

**UNIVERSIDAD MAYOR DE SAN ANDRÉS**  
**FACULTAD DE CIENCIAS PURAS Y NATURALES**  
**CARRERA DE INFORMÁTICA**



**PROYECTO DE GRADO**

**“FRAMEWORK DE AUTOMATIZACIÓN DE PRUEBAS  
FUNCIONALES PARA DISEÑADOR DE PROCESOS DE  
PROCESSMAKER v3  
CASO: COLSER SRL.”**

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA  
MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS

**POSTULANTE: ROBERTO CARLOS GÓNGORA ADUVIRI**  
**TUTOR METODOLÓGICO: Lic. JAVIER HUGO REYES PACHECO**  
**ASESOR: M. Sc. ALDO RAMIRO VALDEZ ALVARADO**

**LA PAZ – BOLIVIA**  
**2016**



**UNIVERSIDAD MAYOR DE SAN ANDRÉS  
FACULTAD DE CIENCIAS PURAS Y NATURALES  
CARRERA DE INFORMÁTICA**



**LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.**

**LICENCIA DE USO**

El usuario está autorizado a:

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

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

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

***DEDICATORIA***

*A mis Padres Raúl y Josefina, quienes nunca  
perdieron la fé en mi, con mucho cariño  
les dedico este logro.*

*De manera especial a mi esposa Mary  
y a mi hijo Fabricio quienes son la  
razón de mi vida.*

## **AGRADECIMIENTO**

Primeramente, agradezco a la Universidad Mayor de San Andrés, por haberme acogido todos estos años y ser parte de la Carrera de Informática, así como también a todos los docentes que dedicaron tiempo y esfuerzo en mi formación profesional.

Agradezco también a mi Asesor M. Sc. Aldo Ramiro Valdez Alvarado, por haberme dado la oportunidad de recurrir a su capacidad y conocimiento, sus consejos y todo el apoyo brindado para lograr este objetivo.

Agradezco también a mi Tutor metodológico Lic. Javier Hugo Reyes Pacheco, por guiarme en esta última parte de mi vida universitaria y su ayuda incondicional.

Mi agradecimiento también va dirigido al Gerente de Ingeniería de Processmaker Ing. Mauricio Veliz, por haberme aceptado y apoyado en la realización de este proyecto en esta prestigiosa empresa.

De manera especial a todo el equipo de Quality Assurance de Processmaker, quienes me guiaron y apoyaron para lograr este objetivo.

A mis hermanos que me apoyaron constantemente a lograr esta meta, muchas gracias por su ayuda y ejemplo que me dieron en mi vida estudiantil.

## **RESUMEN**

La notación para el modelado de procesos de negocio es una forma estándar y gráfica de modelar procesos de negocios. La meta fundamental de BPMN es proporcionar una notación estándar que sea fácilmente comprensible por todos los interesados.

Existen herramientas llamadas BPMS, que construyen aplicaciones BPM (gestión de procesos de negocio), siguiendo la notación estándar BPMN. Processmaker es una solución para automatizar e implementar la gestión de procesos de negocio.

Processmaker cuenta con tres módulos importantes: Diseño de procesos, Ejecución de procesos y Administración de la herramienta. En su nueva versión 3.x se tienen muchos cambios importantes en el primer módulo que necesita aplicación de casos de prueba para garantizar su correcto funcionamiento. Para esto se necesita la intervención de gente especializada en el manejo de la herramienta, además de tecnologías adicionales que ayuden al equipo de Quality Assurance a cumplir con el objetivo de liberar un producto que cumpla con las expectativas de los usuarios.

Para que el equipo de Quality Assurance cumpla con sus objetivos se necesita aplicar pruebas funcionales bien definidas y estructuradas y la mejor manera de aplicar un set de pruebas repetitivas es la Automatización.

La Automatización de Pruebas funcionales aplicada sobre el diseñador de procesos de Processmaker es una herramienta que hace uso de un lenguaje de programación orientado a objetos (JAVA) trabajando juntamente a MAVEN y utilizando un sistema de integración continua que nos permite programar las pruebas a realizar para verificar el estado del módulo en cuestión el cual está en constante desarrollo.

## **SUMMARY**

Business Process Model and Notation is a standard and graphical way of modeling business processes. The main goal of BPMN is to provide a standard notation that is easily understood by all stakeholders.

There are tools called BPMS (Business Process Management Suite), which build BPM (business process management) applications, following BPMN standard notation. Processmaker is a solution for automating and implementing business process management.

Processmaker has three important modules: Process Design, Process Execution and Tool Management. In its new version 3.x there are many important changes in the first module that needs application of test cases to guarantee its correct operation. This requires the intervention of people specialized in the management of the tool, as well as additional technologies that help the Quality Assurance team to meet the goal of releasing a product that meets the expectations of users.

For the Quality Assurance team to meet its objectives, it is necessary to apply well-defined and structured functional tests and the best way to apply a set of repetitive tests is Automation.

The Functional Testing Automation applied to the Processmaker process designer is a tool that makes use of an object oriented programming language (JAVA) working together with MAVEN and using a continuous integration system that allows us to program the tests to be performed for Verify the status of the module in question which is in constant development.

## ÍNDICE

### CAPÍTULO 1 MARCO INTRODUCTORIO

1.1	INTRODUCCIÓN.....	1
1.2	ANTECEDENTES .....	2
1.2.1	ANTECEDENTES INSTITUCIONALES .....	2
1.2.2	PROYECTOS SIMILARES .....	4
1.3	PLANTEAMIENTO DEL PROBLEMA .....	4
1.3.1	PROBLEMA CENTRAL .....	4
1.3.2	PROBLEMAS SECUNDARIOS .....	5
1.4	DEFINICIÓN DE OBJETIVOS.....	6
1.4.1	OBJETIVO GENERAL.....	6
1.4.2	OBJETIVOS ESPECÍFICOS .....	6
1.5	JUSTIFICACIÓN DEL PROYECTO.....	7
1.5.1	JUSTIFICACIÓN SOCIAL.....	7
1.5.2	JUSTIFICACIÓN ECONÓMICA .....	7
1.5.3	JUSTIFICACIÓN TÉCNICA.....	8
1.6	ALCANCES Y LÍMITES .....	9
1.6.1	ALCANCES .....	9
1.6.2	LÍMITES.....	9
1.7	APORTES .....	10
1.8	METODOLOGÍA.....	10
1.8.1	SCRUM .....	10
1.8.2	HISTORIAS DE USUARIO .....	10

### CAPÍTULO 2 MARCO TEÓRICO

2.1	INGENIERÍA DE SOFTWARE .....	12
2.2	METODOLOGÍAS DE DESARROLLO.....	13
2.3	METODOLOGÍAS ÁGILES .....	15
2.4	METODOLOGÍA ÁGIL SCRUM .....	15
2.4.1	INTRODUCCIÓN.....	15

2.4.2	COMPONENTES DE SCRUM.....	18
2.4.2.1	LAS REUNIONES .....	19
2.4.2.2	LOS ROLES .....	19
2.4.3	LOS ELEMENTOS DE SCRUM.....	21
2.4.3.1	PRODUCT BACKLOG .....	21
2.4.3.2	SPRINT BACKLOG .....	24
2.4.3.3	INCREMENTO .....	24
2.4.4	FASES DE SCRUM .....	24
2.4.4.1	PRE-GAME.....	25
2.4.4.2	GAME.....	27
2.4.4.3	POST-GAME.....	30
2.4.5	DESARROLLO DE LAS FASES DE UN PROYECTO EN SCRUM.....	30
2.4.5.1	PREPARACIÓN DEL PROYECTO.....	30
2.4.5.2	PLANIFICAR UN SPRINT .....	31
2.4.5.3	DESARROLLO DEL SPRINT .....	34
2.5	ROADMAP .....	35
2.5.1	COMBINACIÓN DE ROADMAP Y BACKLOG DE SCRUM.....	37
2.5.2	HISTORIAS DE USUARIO .....	39
2.6	ASEGURAMIENTO DE LA CALIDAD DE SOFTWARE(QA).....	40
2.6.1	PRUEBAS FUNCIONALES DE SOFTWARE.....	41
2.6.2	PRUEBAS DE REGRESIÓN.....	43
2.6.3	CASOS DE PRUEBA .....	44
2.7	PATRÓN DE DISEÑO: OBJETOS DE PÁGINA (PAGE OBJECT PATTERN)	45
2.8	HERRAMIENTAS DE DESARROLLO .....	48
2.8.1	JAVA .....	48
2.8.2	MAVEN.....	49
2.8.3	SELENIUM .....	50
2.8.3.1	SELENIUM WEBDRIVER.....	51
2.8.3.2	SELENIUM GRID.....	52
2.9.4	SISTEMA DE INTEGRACIÓN CONTÍNUA: JENKINS .....	53



## CAPÍTULO 3 MARCO APLICATIVO

3.1	INTRODUCCIÓN.....	54
3.2	PRE – GAME.....	54
3.2.1	ROLES DEL PROYECTO.....	55
3.2.2	ANÁLISIS DE REQUERIMIENTOS.....	56
3.2.3	ELECCIÓN DEL ENTORNO DE DESARROLLO.....	59
3.2.4	ESPECIFICACIÓN DE LAS HISTORIAS DE USUARIO.....	61
3.2.4.1	HISTORIA DE USUARIO: ESTRUCTURA DE ARCHIVOS Y CLASES ....	62
3.2.4.2	HISTORIA DE USUARIO: CREACIÓN DE CLASES BASE Y ABSTRACTAS.....	63
3.2.4.3	HISTORIA DE USUARIO: CREACIÓN DE CLASES “WRAPPER” EXTJS66	
3.2.4.4	HISTORIA DE USUARIO: CREACIÓN DE CLASES “WRAPPER” PMUI.69	
3.2.4.5	HISTORIA DE USUARIO: CONSTRUCCIÓN DE PAGE-OBJECTS PARA DISEÑADOR DE PROCESOS.....	77
3.2.4.6	HISTORIA DE USUARIO: ESTRUCTURA DE TESTING SCRIPTS.....	86
3.2.4.7	HISTORIA DE USUARIO: ARMADO DE INFRAESTRUCTURA.....	91
3.2.4.8	HISTORIA DE USUARIO: IMPLEMENTACIÓN Y ESTABILIZACIÓN DEL FRAMEWORK.....	93
3.2.5	DEFINICIÓN DE PRODUCT BACKLOG (PILA DE PRODUCTOS).....	95
3.2.6	DEFINICIÓN DEL CRONOGRAMA DE TRABAJO.....	97
3.2.7	ANÁLISIS DE RIESGOS.....	98
3.3	GAME.....	99
3.3.1	SPRINT 1: DEFINICIÓN DE ESTRUCTURA DE ARCHIVOS Y CREACIÓN DE CLASES.....	99
3.3.1.1	PLANIFICACIÓN DEL SPRINT.....	99
3.3.1.2	DESARROLLO DEL SPRINT.....	100
3.3.1.3	REVISIÓN DEL SPRINT.....	101
3.3.2	SPRINT 2: CREACIÓN DE CLASES WRAPPER EXTJS.....	103
3.3.2.1	PLANIFICACIÓN DEL SPRINT.....	103
3.3.2.2	DESARROLLO DEL SPRINT.....	104
3.3.2.3	REVISIÓN DEL SPRINT.....	104

3.3.3	SPRINT 3: CREACIÓN DE CLASES WRAPPER PMUI.....	108
3.3.3.1	PLANIFICACIÓN DEL SPRINT.....	108
3.3.3.2	DESARROLLO DEL SPRINT.....	110
3.3.3.3	REVISIÓN DEL SPRINT.....	110
3.3.4	SPRINT 4: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE I).....	114
3.3.4.1	PLANIFICACIÓN DEL SPRINT.....	114
3.3.4.2	DESARROLLO DEL SPRINT.....	115
3.3.4.3	REVISIÓN DEL SPRINT.....	116
3.3.5	SPRINT 5: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE II).....	117
3.3.5.1	PLANIFICACIÓN DEL SPRINT.....	117
3.3.5.2	DESARROLLO DEL SPRINT.....	118
3.3.5.3	REVISIÓN DEL SPRINT.....	119
3.3.6	SPRINT 6: DISEÑO DE TESTING SCRIPTS.....	121
3.3.6.1	PLANIFICACIÓN DEL SPRINT.....	121
3.3.6.2	DESARROLLO DEL SPRINT.....	122
3.3.6.3	REVISIÓN DEL SPRINT.....	122
3.3.7	SPRINT 7: ARMADO DE INFRAESTRUCTURA Y ESTABILIZACIÓN DEL FRAMEWORK.....	124
3.3.7.1	PLANIFICACIÓN DEL SPRINT.....	124
3.3.7.2	DESARROLLO DEL SPRINT.....	125
3.3.7.3	REVISIÓN DEL SPRINT.....	125
3.3.8	DESARROLLO DE LOS SPRINTS.....	125
3.3.8.1	DISEÑO DE DIAGRAMA DE CLASES.....	125
3.3.8.2	CREACIÓN DE CLASES WRAPPER.....	128
3.3.8.2.1	CLASES WRAPPER EXTJS.....	129
3.3.8.2.2	CLASES WRAPPER PMUI.....	133
3.3.8.3	USO DE PATRÓN DE DISEÑO PAGE-OBJECT.....	137
3.3.8.3.1	ESTRUCTURA DE UNA PÁGINA.....	137
3.3.8.4	SCRIPTS DE TESTEO.....	147
3.3.8.4.1	ESTRUCTURA DE UN SCRIPT DE TESTEO.....	147

3.3.8.5	SELENIUM GRID.....	150
3.3.8.5.1	SELENIUM HUB .....	151
3.3.8.5.2	NODOS DE TESTEO.....	152
3.3.8.6	SERVIDOR DE INTEGRACIÓN CONTINUA .....	153
3.3.8.6.1	INSTALACION DE JENKINS .....	153
3.4	POST – GAME.....	155
<b>CAPÍTULO 4 CALIDAD DE SOFTWARE</b>		
4.1	INTRODUCCIÓN.....	157
4.2	ISO 9126.....	157
4.2.1	FUNCIONALIDAD .....	157
4.2.2	FIABILIDAD.....	161
4.2.3	USABILIDAD .....	164
4.2.4	MANTENIBILIDAD.....	166
4.2.5	PORTABILIDAD .....	168
4.3	CALIDAD GLOBAL .....	169
<b>CAPÍTULO 5 ANÁLISIS DE COSTO BENEFICIO</b>		
5.1	INTRODUCCIÓN.....	170
5.1	COCOMO II.....	170
5.1.1	COSTOS DEL SOFTWARE DESARROLLADO.....	170
5.1.2	COSTOS DE LA IMPLEMENTACIÓN DEL SISTEMA.....	173
<b>CAPÍTULO 6 CONCLUSIONES Y RECOMENDACIONES</b>		
6.1	CONCLUSIONES.....	176
6.2	RECOMENDACIONES .....	177

## ÍNDICE DE FIGURAS

Figura 2.1 Elementos Básicos de una metodología.....	13
Figura 2.2 Fases del ciclo de desarrollo de Scrum .....	18
Figura 2.3 Reuniones Diarias - Daily Scrum Meeting .....	18
Figura 2.4 Elementos de Scrum.....	21
Figura 2.5 Historia de usuario .....	23
Figura 2.6 Scrum Methodologies .....	25
Figura 2.7 Sprint Backlog.....	32
Figura 2.8 Técnica "Planning Poker" .....	33
Figura 2.9 Reuniones de Sprint .....	35
Figura 2.10 El camino de ruta de un proyecto.....	36
Figura 2.11 Combinación de Roadmap y Backlog de Scrum .....	38
Figura 2.12 Historia de Usuario .....	40
Figura 2.13 Pruebas de Caja Negra .....	42
Figura 2.14 Ejemplo de Formulario básico .....	47
Figura 3.1 Fases del Marco Aplicativo.....	54
Figura 3.2 Roadmap Proyecto Framework de automatización de pruebas Funcionales .....	58
Figura 3.3 Matriz Roadmap para el desarrollo de software .....	58
Figura 3.4 Diagrama de clases (Page-objects).....	126
Figura 3.5 Diagrama de clases (Testing Cases).....	127
Figura 3.6 Clase Envoltorio para un Entero .....	128
Figura 3.7 Uso de la clase Envoltorio.....	128
Figura 3.8 Elementos de creación de un proceso BPMN (Processmaker) .....	130
Figura 3.9 TextBox y su respectivo elemento HTML.....	131
Figura 3.10 ExtJSTextBox wrapper .....	131
Figura 3.11 TextArea y su respectivo elemento HTML.....	132
Figura 3.12 ExtJSTextArea wrapper .....	132
Figura 3.13 Button y su respectivo elemento HTML.....	132
Figura 3.14 ExtJSButton wrapper .....	133
Figura 3.15 Elementos de configuración de un proceso BPMN (Processmaker) .....	134
Figura 3.16 RadioField y su respectivo elemento HTML.....	135
Figura 3.17 PmuiUIRadioButtonGroupField wrapper .....	135
Figura 3.18 TabPanel y su respectivo elemento HTML.....	136
Figura 3.19 PmuiUiTabPanel wrapper .....	136
Figura 3.20 Button pmUI y su respectivo elemento HTML.....	136
Figura 3.21 Button pmUI wrapper .....	137

Figura 3.22 Estructura de una página bajo el patrón Page Object Pattern .....	138
Figura 3.23 Página de Creación de Variable .....	139
Figura 3.24 Creación de Variable bajo el patrón Page Object Pattern .....	139
Figura 3.25 Primera sección de Creación de Variables bajo el patrón Page Object Pattern .....	140
Figura 3.26 Segunda sección de Creación de Variables bajo el patrón Page Object Pattern .....	140
Figura 3.27 Campo Variable Name .....	141
Figura 3.28 Asignación de Variable Name a su respectivo Objeto .....	141
Figura 3.29 Campo Variable Type .....	142
Figura 3.30 Asignación de Variable Type a su respectivo Objeto .....	142
Figura 3.31 Campo Database Connection .....	142
Figura 3.32 Asignación de Database Connection a su respectivo Objeto .....	143
Figura 3.33 Campo SQL .....	143
Figura 3.34 Asignación de SQL field a su respectivo Objeto .....	143
Figura 3.35 Campo Define Accepted variable values .....	144
Figura 3.36 Asignación de Accepted Variable Values a su respectivo Objeto .....	144
Figura 3.37 Cancel Button .....	144
Figura 3.38 Asignación de Cancel Button a su respectivo Objeto .....	145
Figura 3.39 Save Button .....	145
Figura 3.40 Asignación de Save Button a su respectivo Objeto .....	145
Figura 3.41 Método para establecer valores .....	146
Figura 3.42 Métodos para recuperar valores .....	146
Figura 3.43 Métodos para ejecutar acciones sobre los botones .....	147
Figura 3.44 Estructura de un Script de Testeo .....	148
Figura 3.45 Estructura de un Assert .....	149
Figura 3.46 Script de testeo para Creación de Variables .....	149
Figura 3.47 Comandos para la creación de variables .....	150
Figura 3.48 Asserts para la verificación de creación de variables .....	150
Figura 3.49 Configuración simple para ejecución de Pruebas .....	151
Figura 3.50 Envío de Scripts de testeo a un Hub .....	152
Figura 3.51 Comando para iniciar el Hub en el servidor .....	152
Figura 3.52 Comando para iniciar un nodo y registrar en el hub .....	152
Figura 3.53 Envío de Scripts de testeo de un Hub a los nodos .....	153
Figura 3.54 Jenkins y plugins necesarios para automatización .....	154
Figura 3.55 Componentes del Framework de Automatización .....	154
Figura 3.56 Compilación de tests para iniciar el testeo .....	155

Figura 3.57 Salida de consola.....	155
Figura 3.58 Resultados de pruebas .....	156

## ÍNDICE DE TABLAS

Tabla 2.1 Concepción inicial del producto que tienen los accionistas o interesados .....	27
Tabla 2.2 Fase de desarrollo .....	30
Tabla 3.1 Definición de Roles .....	55
Tabla 3.2 Historia de usuario 001 – Estructura de archivos y clases .....	62
Tabla 3.3 Tarea 1 de historia de usuario 001 – Crear archivo de configuración pom.xml ..	63
Tabla 3.4 Tarea 2 de historia de usuario 001 – Crear archivo de configuración .conf.....	63
Tabla 3.5 Historia de usuario 002 – Creación de clases base y abstractas .....	64
Tabla 3.6 Tarea 1 de historia de usuario 002 – Crear clase de invocación de Driver .....	64
Tabla 3.7 Tarea 2 de historia de usuario 002 – Crear clase de configuración de Driver.....	65
Tabla 3.8 Tarea 3 de historia de usuario 002 – Crear clase para eventos de espera “WaitTool” .....	65
Tabla 3.9 Tarea 4 de historia de usuario 002 – Crear clase abstracta de página “Page” .....	65
Tabla 3.10 Historia de usuario 003 - Creación de clases “Wrapper” ExtJS.....	66
Tabla 3.11 Tarea 1 de historia de usuario 003 – Crear clase “wrapper” ExtjsTextBox.....	67
Tabla 3.12 Tarea 2 de historia de usuario 003 – Crear clase “wrapper” ExtjsTextArea.....	67
Tabla 3.13 Tarea 3 de historia de usuario 003 – Crear clase “wrapper” ExtjsDropDown... ..	68
Tabla 3.14 Tarea 4 de historia de usuario 003 – Crear clase “wrapper” ExtjsButton .....	68
Tabla 3.15 Tarea 5 de historia de usuario 003 – Crear clase “wrapper” ExtjsGridPanel.....	69
Tabla 3.16 Tarea 6 de historia de usuario 003 – Crear clase “wrapper” ExtjsMessageWindow .....	69
Tabla 3.17 Historia de usuario 004 - Creación de clases “Wrapper” pmUI.....	70
Tabla 3.18 Tarea 1 de historia de usuario 004 – Crear clase “wrapper” TextBoxField.....	71
Tabla 3.19 Tarea 2 de historia de usuario 004 – Crear clase “wrapper” PasswordField.....	71
Tabla 3.20 Tarea 3 de historia de usuario 004 – Crear clase “wrapper” TextAreaField.....	72
Tabla 3.21 Tarea 4 de historia de usuario 004 – Crear clase “wrapper” DropDownField... ..	72
Tabla 3.22 Tarea 5 de historia de usuario 004 – Crear clase “wrapper” CheckGroupField ..	73
Tabla 3.23 Tarea 6 de historia de usuario 004 – Crear clase “wrapper” RadioField .....	73
Tabla 3.24 Tarea 7 de historia de usuario 004 – Crear clase “wrapper” DateTimeField.....	74
Tabla 3.25 Tarea 8 de historia de usuario 004 – Crear clase “wrapper” ContextMenu .....	74
Tabla 3.26 Tarea 9 de historia de usuario 004 – Crear clase “wrapper” Button .....	75
Tabla 3.27 Tarea 10 de historia de usuario 004 – Crear clase “wrapper” GridPanel.....	75
Tabla 3.28 Tarea 11 de historia de usuario 004 – Crear clase “wrapper” TabPanel .....	76
Tabla 3.29 Tarea 12 de historia de usuario 004 – Crear clase “wrapper” TreePanel.....	76
Tabla 3.30 Tarea 13 de historia de usuario 004 – Crear clase “wrapper” MessageWindow	77
Tabla 3.31 Tarea 14 de historia de usuario 004 – Crear clase “wrapper” Window .....	77

