 días	0 días	0%	0 días	7 días	vie 05/09/14	lun 15/09/14	REQ-DIKAPSA-07
	Desarrollo Catálogo Historial Tareas - Tareas - Involucrados Tareas - Archivos Adjuntos	SP3	Alta	17 días	0 días	0%	0 días	17 días	mar 16/09/14	mié 08/10/14	REQ-DIKAPSA-07
SP4	Cuarto Sprint (Desarrollo Requerimientos Control ODT)		Alta	34 días?	0 días	0%	0 días	34 días?	jue 09/10/14	mar 25/11/14	
	Código supervisor en etapas de ODT	SP4	Alta	6 días	0 días	0%	0 días	6 días	jue 09/10/14	jue 16/10/14	REQ-DIKAPSA-08
	Código supervisor de tiempos de ODT en proceso	SP4	Alta	5 días	0 días	0%	0 días	5 días	vie 17/10/14	jue 23/10/14	REQ-DIKAPSA-09
	Monitor de tiempos entregas ODT	SP4	Alta	10 días	0 días	0%	0 días	10 días	vie 24/10/14	jue 06/11/14	REQ-DIKAPSA-10
	Alertas en etapas Iniciales de proceso	SP4	Alta	4 días	0 días	0%	0 días	4 días	vie 07/11/14	mié 12/11/14	REQ-DIKAPSA-11
	Ver ODT en área producción	SP4	Alta	5 días	0 días	0%	0 días	5 días	jue 13/11/14	mié 19/11/14	REQ-DIKAPSA-12
	Enviar archivos	SP4	Alta	4 días	0 días	0%	0 días	4 días	jue 20/11/14	mar 25/11/14	REQ-DIKAPSA-13
SP5	Quinto Sprint (Desarrollo de Módulo de Seguridad, Aditorial y Parámetros)		Media	34 días	0 días	0%	0 días	34 días	mié 26/11/14	lun 12/01/15	
	Desarrollo de Catálogo de Usuarios - Grupos - Grupos Permisos	SP5	Media	11 días	0 días	0%	0 días	11 días	mié 26/11/14	mié 10/12/14	REQ-DIKAPSA-14
	Desarrollo de Catálogo de Auditorías - Log Accesos	SP5	Media	11 días	0 días	0%	0 días	11 días	jue 11/12/14	jue 25/12/14	REQ-DIKAPSA-14
	Desarrollo de Catálogo de Parámetros Colores - Formatos - Tipos Pagos	SP5	Media	12 días	0 días	0%	0 días	12 días	vie 26/12/14	lun 12/01/15	REQ-DIKAPSA-14
SP6	Sexto Sprint (Elaboración de Reportes)		Media	34 días	0 días	0%	0 días	33 días	mar 13/01/15	jue 26/02/15	
	Reporte ODT tomadas	SP6	Media	5 días	0 días	0%	0 días	5 días	mar 13/01/15	lun 19/01/15	REQ-DIKAPSA-15
	Reporte por producto cantidad	SP6	Media	6 días	0 días	0%	0 días	6 días	mar 20/01/15	mar 27/01/15	REQ-DIKAPSA-16
	Reporte de tiempos en etapas	SP6	Media	6 días	0 días	0%	0 días	6 días	mié 28/01/15	mié 04/02/15	REQ-DIKAPSA-17
	Reporte ODT entregadas tarde	SP6	Media	5 días	0 días	0%	0 días	5 días	jue 05/02/15	mié 11/02/15	REQ-DIKAPSA-18
	Reporte de compras por cliente y categoría	SP6	Media	6 días	0 días	0%	0 días	6 días	jue 12/02/15	jue 19/02/15	REQ-DIKAPSA-19
	Reportes por vendedores	SP6	Media	5 días	0 días	0%	0 días	5 días	vie 20/02/15	jue 26/02/15	REQ-DIKAPSA-20
SP7	Séptimo Sprint (Pruebas)		Media	34 días	0 días	0%	0 días	34 días	vie 27/02/15	mié 15/04/15	
	Ejecución de pruebas de los sprint del sistema	SP7	Media	34 días	0 días	0%	0 días	34 días	vie 27/02/15	mié 15/04/15	
SP8	Octavo Sprint (Publicación)		Media	34 días	0 días	0%	0 días	34 días	vie 27/02/15	mié 15/06/15	
	Puesta en producción integración de Sprints	SP8	Media	34 días	0 días	0%	0 días	34 días	vie 16/04/15	mié 02/06/15	

Figura 4. 2 Planificación de pila del Spring

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.5. Implementación de la Arquitectura de la Solución

En esta sección se procede a describir tanto la implementación del diseño de datos como la arquitectura del sistema, lo que permitirá conocer y comprender el ambiente en el que se encuentra ejecutándose. Los estándares y tecnologías usadas en su totalidad son de software libre por lo que se puede incrementar y permitir la compatibilidad e interoperabilidad entre distintos componentes tanto de hardware y software, ya que cualquier analista con el conocimiento técnico necesario y recursos puede modificar y construir más módulos.

4.5.1. Implementación de Diseño de Datos

Para el desarrollo del sistema se centra en la realización de una base de datos parametrizable que almacene información sobre las ventas de órdenes de trabajo, y el registro de información que muestre el avance del proceso de manufactura en cada una de las tareas a realizarse en dicha línea de producción.

El Modelo Entidad Relación para el sistema de gestión de procesos para la empresa DIKAPSA consta de tablas relacionadas entre si para dar funcionalidad y servir a una interfaz totalmente amigable que permita principalmente las siguientes operaciones.

- Mantenimiento de todos los objetos representados en la base de datos.
- Gestión de Clientes, Productos, Acabados, Líneas de Producción, Procesos y Tareas.
- Gestión de avances de tareas en procesos de producción de un producto.

Capítulo 4: Desarrollo e Implementación del Sistema

4.5.2. Arquitectura del Sistema

Es importante conocer la arquitectura del sistema implementada para así de esta forma entender, interpretarlo y mantenerlo. El sistema maneja un patrón de diseño modelo, vista, controlador con una arquitectura n capas divide al sistema en distintas unidades funcionales; el cual le permite tener una escalabilidad al futuro con la implementación de varios módulos.

Esta división hace que sea fácilmente mantenerle y extensible tanto con responsabilidades en capa del cliente, presentación, lógica del negocio.

Basada en la edición estándar de la plataforma Java EE, los contenedores nos provee un entorno de ejecución web programa un vez, despliega y ejecuta donde se desee. También nos brinda los servicios de seguridad, transacciones, administración del ciclo de vida y persistencia.

El Sistema SIGPROPED es una aplicación web que utiliza múltiples APIs Java EE.

Diseño Gráfico de la Arquitectura del Proyecto SIGPROPED

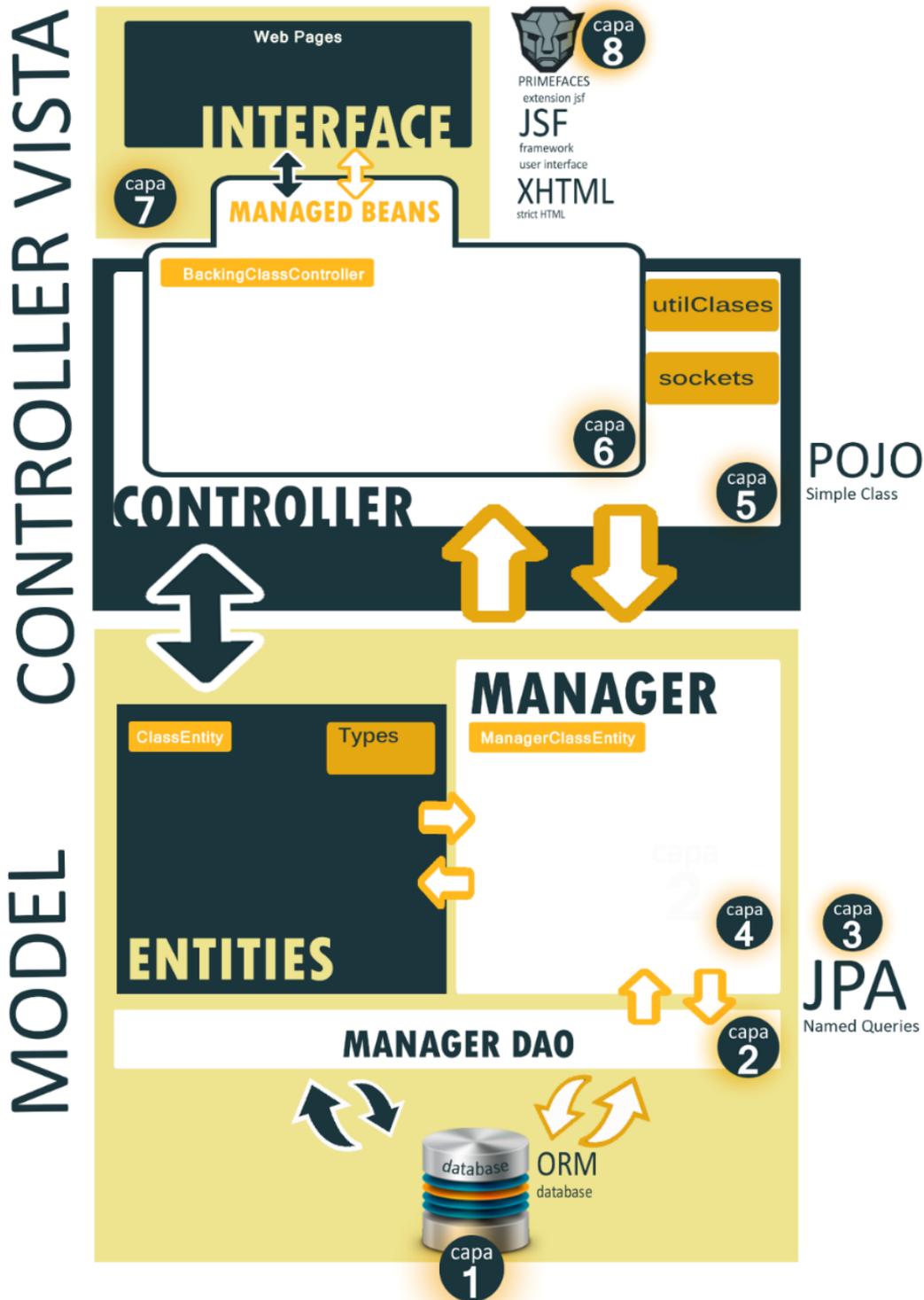


Figura 4. 4 Arquitectura del Sistema.
Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.5.2.1. Modelo

- **Java Persistent API:** Estándar de Java encargado de automatizar dentro de lo posible la persistencia de nuestros objetos en base de datos.
- **Manager DAO:** Contiene detalles específicos de operación o aplicación como el Objeto de Acceso a Datos.
- **Entities:** Contiene las entidades, objetos de persistencia, tiendas como registros en la base de datos.
- **Entity Manager:** Contiene las interfaces, que gestiona la persistencia de objetos. Funciona como instancia de consulta.

4.5.2.2. Controller

- **ManagerBeans:** Contiene los beans que hace referencian en el fichero de configuración de JSF en el apartado de managed beans, Estos beans están gestionados por el controlador JSF. Pues se encarga de su construcción y destrucción automáticas cuando es necesario.
- **BackingBeans:** Estas clases Java se asocian a los formularios JSF se les denomina backend beans ya que son los beans (clases Java) que están detrás del formulario.

4.5.2.3. Vista

- **Web Pages Interface:** contiene los formularios de la página web que entra en contacto con el usuario. Construido con XHTML que es básicamente un HTML expresado como XML válido.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6. Sprint

Se define un formato de asignación del sprint que servirá para cada uno de los Sprint como herramienta para la gestión de las tareas que corresponden a cada Sprint así como la fecha de inicio juntamente al número de días y en que jornadas se desarrollará cada uno, el permanente cambio en los estados de cada una de las tareas de los sprint exige a contar con recurso de estados el ayudara a conocer el estado en el que se encuentra dicha tarea al pasar por fases de Análisis, Codificación, Prototipado, Pruebas y Reuniones excluyendo en las jornadas los días festivos.

Consignación de la información en la tabla de configuración:

- **Nro. de Sprint:** Asignación del identificador del sprint.
- **Inicio:** Fecha de Inicio del Proyecto.
- **Días:** Número de días laborables del Sprint.
- **Tareas Tipos:** Tipos en los que se clasifican las diferentes tareas.
- **Tareas Estado:** Estados de una tarea durante el desarrollo en la etapa que se encuentra.
- **Equipo:** Lista de nombres que integra el equipo de trabajo.

Capítulo 4: Desarrollo e Implementación del Sistema

Tabla 4. 2
Iteraciones del Sprint

Proyecto			
SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA			
Nº de Sprint	Inicio	Días	Jornada
#	12-feb.-14	31	8
TAREAS		EQUIPO	FESTIVOS
TIPOS	ESTADOS		
Análisis	Pendiente	Carlos Burgos	
Codificación	En curso		
Prototipado	Terminada		
Pruebas	Eliminada		
Reunión			

Nota. Recuperado de <http://www.navegapolis.net/>

4.6.1. Sprint 1

En el primer sprint se procede a realizar las tareas del mismo que dejará definido la implementación del módulo comercial y todos sus catálogos. Como son; La creación del catálogo Clientes, Creación del catálogo de generación de una nueva orden de trabajo con sus detalles de artículos a manufacturarse y sus acabados por producto, así como también el desarrollo del catálogo de entregas.

4.6.1.1. Priorización

Para este primer sprint las tareas a desarrollarse son consideradas de acuerdo al nivel de prioridad dada en la Pila del Producto y su relevancia en el proceso del sistema para la ejecución del usuario en la generación de una orden de producción, por cuanto es necesario contar con los catálogos de registros.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.1.2. Asignación del Sprint

Tabla 4. 3

Asignación del sprint 1

SPRINT	INICIO	FIN	DURACIÓN		
1	21/05/14	02/07/14	31		
Tareas pendientes					6
Horas de trabajo pendientes					70
PILA DEL SPRINT					
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-01	Desarrollo de Catálogo de Clientes	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-01	Desarrollo de Catálogo de ODT	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-01	Desarrollo de Catálogo de Detalle ODT	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-01	Desarrollo de Catálogo de Acabados	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-01	Desarrollo de Catálogo de Maquinas	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-01	Desarrollo de Catálogo de Materiales	Media	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.1.3. Prototipo interfaz gráfica

Se presenta el siguiente prototipo de interfaz gráfica usada para la administración de todos los catálogos de gestión, Cada menú del sistema contará con un despliegue de una lista de datos en las que el usuario podrá Crear un nuevo registro, Modificar el registro deseado, Ver a detalle el registro y Eliminar un registro.

Cada botón contará con su respectivo evento hacia una ventana emergente para su respectiva gestión.

Capítulo 4: Desarrollo e Implementación del Sistema

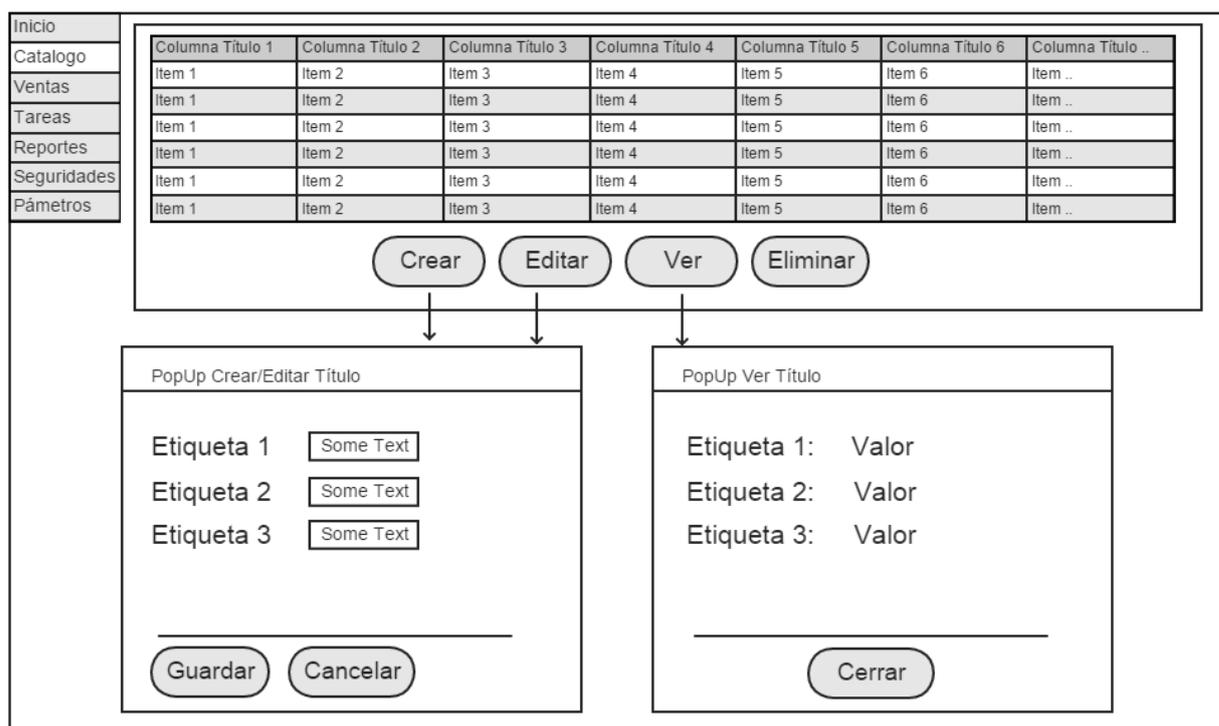


Figura 4. 5 Interfaz de Gestión de Módulos

Fuente: El Autor (Carlos Burgos)

4.6.2. Sprint 2

Contando una fuente de registros para la formulación y generación de una nueva orden de trabajo con cada uno de sus artículos a manufacturarse y sus detalles de imprenta para cada artículo se procedió a realizar el sprint numero dos el cual generará la toma de órdenes de trabajo estandarizadas, reduciendo el tiempo del proceso de venta y negociación al igual que se permite establecer precios manuales con cálculos de totales y saldos de una orden general. También se consideró indispensable la implantación de la impresión de dicha orden de trabajo en tamaño de ticket para el ahorro de recursos y contribuir con el medio ambiente.

4.6.2.1. Priorización

Las tareas elegidas para el actual sprint y su orden de priorización fueron consideradas siguiendo el orden de ejecución en el sistema de gestión de procesos, después

Capítulo 4: Desarrollo e Implementación del Sistema

de haber desarrollado el catálogo de clientes y sus demás verificaciones de los mismos, procedemos a la codificación de las nuevas órdenes de trabajo que es el núcleo para poder poner en marcha cualquier control y monitoreo de una línea de producción y sus procesos operativos.

4.6.2.2. *Asignación del Sprint*

Tabla 4. 4

Asignación del sprint 2

SPRINT	INICIO	FIN	DURACIÓN
2	08/07/14	19/08/14	31

Tareas pendientes	5
Horas de trabajo pendientes	60

PILA DEL SPRINT					
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-02	Toma de ODT Estandarizadas	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-03	Impresión Secuencias ODT	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-04	Precios manuales automáticos	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-05	Anticipo total de la Orden	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-06	Hora tomada una orden de producción	Baja	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.2.3. *Prototipo interfaz gráfica*

Se define un prototipo de interfaz para la gestión de ventas y generar una nueva orden de trabajo, en la que contará con una vista amigable con el usuario, fácil de usar y rápida en la que permite tomar una orden de trabajo estandarizada con un número de identificación única que diferencia de otras, por lo tanto no podrá haber dos órdenes de trabajo del mismo año con el mismo número. Dentro de este módulo el usuario de ventas podrá asignar precios manuales según varía el monto al tamaño o formato del producto deseado por el cliente y

Capítulo 4: Desarrollo e Implementación del Sistema

de igual forma se obtendrá el cálculo del importe de la venta automáticamente incluido el valor que desee abonar el cliente.

Luego de haber generado una orden de trabajo se obtendrá un documento impreso detallando la información de los artículos solicitados a manufacturarse por el cliente en la que incluye horas y fechas actuales en la que está siendo generada dicha orden de trabajo.

Fecha: 01/01/15 Vendedor: Usuario Vendedor

Cliente: Lista Clientes Cédula / RUC: 000000001 Nombre / Razón Social: Carlos Burgos / UTN

Productos: Lista Productos + Añadir Productos

Fecha Entrega: 01/01/15 Observaciones Orden de Trabajo: Observaciones generales de la Orden de Trabajo

Forma de Pago: Formas de Pago

Lugar Entrega: Lugar Entrega

COD	PROD	DESCRIPC	DTLL IMP	PRE UNIT	CANT	SUBTOTAL	QUITAR
....	-
					TOTAL	
					ABONO	
					SALDO	

Calcular Valores
Finalizar Venta

Figura 4. 6 Interfaz de Gestión de Ventas
Fuente: El Autor (Carlos Burgos)

4.6.3. Sprint 3

Para este sprint se procede a crear cada uno de los catálogos que servirán como fuente de datos en la formulación de una nueva secuencia de procesos que debe seguir en una determinada línea de producción, por lo que es importante contar con el levantamiento del proceso de la empresa, así como también sus departamentos, etapas de un proceso y cada uno de los acabados.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.3.1. Priorización

La relevancia que presentan cada una de las tareas es considerada por su importancia dentro de la funcionalidad durante un flujo de trabajo, porque se ha visto necesario tener un registro de todas las fuentes necesarias para la formulación de un flujo de trabajo para una determinada línea de producción.

4.6.3.2. Asignación del Sprint

Tabla 4. 5
Asignación Sprint 3

SPRINT	INICIO	FIN	DURACIÓN
3	23/08/14	03/09/14	31

Tareas pendientes	5
Horas de trabajo pendientes	55

PILA DEL SPRINT					
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-07	Desarrollo Catálogo Líneas de Producción – Productos	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-07	Desarrollo Catálogo Tipos Acabados - Acabados	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-07	Desarrollo Catálogo Departamentos – Etapas – Procesos - Secuencias Procesos	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-07	Desarrollo Catálogo Historial Tareas – Tareas - Involucrados Tareas – Archivos Adjuntos	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-07	Hora tomada una orden de producción	Baja	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.3.3. Prototipo interfaz gráfica

Se considera catálogos a cada fuente de datos imprescindible que debe ser registrada en el sistema para entrar en marcha

Capítulo 4: Desarrollo e Implementación del Sistema

Productos, dentro de esta vista el usuario podrá crear, listar, modificar y eliminar de la lista de productos que ofrece al público al igual que le permitirá asignar precios unitarios.

Líneas de producción, cada producto pertenece a una línea de producción por lo que se requiere clasificar ya que su proceso de manufactura es distinto.

Acabados, los productos mantienen varios tipos de acabados y estos se distinguen por la línea de producción en lo que corresponde a material por lo que crea las vistas para gestionar acabados sus tipos.

- **Departamentos**, dentro de la empresa existen tres tipos de departamentos; Ventas, Diseño y Producción. Es importante contar con un registro de estos por la participación del personal en la un línea de producción.
- **Etapas**, para línea de producción se define tres tipos de etapas como son; pre-prensa, impresión y acabados y entrega.
- **Procesos**, cada línea de producción mantiene procesos diferentes en un orden distinto por lo que se considera registrar los existentes para cada línea.
- **Secuencias Procesos**, se debe estructurar a cada línea de producción su proceso lineal secuencial a seguir.

De tal forma se presenta el prototipo de interfaz gráfica estandarizada para los catálogos de todos del sistema.

Capítulo 4: Desarrollo e Implementación del Sistema

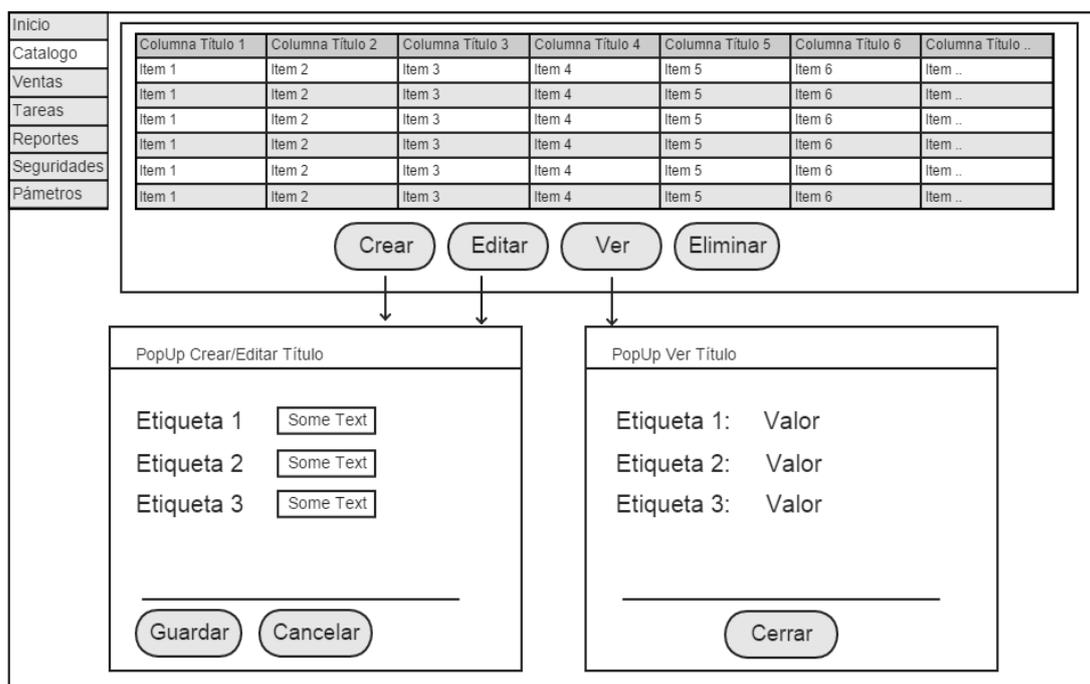


Figura 4. 7 Interfaz de Gestión de Catálogo Gestión de Procesos

Fuente: El Autor (Carlos Burgos)

Para la vista del inicio de producción de una orden de trabajo, se presenta el prototipo en el que cuenta con una lista de todas las órdenes de trabajo junto al número y el cliente. El usuario podrá seleccionar una de dicha lista y establecer un nivel de prioridad que va desde; baja, normal, alta y urgente dependiendo la fecha de entrega. Una vez asignado la prioridad el usuario procederá a iniciar la producción y la orden de trabajo será retirada de la lista de pendientes a ser iniciada.

Capítulo 4: Desarrollo e Implementación del Sistema

<input type="text"/>		Orden Nro	Prioridad <input type="button" value="▲"/>	<input type="button" value="Iniciar"/>
Cliente 1	Orden Nro 1	Fecha Generada	<input type="text" value="Search"/>	
Cliente 2	Orden Nro 2	Fecha Entrega		
Cliente 3	Orden Nro 3	Cliente		
Cliente 4	Orden Nro 4	Oberservaciones		
Cliente 5	Orden Nro 5			
Cliente 6	Orden Nro 6			
Cliente 7	Orden Nro 7			
Cliente 8	Orden Nro 8			
Cliente 9	Orden Nro 9			
Cliente 10	Orden Nro 10			
Cliente 11	Orden Nro 11			
Cliente 12	Orden Nro 12			

Producto	Cantidad	Descripcion	Acabados
Producto 1	Cantidad 2	Descripcion 3	Acabados 4
Producto 2	Cantidad 4	Descripcion 5	Acabados 6

Figura 4. 8 Interfaz de Gestión de Inicio de Orden de Producción

Fuente: El Autor (Carlos Burgos)

Dentro de la vista de gestión de tareas de un artículo en su procesos de producción se considera contar con un control de actividades por proceso, de tal manera que se pueda conocer en qué proceso de la etapa de manufactura se encuentra dicho producto, al igual que debe contar con la asignación de tareas a usuarios del mismo departamento para que pueda ser completada.

Cada evento o acción proporcionada por el usuario durante una tarea específica en un proceso de producción será registrada por el sistema en la que permitirá conocer que cuando el usuario solicito completar la tarea, fecha y hora en la que el usuario terminó la tarea, y de igual manera cuando una tarea es asignada a un usuario. También se verá reflejado los avances del proceso lineal de manufactura por medio de un diagrama generado en tiempo real concluyendo cuando termina todos los procesos y se realiza la entrega al cliente.

Capítulo 4: Desarrollo e Implementación del Sistema

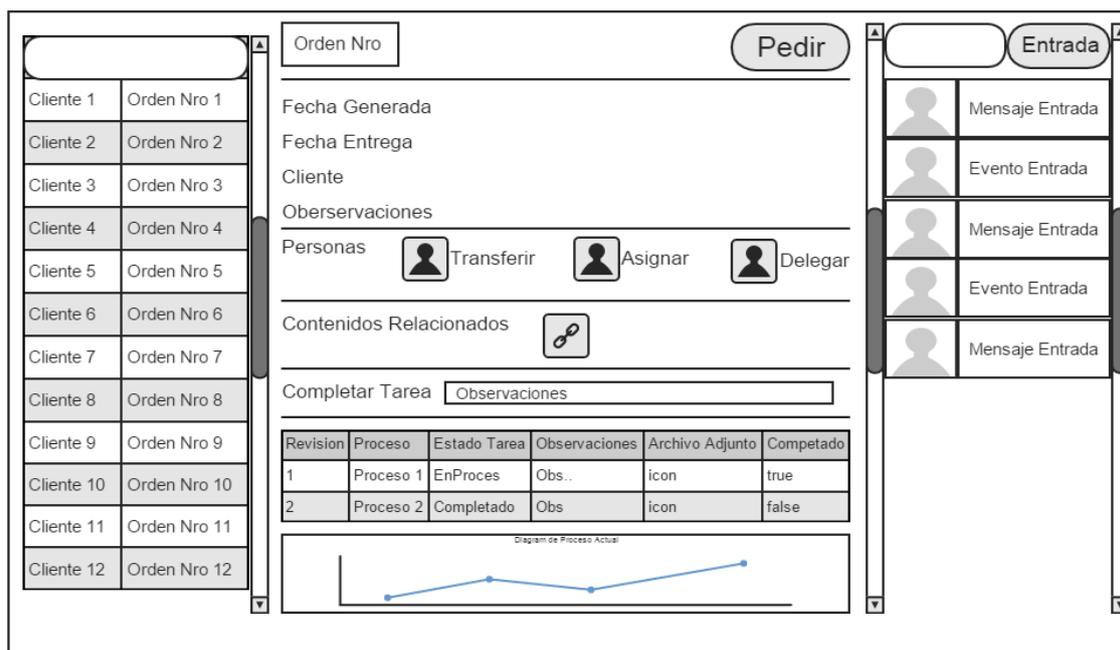


Figura 4. 9 Interfaz de Gestión de Historial Tareas Procesos ODT

Fuente: El Autor (Carlos Burgos)

4.6.4. Sprint 4

En este sprint se realiza el monitoreo general y detallado una orden de trabajo por su proceso de producción durante su metamorfosis hasta que finalmente se realiza la entrega al cliente en sus dos tipos ya sea parcial o total, al igual que durante su flujo de proceso en cada tarea se controla tiempos de entradas y salidas que facilitaran la apreciación real de tiempos en cuanto respecta a entregas de cada producto de la orden.

4.6.4.1. Priorización

Las tareas enlazadas a este evento fueron clasificadas por el desencadenamiento que presenta con el sprint anterior de contar con la estructura del proceso de una línea de producción para iniciar el control y monitoreo.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.4.2. Asignación del Sprint

Tabla 4. 6
Asignación Sprint 4

SPRINT	INICIO	FIN	DURACIÓN
4	09/10/14	20/11/14	31

Tareas pendientes	5
Horas de trabajo pendientes	54

PILA DEL SPRINT					
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-08	Código supervisor en etapas de ODT	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-09	Código supervisor de tiempos de ODT en proceso	Alta	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-10	Monitor de tiempos entregas ODT	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-11	Alertas en etapas Iniciales de proceso	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-12	Ver ODT en área producción	Baja	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.4.3. Prototipo interfaz gráfica

Dentro de prototipo de monitor de órdenes de trabajo tomadas es importante conocer en el departamento de producción todas las órdenes que está siendo generadas constantemente con su detalle de artículos a manufacturarse y sus respectivos acabados, por lo que se ha considerado desarrollar esta vista Atmosphere es un framework java que permitirá la creación asíncrona del módulo ventas y la vista del monitor de órdenes de trabajo, para tener en tiempo real el listado desde la fuente de datos realizada por el servidor sin la necesidad que el usuario lo actualice.

Capítulo 4: Desarrollo e Implementación del Sistema

Lista Actual de Ordenes de Trabajos tomadas en Ventas							
Nro ODT	CED CLIENT	FECHA INIC	FECHA FIN	FECHA INIC	LUGAR ENTRG	VENDEDOR	VER
.....	Ver Detalle
.....	Ver Detalle

Figura 4. 10 Interfaz de Control de Orden de Trabajo

Fuente: El Autor (Carlos Burgos)

4.6.5. Sprint 5

Para este sprint se procede a realizar las tareas de seguridad del sistema, necesitando tener una clasificación de las funciones de los empleados acuerdo sus roles y participación durante el tratamiento de un flujo de procesos de una determinada línea de producción, de igual forma considerando la utilización de parámetros para una mejor funcionamiento del sistema en ciertos módulos y catálogos.

4.6.5.1. Priorización

Su importancia surge a partir de la importancia de contar con determinada acción que deben realizar los empleados en ciertas tareas de determinados productos, y las acciones realizadas por lo que se considera un sprint de aumento.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.5.2. Asignación del Sprint

Tabla 4. 7
Asignación sprint 5

SPRINT	INICIO	FIN	DURACIÓN
5	26/11/14	07/01/15	31

Tareas pendientes	3
-------------------	---

Horas de trabajo pendientes	50
-----------------------------	----

PILA DEL SPRINT

Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-14	Desarrollo de Catálogo de Usuarios - Grupos - Grupos Permisos	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-14	Código supervisor de tiempos de ODT en proceso	Media	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-14	Desarrollo de Catálogo de Auditorias - Log Accesos	Media	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.5.3. Prototipo interfaz gráfica

El mantenimiento de usuarios, grupos y roles es imprescindible por lo que se presenta el prototipo de la interfaz este debe permitir crear nuevos usuarios, deshabilitar usuarios que ya no trabajen en la empresa y dar de baja a los mismo. De igual forma posterior se contará con auditorias para los módulos especificados por el Product Owner.

Capítulo 4: Desarrollo e Implementación del Sistema

Nombre (Usuario) →

Nombre (Usuario)

Nombre (Usuario)

Nombre (Usuario)

NOMBRE USUARIO

ID

NOMBRES

RAZÓN SOCIAL

DIR

TELF

CORREO

USUARIO

CLAVE

ID	GRUPO	ACCIÓN

Figura 4. 11 Interfaz de Gestión de Usuarios

Fuente: El Autor (Carlos Burgos)

4.6.6. Sprint 6

Para este sprint se eligió realizar las tareas que presentan menor prioridad dentro de nuestro sistema, como se puede observar las tareas especificadas en la asignación no influyen en el proceso de la generación de una nueva venta y en el proceso de monitoreo y control de orden de trabajo.

4.6.6.1. Priorización

Las tareas elegidas para este sprint son las de menor prioridad para el funcionamiento del sistema, son tareas de gestión mas no de las del núcleo del sistema que son las de la Toma de una nueva orden de trabajo y monitoreo y control de flujo de proceso de una línea de producción en un determinado producto.

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.6.2. Asignación del Sprint

Tabla 4. 8

Asignación del sprint 6

SPRINT	INICIO	FIN	DURACIÓN
6	13/01/15	24/02/15	31

Tareas pendientes	6
Horas de trabajo pendientes	55

PILA DEL SPRINT

Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
REQ-DIKAPSA-15	Reporte ODT tomadas	Baja	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-16	Reporte por producto cantidad	Baja	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-17	Reporte de tiempos en etapas	Baja	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-18	Reporte ODT entregadas tarde	Baja	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-19	Reporte de compras por cliente y categoría	Baja	Pruebas	Terminada	Carlos Burgos
REQ-DIKAPSA-20	Reportes por vendedores	Baja	Pruebas	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.6.3. Prototipo interfaz gráfica

Para generar dichos reportes establecidos en los requerimientos se presentan de varias alternativas que podrán ser en listas de tablas o informes gráficos como se muestra a continuación. De tal forma que le permita conocer al usuario en tiempo real la información y generarlas de igual forma dichos informes.