Tabla 3.32 Historia de usuario 005 – Construcción de Page-objects para Diseñador de Procesos .....	78
Tabla 3.33 Tarea 1 de historia de usuario 005 – Crear clase ProcessDesignerBPMN .....	79
Tabla 3.34 Tarea 2 de historia de usuario 005 – Crear paquete VariableBPMN .....	80
Tabla 3.35 Tarea 3 de historia de usuario 005 – Crear paquete MessageTypeBPMN .....	80
Tabla 3.36 Tarea 4 de historia de usuario 005 – Crear paquete DynaformBPMN .....	81
Tabla 3.37 Tarea 5 de historia de usuario 005 – Crear paquete InputDocBPMN .....	81
Tabla 3.38 Tarea 6 de historia de usuario 005 – Crear paquete OutputDocBPMN .....	82
Tabla 3.39 Tarea 7 de historia de usuario 005 – Crear paquete TriggerBPMN .....	83
Tabla 3.40 Tarea 8 de historia de usuario 005 – Crear paquete ReportTableBPMN .....	83
Tabla 3.41 Tarea 9 de historia de usuario 005 – Crear paquete DatabaseConnectionBPMN .....	84
Tabla 3.42 Tarea 10 de historia de usuario 005 – Crear paquete TemplateBPMN .....	84
Tabla 3.43 Tarea 11 de historia de usuario 005 – Crear paquete PublicFileBPMN .....	85
Tabla 3.44 Tarea 12 de historia de usuario 005 – Crear paquete PermissionBPMN .....	86
Tabla 3.45 Tarea 13 de historia de usuario 005 – Crear paquete SupervisorBPMN .....	86
Tabla 3.46 Historia de usuario 006 – Diseño y estructura de Testing Scripts .....	87
Tabla 3.47 Tarea 1 de historia de usuario 006 – Crear clase BPMNDrawElements .....	88
Tabla 3.48 Tarea 2 de historia de usuario 006 – Crear clase BPMNProcessProperties .....	88
Tabla 3.49 Tarea 3 de historia de usuario 006 – Crear clase AssignmentRules .....	89
Tabla 3.50 Tarea 4 de historia de usuario 006 – Crear clase BPMNDynaformManagement .....	89
Tabla 3.51 Tarea 5 de historia de usuario 006 – Crear clase InputDocumentManagement .....	90
Tabla 3.52 Tarea 6 de historia de usuario 006 – Crear clase TriggersManagement .....	90
Tabla 3.53 Tarea 7 de historia de usuario 006 – Crear clase DatabaseConnectionManagement .....	91
Tabla 3.54 Historia de usuario 007 – Armado de Infraestructura .....	92
Tabla 3.55 Tarea 1 de historia de usuario 007 – Instalar Servidor de Integración Continua .....	92
Tabla 3.56 Tarea 2 de historia de usuario 007 – Instalar Selenium HUB .....	92
Tabla 3.57 Tarea 3 de historia de usuario 007 – Instalar Nodos de Testeo .....	93
Tabla 3.58 Historia de usuario 008 – Implementación y Estabilización del Framework .....	94
Tabla 3.59 Tarea 1 de historia de usuario 008 – Modificar archivos de configuración del Framework .....	94
Tabla 3.60 Tarea 2 de historia de usuario 008 – Modificar archivos base y herramientas de ayuda .....	95
Tabla 3.61 Product Backlog – Requerimientos del Framework de Automatización de Pruebas Funcionales .....	97



Tabla 3.62 Análisis de Riesgo .....	99
Tabla 3.63 Sprint 1 – Planificación .....	100
Tabla 3.64 Sprint 1 – Prueba: Ejecución del archivo pom.xml.....	102
Tabla 3.65 Sprint 1 – Prueba: Invocación de un navegador mediante el driver.....	102
Tabla 3.66 Sprint 2 – Planificación .....	103
Tabla 3.67 Sprint 2 – Prueba: Creación y/o instanciación de un Wrapper ExtJS de tipo TextBox, TextArea, DropDown .....	105
Tabla 3.68 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo Button .....	106
Tabla 3.69 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo GridPanel .....	107
Tabla 3.70 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo MessageWindow .....	108
Tabla 3.71 Sprint 3 – Planificación .....	110
Tabla 3.72 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI para los campos del formulario.....	111
Tabla 3.73 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo ContextMenu .....	112
Tabla 3.74 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo Button .....	113
Tabla 3.75 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo Panel .....	114
Tabla 3.76 Sprint 4 – Planificación .....	115
Tabla 3.77 Sprint 4 – Prueba: Creación y/o instanciación de páginas de Sprint 4.....	117
Tabla 3.78 Sprint 5 – Planificación .....	118
Tabla 3.79 Sprint 5 – Prueba: Creación y/o instanciación de páginas de Sprint 5.....	121
Tabla 3.80 Sprint 6 – Planificación .....	122
Tabla 3.81 Sprint 6 – Prueba: Creación de Script de Testeo.....	123
Tabla 3.82 Sprint 7 – Planificación .....	124
Tabla 4.1 Factor de Ponderación para la funcionalidad .....	158
Tabla 4.2 Pesos de los puntos función.....	159
Tabla 4.3 Factores de evaluación .....	160
Tabla 4.4 Escala de Punto Función .....	160
Tabla 4.5 Mediciones de fiabilidad .....	162
Tabla 4.6 Preguntas de facilidad de uso .....	164
Tabla 4.7 Escala de Evaluación para la usabilidad.....	165
Tabla 4.8 Resultados de la encuesta de usabilidad.....	165

Tabla 4.9 Valores IMS .....	167
Tabla 4.10 Calidad global.....	169
Tabla 5.1 Factor LCD/PF del lenguaje de programación.....	171
Tabla 5.2 Tipos de proyectos de Software .....	172
Tabla 5.3 Estimación de rendimiento .....	173
Tabla 5.4 Análisis de costo.....	174

# CAPÍTULO 1 MARCO INTRODUCTORIO

## 1.1 INTRODUCCIÓN

Las grandes empresas de desarrollo de software generan muchos proyectos en base a los requerimientos del cliente y a la tendencia de tecnologías actuales, la necesidad de ser ágiles impulsa a sacar productos más rápido al mercado y adelantarse a la competencia, mejorando para ello la calidad de su proceso, producto y equipos de software. Una de las claves para agilizar este proceso es detectar errores antes, en puntos del desarrollo que cueste menos solucionarlos y así desarrollar con más seguridad. En éste contexto las pruebas funcionales forman parte importante del desarrollo de software cuyo objetivo es el control de calidad.

Se denominan pruebas funcionales, a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados, es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas y al dar conformidad sobre esta el paso siguiente es el pase a producción.

A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, ya que los testers o analistas de pruebas, no enfocan su atención a como se generan las respuestas del sistema, básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas.

Las pruebas funcionales en la mayoría de los casos son realizadas manualmente por un analista de pruebas, también es posible automatizar este tipo de pruebas utilizando herramientas de testeo las cuales permiten generar scripts conforme se genere interacciones con el aplicativo a probar.

Este es el motivo por el que la automatización de pruebas es una buena opción, útil para crear un entorno de desarrollo satisfactorio. Las pruebas funcionales garantizan que la aplicación hace lo que se supone que debe hacer. Incluso cuando el código interno de la aplicación cambia constantemente, las pruebas automatizadas permiten garantizar que los cambios no introducen incompatibilidades en el funcionamiento de la aplicación. Este tipo de pruebas obligan a los programadores a crear pruebas en un formato estandarizado y muy rígido que pueda ser procesado por un Framework de pruebas. En ocasiones, las pruebas automatizadas pueden reemplazar la documentación técnica de la aplicación, ya que ilustran de forma clara el funcionamiento de la aplicación. Un buen conjunto de pruebas muestra la salida que produce la aplicación para una serie de entradas de prueba, por lo que es suficiente para entender el propósito de cada método.

ProcessMaker Inc. es desarrollador de ProcessMaker Workflow y BPM Suite. ProcessMaker es software que permite que organizaciones simplifiquen su workflow a través de la captura y automatización de procesos de negocio. ProcessMaker tiene una red de socios globales que pueden implementar y dar soporte de software de ProcessMaker para organizaciones de todos los tamaños.

La implementación de un Framework de Automatización de pruebas funcionales ayudaría a la empresa y a la herramienta en general a detectar errores durante su desarrollo y por consiguiente a crear un producto más confiable.

## **1.2 ANTECEDENTES**

### **1.2.1 ANTECEDENTES INSTITUCIONALES**

#### **COLSER SRL.**

Fundado en octubre de 2000, teniendo oficinas en:

- Oficina Central: Brooklyn, Nueva York – USA

- Oficinas Regionales: Bogotá (Colombia), Lima (Perú), La Paz (Bolivia)

- Red de Socios: Socios estratégicos en 15 países en 5 continentes

Ediciones de Producto Empresariales y Comunitarias. Más de 100 clientes comerciales incluyendo varias empresas, más de 140000 descargas, soporte para 15 lenguajes, finalista al premio SourceForge.NET community Choice, mejor proyecto comercial de software libre y mejor diseño visual.

**Política.** En COLSER SRL. Bolivia. Desarrollamos tecnología y brindamos servicios relacionados con ‘Comercial Open Source’ para Business Process Management – workflow, para nuestros clientes y socios estratégicos con el principal compromiso de cumplir con sus requerimientos, y adicionalmente brindar información actualizada a nuestra Comunidad.

- Ofrecemos a nuestros clientes socios estratégicos servicios ajustados a sus necesidades de automatización y mejora en sus procesos de negocios.

- Tenemos el compromiso de mejorar continuamente ProcessMaker y los servicios relacionados conforme al sistema de gestión de calidad.

- Contamos con personal continuamente capacitado, motivado y comprometido con el éxito de la empresa.

**Misión.** Desarrollar y brindar tecnología “Comercia Open Source” para BPM – workflow que permita actualizar y mejorar procesos de negocios a través de un diseño intuitivo, fácil uso e integración y cumplimiento estándares de BPM. Establecer una red de socios estratégicos y distribuidores calificados que extiendan los servicios de COLSER SRL. A clientes en diferentes regiones del mundo.

**Visión.** Ser el proveedor líder de soluciones de BPM – workflow bajo el modelo ‘Comercial Open Source’, para pequeños y medianos negocios, entidades gubernamentales, desarrolladores, consultores, usuarios finales y para integraciones con otros proyectos y productos Open Source.

## 1.2.2 PROYECTOS SIMILARES

Processmaker Inc. utilizó la automatización en su versión 2.x.x, pero ésta solamente contaba con parte del paradigma implementado en algunas funcionalidades sin ningún tipo de reporte e integración con otras herramientas. En la nueva versión, actualmente en constante cambio y desarrollo se pretende implementar un framework de automatización de pruebas funcionales bien definidas orientadas al sector de Diseño de procesos BPMN.

Varias empresas internacionales, vieron la necesidad y la oportunidad de ofrecer servicios de Testing automatizado, esto para brindar un servicio de pruebas de software satisfactorio, una de ellas que vale la pena mencionar:

**GlobeTesting.** GlobeTesting es una compañía tecnológica especializada en pruebas de software y metodologías de ciclo de vida de software (ALM).

Con sede central en Berlín, GlobeTesting opera en Alemania, España y Suiza, garantizando a sus clientes un servicio de calidad dirigido a cualquier perfil de empresa, independiente del mercado en el que opere.

El modelo de distribución de GlobeTesting permite a sus clientes beneficiarse de una solución de pruebas unificada, automatizada y rentable.

## 1.3 PLANTEAMIENTO DEL PROBLEMA

### 1.3.1 PROBLEMA CENTRAL

Processmaker Inc. en la actualidad está en constante desarrollo del producto Processmaker v3 el cual, como todo software de calidad necesita un ciclo muy bien definido para su desarrollo desde la fase inicial hasta la fase final. Las pruebas de software forman parte de este ciclo y se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad del producto. En específico podríamos hablar de pruebas funcionales que van orientadas a probar y verificar las funcionalidades, dicho de

otro modo, son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y, sobre todo, lo que se ha especificado.

Un producto en constante desarrollo tiene cambios frecuentes en su código, un cambio puede “romper” partes del código que antes funcionaba y que parecía que no tenían relación alguna con la modificación que se ha realizado. Incluso puede hacer que errores que ya se hayan solucionado vuelvan a aparecer. Es por eso que se realizan pruebas de regresión las cuales incluyen pruebas funcionales orientadas a comprobar que el código que se ha modificado se comporta como se desea, es decir, que no produzca ningún error(bug) y que el cambio no hay causado otros problemas en otros sitios del código, que funcionaban correctamente la última vez que se probó. La mejor forma para volver a ejecutar pruebas funcionales con las que se detectaron errores que ya se solucionaron es la automatización.

**¿Cómo resolver la recurrente y desorganizada ejecución manual de pruebas, que muchas veces se realiza de forma incompleta, además de que emplea tiempo y recursos humanos para su ejecución?**

### **1.3.2 PROBLEMAS SECUNDARIOS**

- Se detectan errores en el código del diseñador Processmaker en una fase avanzada del ciclo de desarrollo, por consiguiente, se ignoran muchos problemas en la funcionalidad e interfaz.
- No se hace una cobertura de todos los casos de prueba debido a la cantidad de éstos, por lo tanto, no se tiene un producto probado al 100%.
- No se cuenta con un informe de Processmaker según sus características, esto, impide a los dueños del producto, clientes e interesados tomar decisiones y modificar el camino que seguirá esta herramienta.
- Processmaker no tiene un plan de pruebas definido para cada regresión, por lo tanto, no se realizan las mismas pruebas en cada fase de desarrollo del producto.

- Los ambientes que soporta Processmaker no están definidos formalmente, esto impide ofrecer al cliente un informe de compatibilidad del producto.
- No se cuenta con un servicio de integración continua, en consecuencia, no se puede monitorizar los cambios recientes en la herramienta y administrar ejecuciones de pruebas.

## **1.4 DEFINICIÓN DE OBJETIVOS**

### **1.4.1 OBJETIVO GENERAL**

Desarrollar e implementar un Framework de Automatización de pruebas funcionales que realice un trabajo repetitivo y completo usando un plan de pruebas asignadas para el diseñador de Processmaker v3.

### **1.4.2 OBJETIVOS ESPECÍFICOS**

- Implementar un sistema de testeo “crossbrowsing”, que permita garantizar la correcta ejecución de Processmaker en los diferentes navegadores y sus respectivas versiones.
- Crear un Framework de automatización de pruebas extensible y escalable, es decir, con miras a un futuro crecimiento y adaptable a las circunstancias cambiantes.
- Implementar un sistema/servicio de integración continua, el cual permita la ejecución programada del Framework de Automatización de pruebas funcionales después de obtener los últimos cambios en el desarrollo de Processmaker.
- Elaborar un sistema de testeo organizado por características de Processmaker que permita evaluar todas las funcionalidades y brinde información a todos los interesados acerca de la estabilidad de la herramienta.
- Construir un sistema de testeo multiplataforma, es decir, un sistema que pueda ejecutar sus pruebas en diferentes sistemas operativos, servidores web y gestores



base de datos, dando lugar a una definición formal de servicios compatibles con la herramienta Processmaker.

## **1.5 JUSTIFICACIÓN DEL PROYECTO**

### **1.5.1 JUSTIFICACIÓN SOCIAL**

Dado que es un Framework de Automatización de pruebas, se elimina la necesidad de ejecutar las pruebas funcionales de Processmaker de forma manual, es decir, no se requiere el factor humano para verificar y validar que la herramienta funciona de acuerdo a lo esperado por todos los interesados, por lo tanto, ayuda al Departamento de Quality Assurance dentro del área de Ingeniería de la empresa COLSER SRL. a cumplir con su aporte dentro del proceso de desarrollo del producto Processmaker.

El personal que es beneficiado con este proyecto es asignado a otras tareas que ayudan a la organización y de la misma forma se emplea el tiempo a capacitación constante mejorando la calidad de profesionales que cuenta la empresa en cuestión

### **1.5.2 JUSTIFICACIÓN ECONÓMICA**

El presente proyecto reducirá los gastos de la empresa COLSER SRL. en el desarrollo del producto Processmaker al obtener resultados certeros y confiables al momento de realizar las distintas pruebas sobre la herramienta, esto, sin necesidad de contar con servicio de personal adicional. En consecuencia, se puede invertir recursos de la empresa en otras áreas de desarrollo, obteniendo mejores resultados y una optimización de recursos económicos.

La obtención de resultados fiables y oportunos por parte del Framework de Automatización ayudará a la empresa y asociados a la toma de decisiones importantes sobre el producto Processmaker, el camino que tomará y los cambios que se necesitarán, optimizando el tiempo de desarrollo y de la misma forma optimizando el aspecto económico.

Cabe mencionar que el Framework de Automatización de Pruebas Funcionales se desarrollará bajo herramientas de software libre, lo cual no generará costo alguno para llevar a cabo su realización.

### 1.5.3 JUSTIFICACIÓN TÉCNICA

COLSER SRL. al ser una empresa de desarrollo de software y servicios orientados a ‘Comercial Open Source’ para Business Process Management – workflow, cuenta con el hardware y software necesario para el desarrollo e implementación de un Framework de Automatización de Pruebas funcionales, citando: Servidores de aplicaciones, bases de datos, equipos clientes, etc. todo esto es suficiente para la planificación, desarrollo y puesta en marcha del proyecto.

Se hará uso de un lenguaje de programación con licencia pública general GNU como es JAVA y se aplicará el patrón de diseño **Page object Pattern** el cual es un patrón para crear repositorio de objetos de elementos de la interfaz web. Bajo este modelo, para cada página web en la aplicación debe haber una clase correspondiente. Esta clase encuentra los WebElements de la página web y también contiene métodos que realizan operaciones en esos WebElements.

De la misma forma el proyecto se construirá bajo **Selenium WebDriver** el cual es un servidor escrito en Java que acepta comandos al navegador vía HTTP. Esto hace posible escribir pruebas automatizadas para aplicaciones web, en cualquier lenguaje de programación lo que permite una mejor integración de Selenium a entornos de prueba existentes. Selenium WebDriver acepta comandos (enviados en Selenese o vía el API de cliente) y los envía a un navegador. Esto se implementa a través de un controlador del navegador específico para cada navegador que envía los comandos y trae los resultados de regreso.

La mayoría de los controladores de navegador lanzan y acceden a la aplicación de navegador (como Mozilla Firefox o Internet Explorer), pero también hay un controlador

para HtmlUnit que simula un navegador. A diferencia de Selenium 1, donde el servidor Selenium RC era indispensable, en Selenium WebDriver no se requiere de un servidor especial para ejecutar las pruebas, en vez de ello WebDriver inicia una instancia del navegador y lo controla; sin embargo, puede usarse Selenium Grid para ejecutar pruebas en sistemas remotos

## **1.6 ALCANCES Y LÍMITES**

### **1.6.1 ALCANCES**

- El Framework realizará exclusivamente pruebas funcionales
- El alcance de la automatización abarca el área de diseño de procesos BPMN.
- Agrupará un conjunto de pruebas y generará suites para enfocar el testeo en algún sector en específico mediante configuración.
- Realizará un testing completo de todas las funcionalidades del Diseñador de procesos, usando solamente la red intranet de la empresa, vale decir, solamente podrá ejecutarse con los servidores propuestos por Processmaker Inc.
- Realizará un testing de tipo Cross Browsing en los tres principales navegadores.

### **1.6.2 LÍMITES**

- El Framework se limita solamente al Diseñador de procesos de Processmaker.
- No modificará el código del sistema a testear.
- No realizará pruebas Unitarias del código.
- No realizará pruebas de integración
- No realizará pruebas de estrés o performance del sistema web
- El framework se limita a ejecutar constantemente tareas de testeo con la ayuda de una herramienta de integración continua.

## **1.7 APORTES**

La automatización de pruebas funcionales por lo general apunta a sistemas grandes con muchos casos de pruebas, Processmaker Inc. con su producto en constante desarrollo y mejora (Processmaker v3) contará con una herramienta que ayudará no solamente al proceso de desarrollo sino también al equipo de QA (Quality Assurance) a desempeñar un mejor trabajo con óptimos resultados.

## **1.8 METODOLOGÍA**

La Metodología de desarrollo a usar es Scrum y asimismo como instrumento principal y técnica para detallar los requerimientos del usuario final se usarán las Historias de Usuario.

### **1.8.1 SCRUM**

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente en equipo, y obtener el mejor resultado posible de un proyecto.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales. Esta metodología de desarrollo será ampliamente descrita en el Capítulo II.

### **1.8.2 HISTORIAS DE USUARIO**

Una historia de usuario es una representación de un requisito escrito en una o dos frases utilizando el lenguaje común del usuario. Las historias de usuario son utilizadas en

las metodologías de desarrollo ágiles para la especificación de requisitos (acompañadas de las discusiones con los usuarios y las pruebas de validación).

Son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requisitos cambiantes.



## CAPÍTULO 2 MARCO TEÓRICO

### 2.1 INGENIERÍA DE SOFTWARE

La ingeniería de software abarca las técnicas y métodos para desarrollar y mantener software de calidad. La ingeniería de software intenta resolver todo tipo de problemas que se presentan en la actualidad.

Existen varias definiciones de ingeniería de software:

Ian Sommerville (2004), quien define, “la ingeniería de software es una disciplina de la ingeniería que comprende todos los aspectos de la producción de software desde las etapas iniciales de la especificación del sistema, hasta el mantenimiento de este después de que se utiliza”.

La ingeniería de software comprende un conocimiento amplio de teorías, métodos y herramientas que permiten el desarrollo de software eficiente, de calidad y confiable.

Según Roger S. Pressman(2001) la ingeniería de software, está compuesto por tres fases:

- Fase de definición, en esta fase se analiza la viabilidad del software, que permitan que existan base y fundamentos de costo beneficio para el desarrollo del producto. En esta fase una de las características importantes son los requerimientos y su posterior análisis, de esta manera el software sea confiable.
- Fase de desarrollo, en esta fase se estructuran las bases de datos, la funcionalidad de procesos dando solución a los requerimientos analizados en la fase de definición. Otra de las tareas es el diseño de las interfaces y elección del lenguaje de

programación o lenguajes no procedimentales; teniendo en cuenta las diferentes metodologías ágiles a emplearse.

- Fase de mantenimiento, después de finalizar el desarrollo de software, la fase de mantenimiento es donde se presentan cambios, como la prevención y corrección de errores, adaptaciones y mejoras, de acuerdo a los nuevos requerimientos del usuario final.

## 2.2 METODOLOGÍAS DE DESARROLLO

Una metodología se define como una disciplina que indicara el conjunto de métodos y técnicas que pueden ser utilizados en cada fase del ciclo de vida del desarrollo del proyecto.

Estos elementos se muestran en la siguiente figura:



Figura 2.1 Elementos Básicos de una metodología  
Fuente: Metodología de Scrum, 2014

- Fases, en este punto se marcarán las diferentes actividades que hay que realizar por cada fase.
- Métodos, se tiene que identificar el modo en el que se realizará el proceso de desarrollo del producto software.
- Técnicas y herramientas, indicarán como se debería resolver cada tarea y que herramientas se podría utilizar.
- Documentación, se tiene muy claro que documentación se va a entregar durante todas las fases, esta documentación se debe realizar de manera exhaustiva y completa, usando valores de entrada que se van generando, que servirán para recoger resultados y tomar decisiones de las diferentes situaciones planteadas.
- Control y evaluación, este paso también se debe realizar en todo el desarrollo del software. Consiste en comprobar y aceptar/denegar todos los resultados que se vayan obteniendo y poder replantear, si es necesario, una nueva planificación de las tareas asignadas.

Existen varias metodologías con las cuales se pueden realizar software de calidad como:

- Métodos evolutivos, el desarrollo de software mediante metodologías evolutivas, se hace de una versión inicial, que posteriormente va mejorando durante el ciclo de frecuentemente, estos cambios se deben controlar con documentación de las diferentes versiones existentes de forma iterativa. Existen diferentes métodos evolutivos. Construcción de Prototipos, Modelo Espiral, Modelo de desarrollo concurrente.
- Métodos incrementales, estos modelos permiten entregar con frecuencia avances de desarrollo de software. Aplica secuencia lineal produce incrementos de software. En este modelo el primer incremento es un producto esencial.



## 2.3 METODOLOGÍAS ÁGILES

La ingeniería de software ágil combina una filosofía y un conjunto de directrices de desarrollo. La filosofía busca la satisfacción del cliente y la entrega temprana del software incremental; equipo de proyectos pequeños, y con alta motivación.

La ventaja de los métodos ágiles, es la facilidad de realizar cambios que intervienen en el desarrollo. Las metodologías ágiles más conocidos son las siguientes:

- SCRUM.
- Extreme Programming(XP).
- Metodologías Crystal.
- Open Up.
- Proceso Unificado Rational (RUP).
- Melé.

## 2.4 METODOLOGÍA ÁGIL SCRUM

### 2.4.1 INTRODUCCIÓN

En el año 1986 Takeuchi y Nonaka publicaron el artículo “The new Product Development Game” el cual dará a conocer una nueva forma de gestionar proyectos en la que la agilidad, flexibilidad, y la incertidumbre son los elementos principales.

**Nonaka y Takeuchi** se fijaron en empresas tecnológicas que, estando en el mismo entorno en el que se encontraban otras empresas, realizaban productos en menos tiempo, de buena calidad y menos costes.

Observando a empresas como Honda, HP, Canon etc, se dieron cuenta de que el producto no seguía unas fases en las que había un equipo especializado en cada una de

ellas, si no que se partía de unos requisitos muy generales y el producto lo realizaba un equipo multidisciplinario que trabaja desde el comienzo del proyecto hasta el final.

Se comparó esta forma de trabajo en equipo, con la colaboración que hacen los jugadores de Rugby y la utilización de una formación denominada **SCRUM**.

Scrum aparece como una práctica destinada a los productos tecnológicos y será en 1993 cuando realmente **Jeff Sutherland** aplique un modelo de desarrollo de Software en Ease/Corporation.

En 1996, **Jeff Sutherland y Ken Schwaber** presentaron las prácticas que se usaban como proceso formal para el desarrollo del software y que pasarían a incluirse en la lista de Agile Alliance.

Scrum es adecuado para aquellas empresas en las que el desarrollo de los productos se realiza en entornos que se caracterizan por tener:

- **Incertidumbre:** Sobre esta variable se plantea el objetivo que se quiere alcanzar sin proporcionar un plan detallado del producto. Esto genera un reto y da una autonomía que sirve para generar una “**tensión**” adecuada para la motivación de los equipos.
- **Auto-organización:** Los equipos son capaces de organizarse por sí solos, no necesitan roles para la gestión, pero tienen que reunir las siguientes características.
  - o **Autonomía:** Capaces de encontrar soluciones adecuadas
  - o **Auto-superación:** Las soluciones iniciales sufren cambios
  - o **Auto-enriquecimiento:** Enriquecimiento de forma mutua, aportando soluciones que puedan complementarse.
- **Control Moderado:** Se establece un control suficiente para evitar descontroles. Se basa en crear un escenario de “autocontrol entre iguales” para no impedir la creatividad y espontaneidad de los miembros del equipo.

- **Transmisión del conocimiento:** Todo el mundo aprende de todo el mundo. Las personas pasan de unos proyectos a otros y así comparten sus conocimientos a lo largo de la organización. Scrum al ser una metodología de desarrollo ágil tiene como base la idea de creación de ciclos breves para el desarrollo, que comúnmente se llaman **iteraciones** y que en Scrum se llama “**Sprint**”. Para entender el ciclo de desarrollo es necesario conocer las 5 fases que definen el ciclo del desarrollo ágil:
- **Concepto:** Se define de forma general las características del producto y se asigna el equipo que se encargará de su desarrollo.
- **Especulación:** En esta fase se hacen disposiciones con la información obtenida y se establecen los límites que marcan el desarrollo del producto, tales como costes y agendas.

Se construirá el producto a partir de las ideas principales y se comprueban las partes realizadas y su impacto en el entorno. Esta fase se repite en cada iteración y consiste, en rasgos generales, en:

- Desarrollar y revisar los requisitos generales.
  - Mantener la lista de las funcionalidades que se esperan.
  - Plan de entrega. Se establecen las fechas de las versiones, hitos e interacciones y se medirá el esfuerzo realizado en el proyecto.
- **Exploración:** Se incrementa el producto en el que se añaden las funcionalidades de la fase de especulación.
  - **Revisión:** El equipo revisa todo lo que se ha construido y se contrasta con el objetivo deseado.
  - **Cierre:** Se entrega en la fecha acordada una versión del producto deseado. Al tratarse de una revisión, el cierre no indica que se ha finalizado el proyecto, sino que seguirá habiendo cambios, denominados “mantenimiento”, que hará que el producto final se acerque al producto final deseado.



Figura 2.2 Fases del ciclo de desarrollo de Scrum  
Fuente: Metodología de Scrum, 2014

Scrum gestiona estas iteraciones a través de reuniones diarias, uno de los elementos fundamentales de ésta metodología.



Figura 2.3 Reuniones Diarias - Daily Scrum Meeting  
Fuente: Metodología de Scrum, 2014

## 2.4.2 COMPONENTES DE SCRUM

Para entender todo el proceso de desarrollo Scrum, se describirá de forma general las fases y roles.

Scrum se puede dividir de forma general en 3 fases, que podemos entender como **reuniones**. Las cuales forman parte de los artefactos de esta metodología junto con los roles y los elementos que lo forman.

#### 2.4.2.1 LAS REUNIONES

- **Planificar el Backlog:** Se definirá un documento en el que se reflejarán los requisitos del sistema por prioridades.

En esta fase se definirá también la planificación del **Sprint 0**, en la que se decidirá cuáles van a ser los objetivos y el trabajo que hay que realizar para ésta iteración. Se obtendrá además en esta reunión un **Sprint Backlog**, que es la lista de tareas y que es el objetivo más importante del Sprint.

- **Seguimiento del Sprint:** En esta fase se hacen reuniones diarias en las que las tres preguntas principales para evaluar el avance más importante del Sprint.

- ¿Qué trabajo se realizó desde la reunión anterior?
- ¿Qué trabajo se hará hasta una nueva reunión?
- Inconvenientes que han surgido y que hay que solucionar para continuar.

- **Revisión del Sprint:** Cuando se finaliza el Sprint se realizará una revisión del incremento que se ha generado. Se presentarán los resultados finales y una demo o versión, esto ayudara a mejorar el **feedback** con el cliente.

#### 2.4.2.2 LOS ROLES

Los roles se dividen en 2 grupos:

##### a) Los comprometidos

Son los que están comprometidos con el proyecto y el proceso de Scrum.

- **Scrum Master:** Es el encargado de comprobar que el modelo y la metodología funciona. Eliminará todos los inconvenientes que hagan que el proceso no fluya e interactuara con el cliente y con los gestores.
- **Product Owner:** Es la persona que toma las decisiones, y es la que realmente conoce el negocio del cliente y su visión del producto. Se encarga de escribir las ideas del cliente, las ordena por prioridad y las coloca en el Product Backlog.
- **Scrum Team:** Suele ser un equipo pequeño de unas 5-9 personas y tienen autoridad para organizar y tomar decisiones para conseguir su objetivo. Está involucrado en la estimación del esfuerzo de las tareas del Backlog.
- **Customer:** El cliente participa en las tareas que involucran la lista de Product Backlog.
- **Management:** Es el responsable de tomar decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto. Participa en la selección de objetivos requerimientos y en la selección del Scrum Owner, Tiene la responsabilidad de controlar el progreso y trabaja junto con el Scrum master en la reducción de la lista de Product Backlog.

#### b) Involucrados

Aunque no son parte del proceso de Scrum, es necesario en la retroalimentación de la salida del proceso y así poder revisar y planificar cada sprint.

- **Usuarios:** Es el destinatario final de producto.
- **Stakeholders:** las personas a las que el proyecto les producirá un beneficio. Participan durante las revisiones del Sprint.
- **Managers:** Toma las decisiones finales participando en la selección de los objetivos y de los requisitos.

### 2.4.3 LOS ELEMENTOS DE SCRUM

Los Elementos del scrum se dividen en la pila de tareas por producto, reunión diaria y las tareas por sprint.

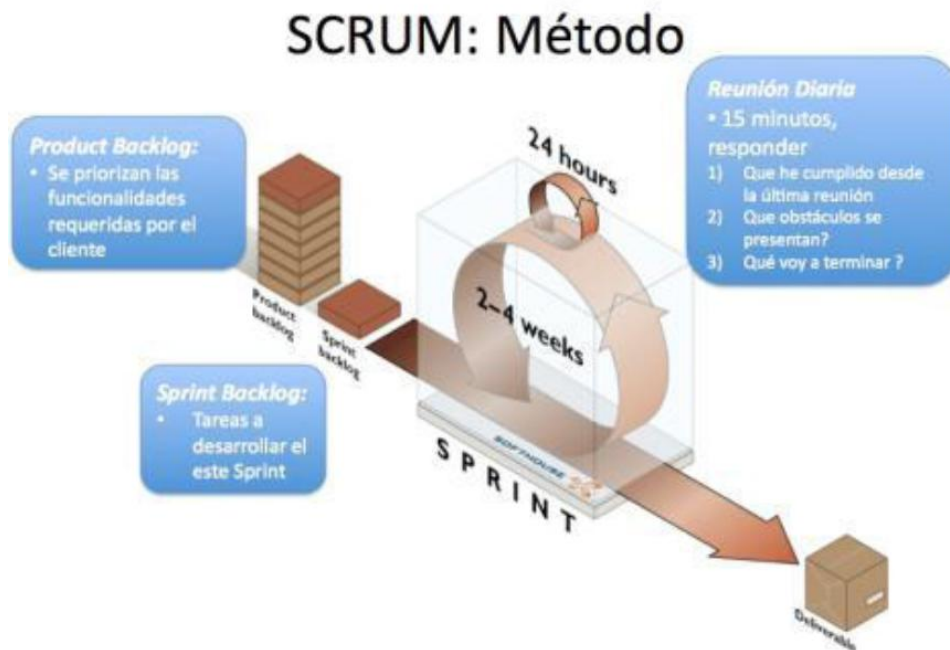


Figura 2.4 Elementos de Scrum  
Fuente: Metodología de Scrum, 2014

Los elementos que forman a Scrum son:

#### 2.4.3.1 PRODUCT BACKLOG

Es el inventario en el que se almacenan todas las funcionalidades o requisitos en forma de lista priorizada. Estos requisitos serán los que tendrá el producto o los que va adquiriendo en el transcurso de las iteraciones.

La lista será gestionada y creada por el cliente con la ayuda del Scrum Master, quien indicara el coste estimado para completar un requisito, y además contendrá todo lo que aporte un valor final al producto.

- Contendrá los objetivos del producto, se suelen usar para expresar las historias de usuario.
- En cada objetivo, se indicará el valor que le da el cliente y el coste estimado; de esta manera, se realiza la lista, priorizando por valor y coste, se basará en el Rol.
- La lista ha de incluir los posibles riesgos e incluir las tareas necesarias para resolverlos.

Es necesario que antes de empezar el Sprint se definan cuáles van a ser los objetivos del producto y tener la lista de los requisitos ya definida. No es necesario que se muy detallada, simplemente deberá contener los requisitos principales para que el equipo pueda trabajar. Se entiende que un producto esté completado si:

- Asegura que se pueda realizar un entregable para realizar una demostración de los requisitos y ver que se han cumplido.
- Incluirá todo lo necesario para indicar que está realizando el producto que el cliente desea.

Finalmente, el producto Backlog irá evolucionando mientras el producto exista en el mercado. Esta es la forma para evolucionar y tener un valor de producto para el cliente suficiente para ser competitivo.

#### **a) Las Historias de Usuario**

Son las descripciones de las funcionalidades que va a tener el software. Estas historias de usuario, serán el resultado de la colaboración entre el cliente y el equipo, e irán evolucionando durante toda la vida del proyecto.



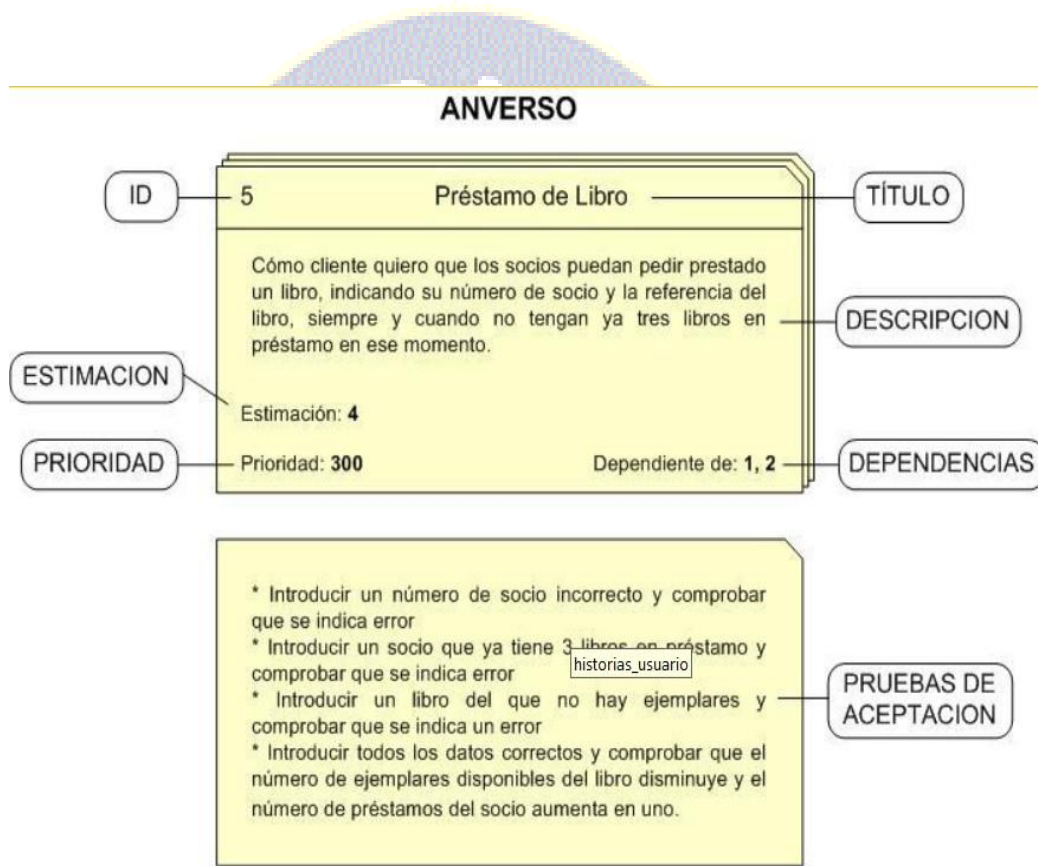


Figura 2.5 Historia de usuario  
Fuente: Elaboración Propia

### b) Formato de Pila Del Producto (Product Backlog)

En Scrum la preferencia por tener documentación en todo momento es menos estricta, se encuentra más necesario el mantener una comunicación directa con el equipo, por eso se usa como herramienta el Backlog.

Aunque no hay ningún producto especial a la hora de confeccionar la lista, es conveniente incluya información relativa:

- Identificar para la funcionalidad.
- Descripción para la funcionalidad.
- Sistema de priorización u orden.

- Estimación.

### **2.4.3.2 SPRINT BACKLOG**

Es la lista de tareas que elabora el equipo durante la planificación de un sprint. Se asigna las tareas a cada persona y el tiempo que queda para terminarlas.

De esta manera el proyecto se descompone en unidades más pequeñas y se puede determinar en qué tareas no se está avanzando e intentar eliminar el problema.

### **2.4.3.3 INCREMENTO**

Representa los requisitos que se han completado en una iteración y que son perfectamente operativos. Según los resultados que se obtengan, el cliente puede ir haciendo los cambios necesarios y replantearlos en el proyecto.

## **2.4.4 FASES DE SCRUM**

Para palacios y Ruata (2010), el proceso de desarrollo de Scrum se compone en cinco actividades: revisión de los planes de los reléase, distribución y ajuste de los estandartes del producto, sprint, revisión del sprint y cierre.

Para WikiUDO (2013), de manera general el proceso de desarrollo de Scrum se compone en cinco fases importantes: Planes de lanzamiento, distribución, revisión, ajuste de los estándares del producto revisión del sprint y cierre.

Para Schwaber (1995), el modelo original presentado para el desarrollo está compuesto de tres fases: Pre-Game, Game y Post-Game.

Para desarrollar el presente proyecto se aplicarán las fases originales de Scrum, vamos a describir las fases.

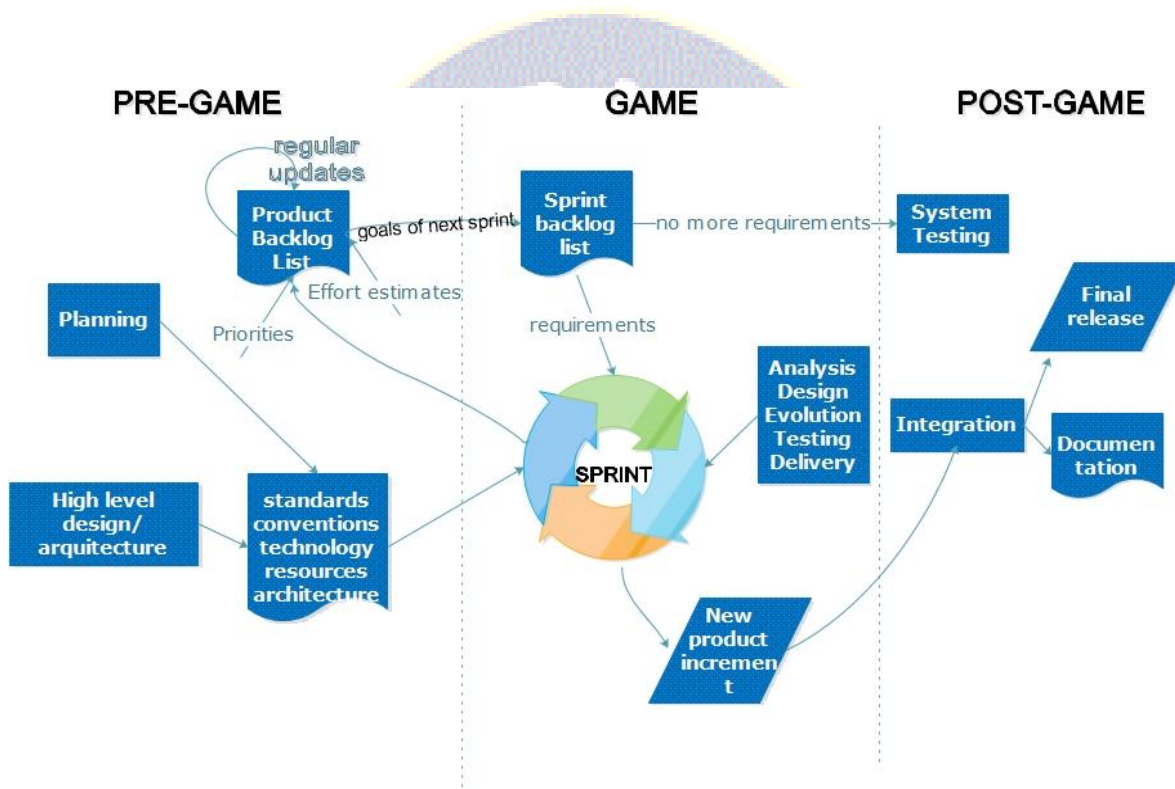


Figura 2.6 Scrum Methodologies  
Fuente: Scrum Manage, Body of knowledge, 2010

#### 2.4.4.1 PRE-GAME

La fase de Pre-Game incluye dos sub fases: Planning (planeación) y Architecture (arquitectura).