Informe Gráfico

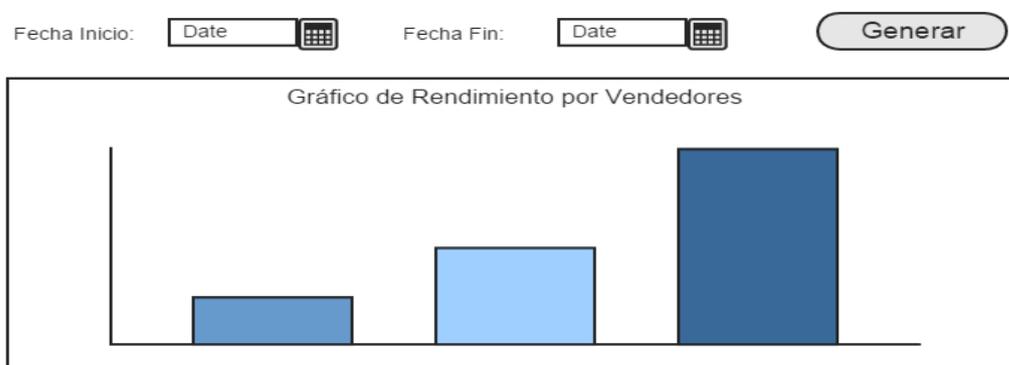


Figura 4. 12 Interfaz de Gráfica de Reporte

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.6.7. Sprint 7

Durante este sprint se corrigen los problemas existentes de los anteriores Sprints para su correcto funcionamiento.

4.6.7.1. Priorización

La priorización de este sprint se presenta de la lista de errores presentados en la funcionalidad del sistema durante su ejecución.

4.6.7.2. Asignación del Sprint

Tabla 4. 9

Asignación del sprint 7

SPRINT	INICIO	FIN	DURACIÓN
7	27/02/15	10/04/15	31

Tareas pendientes		1			
Horas de trabajo pendientes		30			
PILA DEL SPRINT					
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable
	Ejecución de pruebas de los Sprint del sistema	Media	Producción	Terminada	Carlos Burgos

Nota. Elaborado por el Autor Carlos Burgos.

4.6.8. Sprint 8

En este Sprint se presenta la puesta en marcha del Sistema de Gestión de Procesos de Producción por el cual se da inicio en ambiente productivo a la utilización plena del software desarrollado y la integración de todos los Sprints para ser utilizado por el usuario final de DIKAPSA.

Para la implantación de considera los factores externos de puesta en marcha que puede tener una influencia directa sobre el proceso, los cuales incluso puede generar riesgos

Capítulo 4: Desarrollo e Implementación del Sistema

que deben ser gestionados adecuadamente, por cuanto se debe describir el entorno adecuadamente al que está inscrita la migración de datos.

La puesta en producción se mide por los indicadores de disponibilidad, cargas de información, operaciones, integración, interfaces, uso del sistema y la concurrencia

Es indispensable contar con servidores de ambiente de producción en las cuentas con el sistema operativo de preferencia de la empresa en condiciones iniciales ya sea LINUX/WINDOWS para los dos casos las versiones SERVER de 64 Bits.

4.6.8.1. Priorización

Para este sprint el nivel de priorización se determina por la finalización de todo el sprint antecesores y fase de pruebas para contar con ambiente de producción totalmente nativo para el sistema

4.6.8.2. Asignación del Sprint

Tabla 4. 10

Asignación del sprint 8

SPRINT	INICIO	FIN	DURACIÓN
8	13/04/15	23/05/15	31

					Tareas pendientes	1
					Horas de trabajo pendientes	30
PILA DEL SPRINT						
Backlog ID	Tarea	Prioridad	Tipo	Estado	Responsable	
	Puesta en producción integración de Sprints	Media	Producción	Terminada	Carlos Burgos	

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 4: Desarrollo e Implementación del Sistema

4.7. Incremento e implementación de los Sprints

4.7.1. Entregable Creación, Modificación, Vista y Eliminación de Clientes

Para el catálogo de clientes es imprescindible un contar con una tabla en nuestra fuente de datos que nos permita gestionar y enriquecer para llevar un control de los posibles clientes potenciales.

4.7.1.1. Modelo Físico



Figura 4. 13 Modelo físico creación y modificación de clientes

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.1.2. Diagrama de Clases

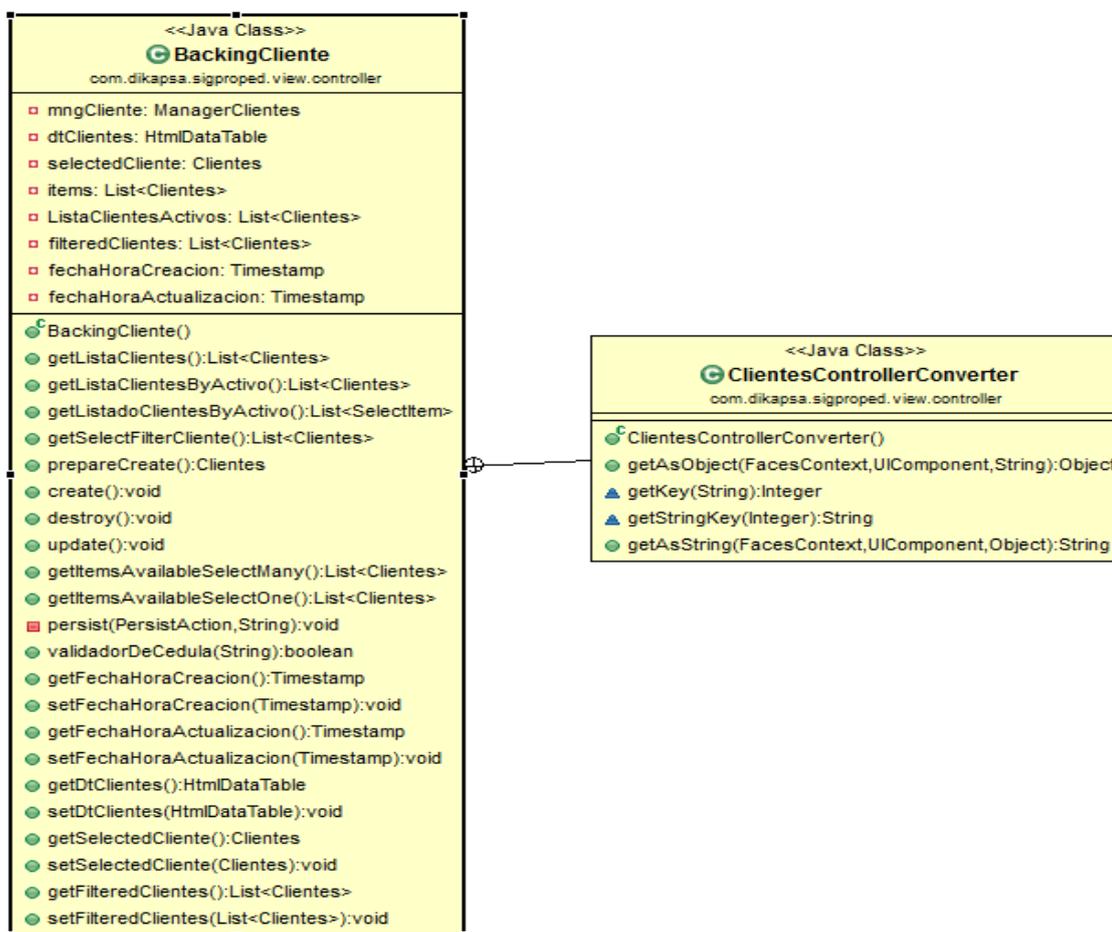


Figura 4. 14 Diagrama de clases de creación y modificación de clientes

Fuente: El Autor (Carlos Burgos)

4.7.1.3. Interfaz

Se cumple con lo propuesto en el prototipo, contando en la mayoría de los catálogos al igual que en clientes del sistema con las funciones básicas como son; Crear, Modificar, Actualizar y Borrar.

Capítulo 4: Desarrollo e Implementación del Sistema

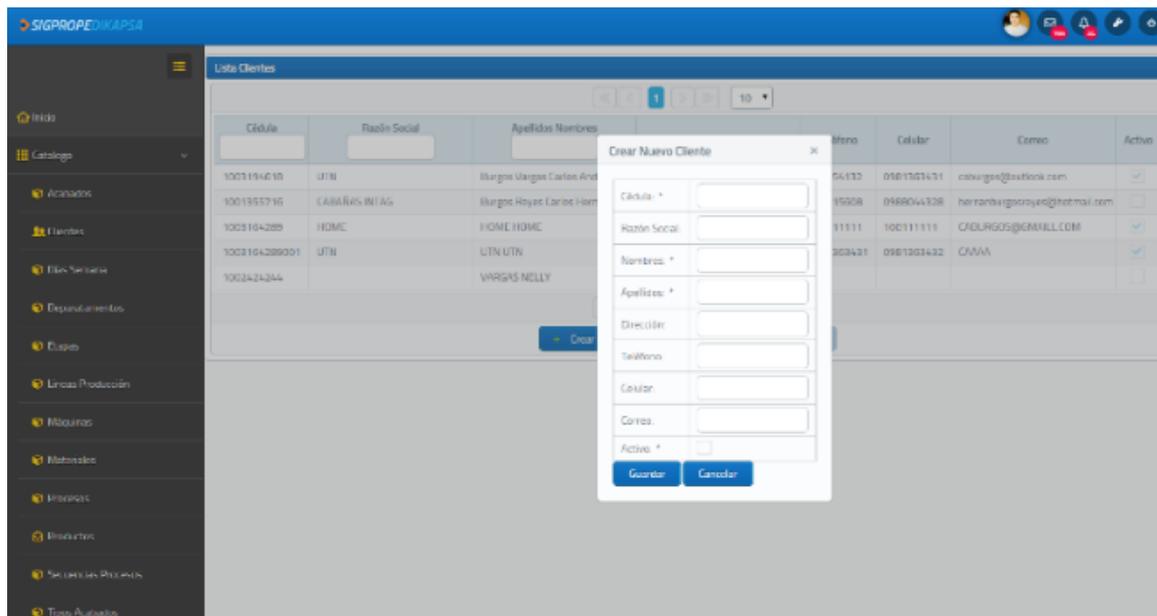


Figura 4. 15 Interfaz de creación y modificación del cliente

Fuente: El Autor (Carlos Burgos)

4.7.2. Entregable Toma de Orden de Trabajo Estandarizadas

Estandarizar la toma de la ordenes de trabajo en un punto de ventas de la empresa, implicó dependencias desde nuestra fuente de datos tales como; Contar previamente con el módulo de productos y acabados con sus respectivas líneas de producción y sus procesos determinados en el sistema, Registro de clientes o registros existentes con información relevante de los mismos y por último tener el registro de los usuarios vendedores que pueden participar en una venta y negociación en la toma de una orden de trabajo.

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.2.1. Modelo Físico

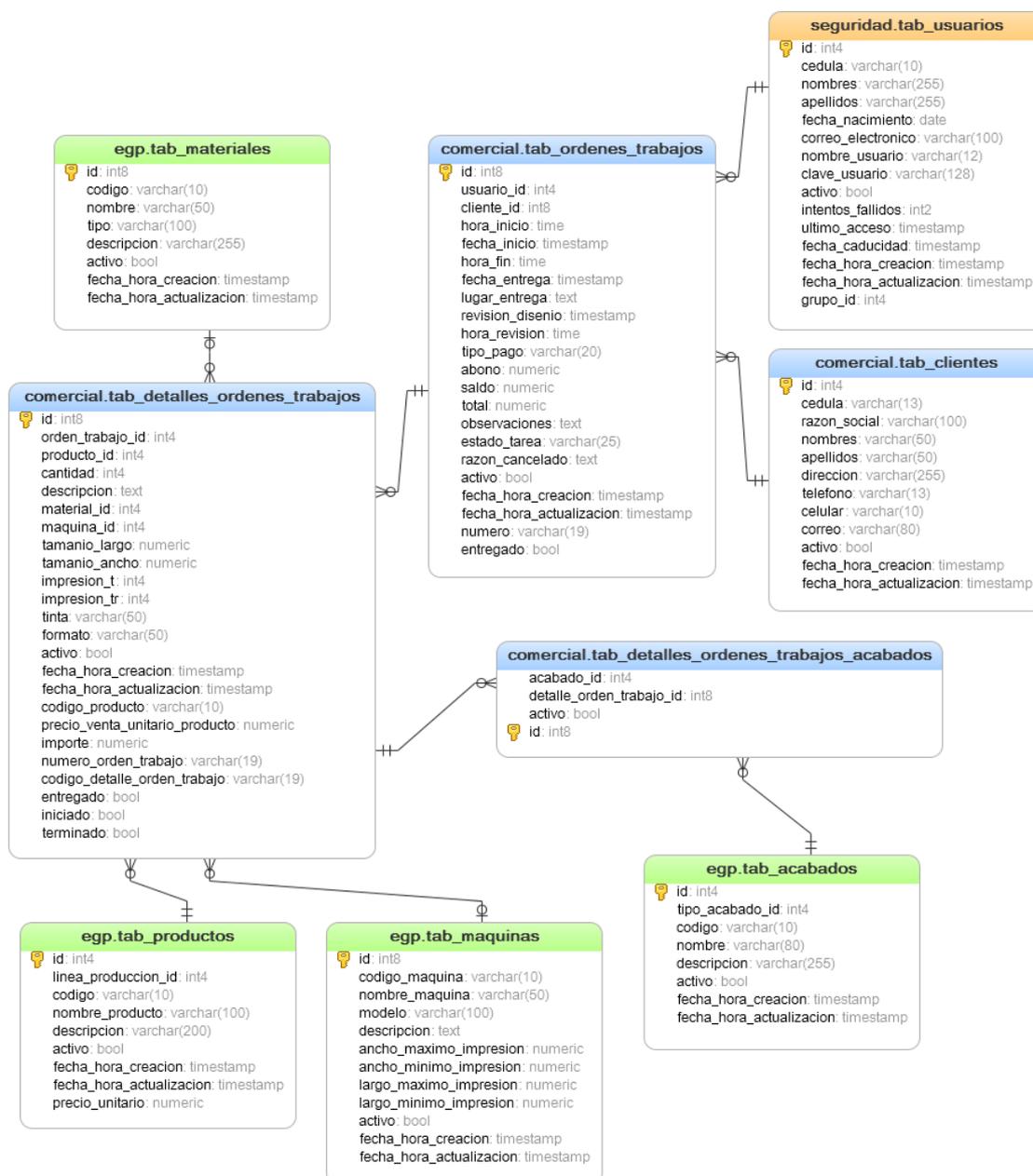


Figura 4. 16 Modelo físico creación nueva orden de trabajo

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.2.2. Diagrama de Clases

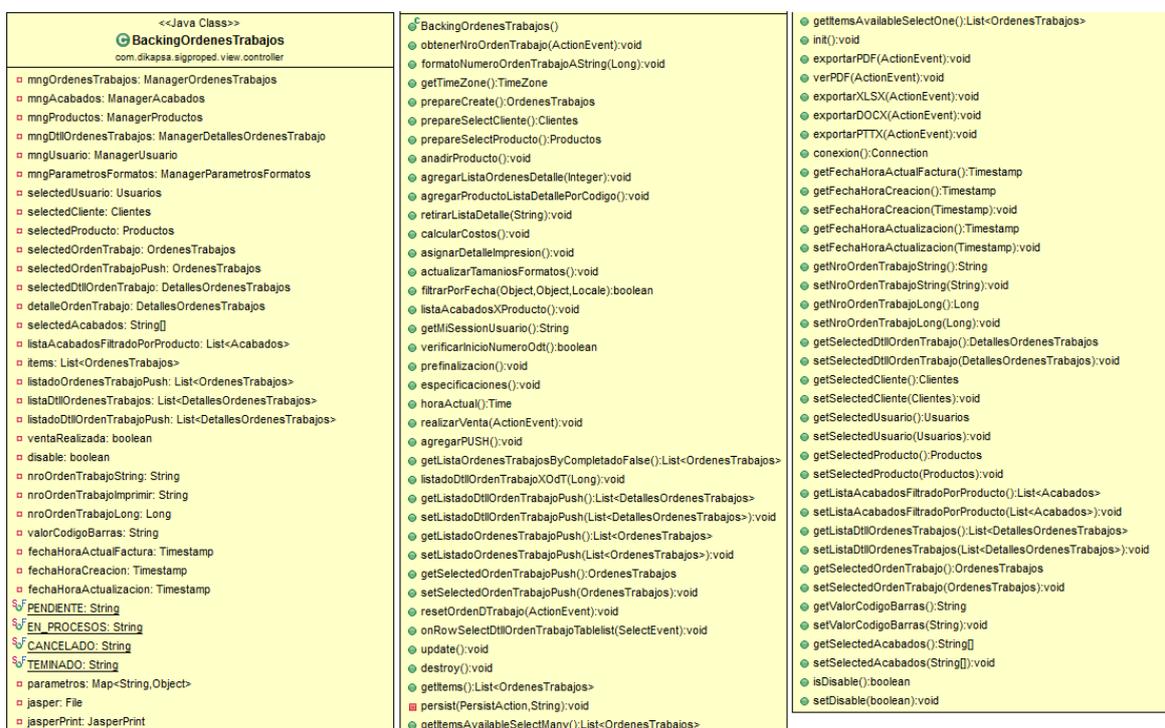


Figura 4. 17 Diagrama de clases de creación nueva orden de trabajo

Fuente: El Autor (Carlos Burgos)

4.7.2.3. Interfaz

El módulo de ventas es uno de los más relevantes dentro del sistema por lo que se puso énfasis en su desarrollo ya que debe estar validado muchos campos y filtros al momento de generar una orden de trabajo como son; Número de la orden de trabajo único, se mostró en el sistema automáticamente al vendedor que se encuentra iniciado sesión, filtros de búsqueda avanzada tanto para clientes y productos, maestros detalle de una orden de trabajo, cálculo de valores automáticos, descripción y observaciones de la orden de trabajo y sus detalles, fijación de medidas y acabados de un artículo, entre otros.

Capítulo 4: Desarrollo e Implementación del Sistema

The screenshot shows the 'Orden de Trabajo' (Work Order) creation interface in the SIGPROPE WAPSA system. The interface is divided into several sections:

- Header:** 'SIGPROPE WAPSA' logo and user profile icons.
- Form Fields:**
 - Nro Orden Trabajo:** Input field for the work order number.
 - Fecha:** Set to '09/12/2015'.
 - Vendedor:** Set to 'Admin Admin'.
 - Cliente:** Includes a dropdown for 'Selección Cliente...', a 'Buscar' button, and fields for 'Cédula/RUC' and 'Nombre / Razón Social'.
 - Producto:** Includes a dropdown for 'Selección un producto.', a '+ Agregar' button, and a 'Buscar' button.
 - Fecha Entrega:** Input field with a calendar icon.
 - Forma de Pago:** Dropdown menu.
 - Lugar Entrega:** Input field.
 - Observaciones generales Orden de Trabajo:** Text area with a note: '50 caracteres necesarios' and 'Especifique las observaciones generales de la Orden de Trabajo'.
- Detalle de productos:** A table with columns: Código, Producto, Descripción, Detalle Impresión, Precio Unitario, Cantidad, Total, Retirar. The table is currently empty, showing 'Lista sin productos.' and summary rows for 'TOTAL:', 'AFCNO:', and 'SALDO:'. A 'Valor Abono: 0.00' is displayed on the right.
- Buttons:** 'Actualizar monto de compra' and 'Finalizar Venta'.