- a) **Planning.** Consiste en la definición del sistema que será construido. Para esto se crea la lista Product Backlog a partir del conocimiento que actualmente se tiene del sistema. En ella se expresan los requerimientos priorizados y a partir de ella se estima el esfuerzo requerido.

La Product Backlog List es actualizada constantemente con ítems nuevos y más detallados, con estimaciones más precisas y cambios en la prioridad de los ítems.

**Tareas:**

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
<p><b>Crear el Product Backlog List y controlar su consistencia</b></p>	<p>Posibles elementos de esta lista son requerimientos técnicos y del negocio, funciones, errores a reparar, defectos, mejoras y actualizaciones tecnológicas requeridas.</p> <p>Para esto se agregan, modifican, eliminan, especifican y priorizan sus elementos</p>	<p>Product Owner</p>	<p>Requerida</p>
<p><b>Priorizar el Product Backlog List</b></p>	<p>Esta actividad se basa en considerar que elementos tienen más o menos influencia en el éxito del proyecto en un momento dado; considerando que los elementos con mayor prioridad se realizan primero.</p>	<p>Product Owner</p>	<p>Requerida</p>
<p><b>Effort Estimation</b></p>	<p>Es un proceso iterativo que reúne toda la información que haya acerca un elemento para tener un mayor nivel de precisión en la estimación.</p> <p>Siempre se mide el esfuerzo que falta para cumplir con el / los objetivos tanto a nivel de la lista Product Backlog como para el Sprint Backlog (lo que resta).</p>	<p>Product Owner</p>	<p>Requerida</p>
<p><b>Design Review</b></p>		<p>Product</p>	<p>Requerida</p>

<b>Meeting</b>	En esta instancia se comunica el diseño a los interesados para revisar el cumplimiento de los ítems especificados en el Product Backlog	Owner	
----------------	---	-------	--

Tabla 2.1 Concepción inicial del producto que tienen los accionistas o interesados  
Fuente: Elaboración Propia

**Verificación:** Deben estar realizadas todas las tareas requeridas.

**Salida:** Product Backlog List, Arquitectura.

- b) **Architecture / High level Design.** El diseño de alto nivel del sistema se planifica a partir de los elementos existentes en la Product Backlog List. En caso de que el producto a construir sea una mejora a un sistema ya existente, se identifican los cambios necesarios para implementar los elementos que aparecen en la lista Product Backlog y el impacto que pueden tener estos cambios. Se sostiene una Design Review Meeting para examinar los objetivos de la implementación y tomar decisiones a partir de la revisión. Se preparan planes preliminares sobre el contenido de cada release.

#### 2.4.4.2 GAME

La fase de **Development (desarrollo)** también llamada Game Phase (fase de Juego) es la parte ágil de Scrum.

En esta fase se espera que ocurran cosas impredecibles. Para evitar el caos Scrum define prácticas para observar y controlar las variables técnicas y del entorno, así también como la metodología de desarrollo que hayan sido identificadas y puedan cambiar. Este control se realiza durante los Sprints. Dentro de variables de entorno encontramos: tiempo, calidad, requerimientos, recursos, tecnologías y herramientas de implementación. En lugar de

tenerlas en consideración al comienzo del desarrollo, Scrum propone controlarlas constantemente para poder adaptarse a los cambios.

**Desarrollo de sprints:** Desarrollo de la funcionalidad de la nueva versión con respeto continuo a las variables de tiempo, requisitos, costo y competencia. La interacción con estas variables define el final de esta fase. El sistema va evolucionando a través de múltiples iteraciones de desarrollo o sprints.

- **Planificación:** Antes de comenzar un Sprint, se lleva a cabo dos reuniones consecutivas, en la primera se clarifica y se prioriza nuevamente el product Backlog del producto. En la segunda reunión se deben considerar como alcanzar los requerimientos y crear el Backlog del sprint.
- **Desarrollo del Sprint:** El trabajo se organiza en iteraciones en no más de 30 días, el sprint es el desarrollo de la nueva funcionalidad del producto. Esta fase provee la siguiente documentación: el sprint Backlog, los responsables y la duración de cada actividad.
- **Revisión del Sprint:** Al final de cada iteración se lleva a cabo una reunión de revisión se presenta la nueva funcionalidad del producto, diseño, ventajas, inconvenientes y esfuerzos del equipo.

**Entrada:** Product Backlog List

Nombre	Descripción	Responsables	Requerida
<b>Sprint</b>		Scrum Master	Requerida
<b>Planning Meeting</b>	<p>En una reunión organizada por el Scrum Master, que se realiza en dos fases.</p> <ul style="list-style-type: none"> <li>- La primera fase tiene como objetivo establecer que ítems del product Backlog List van a ser realizados durante el</li> </ul>	<p>Product Owner</p> <p>Scrum Team</p> <p>Management</p>	

	<p>Sprint. Esto se realiza a partir de que el Scrum Team considera que puede construir durante el sprint.</p>	Customer	
	<ul style="list-style-type: none"> <li>- En la segunda fase se decide cómo van a alcanzar los objetivos del Sprint. En esta fase se crea el Sprint Backlog, indicando que tareas debe desempeñar el equipo para cumplir con los dichos objetivos.</li> </ul>	Scrum Team Scrum Master Product Owner	
<b>Daily Scrum Meeting</b>	<p>Las reuniones se realizan en el mismo lugar y a la misma hora cada día. Idealmente en la mañana para definir el trabajo para el día. Tienen una duración de 15 minutos y los participantes se quedan parados. Estas reuniones no se realizan para resolver problemas. En ellas se realizan tres preguntas.</p> <ul style="list-style-type: none"> <li>- ¿Qué hiciste ayer?</li> <li>- ¿Qué harás hoy?</li> <li>- ¿Qué obstáculos ves en tu camino?</li> </ul> <p>Los participantes son clasificados según el compromiso que tengan con las actividades del proyecto en dos categorías: gallinas y cerdos. Los cerdos son los que están más comprometidos y por tanto son los que pueden hablar y brindan opiniones. Esto ayuda a evitar reuniones innecesarias.</p> <p>Estas reuniones no pueden ser sustituidas por reportes o vía correo por dos motivos:</p> <ul style="list-style-type: none"> <li>- El equipo entero ve el paisaje entero cada día.</li> </ul>	Scrum Team	Requerida

	- Es un elemento de presión para que el individuo haga lo que dijo que va a hacer.		
<b>Sprint Review Meeting</b>	Es una reunión informal que tienen como regla que su preparación no pueda tomar más de 2 horas. En ella el equipo presenta lo que ha logrado durante el Sprint. Generalmente toma la forma de un demo de las nuevas características o la arquitectura.  Los demos deben ser totalmente funcionales.	Customer Management Product Owner Otros interesados	Requerida

Tabla 2.2 Fase de desarrollo  
Fuente: Elaboración Propia

**Verificación:** Durante un sprint se puede acortar funcionalidades, pero la fecha de entrega debe ser respetada

**Salida:** Incremento del producto.

#### 2.4.4.3 POST-GAME

Contiene el cierre del release. Para ingresar a esta fase se debe llegar a un acuerdo respecto a las variables del entorno por ejemplo que los requerimientos fueron completados. El sistema está listo para ser liberado y es en esta etapa en la que se realiza integración, pruebas del sistema y documentación.

### 2.4.5 DESARROLLO DE LAS FASES DE UN PROYECTO EN SCRUM

#### 2.4.5.1 PREPARACIÓN DEL PROYECTO

Conocido como sprint 0, es la fase inicial en la que se intenta completar el caso de negocio con la finalidad de tomar decisiones que agreguen valor agregado al producto. Es



aconsejable no perder tiempo en buscar estimaciones exactas, es mejor invertir en el desarrollo del producto.

- Definir el proyecto.
- Definir tiempo de culminación.
- Definición del Backlog inicial.
- Dedicación de los entregables.

De la misma manera, está sujeto a unas determinadas condiciones determinadas por el equipo de trabajo que serán:

- Tiempo para un entregable.
- La estimación inicial.
- Selección del Backlog del producto.

Se hacen unas reuniones iniciales con todos los roles del equipo para tratar:

- Dimensión del proyecto.
- Revisión del Backlog.
- Organización del equipo y horario para establecer reuniones de control.

#### **2.4.5.2 PLANIFICAR UN SPRINT**

Denominado también “Sprint Planning Meeting”, tiene como finalidad realizar una reunión, en la que participarán el Producto Owner, el Scrum Master y el equipo, con la intención de seleccionar la lista Backlog de producto de las funcionalidades sobre las que se va a trabajar, y que darán valor al producto.

##### **Primera parte de la reunión:**

- El equipo selecciona los ítems para transformarlos en entregables.

- El equipo hace sugerencias, pero es el Product Owner el que decidirá si formarán parte del Sprint.
- El equipo seleccionará el elemento a implementar, de los seleccionados por el Product Owner para ese Sprint.

**Segunda parte de la reunión:**

- El equipo hace las preguntas necesarias que tengan sobre el Product Backlog al Product Owner.
- El equipo se encargará de encontrar la solución adecuada para transformar para la parte seleccionada de una funcionalidad entregable.

El resultado de la segunda parte de la reunión es una lista denominada “Sprint Backlog” con las tareas, estimaciones y las asignaciones de trabajo el equipo para poder empezar a desarrollar la funcionalidad.

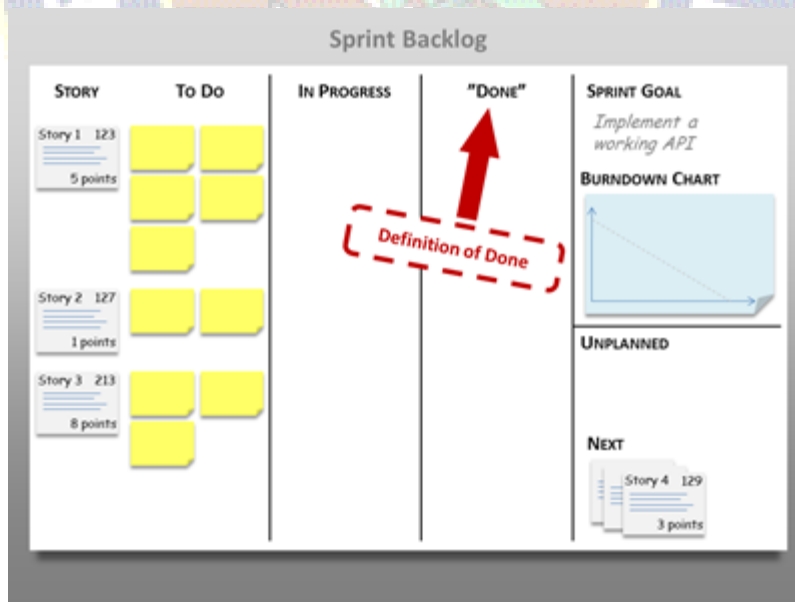


Figura 2.7 Sprint Backlog  
Fuente: Metodología de Scrum, 2014

## a) La Estimación De Sprint

### i. Planning Poker

Realizar la estimación para los ítems seleccionados, es una tarea que debería involucrar a todos los miembros del equipo, para ello se usará la técnica de **Planning Poker**.

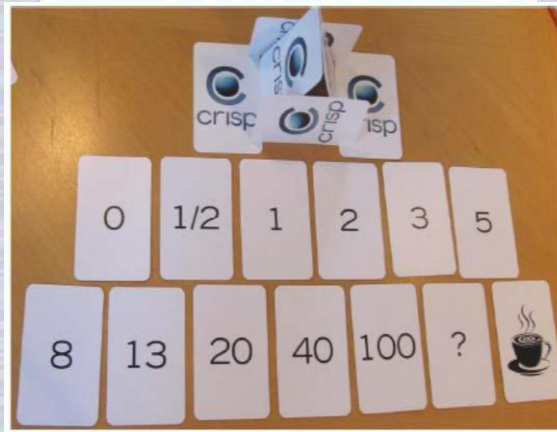


Figura 2.8 Técnica "Planning Poker"  
Fuente: Metodología de Scrum, 2014

Cada miembro del equipo tiene una baraja de 13 cartas, se propone una historia, el miembro del equipo selecciona una carta y la coloca boca abajo, cuando todos los miembros han seleccionado su carta, se le da vuelta al mismo tiempo. Se comprueban las estimaciones, y si hay mucha diferencia se discute sobre la razón y se ponen en común las ideas, y se realiza nuevamente la selección.

### ii. Mantener el Backlog del Sprint

El equipo tiene que mantener actualizado el Backlog del Sprint para poder tener feedback y tomar cualquier decisión de manera rápida.

### 2.4.5.3 DESARROLLO DEL SPRINT

En los sprints, el equipo trabaja para conseguir un incremento del producto, que será productivo para el Product Owner y los Stakeholders. El tiempo más conveniente está entre 2 y 4 semanas.

#### Reuniones del Sprint:

Durante la ejecución del Sprint se van a realizar 3 reuniones:

- Reunión de planificación (Sprint Planning Meeting). Una vez definidos, el equipo comienza su desarrollo, el equipo puede realizar consultas fuera del Sprint, el equipo se auto gestionará solo, si durante el sprint no es viable, se puede realizar una nueva planificación, el equipo no se comprometerá a cumplir el Backlog, realizará una consulta al Product Owner que ítems eliminar.
- Reunión diaria (Daily Meeting). En esta reunión los componentes del equipo comparten información relativa al desarrollo. La reunión no podrá consumir más de 15 minutos y cada miembro contestará a tres preguntas básicas.
  - ¿Qué se ha hecho de nuevo con respecto a la última reunión diaria?
  - ¿Qué será lo siguiente a realizar?
  - ¿Qué problema hay para realizarlos?
- Reunión de revisión del Sprint (Sprint Review Meeting). En esta reunión los desarrolladores presentan el producto entregable que han implementado y los gestores, clientes, usuarios y Product Owner analizan esa entrega y escuchan al equipo sobre los problemas que se han presentado durante el proceso. Si la funcionalidad no está acabada no se puede presentar, no se debe mostrar artefactos que no son funcionalidades para no equivocar a los stakeholders.

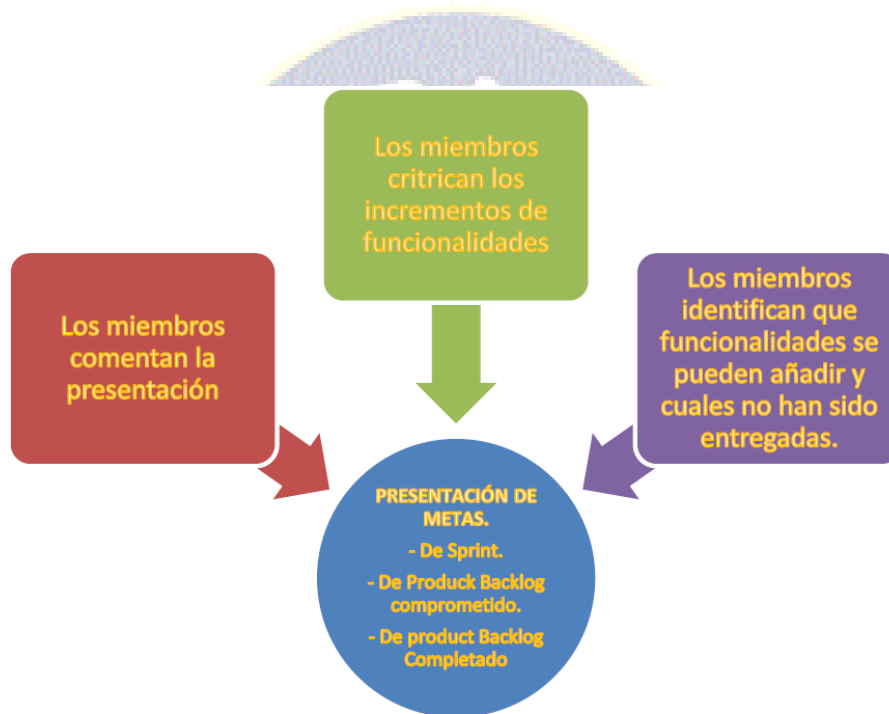


Figura 2.9 Reuniones de Sprint  
 Fuente: Manuel Trigas Gallego, Gestión de Proyectos Informáticos, 2014

- **Reunion retrospectiva (Sprint retrospective Meeting).**

En esta reunión, el equipo debatirá temas relacionado con el sprint reciente finalizado y los cambios que se podrían hacer para mejorar el próximo retrospectiva Sprint que sea más productivo.

## 2.5 ROADMAP

El proceso de **roadmap** (planificación Hoja de Ruta PGR), en gestión de productos.

El manejo de la situación actual de una empresa y su relación con el mercado es tan importante como el enfoque integral de los proyectos futuros. Estos proyectos son parte esencial de una empresa en funcionamiento con el fin de tener éxito.



Figura 2.10 El camino de ruta de un proyecto  
Fuente: entrepreneural-insights, 2015

Los empresarios deben generar estrategias generales de la empresa, bien definidas las intenciones y los objetivos, los empresarios debe definir también las formas en las que se logran estos objetivos y establecer los plazos aproximados.

Varios tipos de planes de trabajo relacionados con el negocio surgieron como una herramienta para cumplir con los fines antes mencionados.

Todos los tipos de planes de trabajo son, por tanto, las herramientas utilizadas por los empresarios con el fin de presentar los datos analíticos recogidos a través del examen exhaustivo de todas las propiedades internas y externas que se correlacionan en los asuntos de negocios. Estas presentaciones de información sobre el estado actual se utilizan para la construcción de una propuesta de plan de acción futura en relación con las áreas específicas que están construidos.

Por tanto, las hojas de ruta de productos integran factores como las tendencias del mercado y la segmentación de los clientes de acuerdo a sus necesidades y preferencias, así como el desarrollo de la tecnología propuesta en relación con las características y las futuras versiones respecto a determinados productos o líneas de productos. Con ésta información, recopilada a través de la colaboración con diversos departamentos de la organización, así como los clientes, los evaluadores y socios, equipos de gestión de

producto y gestores crear una vía de producto que parece más plausible y rentable en el momento de su elaboración.

### **Pasos de la creación del roadmap**

- Recopilación de la información.
- Organizar
- Diseño
- Uso interno
- Entrega externa

En la ingeniería de software un RoadMap (que podría traducirse como hoja de ruta) es una planificación del desarrollo de un software con los objetivos a corto y largo plazo, y posiblemente incluyendo unos plazos aproximados de consecución de cada uno de estos objetivos. Se suele organizar en hitos o "milestones", que son fechas en las que supuestamente estará finalizado un paquete de nuevas funcionalidades.

Para los desarrolladores de software, se convierte en una muy buena práctica generar un Roadmap, ya que de esta forma documentan el estado actual y posible futuro de su software, dando una visión general o específica de hacia a dónde apunta a llegar el software.

La expresión Roadmap se utiliza para dar a conocer el "trazado del camino" por medio del cual vamos a llegar del estado actual al estado futuro. Es decir, la secuencia de actividades o camino de evolución que nos llevará al estado futuro.

### **2.5.1 COMBINACIÓN DE ROADMAP Y BACKLOG DE SCRUM**

Existen escenarios donde una organización mediana o grande tiene múltiples equipos que tienen poca o ninguna relación entre ellos. Esto puede pasar en una empresa

que realice proyectos para clientes o bien, en menor grado, en una que desarrolle un producto complejo con equipos encargándose de componentes o subproductos.

En estos escenarios, especialmente si no existe un departamento de procesos o una tradición en la gestión de procesos, suele ser más sencillo extender el modelo monoequipo de SCRUM a múltiples equipos debido a la facilidad que éste introduce para el seguimiento de los equipos (con las herramientas roadmap y sprint backlog).

La hoja de ruta de productos (roadmap) es una herramienta estratégica de planificación que muestra cómo es probable que crezca el producto a través de varias versiones. Esto crea una continuidad de propósitos, facilita la colaboración de las partes interesadas, ayuda a adquirir financiamiento, y hace que sea más fácil para coordinar el desarrollo y lanzamiento de productos diferentes.

Aplicado correctamente, las dos herramientas se complementan muy bien. La hoja de ruta de producto (roadmap) cuenta con un historial de largo plazo sobre el crecimiento probable del producto, proporciona un apoyo para el Backlog, que contiene los detalles necesarios para crear el producto.

Para emplear con éxito ambas herramientas, se debe decidir si el Backlog del producto debe ser derivado de su plan de producto (roadmap) o al revés.

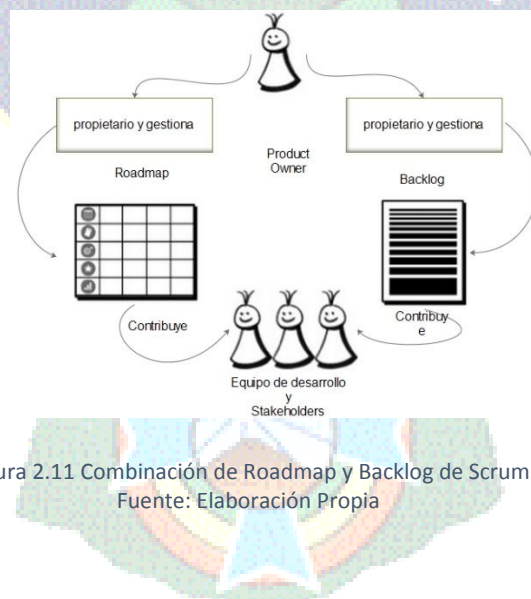


Figura 2.11 Combinación de Roadmap y Backlog de Scrum  
Fuente: Elaboración Propia



## 2.5.2 HISTORIAS DE USUARIO

Las historias de usuario son utilizadas como herramienta para dar a conocer los requerimientos del sistema al equipo de desarrollo. Son pequeños textos en los que el cliente describe una actividad que realizará el sistema, la redacción de los mismos se realiza bajo la terminología del cliente en este caso la empresa “COLSER SRL.”, no del desarrollador, de forma que sea clara y sencilla sin profundizar en detalles, lo que implica que se debe proporcionar una idea clara.

El propósito de las historias es dar a conocer requerimientos del sistema al equipo de desarrollo a la vez ayudan a identificar roles y actividades que realizarán los distintos usuarios que interactuarán con el sistema de software.

También se pueden usar para estimar el tiempo que el equipo de desarrollo tomara para realizar las entregas, en una entrega se pueden desarrollar una o varias historias de usuario, esto depende del tiempo que demore la implementación de cada una de las mismas.

Se delimitará las historias sólo a los principales actores dentro de la metodología Scrum que serán:

- Gerente de Proyectos
- Gerente de Soporte
- Product Owner
- Scrum Master

Quiénes serán los únicos que cuenten con dichas historias, para poder analizar plasmar lo obtenido en diagramas de clases.

Historia de Usuario			
No.	Usuario:		
Nombre historia:			
Prioridad en negocio: Baja/Media/Alta	Riesgo: Bajo/Medio/Alto	Dificultad:	Iteración:
Descripción:			
Observaciones:			

Figura 2.12 Historia de Usuario  
Fuente: Emilio A. Sanchez, Mejorando la gestión de historias de usuario, 2011

## 2.6 ASEGURAMIENTO DE LA CALIDAD DE SOFTWARE(QA)

En el ámbito del desarrollo de software, el aseguramiento de la calidad (Quality assurance) trata de un conjunto de actividades de evaluación de las distintas etapas del proceso de desarrollo para garantizar que el producto final sea de calidad. El concepto de calidad se presta a múltiples interpretaciones, pero siempre implica que el software satisfaga las necesidades del cliente.

Más allá de las diferencias, un buen plan de QA no puede desconocer la importancia de los estándares. Con esto nos referimos a reglas escritas y no ambiguas sobre los objetivos del producto, las metodologías de diseño y a seguir y convenciones necesarias para guiar la tarea de los programadores (estilos de codificación, estructuras de datos, etc.).

El plan de QA atraviesa el proceso de desarrollo desde el nacimiento de la idea hasta la implementación del software. En las primeras etapas, verifica que los objetivos estén bien planteados y los requerimientos sean precisos. En las fases de diseño y codificación, vigila el cumplimiento de los estándares fijados. Finalmente, revisa que el software en funcionamiento respete los requerimientos pedidos y que la entrega al cliente se haga en las condiciones adecuadas.

El QA se basa en conjunto de pruebas de calidad entre las que se incluyen:

- **Pruebas unitarias.** Se prueba que cada módulo funcione bien por separado.
- **Pruebas de stress.** Se prueba la resistencia de la aplicación enviándole una cantidad de peticiones excesiva, buscando que colapse.
- **Pruebas de integración.** Los módulos probados independientemente durante el testeo unitario se acoplan y se prueban en conjunto.
- **Pruebas de aceptación.** El usuario verifica que el producto satisfaga sus expectativas.
- **Pruebas funcionales.** Se prueba que el software ofrezca las funciones solicitadas.

Las pruebas de QA no sólo son beneficiosas para el usuario final, que recibirá un producto de calidad, sino también para el equipo de desarrollo, que al establecer un control permanente sobre el proceso evitará en buena medida los costos de tener que corregir errores en etapas avanzadas del proyecto.

### 2.6.1 PRUEBAS FUNCIONALES DE SOFTWARE

Se denominan pruebas funcionales o Functional Testing, a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados, es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas y al dar conformidad sobre esta el paso siguiente es el pase a producción.

A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, ya que los testers o analistas de pruebas, no enfocan su atención a como se generan las respuestas del sistema, básicamente el enfoque de este tipo de prueba se basa en

el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas.

Las pruebas Funcionales deben enfocarse en los requisitos funcionales, las pruebas pueden estar basadas directamente en los Casos de Uso (o funciones de negocio), y las reglas del negocio. Las metas de estas pruebas son:

- Verificar la apropiada aceptación de datos,
- Verificar el procesamiento y recuperación y la implementación adecuada de las reglas del negocio.

Este tipo de pruebas están basadas en técnicas de caja negra, que es, verificar la aplicación (y sus procesos internos) mediante la interacción con la aplicación vía GUI y analizar la salida (resultados). Lo que se identifica a continuación es un diseño preliminar de las pruebas recomendadas para cada aplicación.

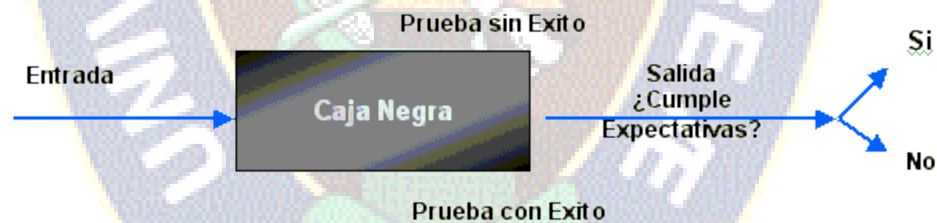


Figura 2.13 Pruebas de Caja Negra  
Fuente: Elaboración propia

La técnica que usa es ejecutar cada caso de uso, flujo de caso de uso, o función, usando datos válidos e inválidos, para verificar lo siguiente:

- Que los resultados esperados ocurran cuando se usen datos válidos.

- Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos.
- Que se aplique apropiadamente cada regla de negocio.

## 2.6.2 PRUEBAS DE REGRESIÓN

Se denominan pruebas de regresión a cualquier tipo de pruebas de software que intentan descubrir errores (bugs), carencias de funcionalidad, o divergencias funcionales con respecto al comportamiento esperado del software, causados por la realización de un cambio en el programa.

Este tipo de cambio puede ser debido a prácticas no adecuadas de control de versiones, falta de consideración acerca del ámbito o contexto de producción final y extensibilidad del error que fue corregido (fragilidad de la corrección), o simplemente una consecuencia del rediseño de la aplicación.

En esta prueba se vuelve a probar el sistema a la luz de los cambios realizados durante el debugging, mantenimiento o desarrollo de la nueva versión del sistema buscando efectos adversos en otras partes.

Las Técnicas para estas pruebas son las siguientes:

- La prueba de regresión es una nueva corrida de casos de prueba previos.
- Se requiere de políticas para ser creada la prueba de regresión y decidir qué casos de prueba incluir, para probar eficientemente.
- La prueba de regresión es un buen candidato para automatización. Desde que estas pruebas se repiten una y otra vez, las herramientas para minimizar el esfuerzo del trabajo son útiles.

- La prueba de viejas funcionalidades es más importante que la de nuevas funcionalidades.
- Aquellos casos de uso (y los casos de prueba asociados) que descubren defectos tempranamente deben ser incluidos en la prueba de regresión.

### **2.6.3 CASOS DE PRUEBA**

Un caso de prueba o test case es, en ingeniería del software, un conjunto de condiciones o variables bajo las cuáles un analista determinará si una aplicación, un sistema software (software system), o una característica de éstos es parcial o completamente satisfactoria.

Se pueden realizar muchos casos de prueba para determinar que un requisito es completamente satisfactorio. Con el propósito de comprobar que todos los requisitos de una aplicación son revisados, debe haber al menos un caso de prueba para cada requisito a menos que un requisito tenga requisitos secundarios. En ese caso, cada requisito secundario deberá tener por lo menos un caso de prueba. Algunas metodologías como RUP recomiendan el crear por lo menos dos casos de prueba para cada requisito. Uno de ellos debe realizar la prueba positiva de los requisitos y el otro debe realizar la prueba negativa.

Si la aplicación es creada sin requisitos formales, entonces los casos de prueba se escriben basados en la operación normal de programas de una clase similar.

Lo que caracteriza un escrito formal de caso de prueba es que hay una entrada conocida y una salida esperada, los cuales son formulados antes de que se ejecute la prueba. La entrada conocida debe probar una precondition y la salida esperada debe probar una postcondición.

Bajo circunstancias especiales, podría haber la necesidad de ejecutar la prueba, producir resultados, y luego un equipo de expertos evaluaría si los resultados se pueden

considerar como "correctos". Esto sucede a menudo en la determinación del número del rendimiento de productos nuevos. La primera prueba se toma como línea base para los subsecuentes ciclos de pruebas/lanzamiento del producto.

Los casos de prueba escritos, incluyen una descripción de la funcionalidad que se probará, la cual es tomada ya sea de los requisitos o de los casos de uso, y la preparación requerida para asegurarse de que la prueba pueda ser dirigida.

Los casos de prueba escritos se recogen generalmente en una suite de pruebas.

Las variaciones de los casos de prueba son comúnmente utilizadas en pruebas de aceptación. La prueba de aceptación es realizada por un grupo de usuarios finales o los clientes del sistema, para asegurarse que el sistema desarrollado cumple sus requisitos. La prueba de aceptación de usuario se distingue generalmente por la incorporación de un trayecto feliz o casos de prueba positivos.

## **2.7 PATRÓN DE DISEÑO: OBJETOS DE PÁGINA (PAGE OBJECT PATTERN)**

El Patrón de diseño Page object Pattern representa las pantallas de una aplicación web como un conjunto de objetos. Dentro de una aplicación web existen áreas las cuales interactúan con las pruebas.

Esto ocurre ya que este patrón de diseño permite separar el comportamiento de una página de los detalles de su implementación, es simplemente una clase que encapsula los detalles de implementación de una página web, permitiendo que la automatización se centre solamente en el comportamiento.

Profundizando aún más la definición, creamos los page objects para agrupar comportamientos comunes y generar una capa de abstracción con la aplicación con el objetivo de reducir el trabajo de mantenimiento y que la estructura de los test reduzca complejidad. Por eso comúnmente tenemos un Page object donde agruparemos aquellos

métodos comunes a todas las páginas y luego vamos creando un page object para cada grupo de elementos que se relacionen de alguna manera (ya sea una barra de menú que se repite en varias páginas, una página con una funcionalidad específica, un popup que aparece en varias páginas, etc...). Lo mismo va a aplicar para los test, ya que podemos centralizar en un TestPage object todo lo relacionado a la configuración de los test y métodos de validación/generación de datos.

En resumen, se puede decir que es un patrón de diseño que puede ser implementado como una las mejores prácticas de Selenium WebDriver. Hay ventajas importantes que caben mencionar:

- Reduce la cantidad de código duplicado.
- Hace los test más entendibles y robustos.
- Mejora la mantenibilidad de los test, es decir si la herramienta bajo testeo presenta algún cambio, el mantenimiento debe ser aplicado en un solo lugar.

En otras palabras:

PAGE = OBJECT

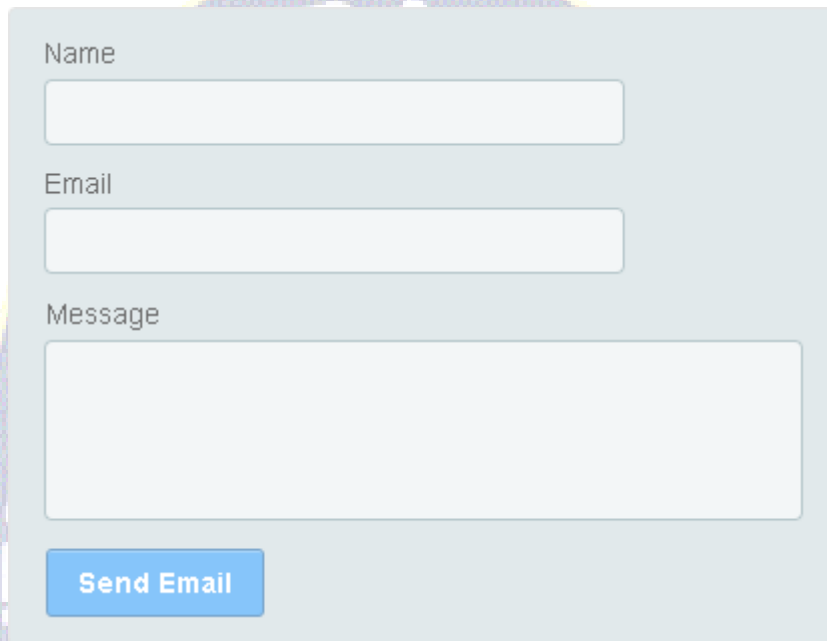
y, por consiguiente:

- Elementos de Página = Atributos
- Servicios de Página = Métodos

Ejemplo:

Tenemos la siguiente pantalla la cual muestra un formulario básico con algunos campos, Name, Email, Message y un botón el cual ejecuta la acción de envío de correo electrónico:





Name

Email

Message

Send Email

Figura 2.14 Ejemplo de Formulario básico  
Fuente: Elaboración propia

Los elementos de esta página serían:

- TextBox “Name”
- TextBox “Email”
- TextArea “Message”
- Button “Send Email”

Y los servicios se detallan a continuación:

- `setName();`
- `setEmail();`
- `setMessage();`
- `clickOnSendButton();`

## 2.8 HERRAMIENTAS DE DESARROLLO

### 2.8.1 JAVA

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

Java es un lenguaje de programación de alto nivel que tiene las siguientes características:

- Orientado a objetos
- Distribuido y dinámico
- Robusto
- Seguro
- Multitarea
- Portable

La mayoría de los lenguajes de programación se caracterizan por ser interpretados o compilados, lo que determina la manera en cómo serán ejecutados en una computadora.

Java tiene la característica de ser al mismo tiempo compilado e interpretado. El compilador es el encargado de convertir el código fuente de un programa en un código intermedio llamado bytecode que es independiente de la plataforma en que se trabaje y que es ejecutado por el intérprete de Java que forma parte de la Máquina Virtual de Java.

## 2.8.2 MAVEN

Maven es una herramienta open-source, que se creó en 2001 con el objetivo de simplificar los procesos de build (compilar y generar ejecutables a partir del código fuente).

Maven se utiliza en la gestión y construcción de software. Posee la capacidad de realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado. Es decir, hace posible la creación de software con dependencias incluidas dentro de la estructura del JAR. Es necesario definir todas las dependencias del proyecto (librerías externas utilizadas) en un fichero propio de todo proyecto Maven, el POM (Project Object Model). Este es un archivo en formato XML que contiene todo lo necesario para que a la hora de generar el fichero ejecutable de nuestra aplicación este contenga todo lo que necesita para su ejecución en su interior.

Sin embargo, la característica más importante de Maven es su capacidad de trabajar en red. Cuando definimos las dependencias de Maven, este sistema se encargará de ubicar las librerías que deseamos utilizar en Maven Central, el cual es un repositorio que contiene cientos de librerías constantemente actualizadas por sus creadores. Maven permite incluso buscar versiones más recientes o más antiguas de un código dado y agregarlas a nuestro proyecto. Todo se hará de forma automática sin que el usuario tenga que hacer nada más que definir las dependencias.

Podríamos decir, que Maven es una herramienta capaz de gestionar un proyecto software completo, desde la etapa en la que se comprueba que el código es correcto, hasta que se despliega la aplicación, pasando por la ejecución de pruebas y generación de informes y documentación.

Para ello, en Maven se definen tres ciclos de build del software con una serie de etapas diferenciadas. Por ejemplo, el ciclo por defecto tiene las etapas de:

- Validación (validate): Validar que el proyecto es correcto.
- Compilación (compile).
- Test (test): Probar el código fuente usando un framework de pruebas unitarias.
- Empaquetar (package): Empaquetar el código compilado y transformarlo en algún formato tipo .jar o .war.
- Pruebas de integración (integration-test): Procesar y desplegar el código en algún entorno donde se puedan ejecutar las pruebas de integración.
- Verificar que el código empaquetado es válido y cumple los criterios de calidad (verify).
- Instalar el código empaquetado en el repositorio local de Maven, para usarlo como dependencia de otros proyectos (install).
- Desplegar el código a un entorno (deploy).

### 2.8.3 SELENIUM

Selenium es un entorno de pruebas de software para aplicaciones basadas en la web. Es un conjunto de utilidades que facilita la labor de obtener juegos de pruebas para aplicaciones web. Para ello nos permite grabar, editar y depurar casos de prueba, que podrán ser ejecutados de forma automática e iterativa posteriormente.

Además de ser una herramienta para registrar acciones, permite editarlas manualmente o crearlas desde cero. Las acciones se basan en el uso de diferentes API's en diferentes lenguajes (PHP, Ruby, JAVA, Javascript, etc). Entre sus principales características podemos nombrar:

- Facilidad de registro y ejecución de los test.

- Referencia a objetos DOM en base al ID, nombre o a través de XPath.
- Auto-completado para todos los comandos.
- Las acciones pueden ser ejecutadas paso a paso.
- Herramientas de depuración y puntos de ruptura (breakpoints).
- Los test pueden ser almacenados en diferentes formatos.

El potencial de esta herramienta puede ser utilizado para la grabación de las pruebas funcionales durante la Generación de pruebas de regresión. Con este servicio se consigue obtener una batería de pruebas automatizadas que podrán ser utilizadas cuando sea necesario repetir las pruebas.

Selenium consta de varias herramientas. Por un lado, Selenium IDE es una extensión para Firefox que registra la actividad en el navegador durante un período determinado; esta actividad se traduce en una serie de comandos que se puede repetir y repetir, incluyendo aserciones y advertencias para realizar el test propiamente dicho. Esta serie de comandos puede exportarse en forma de script en muy diversos lenguajes: HTML, Python, Ruby, Java (JUnit), C# y algunos más.

Este script generado puede editarse para posteriormente ser ejecutado por Selenium WebDriver (antes llamado Selenium RC). Éste implementa una API cliente lista para ser usada con cualquier entorno de testing favorito, que se ejecutará sobre un servidor que gestiona los principales navegadores que se tenga instalado para realizar en ellos las pruebas. Es decir, se podrá ejecutar las pruebas automáticamente en todos los navegadores relevantes, de forma que no se escape ningún detalle de la aplicación.

### **2.8.3.1 SELENIUM WEBDRIVER**

Selenium WebDriver es una herramienta para automatizar los test de aplicaciones Web. Provee un API muy sencilla de usar, que puede usarse desde JUnit, TestNG o desde una clase principal al viejo estilo de JAVA.

Fue diseñado para proveer un API más simple en comparación a su antecesor Selenium-RC; ahora es un API compacto orientado a objetos que permite automatizar aplicaciones web para verificar que trabajen como se espera. Su tarea empieza al recibir comandos enviados a través de un API cliente, luego los envía a un navegador y finalmente devuelve los resultados. Selenium Web Driver hace llamadas directas al navegador usando el soporte nativo de automatización de cada uno de ellos en comparación a Selenium-RC que inyectaba funciones javascript dentro del navegador. Actualmente Selenium provee un API para Java, C#, Ruby y Python.

### **2.8.3.2 SELENIUM GRID**

Selenium Grid es una herramienta para lanzar test de Selenium de forma distribuida. Utiliza internamente Selenium WebDriver para ejecutar los test de integración con distintos navegadores y plataformas.

Dentro de Selenium Grid se encuentra un módulo importante, el Hub. Este módulo es la parte central de Selenium Grid ya que nos permitirá controlar los distintos Servidores encargados de lanzar los test y distribuir los test en los mismos.

Selenium grid permite levantar un servidor hub, al que se pueden conectar tantos nodos como queramos. Cada uno de estos nodos es un servidor de selenium donde se pueden ejecutar las pruebas. Tendríamos de este modo una “granja de servidores” compuesta por una serie de nodos donde manejar el navegador para realizar las pruebas. Cada uno de estos nodos se puede configurar para manejar un determinado tipo de navegador.

Todas las pruebas se lanzan apuntando a un único punto, el hub, y es este hub el encargado de encolar las peticiones y distribuirlas por los distintos nodos.

#### **2.9.4 SISTEMA DE INTEGRACIÓN CONTÍNUA: JENKINS**

La integración continua es una práctica de desarrollo software donde los miembros de un equipo integran su trabajo frecuentemente (como mínimo una vez al día, aunque normalmente se realizan múltiples integraciones diarias).

Cada integración se verifica compilando el código fuente y obteniendo un ejecutable (a esto se le llama build, y debe hacerse de forma automatizada). Además, también se pasan las pruebas y métricas de calidad para detectar los errores tan pronto como sea posible.

Al integrar frecuentemente el código, y con la ayuda de herramientas como Jenkins, se puede saber el estado del software en todo momento.

Jenkins se trata de una herramienta open source de integración continua que permite ejecutar la batería de test de los proyectos en un entorno independiente archivando los resultados de cada build y generando estadísticas. Las ejecuciones se pueden programar periódicamente, realizar bajo demanda o bien mediante disparadores tras realizar un commit en el repositorio del proyecto.