Figura 4. 18 Interfaz de creación de una nueva orden de trabajo

Fuente: El Autor (Carlos Burgos)

4.7.3. Entregable Creación de Secuencias Procesos de una Línea de Producción

Se implementa la estructuración de los procesos lineal y secuencial para cada línea de producción existente, tomando la fuente de registros tales como; departamentos a los que pertenece una línea de producción, etapas de la línea de producción, procesos y acabados de la misma.

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.3.1. Modelo Físico



Figura 4. 19 Modelo físico creación nueva secuencia de procesos

Fuente: El Autor (Carlos Burgos)

4.7.3.2. Diagrama de Clases

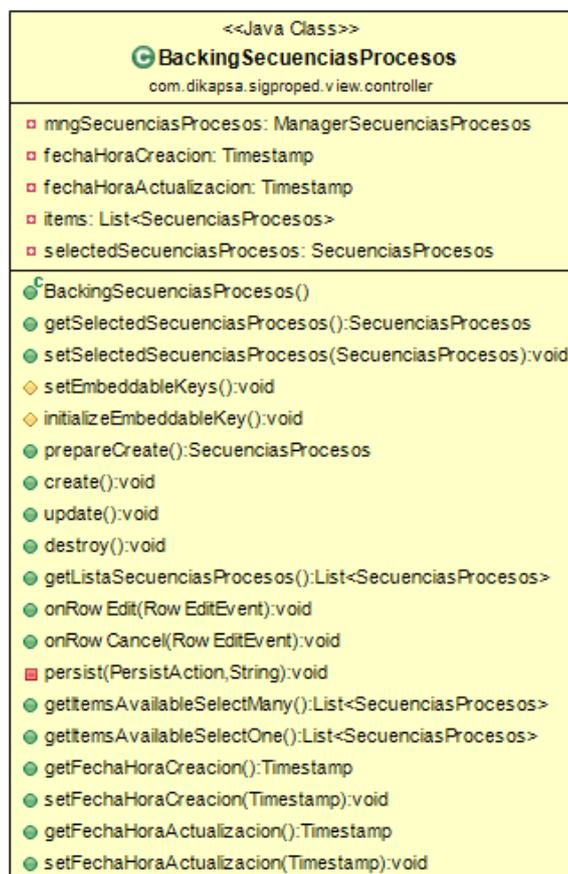


Figura 4. 20 Diagrama de clases de creación nueva secuencias de procesos

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.3.3. Interfaz

Dentro de este catálogo el usuario podrá gestionar cada una de las secuencias de procesos, al igual que realizar un filtro por línea de producción para su fácil edición evitando un posible error de edición con otros registros, se ha resaltado con colores diferentes para su fácil comprensión y distinción unas de otras, al igual cuenta con una paginación para su mejor interacción.

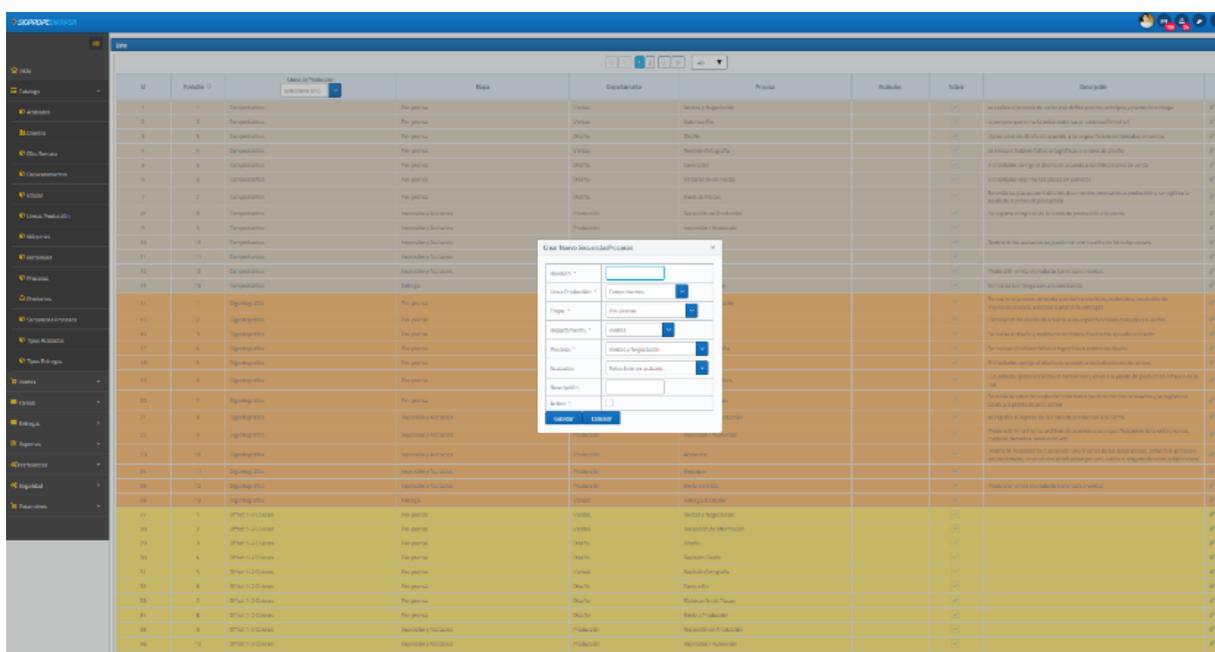


Figura 4. 21 Interfaz de creación de una nueva secuencia de procesos
Fuente: El Autor (Carlos Burgos)

4.7.4. Entregable Ver Orden de Trabajo en Producción

Este entregable es importante para la empresa ya que contar con un monitor que permita visualizar en tiempo real las órdenes de trabajo en la planta de producción, es de gran ayuda a los empleados para una eficaz y optimización manufactura de los artículos. Por lo que se implementa dicho modulo sin observaciones y con una funcionalidad correcta.

Capítulo 4: Desarrollo e Implementación del Sistema

Con las listas de datos obtenidos dentro del módulo se puede filtrar por orden de trabajo, por el nombre del cliente o si el usuario del sistema prefiere por las fechas de inicio de una orden de trabajo en la que fue generada o la fecha de finalización “entrega” de esa orden.

4.7.4.1. Modelo Físico

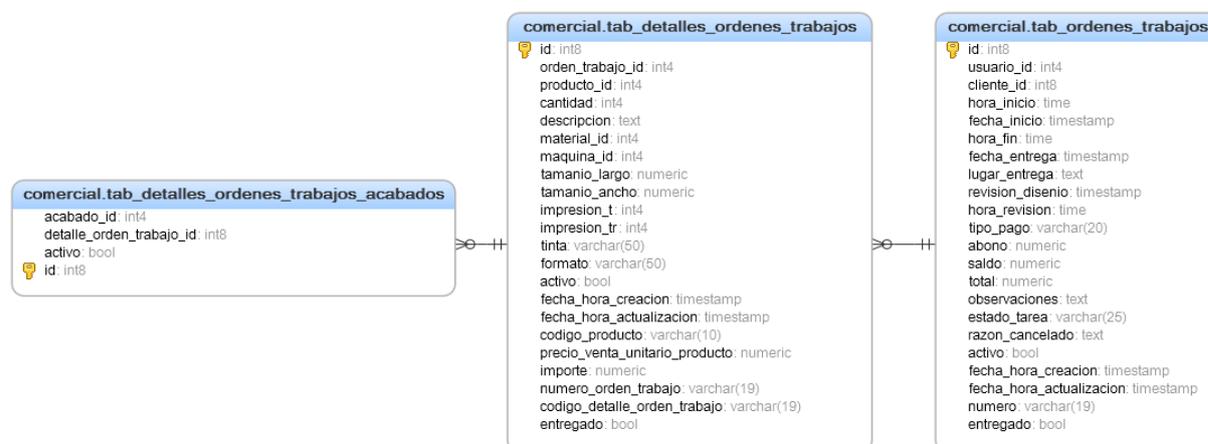


Figura 4. 22 Modelo físico Ver Orden de Trabajo en Producción

Fuente: El Autor (Carlos Burgos)

4.7.4.2. Diagrama de Clases



Figura 4. 23 Diagrama de clases Ver Orden de Trabajo en Producción

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.4.3. Interfaz

Se presenta la implementación del monitor y visualización en tiempo real de las órdenes de trabajo en la planta de producción tomadas en el departamento de ventas. Funcionalidad que servirá para anticipación de materiales de manufactura para los artículos de una orden de trabajo.

The screenshot displays the SIGPROPED system interface. At the top, there is a navigation bar with 'Inicio' and 'Mantenimiento'. Below it, a header reads 'LISTA ORDENES TRABAJOS'. A table lists work orders with columns for 'Número', 'Cliente', 'Fecha Recepción', 'Fecha Entrega', 'Lugar Entrega', 'Observaciones', 'Vendedor', and 'Seleccionar'. Three orders are visible, all from 'HOME HOME' or 'Burgos Vargas Carlos Andrés'. A modal window titled 'LISTA DETALLES ORDENES TRABAJOS' is open, showing a detailed view of order '000000005'. This window includes a table with columns: 'Número Orden', 'Producto', 'Cantidad', 'Descripción', 'Material', 'Máquina', 'Impresión T', 'Impresión Tr', 'Tinta', 'Formato', 'Tamaño Largo', 'Tamaño Ancho', and 'Acabado'. The detailed table shows 'COMPROBANTES A4' with a quantity of 2, material 'LONAMATE', and machine 'IMOFFSETC'. Below this, a smaller table lists 'Acabado' options: 'BRILLO UV', 'REPUJADO', and 'PLASTICADO BRILLANTE'.

Figura 4. 24 Interfaz Ver Orden de Trabajo en Producción

Fuente: El Autor (Carlos Burgos)

4.7.5. Entregable Inicio de Proceso de una Orden de Trabajo

Se procede a la implementación del módulo culminado de gestión de inicio de tareas, en el que el usuario puede iniciar la producción de las órdenes de trabajo a realizarse utilizando una prioridad de manufactura.

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.5.1. Modelo Físico

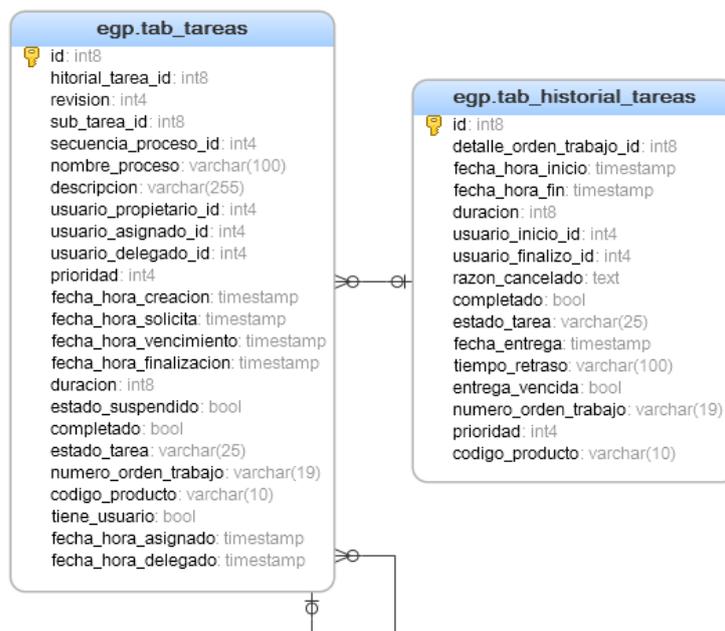


Figura 4. 25 Modelo físico gestión de inicio de una orden de producción
Fuente: El Autor (Carlos Burgos)

4.7.5.2. Diagrama de Clases

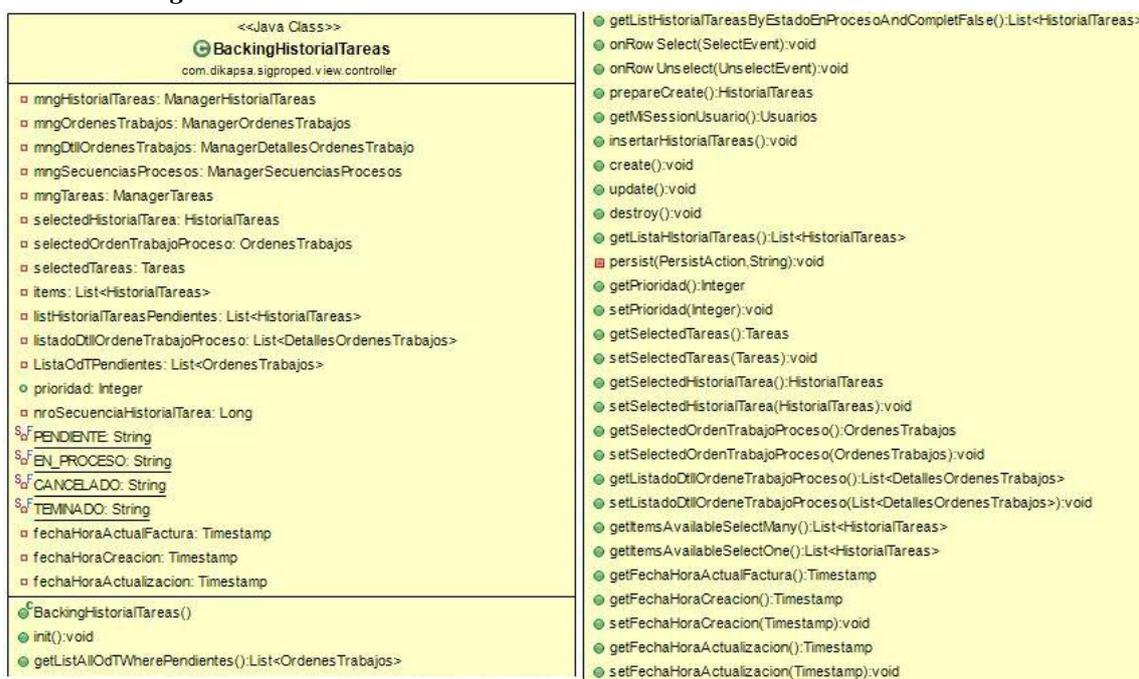


Figura 4. 26 Diagrama de clases Inicio de Procesos de Producción de una Orden de Trabajo
Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.6.1. Modelo Físico

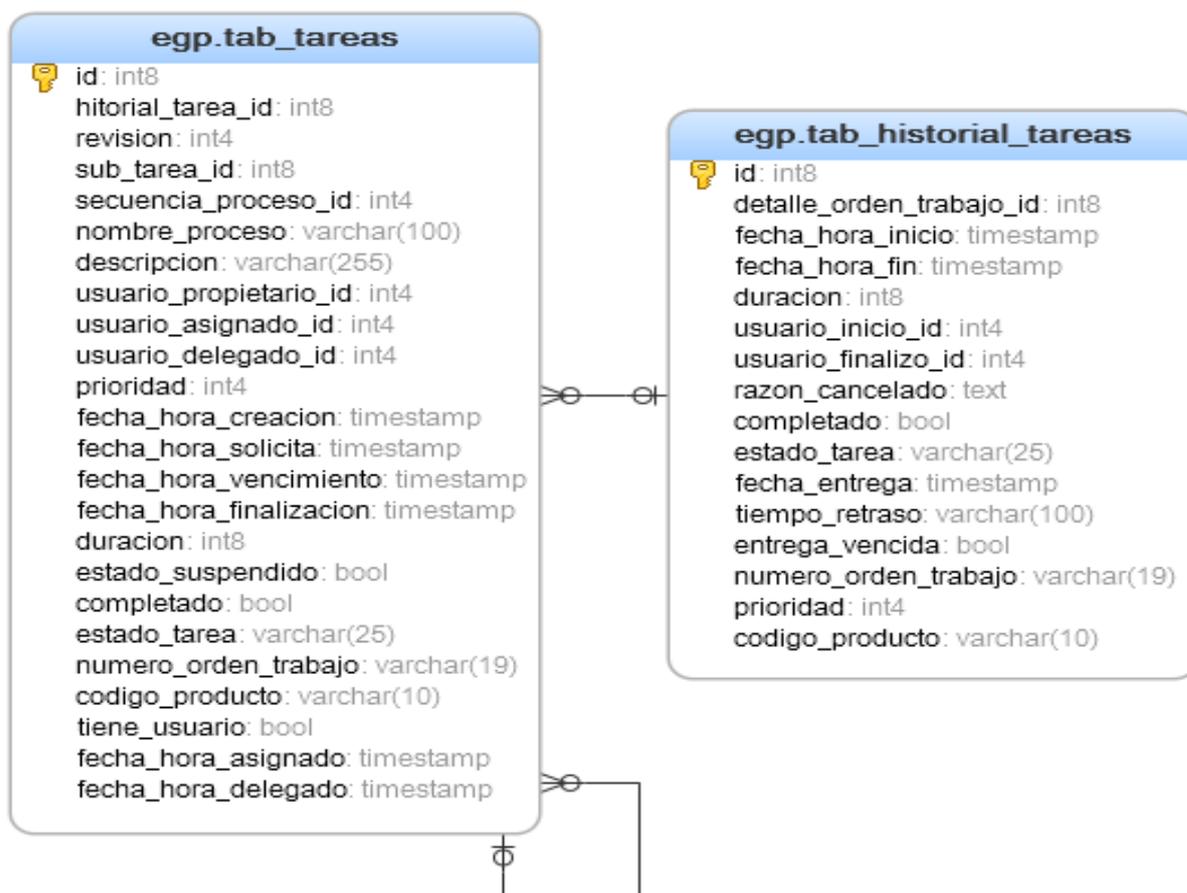


Figura 4. 28 Modelo físico gestión tareas de una orden de producción

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.6.2. Diagrama de Clases



Figura 4. 29 Diagrama de clases gestión tareas de una orden de producción

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.6.3. Interfaz

Para esta interfaz se consideró contar con un control de actividad de eventos del módulo en sí, el cual permite conocer la actividad realizada en cada una de las tareas del proceso al igual que la participación de los empleados en dicha etapa del proceso. Con una interfaz amigable el usuario puede completar la tarea y pasar a realizar la siguiente si perteneciere a su departamento hasta completar su proceso de manufactura.