Jenkins nace en 2011 y es un proyecto que surge como un fork de Hudson, desarrollado inicialmente en 2004 dentro de Sun Microsystems. Actualmente el desarrollo de Jenkins continúa bastante activo y hay gran cantidad de plugins que nos permitirán entre otras cosas integrar Jenkins con repositorios de Github o Bitbucket además de otros servicios de mensajería como Hipchat para recibir las notificaciones del estado de los últimos builds.

## CAPÍTULO 3 MARCO APLICATIVO

### 3.1 INTRODUCCIÓN

En este capítulo se describe la implementación de la metodología Scrum para el desarrollo del proyecto, se aplicará las fases originales de scrum descritas por Schwaber(2002).

La descripción de la implementación de las tareas se apoya en Roadmap, que coadyuvará al Backlog del producto, estimando tareas y fechas de entrega siguiendo un lineamiento acorde a las necesidades de ProcessMaker.



Figura 15.1 Fases del Marco Aplicativo  
Fuente: Elaboración Propia

### 3.2 PRE – GAME

- Roles del Proyecto.
- Análisis de requerimientos, crear el Product Backlog List y controlar su consistencia., siguiendo el lineamiento Roadmap para la crear (historias de usuario).
- Priorizar el Product Backlog List.



- Effort Estimación, definición de cronograma de trabajo.
- Design Review Meeting.

Deben estar todas las tareas requeridas, donde las salidas serán:

- Product Backlog List
- Arquitectura de la herramienta.

### 3.2.1 ROLES DEL PROYECTO

El equipo de trabajo y los roles para el desarrollo del producto software, bajo la metodología SCRUM es conformado, por el Product Owner/Product Manager, Scrum Master, y desarrollador

Mencionando los roles más significativos para el desarrollo del proyecto, citamos:

ROL	PERSONA	CARGO
Product Owner (dueño del producto)	Ing. Ademar Hurtado	QA Team Leader
Scrum Master (Facilitador)	Ing. Mauricio Veliz	Gerente del área de desarrollo Colser SRL.
Quality Engineer/Developer	Roberto Carlos Góngora Aduviri	Desarrollador QA Engineer

Tabla 3.1 Definición de Roles  
Fuente: Elaboración Propia

### 3.2.2 ANÁLISIS DE REQUERIMIENTOS

Como ya se manifestó anteriormente, la información pertinente para el correcto desarrollo de éste proyecto se obtuvo a través de las descripciones y asesoría continua del gerente de Desarrollo Ing. Mauricio Veliz (Scrum Master) y Team Leader de Quality Engineering Ing. Ademar Hurtado, que orientaron el desarrollo del mismo, para crear una herramienta acorde a las necesidades de Testing de Processmaker, además de mejorar la usabilidad y funcionalidad gracias a sus experiencias.

La información acerca de la lógica de negocio se consignó en las historias de usuario; la elaboración de estas se realizó a lo largo del desarrollo de las reuniones que integraron al Product Owner/Product Manager, Scrum Master y al equipo de Quality Engineering quienes son un grupo de desarrolladores/Ingenieros de Calidad que serán los primeros en interactuar con la herramienta y harán uso del Framework como clientes internos y directos del Framework de Automatización de Pruebas funcionales bajo el denominativo “Robotest” y gracias a la flexibilidad de poder unir otras herramientas de planificación con Scrum como lo es el **Roadmap** se fijó un lineamiento el cual se debe seguir para lograr cumplir nuestras expectativas.

#### a) Roadmap mas Scrum Backlog

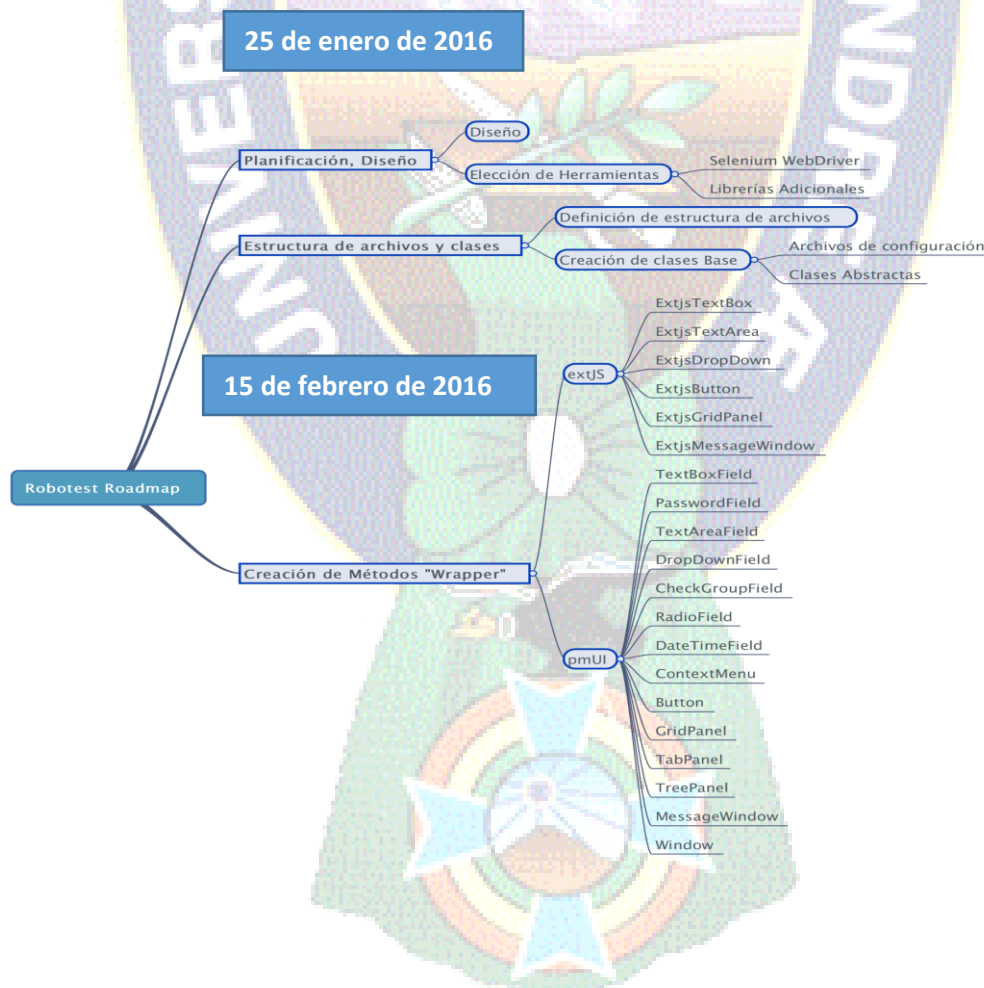
Se sabe que una acumulación de trabajo ordenado no completado (backlog), ayuda a entender que hacer a continuación, pero no siempre da a saber dónde está el proyecto y hacia dónde se dirige, especialmente si se acaba sumergido en un proyecto de gran envergadura con muchas historias y/o tareas creadas, como lo es el presente Framework, debido a la cantidad de módulos que convergen hacia un solo producto.

Para resolver estas situaciones, se usa la gestión del proyecto hoja de ruta (Roadmap) reflejado en un mapa de seguimiento y el backlog de scrum.

Roadmap ayudará a realizar un seguimiento cronológico y a organizar historias de usuario en un modelo útil para ayudar a entender la funcionalidad del producto, en este caso para el Framework de Automatización de pruebas funcionales, además de identificar los huecos y omisiones en el backlog, y planificar de manera efectiva lanzamientos de ciertas partes o en general de las versiones y mejoras del producto.

**b) ROADMAP (camino)**

En la reunión entre, scrum Master, product Owner que es encargado del equipo de Ingenieros de calidad es definido el lineamiento a seguir en el producto, la siguiente hoja de ruta muestra los alcances e intervalos de fechas entre tareas las cuales luego serán definidas como parte de los sprints



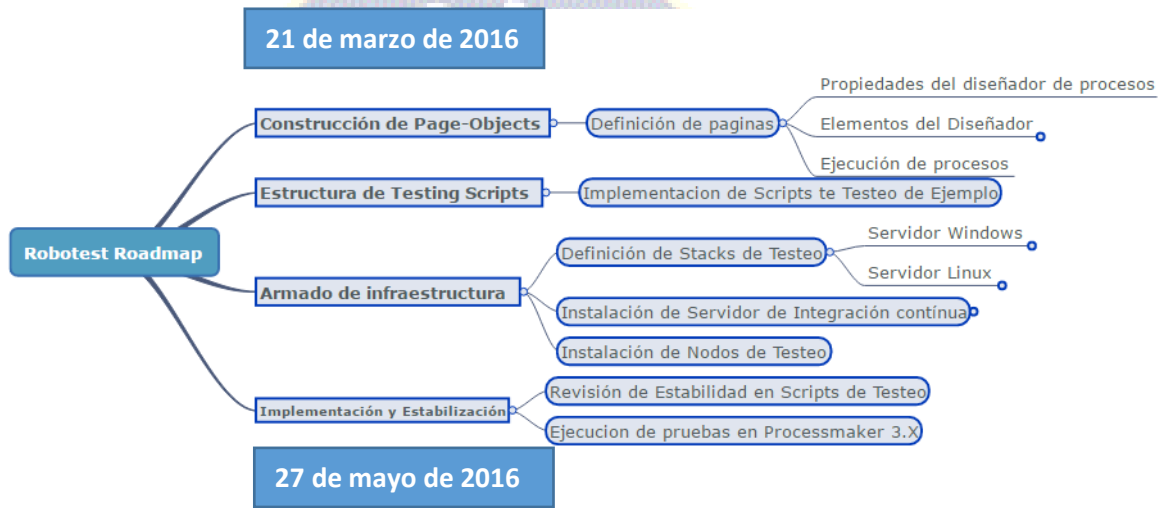


Figura 16.2 Roadmap Proyecto Framework de automatización de pruebas Funcionales  
Fuente: Elaboración Propia

c) **Matriz de Historias de Roadmap.** Se detalla matriz de historias según Roadmap:

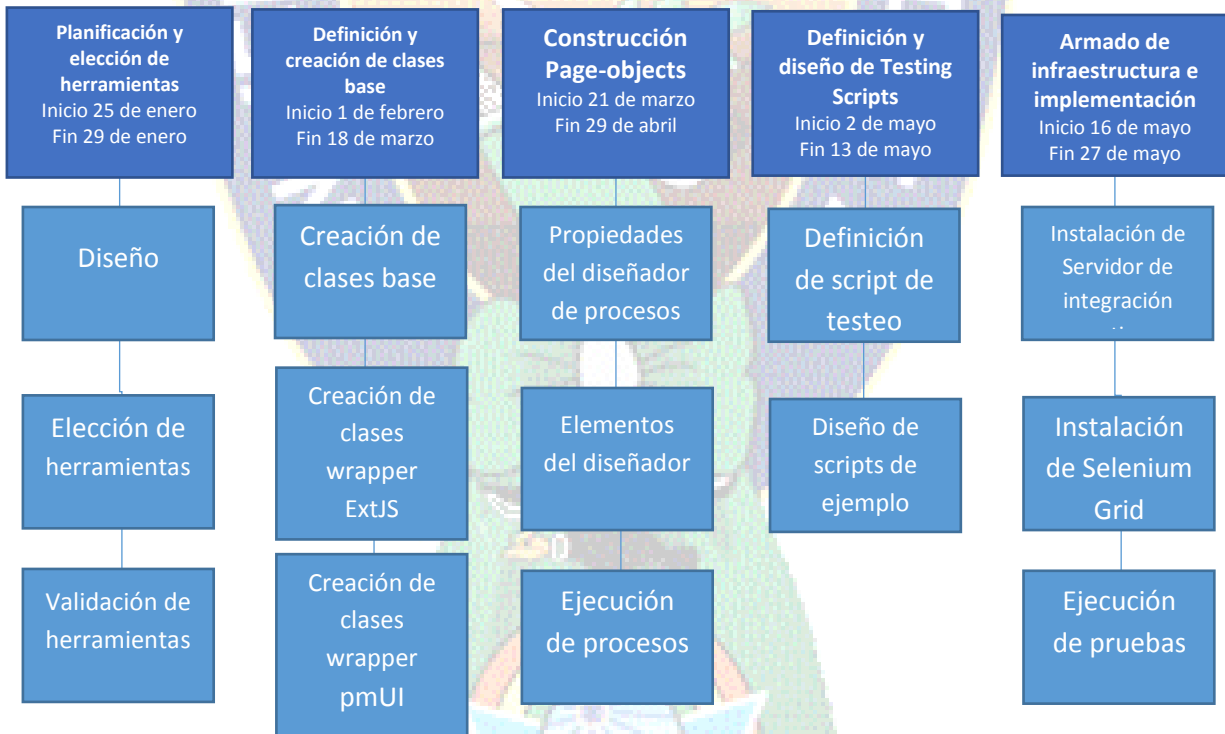


Figura 17.3 Matriz Roadmap para el desarrollo de software  
Fuente: Elaboración Propia

Horizontalmente, se encuentra el título para cada funcionalidad agrupados; verticalmente, las historias principales / cuestiones relacionadas con cada grupo.

Las funcionalidades se priorizan de izquierda a derecha (más importante) a derecha (menos importante). Cada grupo tendrá entonces las historias, la prioridad de las historias es verticalmente.

Aquí se encuentran historias sin la estimación, ideas simples ni confirmadas o mejoras técnicas de fallas y/o problemas que se quieren solucionar con la nueva versión. Lo importante es que ayude a comprender el estado del proyecto a través de un mapa ya establecido.

### 3.2.3 ELECCIÓN DEL ENTORNO DE DESARROLLO

**Selenium** es un conjunto de utilidades que facilita la labor de obtener juegos de pruebas para aplicaciones web. Para ello nos permite grabar, editar y depurar casos de prueba, que podrán ser ejecutados de forma automática e iterativa posteriormente.

Además de ser una herramienta para registrar acciones, permite editarlas manualmente o crearlas desde cero. Las acciones se basan en el uso de diferentes API's en diferentes lenguajes (PHP, Ruby, JAVA, Javascript, etc). Entre sus principales características podemos nombrar:

- Facilidad de registro y ejecución de los test.
- Referencia a objetos DOM en base al ID, nombre o a través de XPath.
- Auto-completado para todos los comandos.
- Las acciones pueden ser ejecutadas paso a paso.
- Herramientas de depuración y puntos de ruptura (breakpoints).
- Los test pueden ser almacenados en diferentes formatos.

El potencial de esta herramienta puede ser utilizado para la grabación de las pruebas funcionales durante la Generación de pruebas de regresión. Con este servicio se consigue obtener una batería de pruebas automatizadas que podrán ser utilizadas cuando sea necesario repetir las pruebas.

De la misma forma tras haber realizado una investigación acerca de entornos de desarrollo, se decidió optar por JAVA, las razones son las siguientes:

- **Lenguaje Simple:** Se lo conoce como lenguaje simple porque viene de la misma estructura de c y c++; ya que c++ fue un referente para la creación de java por eso utiliza determinadas características de c++ y se han eliminado otras.
- **Orientado a Objetos:** Toda la programación en java en su mayoría está orientada a objeto, ya que al estar agrupados estructuras encapsuladas es más fácil su manipulación.
- **Distribuido:** Permite abrir sockets, establecer y aceptar conexiones con los servidores o clientes remotos; facilita la creación de aplicaciones distribuidas ya que proporciona una colección de clases para aplicaciones en red.
- **Robusto:** Es altamente fiable en comparación con c, se han eliminado muchas características con la aritmética de punteros, proporciona numerosas comprobaciones en compilación y en tiempo de ejecución.
- **Seguro:** La seguridad es una característica muy importante en java ya que se han implementado barreras de seguridad en el lenguaje y en el sistema de ejecución de tiempo real.
- **Multiplataforma:** Java es compatible con cualquier sistema operativo que va desde Windows, las diferentes distribuciones de Linux y Mac.
- **Portable:** Por ser independiente de la arquitectura en que se ejecuta (SO), esto hace que su portabilidad sea muy eficiente, sus programas son iguales en cualquiera de las plataformas, esto se debe gracias a su máquina virtual.

- **Interpretado y compilado a la vez:** Java puede ser compilado e interpretado en tiempo real, ya que cuando se construye el código fuente este se transforma a código de máquina.
- **Multitarea:** Java tiene la facilidad de ejecutar varias funciones al mismo tiempo, gracias a su función de multihilos ya que por cada hilo que el programa puede ejecutar en tiempo real muchas funciones al mismo tiempo.
- **Dinámico:** El lenguaje Java es muy dinámico en la fase de enlazado, sus clases solamente actuarán en medida en que sean requeridas o necesitadas, con esto permitirá que los enlaces se puedan incluir incluso desde fuentes muy variadas o desde la red.
- **Applets:** En Java se pueden crear aplicaciones independientes y applets. Independientes porque se pueden comportar como cualquier programa escrito en cualquier lenguaje. Por otra parte, los applets considerados pequeños programas, tienen la capacidad de ejecutar funciones muy complejas o de forma segura.
- **Potente:** Java es considerado de alto rendimiento por ser tan veloz en el momento de correr los programas y por ahorrarse muchas líneas de código.

### 3.2.4 ESPECIFICACIÓN DE LAS HISTORIAS DE USUARIO

Durante las reuniones con el Product Owner y Scrum Master se trazó un lineamiento de objetivos en el **Roadmap** y con los usuarios del Framework (equipo de Quality Engineering de Processmaker), se establecieron las historias de usuario a las cuales se les asigna un número y un nombre con el requerimiento a implantar. El product Owner (Team Leader de Quality Engineering) asigna la prioridad, el riesgo y la dificultad es asignada por el desarrollador y por último se le asigna un número de iteración a resolver dependiendo de los valores anteriormente mencionados.

### 3.2.4.1 HISTORIA DE USUARIO: ESTRUCTURA DE ARCHIVOS Y CLASES

La siguiente tabla muestra la descripción de la historia de usuario para definir la estructura de archivos y clases del proyecto de Automatización:

HISTORIA DE USUARIO			
Numero : <b>001</b>	Nombre de historia de usuario:	Estructura de archivos y clases	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	MUY ALTA	Dificultad:	MEDIA
Riesgo :	MEDIA		
<p>Descripción:</p> <p>El Framework debe ser extensible, es decir, se toma en consideración futuro crecimiento, la estructura de archivos y directorios debe dar lugar a esto y debe ser fácil entender el funcionamiento de la herramienta.</p> <ul style="list-style-type: none"> <li>- Tarea: Crear archivo de configuración <b>pom.xml</b> el cual es la configuración del proyecto y obtiene todas las dependencias, plugins, fuentes, estructura de directorios, etc. del proyecto Maven.</li> <li>- Tarea: Crear archivo <b>.conf</b> el cual será fuente de datos que la ejecución de los Scripts de testeo necesitarán</li> </ul>			
<p>Observaciones: El archivo pom.xml responde a las siglas de Project Object Model, este archivo es la unidad principal de un Proyecto MAVEN.</p>			

Tabla 4 Historia de usuario 001 – Estructura de archivos y clases  
Fuente: Elaboración Propia

Descripción de tareas a realizar para la Historia de Usuario 001

#### a) TAREA 1: Crear archivo de configuración **pom.xml**

TARJETA DE TAREA RT- 1			
Crear archivo de configuración pom.xml			
Fecha:	Tipo de actividad:	✓ Nueva	Fijar Mejorar
Numero de Historia: 001	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora
Referencia Anterior :	Riesgo : Media	Técnicas de estimación: Poker Scrum	
<p><b>Descripción:</b></p> <p>El proyecto de Automatización Robotest debe tener un archivo de configuración pom.xml el cual contiene la configuración básica del proyecto MAVEN.</p>			



**Criterios de Aceptación:**

- Éste archivo deberá contener código XML válido y ejecutable.
- Deberá contener una sección de dependencias del proyecto.
- Deberá contener una sección información acerca del presente proyecto.

Tabla 5 Tarea 1 de historia de usuario 001 – Crear archivo de configuración pom.xml  
Fuente: Elaboración Propia

**b) TAREA 2: Crear archivo de configuración .conf****TARJETA DE TAREA RT-2****Crear archivo de configuración .conf**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 001	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Media	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b> El proyecto de Automatización Robotest debe tener un archivo de configuración, <b>conf</b> el cual contiene datos para la ejecución de Scripts de Testeo.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener configuración de la ejecución, tales como la plataforma en la que se ejecuta el test, navegador y versión del mismo</li> <li>- Deberá contener la configuración del Hub que ejecutará los nodos, cuando se implemente Selenium Grid.</li> </ul>				

Tabla 6 Tarea 2 de historia de usuario 001 – Crear archivo de configuración .conf  
Fuente: Elaboración Propia

### 3.2.4.2 HISTORIA DE USUARIO: CREACIÓN DE CLASES BASE Y ABSTRACTAS

Las siguientes tablas muestran la descripción de la historia de usuario para definir la estructura de archivos y clases del proyecto de Automatización:

HISTORIA DE USUARIO		
Numero : <b>002</b>	Nombre de historia de usuario:	Creación de Clases Base y Abstractas
Fecha :		Desarrollador: Roberto Góngora

Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	MUY ALTA	Dificultad:	ALTA
Riesgo :	MUY ALTA		
<b>Descripción:</b> Las Clases base y Abstractas son los pilares del Framework, todas las clases y funcionalidades se generarán a partir de estas clases, por lo tanto, deben tener funciones que ayuden no solamente al desarrollo de futuras características sino también al desarrollo de Scripts por parte del Ingeniero de Calidad. <ul style="list-style-type: none"> <li>- Tarea: Crear clase de Invocación de Driver para los navegadores “BrowserInstance”.</li> <li>- Tarea: Crear clase de configuración de driver para los navegadores “BrowserSettigns”.</li> <li>- Tarea: Crear clase para eventos de espera “WaitTool”.</li> <li>- Tarea: Crear clase abstracta de página “PAGE”.</li> </ul>			
Observaciones: Ninguna.			

Tabla 7 Historia de usuario 002 – Creación de clases base y abstractas  
Fuente: Elaboración Propia

Descripción de tareas a realizar para la Historia de Usuario 002

a) TAREA 1: Crear clase de invocación de Driver “BrowserInstance”.

TARJETA DE TAREA RT-3					
Crear clase de invocación de Driver “BrowserInstance”					
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar	
Numero de Historia: 002	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora		
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum			
<b>Descripción:</b> El proyecto de Automatización Robotest debe tener una clase “BrowserInstance” la cual hace la invocación al driver del navegador para ejecutar los scripts de testeo.					
<b>Criterios de Aceptación:</b> <ul style="list-style-type: none"> <li>- Deberá contener métodos que ejecuten acciones sobre el navegador, tales como, Cerrar el navegador, abrir una URL específica, etc.</li> </ul>					

Tabla 8 Tarea 1 de historia de usuario 002 – Crear clase de invocación de Driver  
Fuente: Elaboración Propia

b) TAREA 2: Crear clase de configuración de Driver “BrowserSettings”.

TARJETA DE TAREA RT-4					
Crear clase de configuración de Driver “BrowserSettings”					
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar	
Numero de Historia:	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto		

002	Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum
<b>Descripción:</b>		
El proyecto de Automatización Robotest debe tener una clase “BrowserSettings” la cual recupera la configuración del Framework para ejecutar los scripts de testeo.		
<b>Criterios de Aceptación:</b>		
<ul style="list-style-type: none"> <li>- Deberá contener un método para recuperar las configuraciones del Framework y mandar al driver del navegador para su ejecución.</li> </ul>		

Tabla 9 Tarea 2 de historia de usuario 002 – Crear clase de configuración de Driver  
Fuente: Elaboración Propia

c) TAREA 3: Crear clase para eventos de espera “WaitTool”

TARJETA DE TAREA RT-5				
Crear clase para eventos de espera “WaitTool”				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 002	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
El proyecto de Automatización Robotest debe tener una clase “WaitTool” la cual se encarga de ejecutar eventos de espera cuando se ejecuta un navegador. Específicamente espera a que los elementos de una página estén presentes antes de ejecutar funciones sobre los mismos.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener métodos que comprueben la existencia de un objeto antes de ejecutar alguna acción sobre los mismos.</li> </ul>				

Tabla 10 Tarea 3 de historia de usuario 002 – Crear clase para eventos de espera “WaitTool”  
Fuente: Elaboración Propia

d) TAREA 4: Crear clase abstracta de página “Page”.

TARJETA DE TAREA RT-6				
Crear clase abstracta de página “Page”				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 002	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
El proyecto de Automatización Robotest debe tener una clase “Page” la cual es una clase abstracta que será la base de todas las páginas generadas en el Framework				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener métodos y atributos que sea de común uso en todas las páginas del Framework en desarrollo.</li> </ul>				

Tabla 11 Tarea 4 de historia de usuario 002 – Crear clase abstracta de página “Page”  
Fuente: Elaboración Propia

### 3.2.4.3 HISTORIA DE USUARIO: CREACIÓN DE CLASES “WRAPPER” EXTJS

Las clases “wrapper” son clases que contienen métodos que permiten tratar a todos los elementos creados con la librería Extjs (que son parte de Processmaker) como Objetos dentro del Framework, a continuación se detalla en las siguientes tablas la descripción de la historia de usuario que define la creación de estos métodos:

HISTORIA DE USUARIO			
Numero : <b>003</b>	Nombre de historia de usuario:	Creación de clases “Wrapper” ExtJS	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	ALTA	Dificultad:	ALTA
Riesgo :	ALTA		
<b>Descripción:</b>			
Las clases “Wrapper” ExtJS, son clases que permiten tratar a los elementos ExtJS como objetos, éstos objetos son necesarios para crear las páginas usando el Patrón de Objeto-Página.			
Los elementos ExtJS son parte del diseñador de procesos en Processmaker y de la misma forma componen lo que es la sección de ejecución de procesos.			
<ul style="list-style-type: none"> <li>- Tarea: Crear clase “wrapper” ExtjsTextBox</li> <li>- Tarea: Crear clase “wrapper” ExtjsTextArea</li> <li>- Tarea: Crear clase “wrapper” ExtjsDropDown</li> <li>- Tarea: Crear clase “wrapper” ExtjsButton</li> <li>- Tarea: Crear clase “wrapper” ExtjsGridPanel</li> <li>- Tarea: Crear clase “wrapper” ExtjsMessageWindow</li> </ul>			
Observaciones: Ninguna.			

Tabla 12 Historia de usuario 003 - Creación de clases “Wrapper” ExtJS  
Fuente: Elaboración Propia

Descripción de tareas a realizar para la Historia de Usuario 003:

a) TAREA 1: Crear clase “wrapper” ExtjsTextBox

TARJETA DE TAREA RT-7					
Crear clase “wrapper” ExtjsTextBox					
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar	
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora		
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum			
<b>Descripción:</b>					

Se necesita crear una clase de tipo “wrapper” para el campo de tipo ExtjsTextBox creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo TextBox usando el ID.
- Deberá contener un método para obtener el campo Textbox usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 13 Tarea 1 de historia de usuario 003 – Crear clase “wrapper” ExtjsTextBox  
Fuente: Elaboración Propia

b) TAREA 2: Crear clase “wrapper” ExtjsTextArea

**TARJETA DE TAREA RT-8**

**Crear clase “wrapper” ExtjsTextArea**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo ExtjsTextArea creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo TextArea usando el ID.
- Deberá contener un método para obtener el campo TextArea usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 142 Tarea 2 de historia de usuario 003 – Crear clase “wrapper” ExtjsTextArea  
Fuente: Elaboración Propia

c) TAREA 3: Crear clase “wrapper” ExtjsDropDown

**TARJETA DE TAREA RT-9**

**Crear clase “wrapper” ExtjsDropDown**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo ExtjsDropDown creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo DropDown usando el ID.
- Deberá contener un método para obtener el campo DropDown usando el WebElement.
- Deberá contener un método para seleccionar un valor del campo.
- Deberá contener un método para recuperar el valor seleccionado del campo.
- Deberá contener un método para retornar el número de opciones del DropDown.

Tabla 153 Tarea 3 de historia de usuario 003 – Crear clase “wrapper” ExtjsDropDown  
Fuente: Elaboración Propia

#### d) TAREA 4: Crear clase “wrapper” ExtjsButton

TARJETA DE TAREA RT-10				
Crear clase “wrapper” ExtjsButton				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase de tipo “wrapper” para el elemento de tipo ExtjsButton creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método para obtener el elemento button usando el ID.</li> <li>- Deberá contener un método para obtener el elemento button usando el WebElement.</li> <li>- Deberá contener un método para hacer “click” sobre el elemento.</li> <li>- Deberá contener un método para recuperar el texto que contiene el elemento button.</li> </ul>				

Tabla 164 Tarea 4 de historia de usuario 003 – Crear clase “wrapper” ExtjsButton  
Fuente: Elaboración Propia

#### e) TAREA 5: Crear clase “wrapper” ExtjsGridPanel

TARJETA DE TAREA RT-11				
Crear clase “wrapper” ExtjsGridPanel				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase de tipo “wrapper” para el elemento de tipo ExtjsGridPanel creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método para obtener el elemento GridPanel usando el ID.</li> <li>- Deberá contener un método para obtener el elemento GridPanel usando el WebElement.</li> <li>- Deberá contener un método para seleccionar un ítem del elemento GridPanel</li> </ul>				

- Deberá contener un método para hacer “click” sobre un ítem del elemento GridPanel.
- Deberá contener un método para retornar el número de ítems que contiene el GridPanel.
- Deberá contener un método para obtener el número de páginas que tiene el GridPanel.
- Deberá contener un método para cambiar la página actual del GridPanel.

Tabla 175 Tarea 5 de historia de usuario 003 – Crear clase “wrapper” ExtjsGridPanel  
Fuente: Elaboración Propia

f) TAREA 6: Crear clase “wrapper” ExtjsMessageWindow

**TARJETA DE TAREA RT-12**

**Crear clase “wrapper” ExtjsMessageWindow**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 003	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo MessageWindow creado con la biblioteca Javascript Extjs e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento MessageWindow usando el ID.
- Deberá contener un método para obtener el elemento MessageWindow usando el WebElement.
- Deberá contener un método para recuperar el texto del elemento MessageWindow.
- Deberá contener un método para recuperar el número de botones que contiene el elemento MessageWindow.
- Deberá contener un método para hacer “click” sobre los botones del elemento MessageWindow.

Tabla 186 Tarea 6 de historia de usuario 003 – Crear clase “wrapper” ExtjsMessageWindow  
Fuente: Elaboración Propia

### 3.2.4.4 HISTORIA DE USUARIO: CREACIÓN DE CLASES “WRAPPER” PMUI

PMUI es una biblioteca de componentes creado exclusivamente para el uso en la herramienta Processmaker, al igual que ExtJS se caracteriza por contar con varios elementos, campos y widgets que se implementan en las páginas de la interfaz, por lo tanto también son tomadas en cuenta para generar clases “wrapper”, ya que se necesita tratar a los elementos como objetos dentro del presente Framework de Pruebas

automatizadas, a continuación se detalla en las siguientes tablas la descripción de la historia de usuario y tareas que definen la creación de estos métodos.

HISTORIA DE USUARIO			
Numero : <b>004</b>	Nombre de historia de usuario:	Creación de clases “Wrapper” pmUI	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	MUY ALTA	Dificultad:	ALTA
Riesgo :	ALTA		
<b>Descripción:</b>			
Las clases “Wrapper” pmUI, son clases que permiten tratar a los elementos pmUI como objetos, éstos objetos son necesarios para crear las páginas usando el Patrón de Objeto-Página.			
Los elementos pmUI son parte del diseñador de procesos en Processmaker y de la misma forma componen lo que es la sección de ejecución de procesos.			
<ul style="list-style-type: none"> <li>- Tarea: Crear clase “wrapper” TextBoxField</li> <li>- Tarea: Crear clase “wrapper” PasswordField</li> <li>- Tarea: Crear clase “wrapper” TextAreaField</li> <li>- Tarea: Crear clase “wrapper” DropDownField</li> <li>- Tarea: Crear clase “wrapper” CheckGroupField</li> <li>- Tarea: Crear clase “wrapper” RadioField</li> <li>- Tarea: Crear clase “wrapper” DateTimeField</li> <li>- Tarea: Crear clase “wrapper” ContextMenu</li> <li>- Tarea: Crear clase “wrapper” Button</li> <li>- Tarea: Crear clase “wrapper” GridPanel</li> <li>- Tarea: Crear clase “wrapper” TabPanel</li> <li>- Tarea: Crear clase “wrapper” TreePanel</li> <li>- Tarea: Crear clase “wrapper” MessageWindow</li> <li>- Tarea: Crear clase “wrapper” Window</li> </ul>			
Observaciones: Estos elementos no son todos los elementos PMUI, solo se menciona los que se va interactuar con el diseñador de procesos.			

Tabla 19 Historia de usuario 004 - Creación de clases “Wrapper” pmUI  
Fuente: Elaboración Propia

a) TAREA 1: Crear clase “wrapper” TextBoxField

TARJETA DE TAREA RT-13					
Crear clase “wrapper” TextBoxField					
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar	
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora		
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum			



**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo TextBoxField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo TextBox usando el ID.
- Deberá contener un método para obtener el campo Textbox usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 20 Tarea 1 de historia de usuario 004 – Crear clase “wrapper” TextBoxField  
Fuente: Elaboración Propia

## b) TAREA 2: Crear clase “wrapper” PasswordField

**TARJETA DE TAREA RT-14****Crear clase “wrapper” PasswordField**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo PasswordField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo Password usando el ID.
- Deberá contener un método para obtener el campo Password usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 21 Tarea 2 de historia de usuario 004 – Crear clase “wrapper” PasswordField  
Fuente: Elaboración Propia

## c) TAREA 3: Crear clase “wrapper” TextAreaField

**TARJETA DE TAREA RT-15****Crear clase “wrapper” TextAreaField**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo TextAreaField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo TextArea usando el ID.
- Deberá contener un método para obtener el campo TextArea usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 22 Tarea 3 de historia de usuario 004 – Crear clase “wrapper” TextField  
Fuente: Elaboración Propia

## d) TAREA 4: Crear clase “wrapper” DropDownField

**TARJETA DE TAREA RT-16****Crear clase “wrapper” DropDownField**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo DropDownField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo DropDown usando el ID.
- Deberá contener un método para obtener el campo DropDown usando el WebElement.
- Deberá contener un método para seleccionar un valor del campo.
- Deberá contener un método para recuperar el valor seleccionado del campo.
- Deberá contener un método para retornar el número de opciones del DropDown.

Tabla 23 Tarea 4 de historia de usuario 004 – Crear clase “wrapper” DropDownField  
Fuente: Elaboración Propia

## e) TAREA 5: Crear clase “wrapper” CheckGroupField

**TARJETA DE TAREA RT-17****Crear clase “wrapper” CheckGroupField**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el campo de tipo CheckGroupField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo CheckGroup usando el ID.
- Deberá contener un método para obtener el campo CheckGroup usando el WebElement.
- Deberá contener un método para seleccionar un valor del campo.
- Deberá contener un método para dejar de seleccionar un valor del campo.
- Deberá contener un método para recuperar el valor seleccionado del campo.
- Deberá contener un método para retornar el número de opciones del CheckGroup.

Tabla 24 Tarea 5 de historia de usuario 004 – Crear clase “wrapper” CheckGroupField  
Fuente: Elaboración Propia

f) TAREA 6: Crear clase “wrapper” RadioField

TARJETA DE TAREA RT-18				
Crear clase “wrapper” RadioField				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase de tipo “wrapper” para el campo de tipo RadioField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método para obtener el campo Radio usando el ID.</li> <li>- Deberá contener un método para obtener el campo Radio usando el WebElement.</li> <li>- Deberá contener un método para seleccionar un valor del campo.</li> <li>- Deberá contener un método para recuperar el valor seleccionado del campo.</li> <li>- Deberá contener un método para retornar el número de opciones del RadioField.</li> </ul>				

Tabla 25 Tarea 6 de historia de usuario 004 – Crear clase “wrapper” RadioField  
Fuente: Elaboración Propia

g) TAREA 7: Crear clase “wrapper” DateTimeField

TARJETA DE TAREA RT-19				
Crear clase “wrapper” DateTimeField				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase de tipo “wrapper” para el campo de tipo DateTimeField creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.				

**Criterios de Aceptación:**

- Deberá contener un método para obtener el campo DateTime usando el ID.
- Deberá contener un método para obtener el campo DateTime usando el WebElement.
- Deberá contener un método para establecer el valor del campo.
- Deberá contener un método para recuperar el valor del campo.

Tabla 26 Tarea 7 de historia de usuario 004 – Crear clase “wrapper” DateTimeField  
Fuente: Elaboración Propia

## h) TAREA 8: Crear clase “wrapper” ContextMenu

**TARJETA DE TAREA RT-20****Crear clase “wrapper” ContextMenu**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo ContextMenu creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento ContextMenu usando el ID.
- Deberá contener un método para obtener el elemento ContextMenu usando el WebElement.
- Deberá contener un método para seleccionar un ítem del elemento ContextMenu
- Deberá contener un método para recuperar el texto del ítem seleccionado del elemento ContextMenu

Tabla 27 Tarea 8 de historia de usuario 004 – Crear clase “wrapper” ContextMenu  
Fuente: Elaboración Propia

## i) TAREA 9: Crear clase “wrapper” Button

**TARJETA DE TAREA RT-21****Crear clase “wrapper” Button**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo Button creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento button usando el ID.
- Deberá contener un método para obtener el elemento button usando el WebElement.
- Deberá contener un método para hacer “click” sobre el elemento.
- Deberá contener un método para recuperar el texto que contiene el elemento button.

Tabla 28 Tarea 9 de historia de usuario 004 – Crear clase “wrapper” Button  
Fuente: Elaboración Propia

## j) TAREA 10: Crear clase “wrapper” GridPanel

**TARJETA DE TAREA RT-22****Crear clase “wrapper” GridPanel**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo GridPanel creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento GridPanel usando el ID.
- Deberá contener un método para obtener el elemento GridPanel usando el WebElement.
- Deberá contener un método para seleccionar un ítem del elemento GridPanel
- Deberá contener un método para hacer “click” sobre un ítem del elemento GridPanel.
- Deberá contener un método para retornar el número de ítems que contiene el GridPanel.
- Deberá contener un método para obtener el número de páginas que tiene el GridPanel.
- Deberá contener un método para cambiar la página actual del GridPanel.

Tabla 29 Tarea 10 de historia de usuario 004 – Crear clase “wrapper” GridPanel  
Fuente: Elaboración Propia

## k) TAREA 11: Crear clase “wrapper” TabPanel

**TARJETA DE TAREA RT-23****Crear clase “wrapper” TabPanel**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo TabPanel creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento TabPanel usando el ID.
- Deberá contener un método para obtener el elemento TabPanel usando el WebElement.
- Deberá contener un método para seleccionar un ítem del elemento TabPanel
- Deberá contener un método para hacer “click” sobre un ítem del elemento TabPanel.
- Deberá contener un método para retornar el número de ítems que contiene el TabPanel.

Tabla 30 Tarea 11 de historia de usuario 004 – Crear clase “wrapper” TabPanel  
Fuente: Elaboración Propia

l) TAREA 12: Crear clase “wrapper” TreePanel

**TARJETA DE TAREA RT-24**

**Crear clase “wrapper” TreePanel**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo TreePanel creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento TreePanel usando el ID.
- Deberá contener un método para obtener el elemento TreePanel usando el WebElement.
- Deberá contener un método para seleccionar un ítem del elemento TreePanel
- Deberá contener un método para hacer “click” sobre un ítem del elemento TreePanel.
- Deberá contener un método para hacer “Double-click” sobre un ítem del elemento TreePanel.
- Deberá contener un método para retornar el número de ítems que contiene el TreePanel.

Tabla 31 Tarea 12 de historia de usuario 004 – Crear clase “wrapper” TreePanel  
Fuente: Elaboración Propia

m) TAREA 13: Crear clase “wrapper” MessageWindow

**TARJETA DE TAREA RT-25**

**Crear clase “wrapper” MessageWindow**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo MessageWindow creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución

Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento MessageWindow usando el ID.
- Deberá contener un método para obtener el elemento MessageWindow usando el WebElement.
- Deberá contener un método para recuperar el texto del elemento MessageWindow.
- Deberá contener un método para recuperar el número de botones que contiene el elemento MessageWindow.
- Deberá contener un método para hacer “click” sobre los botones del elemento MessageWindow.

Tabla 32 Tarea 13 de historia de usuario 004 – Crear clase “wrapper” MessageWindow  
Fuente: Elaboración Propia

n) TAREA 14: Crear clase “wrapper” Window

**TARJETA DE TAREA RT-26**

**Crear clase “wrapper” Window**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 004	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear una clase de tipo “wrapper” para el elemento de tipo Window creado con la biblioteca Javascript pmUI e implementado en el diseñador de procesos y ejecución Processmaker.

**Criterios de Aceptación:**

- Deberá contener un método para obtener el elemento Window usando el ID.
- Deberá contener un método para obtener el elemento Window usando el WebElement.
- Deberá contener un método para recuperar el texto de la sección “Header” del elemento Window.
- Deberá contener un método para recuperar el número de botones que contiene el elemento Window.
- Deberá contener un método para hacer “click” sobre los botones del elemento Window.

Tabla 33 Tarea 14 de historia de usuario 004 – Crear clase “wrapper” Window  
Fuente: Elaboración Propia

### 3.2.4.5 HISTORIA DE USUARIO: CONSTRUCCIÓN DE PAGE-OBJECTS PARA DISEÑADOR DE PROCESOS

Una vez creadas todas las clases “wrappers” con sus respectivos métodos, se tiene a continuación la tarea de construir las páginas usando el Patrón de diseño Page Object

Pattern el cual hace uso de todas las clases “wrappers” y genera páginas que serán usadas más adelante en el diseño de Scripts de Testeo. A continuación, se detalla en las siguientes tablas la descripción de la historia de usuario que define la construcción de estas páginas.