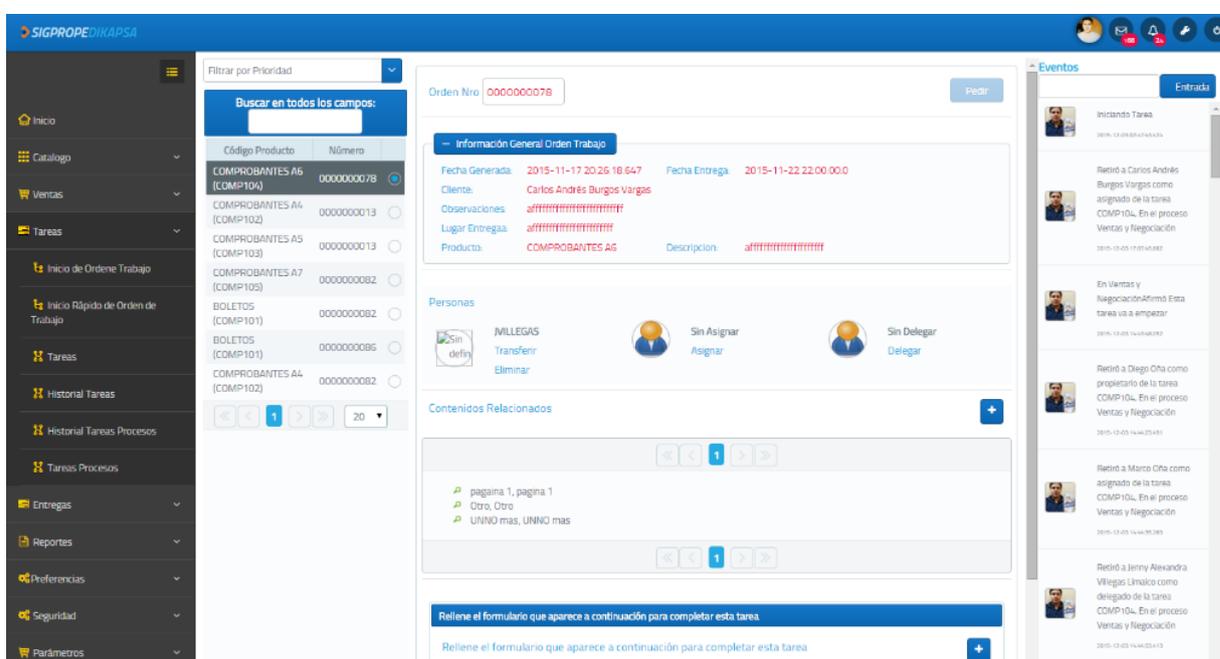


Figura 4. 30 Interfaz gestión tareas de una orden de producción

Fuente: El Autor (Carlos Burgos)

4.7.7. Entregable Entregas Órdenes de Trabajos

Las entregas de las órdenes de trabajo en el sistema le permiten conocer al usuario las tareas que han sido completadas y necesita ser entregadas para culminar en su totalidad

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.7.1. Modelo Físico

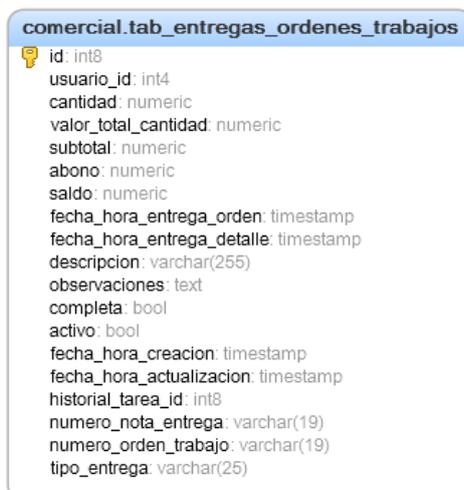


Figura 4. 31 Modelo físico gestión entregas ordenes trabajos

Fuente: El Autor (Carlos Burgos)

4.7.7.2. Diagrama de Clases

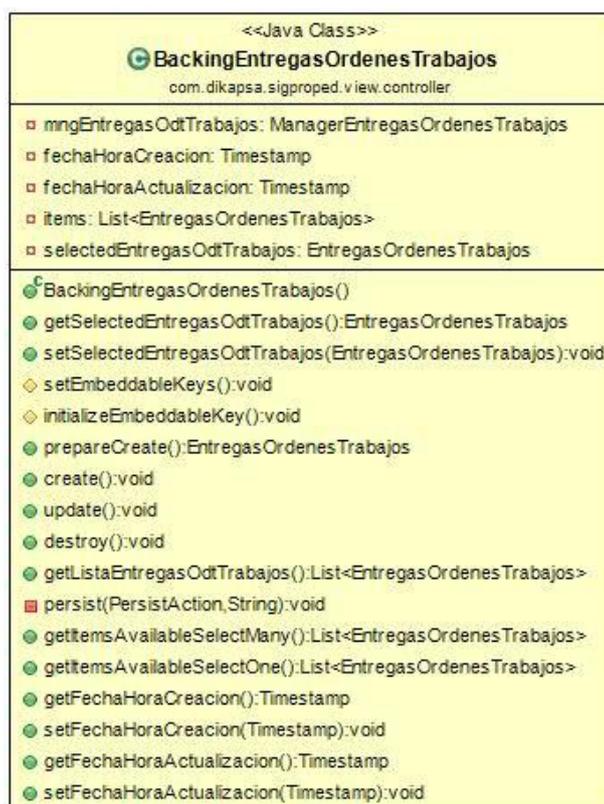


Figura 4. 32 Diagrama de clases gestión entregas ordenes trabajos

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.7.7.3. Interfaz

Esta interfaz se asemeja a la de ventas por tanto se tomó de dicho modulo filtrando los artículos que han sido culminados y se encuentran listos ha se entregados tanto parcial como total.

The screenshot shows the 'Nota Entregas Orden de Trabajo' (Delivery Order Note) interface. It features a sidebar menu on the left with options like 'Inicio', 'Catalogo', 'Ventas', 'Tareas', 'Entregas', 'Entregas Ordenes Trabajos', 'Consultar Entregas', 'Reportes', 'Monitor Ordenes Trabajos', 'Preferencias', 'Seguridad', and 'Parámetros'. The main area contains a form for creating a new order with fields for 'Fecha' (11/02/2016), 'Nro Nota Entrega' (0000000073), and 'Vendedor' (Admin Admin). Below this is a section for 'Artículos Terminados' with a search bar and buttons for '+ Añadir', 'Productos Terminados', and 'Ver Todas'. The 'Artículos a Entregar' section includes a table with columns for 'Número', 'Código', 'Descripción', 'Precio Unitario', 'Cantidad', 'Total', and 'Retirar'. The table contains one row for 'BOLETOS SIN TALONARIO 1 COLOR CART KYMBERLY' with a unit price of 0.01, a quantity of 10000, and a total of 70.00. Summary statistics at the bottom show 'TOTAL ORDEN TRABAJO: 615.00', 'VARLOR ABONADO: 230.00', and 'SALDO PENDIENTE: 0.00'. Buttons for 'Actualizar Pago' and 'Finalizar Entrega' are also visible.

Número	Código	Descripción	Precio Unitario	Cantidad	Total	Retirar
0000000012	COMP101	BOLETOS SIN TALONARIO 1 COLOR CART KYMBERLY	0.01	10000	70.00	
					TOTAL ORDEN TRABAJO:	615.00
					VARLOR ABONADO:	230.00
					SALDO PENDIENTE:	0.00

Figura 4. 33 Interfaz gestión entregas ordenes trabajos

Fuente: El Autor (Carlos Burgos)

4.8. Graficas de registro y seguimiento del avance

Las siguientes representaciones graficas muestran el avance de los temas o tareas asignadas a cada Sprint y como han sido desarrolladas en el orden que se desean y el tiempo en el que se prevé su ejecución.

La línea de velocidad proyecta sobre el eje X la fecha o sprint en el que previsiblemente se completarán las versiones representadas en el eje Y.

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.1. Ejecución avance Sprint 1

4.8.1.1. Gráfica de producto o Burn Up

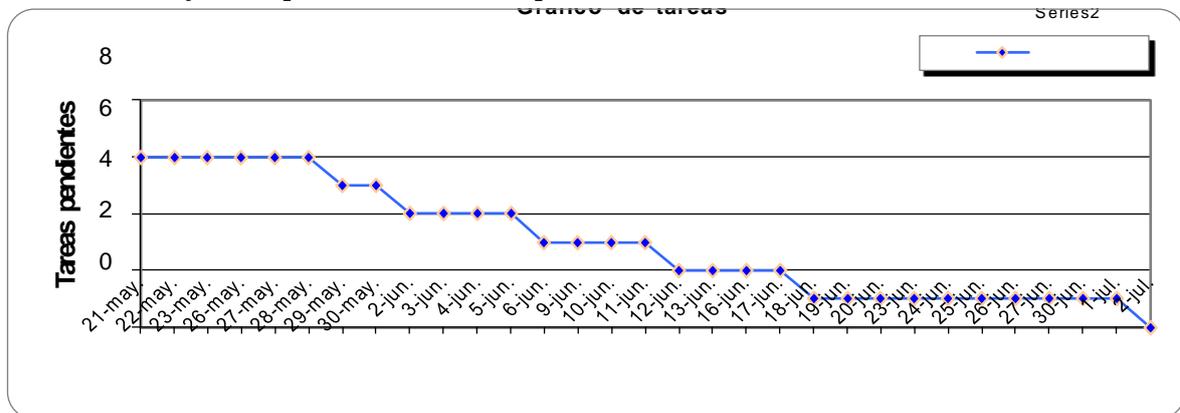


Figura 4. 34 Gráfico de tareas pendientes del Sprint 1

Fuente: El Autor (Carlos Burgos)

4.8.1.2. Gráfica de Avance o Burn Down

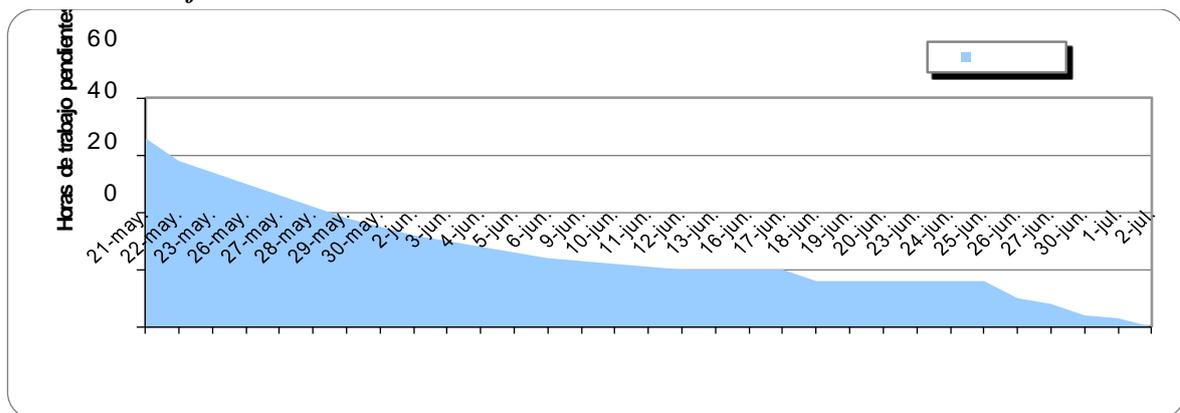


Figura 4. 35 Gráfico de Horas de Trabajo del Sprint 1

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.2. Ejecución avance Sprint 2

4.8.2.1. Gráfica de producto o Burn Up

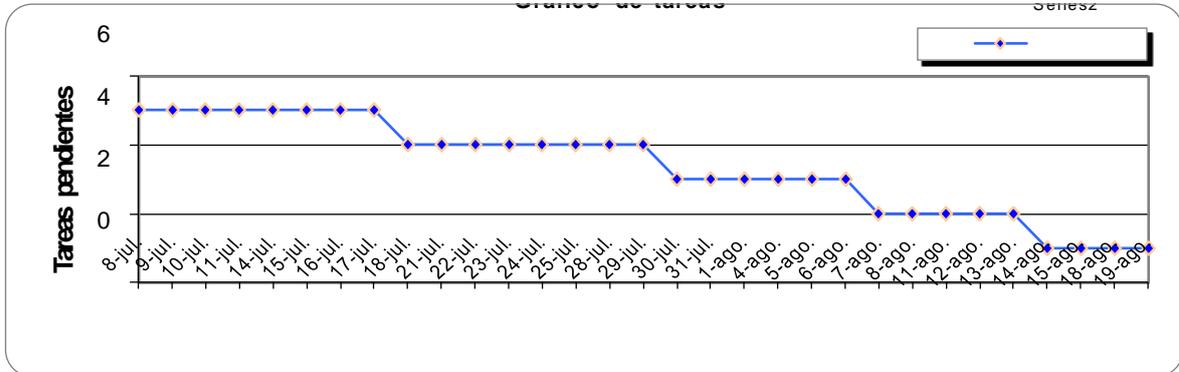


Figura 4. 36 Gráfico de tareas pendientes del Sprint 2

Fuente: El Autor (Carlos Burgos)

4.8.2.2. Gráfica de Avance o Burn Down

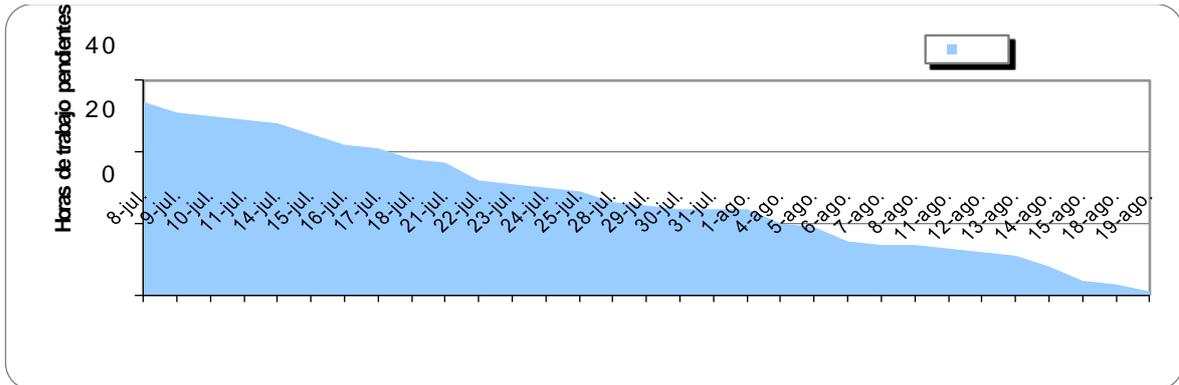


Figura 4. 37 Gráfico de Horas de Trabajo del Sprint 2

Fuente: El Autor (Carlos Burgos)

4.8.3. Ejecución avance Sprint 3

4.8.3.1. Gráfica de producto o Burn Up

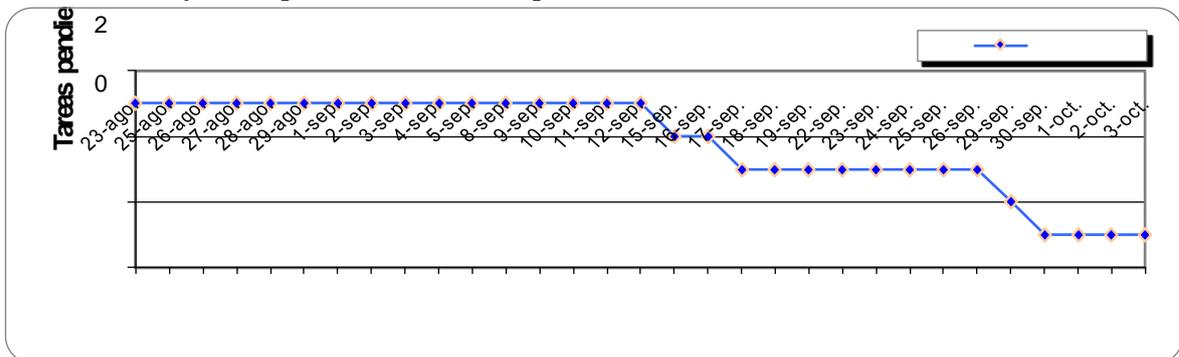


Figura 4. 38 Gráfico de tareas pendientes del Sprint 3

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.3.2. Gráfica de Avance o Burn Down

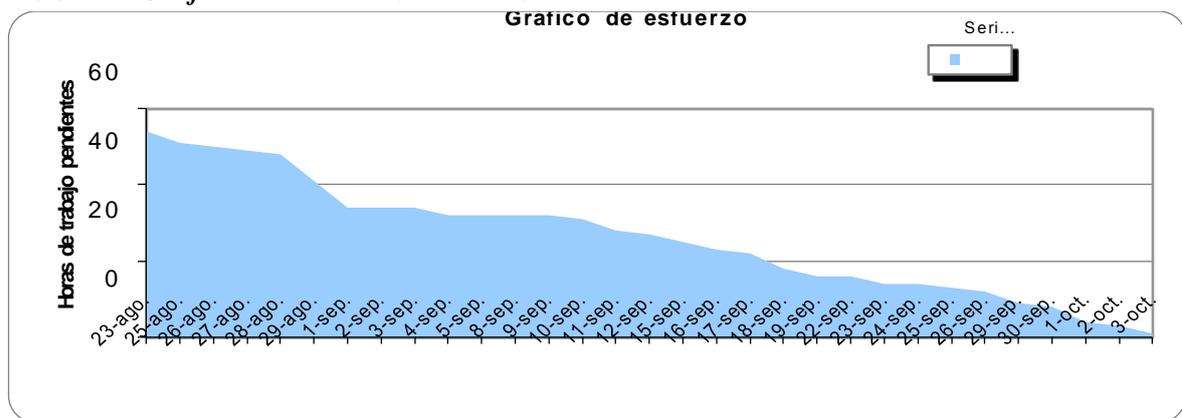


Figura 4. 39 Gráfico de Horas de Trabajo del Sprint 3
Fuente: El Autor (Carlos Burgos)

4.8.4. Ejecución avance Sprint 4

4.8.4.1. Gráfica de producto o Burn Up

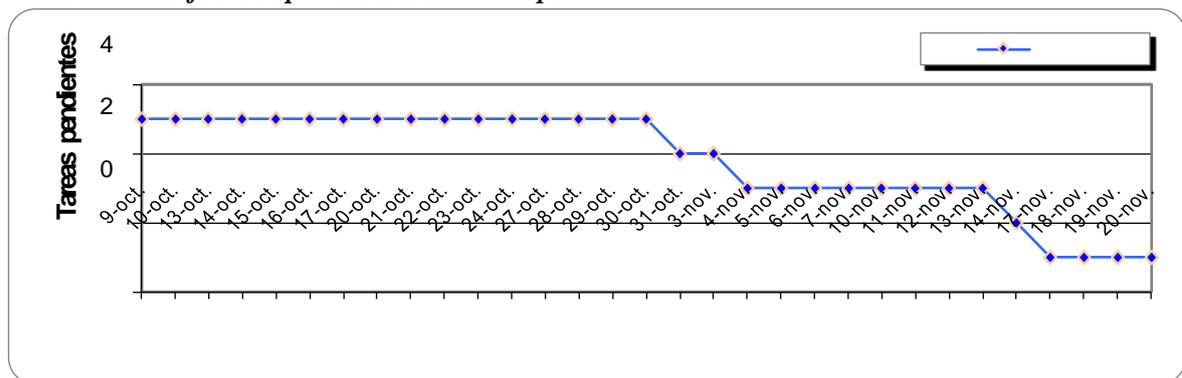


Figura 4. 40 Gráfico de tareas pendientes del Sprint 4
Fuente: El Autor (Carlos Burgos)

4.8.4.2. Gráfica de Avance o Burn Down

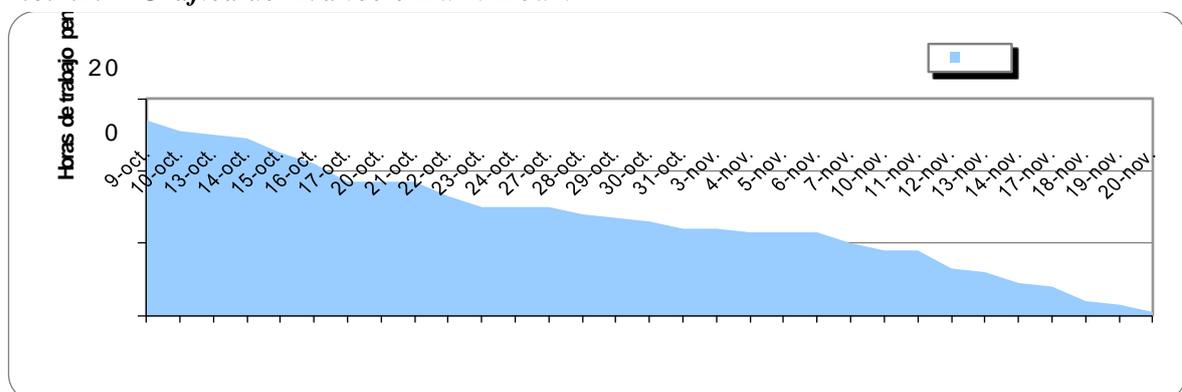


Figura 4. 41 Gráfico de Horas de Trabajo del Sprint 4
Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.5. Ejecución avance Sprint 5

4.8.5.1. Gráfica de producto o Burn Up

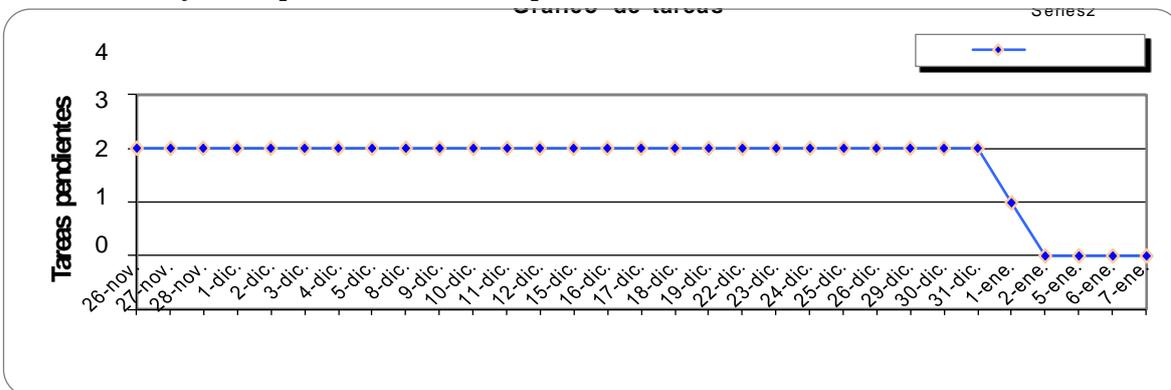


Figura 4. 42 Gráfico de tareas pendientes del Sprint 5

Fuente: El Autor (Carlos Burgos)

4.8.5.2. Gráfica de Avance o Burn Down

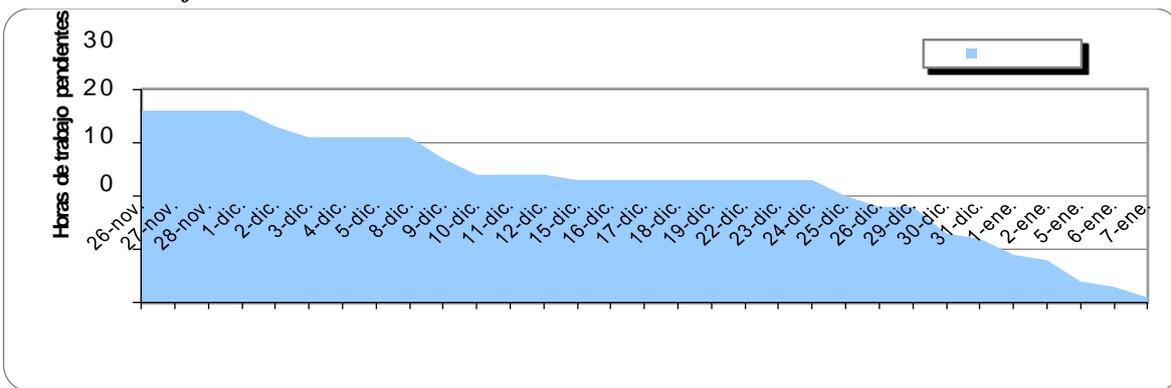


Figura 4. 43 Gráfico de Horas de Trabajo del Sprint 5

Fuente: El Autor (Carlos Burgos)

4.8.6. Ejecución avance Sprint 6

4.8.6.1. Gráfica de producto o Burn Up

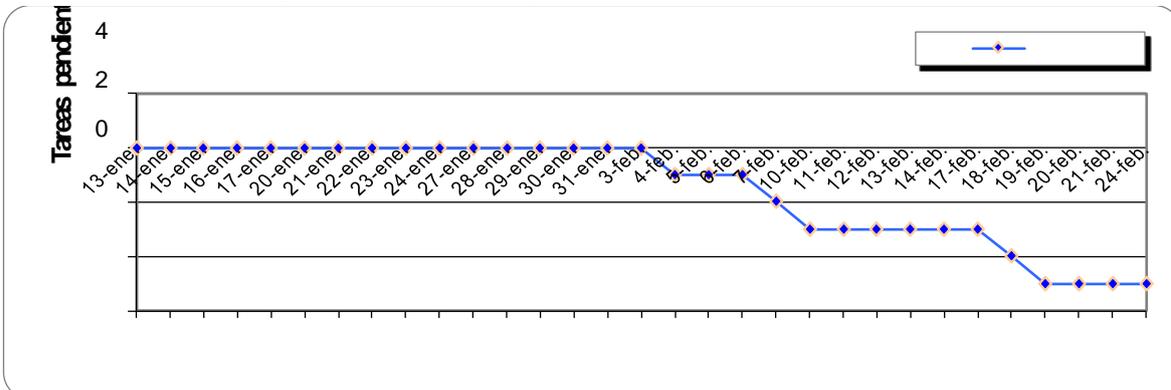


Figura 4. 44 Gráfico de tareas pendientes del Sprint 6

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.6.2. Gráfica de Avance o Burn Down

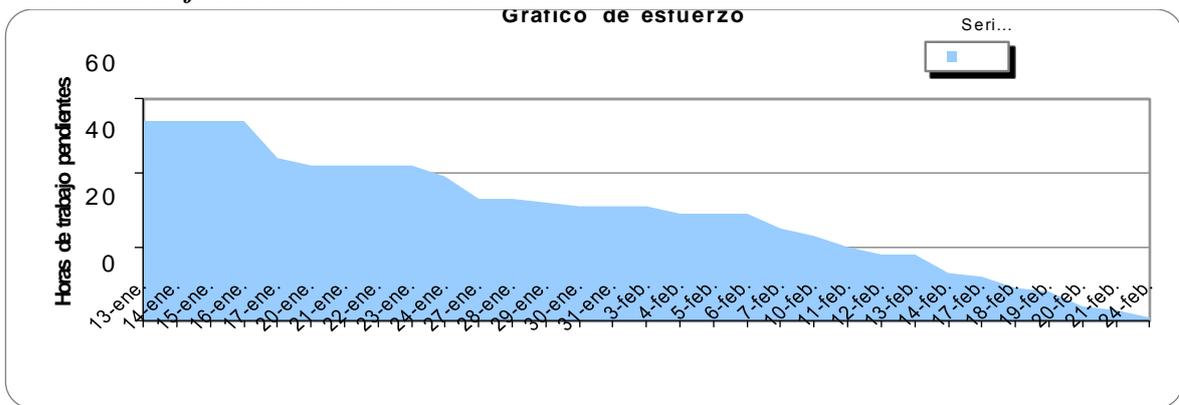


Figura 4. 45 Gráfico de Horas de Trabajo del Sprint 6
Fuente: El Autor (Carlos Burgos)

4.8.7. Ejecución avance Sprint 7

4.8.7.1. Gráfica de producto o Burn Up

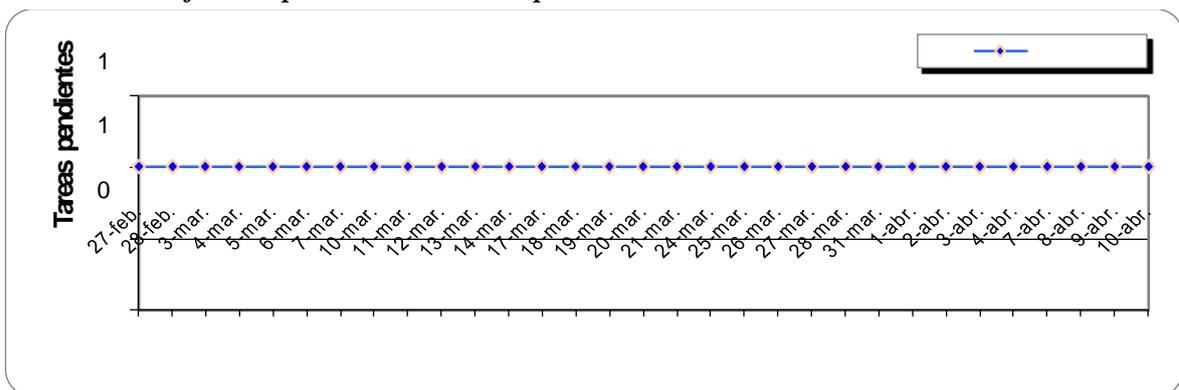


Figura 4. 46 Gráfico de tareas pendientes del Sprint 7
Fuente: El Autor (Carlos Burgos)

4.8.7.2. Gráfica de Avance o Burn Down

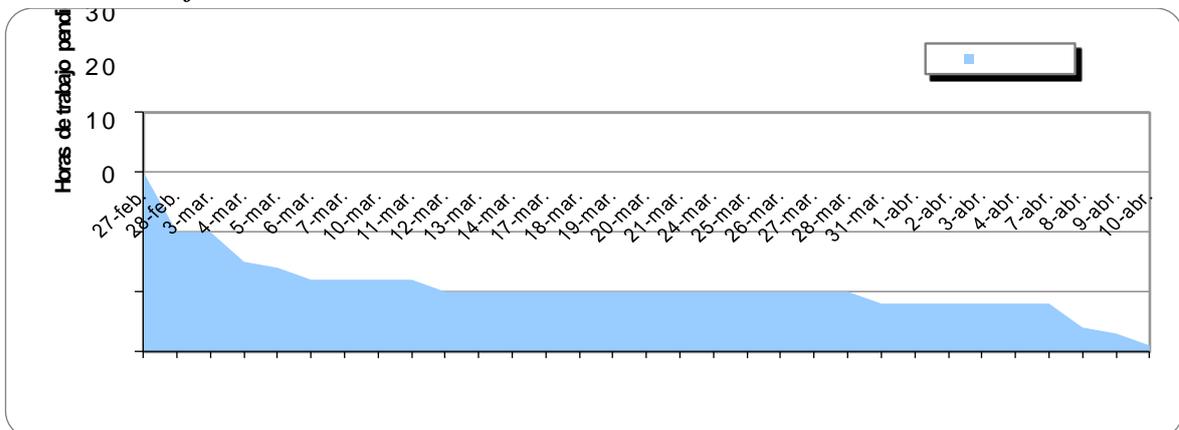


Figura 4. 47 Gráfico de Horas de Trabajo del Sprint 7
Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.8.8. Ejecución avance Sprint 8

4.8.8.1. Gráfica de producto o Burn Up

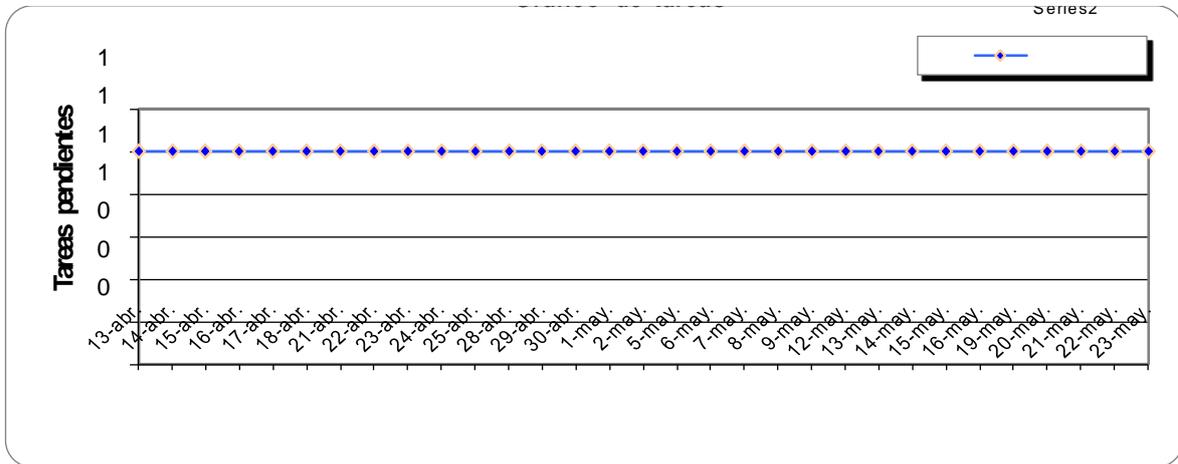


Figura 4. 48 Gráfico de tareas pendientes del Sprint 8

Fuente: El Autor (Carlos Burgos)

4.8.8.2. Gráfica de Avance o Burn Down

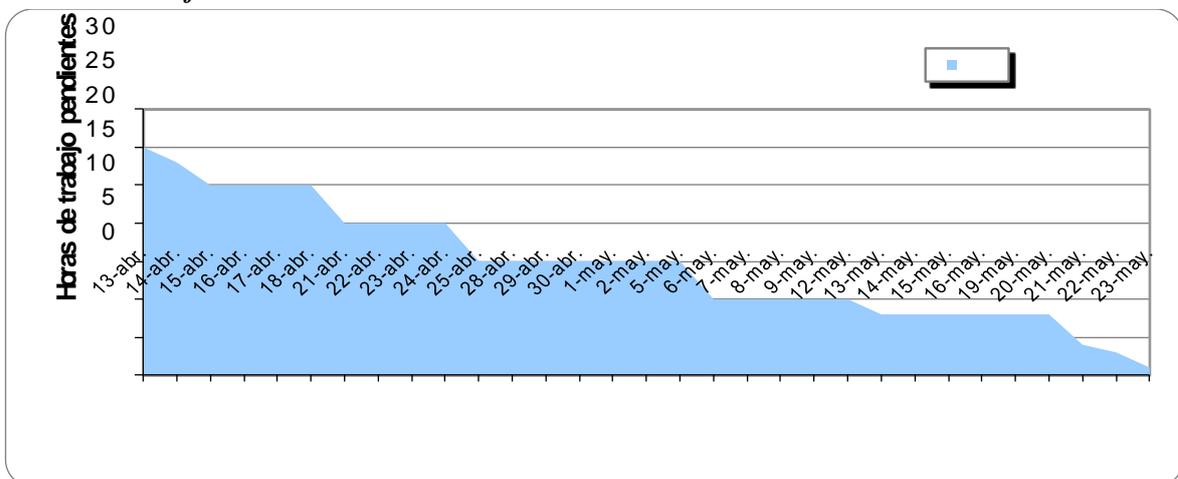


Figura 4. 49 Gráfico de Horas de Trabajo del Sprint 8

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.9. Pruebas

4.9.1. Pruebas de Creación y Modificación de Clientes

Tabla 4. 11

Pruebas de creación y modificación de clientes

Nro. Pruebas	Variables	Se espera	Se obtuvo
1	Datos para guardar un cliente	Llenar la información y el cliente se guardara en la base mientras el sistema muestra un mensaje de confirmación.	El registro se guardó en la base de datos el sistema mostró el mensaje de éxito.
2	Datos para actualizar	Selecciona un registro de la lista de clientes, Los datos se muestran en una ventana flotante, una vez modificado se guarda y el sistema emite un mensaje de confirmación.	El registro del cliente se actualizó correctamente en la base de datos y el sistema mostró el mensaje de confirmación.

Nota. Elaborado por el Autor Carlos Burgos.

4.9.2. Pruebas de Toma de ODT Estandarizadas

Tabla 4. 12

Pruebas de una nueva orden de trabajo

Nro. Pruebas	Variables	Se espera	Se obtuvo
1	Nuevo número de orden de trabajo	Presiona el botón nuevo y se mostrará en el campo número orden en letras rojas el nuevo y único número de orden de trabajo.	Se presionó el botón nuevo y el sistema mostró en el campo número orden el nuevo número de orden.
2	Selección del Cliente	Seleccionar un cliente del listado de clientes y cargará su cedula y sus nombres en un cuadro de texto.	Se seleccionó un cliente de la lista y el sistema cargó los datos del cliente en un cuadro de texto externo.
3	Selección del Producto	Seleccionar de una listado de productos y se presiona el botón añadir y se agrega al detalle de la orden de trabajo	Se seleccionó el producto de la lista, luego se presionó el botón añadir y el sistema agrego al detalle de la orden de trabajo.
4	Iniciar valores del detalle del producto	Ingresar valores manuales y cantidades que se calculen automáticamente al presionar actualizar montos.	Se ingresó el valor unitario y la cantidad, el sistema actualizó el total, el abono y el saldo del detalle del producto.
5	Ingresar datos extras de la orden de trabajo	Se pueda ingresar la fecha de entrega, lugar de entrega y la forma de pago así como también una observación general.	Se ingresó los datos extras de la orden y el sistema inserto en las variables correspondientes los datos.

Capítulo 4: Desarrollo e Implementación del Sistema

6	Confirmar la Finalización de la orden de trabajo.	Se pueda ver los datos de confirmación para la generación de la nueva orden de trabajo, al pulsar finalizar.	Al presionar el botón finalizar el sistema mostro en una ventana flotante la información de la nueva orden de trabajo.
7	Guardar la nueva orden	Al presionar confirmar se guarde la nueva orden de trabajo en la base de datos y se pueda imprimir.	Se confirmó la nueva orden de trabajo y el sistema guardo la nueva orden y activo el botón correspondiente para imprimir.

Nota. Elaborado por el Autor Carlos Burgos.

4.9.3. Pruebas de Creación de Secuencias Procesos de una Línea de Producción

Tabla 4. 13

Pruebas de Secuencias Procesos de una Línea de Producción

Nro. Pruebas	Variables	Se espera	Se obtuvo
1	Datos de Guardado de una Secuencia de Procesos	Poder ingresar la información necesaria y crear un nuevo registro de Secuencias de Procesos.	Se llenó con la información, se guardó en la base de datos y el sistema confirmo la acción.
2	Edición y Eliminación de los datos.	Poder seleccionar un registro de la lista y poder modificar o eliminar.	En cuanto se seleccionó el registro se activó la Edición / Eliminación de la información de la base de datos.

Nota. Elaborado por el Autor Carlos Burgos.

4.9.4. Pruebas de Ver Orden de Trabajo en Producción

Tabla 4. 14

Pruebas de Ver Orden de Trabajo en Producción

Nro. Pruebas	Variables	Se espera	Se obtuvo
1	Actualizar Automáticamente la información	Que cuando se genere una nueva orden de trabajo sea visualizada automáticamente en el departamento de producción.	Durante la generación de una nueva orden de trabajo se guardó la nueva orden en la base de datos y se actualizó el registro de la información en Producción.
2	Ver detalle de nueva orden de trabajo	Poder mirar el detalle de impresión con todos sus acabados.	Por medio de un botón del listado se pudo abrir una ventana de información del detalle de la orden de trabajo.

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 4: Desarrollo e Implementación del Sistema

4.9.5. Pruebas de Inicio de Proceso de una Orden de Trabajo

Tabla 4. 15

Pruebas de Inicio de Proceso de una Orden de Trabajo

Nro. PRUEBAS	VARIABLES	SE ESPERA	SE OBTUVO
1	Poner en marcha una orden de trabajo.	Se pueda iniciar el proceso de producción de una nueva orden de trabajo	Se seleccionó la nueva orden del listado y al pulsar iniciar el sistema puso en marcha cada uno de sus procesos de producción a ser completados.
2	Definir la prioridad	Tener una lista de prioridades la cual se le pueda dar una a la orden de trabajo a ser producida	El sistema cuenta con una lista que permitió seleccionar una prioridad y ordeno su producción por orden prioritario.

Nota. Elaborado por el Autor Carlos Burgos.

4.9.6. Pruebas de Gestión de Tareas de una Orden de Trabajo

Tabla 4. 16

Pruebas de Inicio de Proceso de una Orden de Trabajo

Nro. PRUEBAS	VARIABLES	SE ESPERA	SE OBTUVO
1	Pedir una tarea	Como usuario de un departamento pedir si cierta tarea en proceso a ser completada	Si la tarea pertenece al departamento del empleado puede ser completada con éxito caso contrario no se muestra.
2	Asignar, Delegar tareas a usuarios	Se pueda delegar tareas a los empleados del mismo departamento.	Las tareas han sido asignadas exitosamente en el sistema a los empleados del mismo departamento.
3	Registro de actividades	Todos los eventos realizados en la tarea actual sean registrados por el sistema.	Durante la ejecución de la tarea actual se registra eventos automáticos por el sistema.

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 4: Desarrollo e Implementación del Sistema

4.9.7. Entregas Órdenes de Trabajos

Tabla 4. 17

Pruebas de Entregas Órdenes de Trabajo.

Nro. PRUEBAS	VARIABLES	SE ESPERA	SE OBTUVO
1	Identificación de tareas completadas.	Se pueda realizar entregas de las órdenes de trabajos completadas.	El sistema mostró la lista de las tareas completadas y listas a ser entregadas.
2	Modificar totales	Se pueda ingresar nuevos valores de totales, saldos y abonos en una orden de trabajo a entregar	El sistema permitió modificar los valores y recalcularlos.
3	Entregas parciales o totales.	Poder realizar la entrega de las órdenes de trabajo ya sea en forma parcial o total.	El sistema presenta de forma parcial o total las órdenes de trabajo a ser entregadas.

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 4: Desarrollo e Implementación del Sistema

4.9.8. Pruebas Ejecución de los Sprint del Sistema

4.9.8.1. LISTA DE CHEQUEO: Control de Calidad de Sprints desarrollados

LISTA DE CHEQUEO:

CONTROL DE CALIDAD DE SPRINTS DESARROLLADOS

Item/s inspeccionado/s: Sprint 1, Sprint 2, Sprint 3, Sprint4, Sprint 5, Sprint 6	Fecha: 01/09/2015
Puntos chequeados: <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5	Inspector: Carlos Burgos, Diego Oña
1. Componentes usados	
¿Los componentes usados son correctos?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
¿Se poseen los registros de recepción de los componentes?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
Código de los informes de recepción:	
2. Actividades realizadas	
¿Se siguieron los procedimientos?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
¿Se usaron las revisiones vigentes de los procedimientos?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
¿Se rellenaron los registros y estos son correctos?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
3. Incidencias	
¿Producto final conforme?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
¿Existe alguna incidencia relacionada?	<input type="checkbox"/> SI <input checked="" type="checkbox"/> NO <input type="checkbox"/> N/A
Código incidencias relacionadas:	
4. Tiempos de producción	
¿Existieron retrasos en la fabricación?	<input type="checkbox"/> SI <input checked="" type="checkbox"/> NO <input type="checkbox"/> N/A
¿Hubo máquinas indisponibles?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/P
5. Entrega y logística	
¿Producto correctamente identificado?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
¿Producto conforme a las especificaciones del cliente?	<input checked="" type="checkbox"/> SI <input type="checkbox"/> NO <input type="checkbox"/> N/A
Observaciones	
Todos los Sprint se encuentran correctamente funcionando.	

NOTA: N/A = No aplicable. N/P = No presenciado.

Figura 4. 50 Informe de ejecución de Pruebas

Fuente: El Autor (Carlos Burgos)

Capítulo 4: Desarrollo e Implementación del Sistema

4.9.9. Pruebas de Puesta en Producción e Integración con los Sprints

Tabla 4. 18

Pruebas de puesta en producción e integración con los Sprints

Nombre del indicador	Definición	Umbral para el cumplimiento
Indicador de disponibilidad	SIGPROPED debe tener una disponibilidad de uso de sus funcionalidades medido sobre el tiempo operacional de 8 horas diarias. Esta disponibilidad no cubre aspectos externos no controlables, como caída de fluido electrónico o de red por ejemplo.	>90%
Indicador de carga de información	El sistema pudo ser cargado con la información que se definió en la empresa anteriormente.	=100%
Indicador de operación:	Mide la capacidad del sistema para cumplir las funcionalidades contratadas y aprobadas por la empresa en el plan de pruebas. Se mide por el % de procesos que son concluidos exitosamente dentro del Sistema.	>97%
Indicador de integración	Mide la capacidad del sistema para cumplir las funcionalidades de integración con los demás Sprints. Se mide el % de las tareas concluidas exitosamente.	>97%
Indicador de interfaces	Este indicador mide la estabilidad del Sistema para realizar la carga de información de fuentes externas. Se mide por el % por el número de subidas de información intentados vs los intentos efectivos	>97%
Indicador de uso del sistema	Este indicador mide el número de usuarios trabajando sobre el Sistema.	Mínimo el 80% de los usuarios capacitados en la empresa DIKAPSA.
Indicador de concurrencia	Este indicador mide el número de usuarios concurrentes trabajando sobre el Sistema.	El número de sesiones activas mínima fue el 20% de los empleados a los que se les dio el acceso.

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 5

ANÁLISIS DE IMPACTOS



El presente capítulo describe el impacto ambiental y socio económico que genera la implementación de este proyecto además de un pequeño análisis de costos totales.

Se ha tomado la decisión de realizar un análisis de impactos, es decir reflexionar los aspectos de la ingeniería del sistema que genere en ciertas áreas como son impacto social, económico, ambiental y educativo.

Además se ha construido una escala de valores que permitirá evaluar de una manera cuantitativa el impacto en cada área, se detalla a continuación la escala:

Tabla 5. 1
Escala de Valores

Calificación	Evaluación
Impacto alto Negativo	-3
Impacto medio Negativo	-2
Impacto bajo Negativo	-1
No hay Impacto	0
Impacto bajo Positivo	1
Impacto medio Positivo	2
Impacto alto Positivo	3

Nota. Elaborado por el Autor Carlos Burgos.

Capítulo 5: Análisis de Impactos

5.1. Impactos del Proyecto

Los impactos para el presente proyecto se explican con las siguientes matrices las cuales se detallan así:

- Se asignará para cada área una matriz, donde horizontalmente se ubica los nivel de impacto y verticalmente se determina una serie de indicadores.
- Para cada indicador se le asignará un nivel de impacto ya sea positivo o negativo; luego se encuentra la sumatoria de los niveles de impacto, la misma que será dividida para el número de indicadores establecidos, obteniendo el nivel de impacto del área estudiada.
- Se indica un breve análisis bajo cada matriz donde se argumenta las razones que conllevan a asignar el nivel de impacto.

5.1.1. Impacto económico

Inicialmente se detalla impacto económico que tiene la aplicación del modelo propuesto para así determinar el beneficio que tiene la empresa.

Tabla 5. 2
Nivel de impacto económico

INDICADOR	Nivel de Impacto							TOTAL
	-3	-2	-1	0	1	2	3	
Liquidez empresarial							X	3
Generación de empleo						X		2
Reinversión de utilidades						X		2
TOTAL								7

Nota. Total de impacto económico: $7/3 = 2.33$ y Nivel de impacto económico: medio positivo. Elaborado por el Autor Carlos Burgos.

Capítulo 5: Análisis de Impactos

5.1.1.1. Análisis impacto económico

La empresa DIKAPSA beneficiará económicamente a varios entes, por un lado a los propietarios y por otra parte a los clientes brindándoles buenos productos, la empresa podrá cubrir todas sus obligaciones, es por eso que se dice que tendrá una liquidez positiva.

Se contratará personal calificado y no calificado, los mismos que tendrán que ser capacitados durante todo el proceso de arranque del proyecto, de lo cual se destinará la mayor parte a la reinversión para equipos o maquinaria nueva que beneficien el crecimiento de la empresa.

5.1.2. Impacto ambiental

Además la evaluación del impacto ambiental es hoy en día un parámetro muy importante en la concepción de cualquier proyecto, teniendo en cuenta las influencias positivas o negativas, por todo ello la siguiente tabla demuestra los factores inmersos dentro de la implementación de este proyecto.

Tabla 5. 3

Nivel de impacto ambiental

INDICADOR	Nivel de Impacto							TOTAL
	-3	-2	-1	0	1	2	3	
Conservación del medio ambiente						X		2
Menor nivel de contaminación						X		2
Uso de insumos orgánicos					X			1
Desechos peligrosos						X		2
TOTAL								7

Nota. Total de impacto ambiental: $7/4 = 1.75$ y Nivel de impacto ambiental: medio positivo. Elaborado por el Autor Carlos Burgos.

Capítulo 5: Análisis de Impactos

5.1.2.1. Análisis impacto ambiental

A través de la implementación del sistema en la empresa se ahorrará hojas, ya que no se imprimirá una y otra vez el pedido, sino únicamente una final reduciendo notoriamente el gasto en suministro hojas.

Por otra parte los desechos que despiden los materiales utilizados son la mayoría de ellos amigables con el medio ambiente por lo que no causa daño ambiental y antes ayuda con la conservación del ecosistema.

5.1.3. Impacto social y cultural

En este ítem del proyecto se prevé beneficiar a los usuarios finales o consumidores de la ciudad de Otavalo, brindando un mejor servicio y apoyando a todo los usuarios que necesita los servicios de la empresa DIKAPSA.

Tabla 5. 4

Nivel de impacto social y cultural

INDICADOR	Nivel de Impacto							TOTAL
	-3	-2	-1	0	1	2	3	
Consumo							X	3
Generación de empresas productivas							X	3
Bienestar comunitario							X	3
Servicio							X	3
TOTAL								12

Nota. Total de impacto social y cultural: $12/4 = 3$ y Nivel de impacto social y cultural: Positivo.

Elaborado por el Autor Carlos Burgos.

5.1.3.1. Análisis impacto social y cultural

Como podemos observar mediante la ejecución de este proyecto estamos no solo dando las facilidades a los habitantes de la ciudad de Otavalo de obtener el servicio sino también culturalmente permite el crecimiento del sector.

Capítulo 5: Análisis de Impactos

Notablemente podemos ver que es uno de los impactos más relevantes y positivos beneficiando al usuario, comunidad y el entorno que lo rodea.

5.1.4. Impacto general.

El impacto general es el resumen de los impactos anteriormente vistos dándonos una idea general del impacto que genera la implementación de este proyecto.

Tabla 5. 5

Nivel de impacto general

INDICADOR	Nivel de Impacto							TOTAL
	-3	-2	-1	0	1	2	3	
Impacto económico						X		2
Impacto ambiental						X		2
Impacto social y cultural							X	3
TOTAL								7

Nota. Total de impacto social y cultural: $7/3 = 2,33$ y Nivel de impacto social y cultural: Positivo.

Elaborado por el Autor Carlos Burgos.

5.1.4.1. Análisis impacto general

El impacto a nivel general que dará el proyecto será de 2.33, lo cual constituye un impacto positivo medio; lo que quiere decir que la empresa mejorará notablemente implementando un sistema para la gestión de procesos.

Además generará fuentes de trabajo en la ciudad de Otavalo, hay que recalcar que la implementación del sistema no reducirá el número de empleados, porque el sistema lo permite poner orden más no va a realizar las funciones de los empleados.

5.2. Análisis de Costos

En base a los requerimientos de la empresa DIKAPSA se procedió a solicitar proforma los equipos necesarios para la implementación del sistema, tomando en cuenta que

Capítulo 5: Análisis de Impactos

el diseño del sistema se lo realizó con herramienta libres lo quiere decir que tienen un costo cero dólares.

El análisis que se realizará constituye la técnica matemático-financiero para determinar los beneficios o inversiones que se realizarán para la ejecución de este proyecto, las posibilidades de seguir creciendo.

5.2.1 Costo de mantenimiento.

Los costos de mantenimiento son muy difíciles de estimar con anticipación, varía mucho de una aplicación a otra, por lo que vamos a estimar aproximadamente un monto para el cálculo de costos.

Supondremos que el costo de mantenimiento es una vez por año y es de 500,00 dólares, debido a que se gastará solamente en mantenimiento del equipo que aloja el sistema.

5.2.2 Análisis económico

Para conocer la factibilidad y la rentabilidad del mismo vamos a detallar un análisis económico general donde podremos conocer los costos de inversión y los beneficios del proyecto.

5.2.2.1 Costo de inversión

- **Costo de equipos:** para la implementación del sistema es necesario un computador con las características detalladas en el capítulo 4 considerando los requerimientos antes mencionados.

Capítulo 5: Análisis de Impactos

Tabla 5. 6

Requerimiento hardware

Item	Cantidad	Descripción	Valor U	Subtotal (USD)
1	1	Servidor web PC	800,00	800,00
			IVA 12%	96,00
			TOTAL	896,00

Nota. Elaborado por el Autor Carlos Burgos.

- **Costo de implementación y ejecución:** hace referencia a costo del desarrollo del sistema, la investigación y documentación.

Tabla 5. 7

Costo de elaboración

Item	Cantidad	Descripción	Valor U	Subtotal (USD)
1	300 horas	Recopilación de información	1,50	450,00
2	50 horas	Investigación de campo	3,00	150,00
3	50 horas	Elaboración de planes y cronogramas	3,00	150,00
4	400 horas	Diseño y desarrollo de software	3,00	1200,00
5	100 horas	Redacción de documento	1,50	150,00
			TOTAL	2100,00

Nota. Elaborado por el Autor Carlos Burgos.

Tabla 5. 8

Costo de implementación

Item	Cantidad	Descripción	Valor U	Subtotal (USD)
1	4	Impresiones tesis y demás	30,00	120,00
2	400	Copias de la tesis	0,05	20,00
3	1	Licencia software	0,00	0,00
4	1	Equipo para desarrollo	500,00	500,00
5	100	Transporte ciudad de Ibarra- Otavalo	0,50	50,00
6	1	Varios	200,00	200,00
			TOTAL	890,00

Nota. Elaborado por el Autor Carlos Burgos.

El costo total del proyecto es de 3886,00 dólares de los cuales debemos rescatar que la inversión es mínima para los múltiples beneficios que la empresa DIKAPSA recibirá.

CONCLUSIONES

- Se cumplió a satisfacción con la implementación del sistema de gestión de procesos de producción para la empresa DIKAPSA y su puesta en producción de las dos líneas de producción de más demanda como son las gigantografías y comprobantes o libretines.
- Se concluye que mediante la utilización del presente sistema se disminuye su impacto tanto económico, ambiental social y cultural.
- El estudio de Impacto Ambiental se deduce que el proyecto no tiene incidencia significativa en lo que se refiere al alto consumo de recursos como hojas.
- Se entregó a la empresa DIKAPSA un manual de usuario donde detalla paso a paso el funcionamiento del sistema, cumpliendo con el último requerimiento.
- Se entregó a la persona encargada de la administración un manual técnico para su administración y documentación del mismo.
- La selección de la metodología ágil para el desarrollo de software fue una parte muy importante, ya que en esta metodología, el ciclo de vida del software sigue la secuencia, es decir no se puede continuar a la siguiente etapa si la etapa anterior presenta errores.
- La etapa de pruebas ayudó a detectar los errores que se presentaron como: errores de duplicidad de código, errores de validación de datos y errores de visualización de los datos. De esta manera se pudo corregirlos, para garantizar el correcto funcionamiento del Sistema de gestión de procesos de producción para la empresa DIKAPSA.
- La etapa de implantación permitió a los usuarios familiarizarse con el uso del sistema en el tiempo de una semana, dedicando 15 minutos diarios, su aprendizaje fue satisfactorio, la capacitación tomó una duración de 2 horas.

- Con respecto al administrador del sistema, el mismo fue capacitado en su lugar de trabajo y el tiempo que tomó este fin fueron dos horas, la capacitación incluyó el proceso de respaldos de la base de datos.
- El uso del gestor de base de datos PostgreSQL fue satisfactorio, por la seguridad de la conectividad y el soporte de gran cantidad de datos.
- El lenguaje utilizado Java es muy flexible y es un lenguaje que se ejecuta en el servidor por lo que es invisible al navegador.
- SCRUM por sus características no es válido para cualquier proyecto ni para cualquier persona o equipo de personas. Es más, SCRUM según muchos especialistas de esta metodología, es óptima para equipos de trabajo de hasta 8 personas, aunque hay empresas que han utilizado SCRUM con éxito con equipos más grandes.
- Además SCRUM se puede trabajar prácticamente desde el primer momento y empezar a sacar frutos de ese trabajo para que el administrador de la empresa vaya viendo los avances y se quede satisfecho con lo que se está haciendo y cómo se está haciendo.

RECOMENDACIONES

- Capacitar a los empleados mediante el manual de usuario entregado para garantizar el buen uso del sistema.
- Realizar un mantenimiento o actualización periódica del sistema dependiendo de cómo vaya incrementando sus servicios la empresa.
- Se recomienda al encargado del sistema que en calidad de administrador del “SISTEMA DE GESTIÓN DE PROCESOS DE PRODUCCIÓN PARA LA EMPRESA DIKAPSA, garantizar la integridad y veracidad de los datos periódicamente.
- Se recomienda al administrador del sistema realizar respaldos de la base de datos constantemente con la finalidad de evitar posibles inconvenientes a futuro de pérdidas de valiosa e importante información.
- Se recomienda a los usuarios del Sistema revisar el manual de usuario (Anexo 2) en el caso de presentarse alguna duda con el funcionamiento.
- Debido a que el servidor web apache a más de ser de código abierto es multiplataforma. Por esta razón el sistema es escalable y además permite añadir nuevos módulos al sistema.

GLOSARIO DE TÉRMINOS

- **Actividad:** son los elementos de más bajo nivel que se analizan y adquieren pleno sentido al cuestionar su valor al interior de un proceso.
- **Apache:** es un servidor web y es completamente libre, ya que es un software Open Source y con licencia GPL.
- **ACID:** Atomicidad, Consistencia, Aislamiento y Durabilidad.
- **Bytecode:** un archivo binario que contiene un programa ejecutable similar a un módulo objeto.
- **BSD:** Berkeley Software Distribution o BSD (en español, «distribución de software berkeley») fue un sistema operativo derivado del sistema Unix
- **Backing Beans:** JavaBeans especializados que recogen valores de los componentes UI e implementan métodos listener de eventos.
- **C:** es usado para crear programas de software para computadoras en empresas, hogar, juegos y otras áreas
- **C++:** toma la base del lenguaje C, es un lenguaje de programación multiparadigma.
- **Catalina:** A partir de la versión 4.0, Jakarta Tomcat utiliza el contenedor de servlets Catalina.
- **Cliente:** es la persona, empresa u organización que adquiere o compra de forma voluntaria productos o servicios que necesita o desea para sí mismo, para otra persona o para una empresa u organización.
- **Componente UI** Se trata de un objeto con estado, mantenido por el servidor, que proporciona funcionalidad específica para interactuar con un usuario final. Los Componente UI son JavaBeans con propiedades, métodos y eventos.

- **Converter:** Convierte un valor de un componente a y desde una cadena para mostrarlo. Los converter se asocian a un componente UI.
- **DBMS:** Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos.
- **Desensamblador:** un programa de computador que traduce el lenguaje de máquina a lenguaje ensamblador.
- **DTD:** es un documento que define la estructura de un documento XML, los elementos, atributos, entidades, notaciones.
- **EJB3:** simplificar el desarrollo de aplicaciones Enterprise Java.
- **Events y Listeners:** JSF usa el modelo event/listener de JavaBeans. Los Componentes UI generan eventos y los listeners pueden ser registrados para manejar dichos eventos.
- **Empresa:** es la organización que aplica unas capacidades o recursos para satisfacer determinadas necesidades de sus clientes.
- **Enterprise JavaBeans:** son una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE (ahora JEE) de Oracle Corporation.
- **Encapsulación:** técnica en la que la información a enviar se coloca en el área de datos de un paquete o cuadro. Puede encapsularse el paquete de un protocolo en otro.
- **FTP:** (File Transfer Protocol) Nombre del protocolo estándar de transferencia de ficheros. Su misión es permitir a los usuarios recibir y enviar ficheros de todas las máquinas que sean servidores FTP.
- **GWT:** es un conjunto de herramientas de desarrollo para la construcción y optimización de aplicaciones basadas en navegador complejos.

- **Herencia:** crear clases derivadas de otras existentes, que heredan sus tipos y métodos y pueden contener otros nuevos. Si una nueva clase hereda propiedades de más de una antecesora, se denomina herencia múltiple; no todos los lenguajes orientados a objetos lo permiten.
- **HTTP:** (Hiper Text Transfer Protocol). Es el protocolo de Internet que permite que los exploradores del WWW recuperen información de los servidores.
- **Hostname:** es el programa que se utiliza para mostrar o establecer el nombre actual del sistema también llamado nombre de equipo.
- **Iterativo:** acto de repetir un proceso con el objetivo de alcanzar una meta deseada, objetivo o resultado.
- **JMS:** servicio de mensajes Java, también es conocido como Middleware Orientado a Mensajes (MOM) y es una herramienta universalmente reconocida para la construcción de aplicaciones empresariales.
- **JBoss:** es un servidor de aplicaciones Java EE de código abierto implementado en Java puro.
- **Jakarta:** Apache Tomcat también llamado Jakarta Tomcat o simplemente Tomcat, funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation.
- **JavaBeans:** La clase Java encapsula los valores de instancia y sus comportamientos en una sola unidad llamada objeto.
- **Manufactura:** Proceso de fabricación de un producto que se realiza con las manos o con ayuda de máquinas.
- **Melé:** Amontonamiento de personas o de cosas.
- **Método:** es un bloque de código que contiene una serie de instrucciones.

- **Norma:** es una estandarización con el medio con mayor o menor grado de obligatoriedad
- **Objetos:** entidad existente en la memoria del ordenador que tiene unas propiedades (atributos o datos sobre sí mismo almacenados por el objeto) y unas operaciones disponibles específicas (métodos).
- **OpenOffice:** es una suite ofimática libre (código abierto y distribución gratuita) que incluye herramientas como procesador de textos, hoja de cálculo, presentaciones, herramientas para el dibujo vectorial y base de datos.
- **Pila de producto:** es una lista priorizada de historias de usuario más o menos detalladas.
- **Poliéster:** Resina plástica que se obtiene mediante una reacción química y que es muy resistente a la humedad y a los productos químicos.
- **Prototipado:** es un proceso utilizado para fabricar artículos de plástico, metal o cerámica.
- **Proceso:** cumple un objetivo completo y que agrega valor para el cliente.
- **Procedimiento:** se aplica a las tareas o actividades y las documenta en detalle.
- **Polimorfismo:** Permite al programador generar componentes reutilizables de alto nivel que puedan adaptarse a diferentes aplicaciones mediante el cambio de sus partes de bajo nivel.
- **POJO:** por sus siglas en inglés "Plain Old Java Object", que significa "Un objeto Java simple y plano". Un POJO es una instancia de una clase que no extiende ni implementa interfaces especificadas previamente no ligada a ninguna restricción sirve para enfatizar el uso de clases simples y que no dependen de un framework en especial.

- **Rich Client Platform:** Plataforma de cliente enriquecido es una forma de crear aplicaciones Java que pueden competir con las aplicaciones nativas en cualquier plataforma.
- **SDK:** incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java.
- **Spring Backlog:** es una lista de tareas de la iteración.
- **Skinning:** máscara de Interfaz gráfica.
- **SSL:** Secure Sockets Layer (SSL; en español «capa de conexión segura») son protocolos criptográficos que proporcionan comunicaciones seguras por una red.
- **Socket:** es un punto final de un proceso de comunicación. Es una abstracción que permite manejar de una forma sencilla la comunicación entre procesos.
- **Regla:** es parte de un reglamento interno.
- **Renderer:** Es el responsable de mostrar un componente UI y traducir la entrada del usuario en valores de componentes.
- **Tarea:** es el desarrollo de la actividad en acciones muy específicas.
- **TCP IP:** Transmission Control Protocol/Internet Protocol). Se trata de un estándar de comunicaciones muy extendido y de uso muy frecuente para software de red basado en Unix con protocolos Token-Ring y Ethernet, entre otros.
- **Validador** Es el responsable de asegurar que el valor introducido por un usuario es correcto. Podemos asociar uno o más validadores a un Componente UI.
- **Variable:** es un espacio de memoria reservado para almacenar un valor que corresponde a un tipo de dato soportado por el lenguaje de programación.
- **/bin:** permite el arranque, detención, otros scripts útiles y ejecutables para Unix o Windows.
- **/common:** clases comunes que pueden utilizar Catalina y las aplicaciones web.

- **/conf:** ficheros de configuración y DTDs relacionadas. El más importante es el `server.xml`, principal fichero de configuración del contenedor. ficheros XML y los correspondientes DTD para la configuración de Tomcat.
- **/log:** ficheros de log, es un registro e de eventos durante un rango de tiempo en particular. logs de Catalina y de las aplicaciones.
- **/server:** clases utilizadas solamente por Catalina.
- **/shared:** clases compartidas por todas las aplicaciones web.
- **/webapps:** directorio de despliegue para las aplicaciones web. directorio que contiene las aplicaciones web.
- **/work:** almacenamiento temporal de ficheros y directorios.

LISTA DE REFERENCIAS

- Esteban, A. (2010). *Programación en Java*. Grupo Eidos. Recuperado de:
<http://www.intercambiosvirtuales.org/libros-manuales/programacin-en-java-angel-esteban-grupo>
- Atmosphere/atmosphere*. (2015). GitHub. Recuperado de:
<https://github.com/Atmosphere/atmosphere>. (01/12/2015)
- Chase, R. B. (2005). *Administración de la Producción y Operaciones*. México.
- Complex Project*. (2015). SCRUM Aliance. Recuperado de: www.SCRUMalliance.org.
 (03/12/2015)
- Deemer, P. (2012). *Una introducción básica a la teoría y práctica de Scrum*. (Versión 2.0).
 Projectalis.
- Desarrollo de un sistema que permita la gestión de registro de calificaciones en la escuela fiscal de educación básica NICOLÁS GUILLÉN*. (2015). Tesis Universidad Central del Ecuador. Recuperado de:
<http://translate.google.com/translate?hl=es&sl=en&u=http://www.oracle.com/technetwork/java/javase/jdbc/&prev=search>. (23/07/2015)
- Eclipse*. (2013). Eclipse. Recuperado de:
<http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/887>
 (03/11/2015)
- Eclipse*. (2015). EcuRed. Recuperado de:
http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.
 (18/04/2015)
- El conocimiento es la meta*. (2014). Dataprix. Recuperado de:
<http://www.dataprix.com/ti/manual-introduccion-desarrollo-primefaces>.
 (03/02/2015)

- Entorno de desarrollo integrado.* (2015). EcuRed. Recuperado de: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado. (12/03/2015)
- Estudio de Impacto Económico.* (2015). PricewaterhouseCoopers International Limited (PwCIL). Recuperado de: http://www.pwc.es/es_ES/es/sector-publico/assets/brochure-estudios-impacto-economico.pdf. (10/01/2015)
- Etheridge, D. (2010). *Java: Classes in Java Application and Introduction to Java Programming*. EEUU: Ventus Publishing. ISBN 978-87-7681-495-3.
- Fields, M.A. (2010). *Web Development with JavaServer Pages*. (7° Edición). Greenwich.
- FUOC. Universitat Oberta de Catalunya. (2008). Recuperado de: http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02152.pdf. (28/08/2015)
- Galli, P. (2015). *Linux & Open Source*. Moglen: GPL 3.0 Rewrite Drive Is No Democracy eWeek.com.
- Geiger, K. (1995). *Inside ODBC*. Redmond: Microsoft Press.
- González A. C. (2013). Recuperado de: <http://www.adictosaltrabajo.com/tutoriales/introduccion-jsf-java/> (29/06/2015)
- Hall, M. (2012). *Coreservlets.com*. Recuperado de: <http://www.coreservlets.com/JSF-Tutorial/primefaces/>
- Ipinza, F. (2012). *Administración de las Operaciones Productivas*. México.
- IRreport Diseñador.* (2015). Jaspersoft community. Recuperado de: <http://community.jaspersoft.com/project/ireport-designer>. (23/09/2015)
- JDBC – Introduction.* (2015). TutorialPoints. Recuperado de: <http://www.tutorialspoint.com/jdbc/jdbc-introduction.htm>. (17/07/2015)
- JPA.* (2015). Tutorial Point. Recuperado de: http://www.tutorialspoint.com/es/jpa/jpa_introduction.htm. (14/02/2015)

- JSF + JPA + JasperReports (iReport)*. (2015). Ramki. Java Blog. Recuperado de: <http://www.ramkitech.com/2011/11/jsf-jpa-jasperreports-ireport.html>. (19/09/2015)
- JSF 2 Tutorial Series*. (2015). Customized J2EE Training. Recuperado de: <http://www.coreservlets.com/JSF-Tutorial/jsf2/>. (03/02/2015)
- JSF Introduction*. (2015). JSF Tutorial. Recuperado de: <http://www.java2s.com/Tutorials/Java/JSF/index.htm>. (12/06/2015)
- Kiet, T. PhD. (2013). *Introduction to Web Services with Java*. EEUU: Bookboon, ISBN 978-87-403-0509-8.
- Las reglas de SCRUM*. (2015). SCRUM Manager. Recuperado de: www.SCRUMmanager.net/files/SCRUM_I.pdf. (14/02/2015)
- Marqués, M. (2011). *Bases de Datos*. Castelló España. ISBN: 978-84-693-0146-3.
- Martínez, R. (2013). *PostgreSQL-es*. Recuperado de: http://www.postgreSQL.org.es/sobre_postgreSQL. (25/12/2015)
- Murphey, R. (2013). *Fundamentos de jQuery*. (1º Edición). United States: Creative Commons.
- Noori, H. (2012). *Administración de operaciones y producción: Calidad total y respuesta sensible rápida*. (1º Edición). McGrawHill.
- Oracle. (2010). Recuperado de: <http://www.oracle.com/technetwork/java/javase/jdbc/>. (06/03/2015)
- Pérez, A. O. (2013). *Laboratorio del Departamento de Informática (LDI)*. Recuperado de: <http://www.lab.inf.uc3m.es/~a0080802/RAI/jpa.html>
- Pérez, J. A. (2010). *Gestión por Procesos*. (4º Edición). Madrid: ESIC.
- PostgreSQL. (2011). *PostgreSQL*. Autoedición 1. United States: Autoedición.
- PostgreSQL*. (2015). Hispalinux. Recuperado de: <http://es.tldp.org/PostgreSQL-es/web/navegable/todopostgreSQL/arch-pg.html#AEN16618>. (16/07/2015)

- PostgreSQL. (2015). *Feature Matrix*. Recuperado de: <http://www.postgresql.org/about/featurematrix>. (29/08/2015)
- Programación de redes de telecomunicaciones*. (2015). Programación de Sockets. Recuperado de: <http://programacionderedesdetelecomunicaciones.wikispaces.com/Programacion+d+e+Sockets>. (15/09/2015)
- Quartz*. (2015). Terracotta ORG. Recuperado de: <http://quartz-scheduler.org/documentation/2.2.1/pdf/index>. (14/11/2015)
- Repositorio Académico de la Universidad de Chile. (2015). Recuperado de: <http://www.tesis.uchile.cl/handle/2250/110854>. (28/10/2015)
- Sanchez, A. (2013). *Gestión de Bases de Datos*. (2º Edición). United States: Creative Commons.
- SCRUM & Agile Training*. (2008). Mountain Goat Software. Recuperado de: www.mountaingoatsoftware.com. (05/10/2015)
- The Eclipse Foundation. (2015). *Eclipse*. Recuperado de: <http://www.eclipse.org/home/index.php>. (17/03/2015)
- Título de experto universitario en desarrollo de aplicaciones y servicios con java EE*. (2014). Departamento de Ciencias de la Computación. Recuperado de: <http://www.jtech.ua.es/j2ee/publico/jsf-2012-13/sesion03-apuntes.html>. (03/12/2014)
- Understanding the Atmosphere Framework*. (2015). Async-IO.org. Recuperado de: <http://async-io.org/tutorial.html>. (19/12/2014)
- Vogel, A. (2015). *Eclipse IDE – Tutorial*. (Sección 5). Recuperado de: <http://www.vogella.com/tutorials/Eclipse/article.html>. (01/05/2015)
- White, S., Fisher, M., Cattell, G. (2014). *JDBC API Tutorial and Reference Universal Data Access for the Java Platform*. Hamilton: Addison.

ANEXOS



En esta sección se anexa los manuales técnico y de usuario del sistema. (CD)

Anexo 1

MANUAL TÉCNICO
“SISTEMA DE GESTIÓN DE PROCESOS DE
PRODUCCIÓN PARA LA EMPRESA DIKAPSA



Carlos Andrés Burgos V.

Diciembre 2015

El Anexo 1 se encuentra en el CD entregable.

Anexo 2

MANUAL DE USUARIO
SISTEMA DE GESTIÓN DE PROCESOS DE
PRODUCCIÓN PARA LA EMPRESA DIKAPSA



Carlos Andrés Burgos V.

Diciembre 2015

El Anexo 2 se encuentra en el CD entregable.