HISTORIA DE USUARIO			
Numero : <b>005</b>	Nombre de historia de usuario:	Construcción de Page-objects para diseñador de Procesos	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	MUY ALTA	Dificultad:	ALTA
Riesgo :	ALTA		
<b>Descripción:</b>			
<p>Las clases que son desarrolladas bajo el patrón de diseño “Page-object”, reciben el nombre de página-objetos y son imprescindibles ya que detallan todos los elementos que contienen cada una de las páginas de la herramienta Processmaker.</p> <p>Estas páginas contienen objetos los cuales no son nada más que todas las clases “wrapper” diseñadas, para esto se necesita estructurar y abstraer las páginas del diseñador de procesos.</p> <p>Se necesita agrupar en paquetes(packages) ya que cada una de las páginas de Processmaker necesitan más de una clase para poder abstraer su funcionalidad.</p> <ul style="list-style-type: none"> <li>- Tarea: Crear clase ProcessDesignerBPMN</li> <li>- Tarea: Crear paquete VariableBPMN</li> <li>- Tarea: Crear paquete MessageTypeBPMN</li> <li>- Tarea: Crear paquete DynaformBPMN</li> <li>- Tarea: Crear paquete InputDocBPMN</li> <li>- Tarea: Crear paquete OutputDocBPMN</li> <li>- Tarea: Crear paquete TriggerBPMN</li> <li>- Tarea: Crear paquete ReportTableBPMN</li> <li>- Tarea: Crear paquete DatabaseConnectionBPMN</li> <li>- Tarea: Crear paquete TemplateBPMN</li> <li>- Tarea: Crear paquete PublicFileBPMN</li> <li>- Tarea: Crear paquete PermissionBPMN</li> <li>- Tarea: Crear paquete SupervisorBPMN</li> </ul>			
<p>Observaciones: Contempla todos los elementos de un proceso al momento de Diseñar Processmaker, se excluye CaseTracker ya que este elemento usa otra interfaz ajena al Diseñador de procesos.</p>			

Tabla 34 Historia de usuario 005 – Construcción de Page-objects para Diseñador de Procesos  
Fuente: Elaboración Propia



a) TAREA 1: Crear clase ProcessDesignerBPMN

TARJETA DE TAREA RT-27				
Crear clase ProcessDesignerBPMN				
Fecha:	Tipo de actividad:	<input checked="" type="checkbox"/> Nueva	<input type="checkbox"/> Fijar	<input type="checkbox"/> Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase ProcessDesignerBPMN que contenga todas las propiedades y acciones automatizadas del diseñador de procesos, desde el Dibujado de procesos hasta la configuración del mismo. Se usarán clases “wrapper” de tipo pmUI.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener un método de dibujado y construcción de procesos usando los elementos básicos y avanzados de diseño de procesos BPMN.</li> <li>- Deberá contener un método que verifique la correcta creación de elementos.</li> <li>- Deberá contener un método que realice la configuración automatizada de propiedades de proceso.</li> <li>- Deberá contener un método que verifique la configuración realizada.</li> </ul>				

Tabla 35 Tarea 1 de historia de usuario 005 – Crear clase ProcessDesignerBPMN  
Fuente: Elaboración Propia

b) TAREA 2: Crear paquete VariableBPMN

TARJETA DE TAREA RT-28				
Crear paquete VariableBPMN				
Fecha:	Tipo de actividad:	<input checked="" type="checkbox"/> Nueva	<input type="checkbox"/> Fijar	<input type="checkbox"/> Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Variables”, que contenga las propiedades y acciones automatizadas de las Variables sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Variables, tales como la</li> </ul>				

- creación, edición, y eliminación.
- Deberá contener un método que verifique la correcta creación de Variables.
- Deberá contener un método que devuelva la cantidad de variables creadas para el proceso.
- Deberá contener un método que devuelva las propiedades de una variable creada.

Tabla 36 Tarea 2 de historia de usuario 005 – Crear paquete VariableBPMN  
Fuente: Elaboración Propia

c) TAREA 3: Crear paquete MessageTypeBPMN

TARJETA DE TAREA RT-29				
Crear paquete MessageTypeBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN "MessageTypes", que contenga las propiedades y acciones automatizadas de los Message Types sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de MessageTypes, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de MessageTypes.</li> <li>- Deberá contener un método que devuelva la cantidad de MessageTypes creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un MessageType creado.</li> </ul>				

Tabla 37 Tarea 3 de historia de usuario 005 – Crear paquete MessageTypeBPMN  
Fuente: Elaboración Propia

d) TAREA 4: Crear paquete DynaformBPMN

TARJETA DE TAREA RT-30				
Crear paquete DynaformBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				

Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Dynaforms”, que contenga las propiedades y acciones automatizadas de los formularios sobre los procesos BPMN.

**Criterios de Aceptación:**

- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.
- Deberá contener métodos de administración automatizada de Dynaforms, tales como la creación, edición, y eliminación.
- Deberá contener un método que verifique la correcta creación de Dynaforms.
- Deberá contener un método que devuelva la cantidad de Dynaforms creados para el proceso.
- Deberá contener un método que devuelva las propiedades de un Dynaform creado.

Tabla 38 Tarea 4 de historia de usuario 005 – Crear paquete DynaformBPMN  
Fuente: Elaboración Propia

e) TAREA 5: Crear paquete InputDocBPMN

TARJETA DE TAREA RT-31				
Crear paquete InputDocBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Input Documents”, que contenga las propiedades y acciones automatizadas de los documentos de entrada sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Input Documents, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Input Documents.</li> <li>- Deberá contener un método que devuelva la cantidad de Input Documents creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un Input Document creado.</li> </ul>				

Tabla 39 Tarea 5 de historia de usuario 005 – Crear paquete InputDocBPMN  
Fuente: Elaboración Propia

f) TAREA 6: Crear paquete OutputDocBPMN

TARJETA DE TAREA RT-32				
Crear paquete OutputDocBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Output Documents”, que contenga las propiedades y acciones automatizadas de los documentos de salida sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Output Documents, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Output Documents.</li> <li>- Deberá contener un método que devuelva la cantidad de Output Documents creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un Output Document creado.</li> </ul>				

Tabla 40 Tarea 6 de historia de usuario 005 – Crear paquete OutputDocBPMN  
Fuente: Elaboración Propia

g) TAREA 7: Crear paquete TriggerBPMN

TARJETA DE TAREA RT-33				
Crear paquete TriggerBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Triggers”, que contenga las propiedades y acciones automatizadas de los Triggers sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Triggers, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Triggers.</li> </ul>				

- Deberá contener un método que devuelva la cantidad de Triggers creados para el proceso.
- Deberá contener un método que devuelva las propiedades de un Trigger creado.

Tabla 41 Tarea 7 de historia de usuario 005 – Crear paquete TriggerBPMN  
Fuente: Elaboración Propia

#### h) TAREA 8: Crear paquete ReportTableBPMN

TARJETA DE TAREA RT-34				
Crear paquete ReportTableBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Report Tables”, que contenga las propiedades y acciones automatizadas de las Tablas de Reporte sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Report Tables, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Report Tables.</li> <li>- Deberá contener un método que devuelva la cantidad de Report Tables creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un Report Table creado.</li> </ul>				

Tabla 42 Tarea 8 de historia de usuario 005 – Crear paquete ReportTableBPMN  
Fuente: Elaboración Propia

#### i) TAREA 9: Crear paquete DatabaseConnectionBPMN

TARJETA DE TAREA RT-35				
Crear paquete DatabaseConnectionBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Database Connections”, que contenga las propiedades y acciones automatizadas de las conexiones a base de				

datos sobre los procesos BPMN.

**Criterios de Aceptación:**

- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.
- Deberá contener métodos de administración automatizada de DatabaseConnections, tales como la creación, edición, y eliminación.
- Deberá contener un método que verifique la correcta creación de DatabaseConnections.
- Deberá contener un método que devuelva la cantidad de DatabaseConnections creados para el proceso.
- Deberá contener un método que devuelva las propiedades de un Database Connection creado.

Tabla 43 Tarea 9 de historia de usuario 005 – Crear paquete DatabaseConnectionBPMN  
Fuente: Elaboración Propia

j) TAREA 10: Crear paquete TemplateBPMN

**TARJETA DE TAREA RT-36**

**Crear paquete TemplateBPMN**

Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		

**Descripción:**

Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Templates”, que contenga las propiedades y acciones automatizadas de las plantillas sobre los procesos BPMN.

**Criterios de Aceptación:**

- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.
- Deberá contener métodos de administración automatizada de Templates, tales como la creación, edición, y eliminación.
- Deberá contener un método que verifique la correcta creación de Templates.
- Deberá contener un método que devuelva la cantidad de Templates creados para el proceso.
- Deberá contener un método que devuelva las propiedades de un Template creado.

Tabla 44 Tarea 10 de historia de usuario 005 – Crear paquete TemplateBPMN  
Fuente: Elaboración Propia

k) TAREA 11: Crear paquete PublicFileBPMN

TARJETA DE TAREA RT-37				
Crear paquete PublicFileBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN "Public Files", que contenga las propiedades y acciones automatizadas de los Public Files sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Public Files, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Public Files.</li> <li>- Deberá contener un método que devuelva la cantidad de Public Files creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un Public File creado.</li> </ul>				

Tabla 45 Tarea 11 de historia de usuario 005 – Crear paquete PublicFileBPMN  
Fuente: Elaboración Propia

l) TAREA 12: Crear paquete PermissionBPMN

TARJETA DE TAREA RT-38				
Crear paquete PermissionBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN "Permissions", que contenga las propiedades y acciones automatizadas de los permisos sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Permissions, tales como la creación, edición, y eliminación.</li> </ul>				

- Deberá contener un método que verifique la correcta creación de Permissions.
- Deberá contener un método que devuelva la cantidad de Permissions creados para el proceso.
- Deberá contener un método que devuelva las propiedades de un Permission creado.

Tabla 46 Tarea 12 de historia de usuario 005 – Crear paquete PermissionBPMN  
Fuente: Elaboración Propia

m) TAREA 13: Crear paquete SupervisorBPMN

TARJETA DE TAREA RT-39				
Crear paquete SupervisorBPMN				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 005	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear un paquete de clases relacionadas al elemento de un proceso BPMN “Supervisors”, que contenga las propiedades y acciones automatizadas de los supervisores sobre los procesos BPMN.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener un método de verificación de página, el cual verifica que todos los elementos estén presentes al momento de abrir la página de diseñador de procesos en el navegador.</li> <li>- Deberá contener métodos de administración automatizada de Supervisors, tales como la creación, edición, y eliminación.</li> <li>- Deberá contener un método que verifique la correcta creación de Supervisors.</li> <li>- Deberá contener un método que devuelva la cantidad de Supervisors creados para el proceso.</li> <li>- Deberá contener un método que devuelva las propiedades de un Supervisor creado.</li> </ul>				

Tabla 47 Tarea 13 de historia de usuario 005 – Crear paquete SupervisorBPMN  
Fuente: Elaboración Propia

### 3.2.4.6 HISTORIA DE USUARIO: ESTRUCTURA DE TESTING SCRIPTS

Los Testing Scripts, son clases desarrolladas con el propósito de aplicar herramientas del Framework para ejecutar pruebas funcionales, de este modo se necesita estructurar y formalizar el armado de estas clases las cuales usan Junit para la validación



de aserciones. A continuación, se detalla en las siguientes tablas la descripción de la historia de usuario que define el armado de estas clases.

HISTORIA DE USUARIO			
Numero : <b>006</b>	Nombre de historia de usuario:	Estructura de Testing Scripts	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	ALTA	Dificultad:	MEDIA
Riesgo :	ALTA		
<b>Descripción:</b>			
Los Scripts de testeo son el objetivo principal del Framework de Automatización de pruebas funcionales, son clases que devuelven resultados. Se necesita desarrollar Testing Scripts funcionales que muestren el armado correcto y la forma de obtener resultados usando JUNIT.			
<ul style="list-style-type: none"> <li>- Tarea: Crear clase BPMNDrawElements</li> <li>- Tarea: Crear clase BPMNProcessProperties</li> <li>- Tarea: Crear clase AssignmentRules</li> <li>- Tarea: Crear clase BPMNDynaformManagement</li> <li>- Tarea: Crear clase InputDocumentManegement</li> <li>- Tarea: Crear clase TriggersManagement</li> <li>- Tarea: Crear clase DatabaseConnectionManagement</li> </ul>			
Observaciones: Estas clases de prueba muestran la forma correcta de armar un Script de Testeo, por lo tanto se consideran suficientes los ejemplos generados en esta historia de usuario.			

Tabla 48 Historia de usuario 006 – Diseño y estructura de Testing Scripts  
Fuente: Elaboración Propia

a) TAREA 1: Crear clase BPMNDrawElements

TARJETA DE TAREA RT-40					
Crear clase BPMNDrawElements					
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar	
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora		
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum			
<b>Descripción:</b>					
Se necesita crear una clase BPMNDrawElements que contenga el correcto armado de un Script de testeo orientado al dibujado de procesos y aplicación correcta de una prueba funcional.					
<b>Criterios de Aceptación:</b>					
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> <li>- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y dibujado del proceso.</li> </ul>					

- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.

Tabla 49 Tarea 1 de historia de usuario 006 – Crear clase BPMNDrawElements  
Fuente: Elaboración Propia

b) TAREA 2: Crear clase BPMNProcessProperties

TARJETA DE TAREA RT-41				
Crear clase BPMNProcessProperties				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase BPMNProcessProperties que contenga el correcto armado de un Script de testeo orientado a la configuración de procesos y aplicación correcta de una prueba funcional.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> <li>- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y configuración del proceso.</li> <li>- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.</li> </ul>				

Tabla 50 Tarea 2 de historia de usuario 006 – Crear clase BPMNProcessProperties  
Fuente: Elaboración Propia

c) TAREA 3: Crear clase AssignmentRules

TARJETA DE TAREA RT-42				
Crear clase AssignmentRules				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase AssignmentRules que contenga el correcto armado de un Script de testeo orientado a la asignación de usuarios y aplicación correcta de una prueba funcional.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> </ul>				

- El Script debe iniciar desde la página de Login de Processmaker hasta la asignación de usuarios de un proceso.
- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.

Tabla 51 Tarea 3 de historia de usuario 006 – Crear clase AssignmentRules  
Fuente: Elaboración Propia

d) TAREA 4: Crear clase BPMNDynaformManagement

TARJETA DE TAREA RT-43				
Crear clase BPMNDynaformManagement				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase BPMNDynaformManagement que contenga el correcto armado de un Script de testeo orientado a la creación y armado de un formulario y aplicación correcta de una prueba funcional.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> <li>- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y armado de un formulario.</li> <li>- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.</li> </ul>				

Tabla 52 Tarea 4 de historia de usuario 006 – Crear clase BPMNDynaformManagement  
Fuente: Elaboración Propia

e) TAREA 5: Crear clase InputDocumentManagement

TARJETA DE TAREA RT-44				
Crear clase InputDocumentManagement				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase InputDocumentManagement que contenga el correcto armado de un				

Script de testeo orientado a la creación y configuración de Input Documents y aplicación correcta de una prueba funcional.

**Criterios de Aceptación:**

- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo
- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y configuración de un Input Document.
- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.

Tabla 53 Tarea 5 de historia de usuario 006 – Crear clase InputDocumentManagement  
Fuente: Elaboración Propia

f) TAREA 6: Crear clase TriggersManagement

TARJETA DE TAREA RT-45				
Crear clase TriggersManagement				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear una clase TriggersManagement que contenga el correcto armado de un Script de testeo orientado a la creación de triggers y aplicación correcta de una prueba funcional.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> <li>- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y configuración de triggers.</li> <li>- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.</li> </ul>				

Tabla 54 Tarea 6 de historia de usuario 006 – Crear clase TriggersManagement  
Fuente: Elaboración Propia

g) TAREA 7: Crear clase DatabaseConnectionManagement

TARJETA DE TAREA RT-46				
Crear clase DatabaseConnectionManagement				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 006	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	

Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum
<b>Descripción:</b>		
Se necesita crear una clase DatabaseConnectionManagement que contenga el correcto armado de un Script de testeo orientado a la creación y configuración de conexiones a bases de datos y aplicación correcta de una prueba funcional.		
<b>Criterios de Aceptación:</b>		
<ul style="list-style-type: none"> <li>- Deberá contener descripción de código indicando cada uno de los pasos aplicados al script de testeo</li> <li>- El Script debe iniciar desde la página de Login de Processmaker hasta la creación y configuración de una conexión a bases de datos.</li> <li>- Adjuntar aserciones que verifiquen que la herramienta Processmaker genere los resultados que se esperan.</li> </ul>		

Tabla 55 Tarea 7 de historia de usuario 006 – Crear clase DatabaseConnectionManagement  
Fuente: Elaboración Propia

### 3.2.4.7 HISTORIA DE USUARIO: ARMADO DE INFRAESTRUCTURA

Una vez desarrollado y completado el Framework de Automatización de pruebas funcionales, se necesita una infraestructura sobre la cual se van a ejecutar todas las pruebas de la herramienta Processmaker. Para esto, es necesario definir e instalar una serie de servicios, a continuación, se detalla en las siguientes tablas la descripción de la historia de usuario que define el armado de la infraestructura.

HISTORIA DE USUARIO			
Numero : <b>007</b>	Nombre de historia de usuario:	Armado de Infraestructura	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	MUY ALTA	Dificultad:	MUY ALTA
Riesgo :	MUY ALTA		
<b>Descripción:</b>			
La infraestructura sobre la cual se ejecutará el Framework de Automatización se compone de varios servidores y servicios los cuales hacen que la herramienta tenga una cobertura más amplia sobre las pruebas funcionales que se ejecutarán.			
Se necesita tener un servicio de Integración Continua que ejecute los test Automáticamente, un servidor con Selenium Hub y sus respectivos Nodos, los cuales contendrán los Stacks que soporta Processmaker para su ejecución.			
<ul style="list-style-type: none"> <li>- Tarea: Instalar Servidor de Integración Continua.</li> <li>- Tarea: Instalar Selenium HUB</li> </ul>			

- Tarea: Instalar Nodos de Testeo.
Observaciones: Ninguna

Tabla 56 Historia de usuario 007 – Armado de Infraestructura  
Fuente: Elaboración Propia

a) TAREA 1: Instalar Servidor de Integración Continua

TARJETA DE TAREA RT-47				
Instalar Servidor de Integración Continua				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 007	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear e instalar un servidor de Integración continua “Jenkins” el cual sirva para realizar tareas automatizadas, tales como descargar el código desarrollado para el Framework y ejecutarlo periódicamente, lanzando resultados				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá tener instalado un plugin de integración con GIT.</li> <li>- Deberá tener instalado un plugin de integración con proyectos MAVEN.</li> <li>- Deberá estar configurado correctamente para ejecutar pruebas periódicamente.</li> </ul>				

Tabla 57 Tarea 1 de historia de usuario 007 – Instalar Servidor de Integración Continua  
Fuente: Elaboración Propia

b) TAREA 2: Instalar Selenium HUB

TARJETA DE TAREA RT-48				
Instalar Selenium HUB				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 007	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear e instalar Selenium Hub el cual mandará instrucciones de ejecución a los nodos.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá tener instalado Selenium Server que se ejecute como Hub.</li> </ul>				

Tabla 58 Tarea 2 de historia de usuario 007 – Instalar Selenium HUB  
Fuente: Elaboración Propia

c) TAREA 3: Instalar Nodos de Testeo

TARJETA DE TAREA RT-49				
Instalar Nodos de Testeo				
Fecha:	Tipo de actividad:	✓ Nueva	Fijar	Mejorar
Numero de Historia: 007	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora	
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum		
<b>Descripción:</b>				
Se necesita crear e instalar nodos de ejecución de test, para esto se necesita consultar los Stacks soportados por Processmaker, tales como el tipo de Plataforma sobre el cual se ejecuta la herramienta y navegadores.				
<b>Criterios de Aceptación:</b>				
<ul style="list-style-type: none"> <li>- Deberá tener instalado navegadores (Mozilla Firefox, Google Chrome, Internet Explorer).</li> <li>- Deberá tener instalado sus respectivos drivers para que puedan ejecutar las pruebas.</li> <li>- Deberá tener instalado el servicio de Selenium server para ejecutarse como nodo.</li> </ul>				

Tabla 59 Tarea 3 de historia de usuario 007 – Instalar Nodos de Testeo  
Fuente: Elaboración Propia

### 3.2.4.8 HISTORIA DE USUARIO: IMPLEMENTACIÓN Y ESTABILIZACIÓN DEL FRAMEWORK

Después de terminar con las tareas previas se necesita implementar el Framework junto a Processmaker versión 3.x, seguidamente estabilizar la herramienta de testeo, a continuación, se detalla en las siguientes tablas la descripción de la historia de usuario que define la implementación y estabilización del Framework.

HISTORIA DE USUARIO			
Numero : <b>008</b>	Nombre de historia de usuario:	Implementación y Estabilización del Framework	
Fecha :		Desarrollador: Roberto Góngora	
Usuario :	Quality Engineer	Numero de iteración	Primera
Prioridad :	ALTA	Dificultad:	MEDIA
Riesgo :	MEDIA		
<b>Descripción:</b>			
Se debe Implementar el Framework junto a Processmaker versión 3.x el cual sigue en			

constante desarrollo y mejora continua, lanzar resultados y estabilizar la herramienta de testeo para que devuelva resultados mucho más fiables.

- Tarea: Modificar archivos de configuración del Framework para apuntar al servidor de pruebas donde se encuentra instalado Processmaker en desarrollo.
- Tarea: Modificar archivos base y herramientas de ayuda para ejecución correcta.

Observaciones: Ninguna

Tabla 60 Historia de usuario 008 – Implementación y Estabilización del Framework  
Fuente: Elaboración Propia

- a) TAREA 1: Modificar archivos de configuración del Framework para apuntar al servidor de pruebas

TARJETA DE TAREA RT-50			
Modificar archivos de configuración del Framework para apuntar al servidor de pruebas			
Fecha:	Tipo de actividad:	Nueva	Fijar <input type="checkbox"/> Mejorar <input checked="" type="checkbox"/>
Numero de Historia: 008	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum	
<b>Descripción:</b>			
Se necesita modificar los archivos de configuración del Framework para apuntar al servidor de pruebas donde se encuentra instalado Processmaker en desarrollo.			
<b>Criterios de Aceptación:</b>			
- Modificar el archivo.conf (ahora denominado <b>default.conf</b> ) para que apunte al servidor correcto usando Processmaker en versión desarrollo y ejecutando Selemion HUB.			

Tabla 61 Tarea 1 de historia de usuario 008 – Modificar archivos de configuración del Framework  
Fuente: Elaboración Propia

- b) TAREA 2: Modificar archivos base y herramientas de ayuda para ejecución correcta.

TARJETA DE TAREA RT-51			
Modificar archivos base y herramientas de ayuda para ejecución correcta			
Fecha:	Tipo de actividad:	Nueva	Fijar <input type="checkbox"/> Mejorar <input checked="" type="checkbox"/>
Numero de Historia: 008	Prioridad: Alta	Usuario: Quality Engineer	Desarrollador: Roberto Góngora
Referencia Anterior :	Riesgo : Alta	Técnicas de estimación: Poker Scrum	
<b>Descripción:</b>			
Se necesita modificar los archivos base y herramientas de ayuda para ejecución correcta, previa verificación de los mismo.			
<b>Criterios de Aceptación:</b>			



- Verificar el correcto comportamiento de los archivos base y clases abstractas y modificar para una mejor ejecución usando la infraestructura de Automatización.

Tabla 62 Tarea 2 de historia de usuario 008 – Modificar archivos base y herramientas de ayuda  
Fuente: Elaboración Propia

### 3.2.5 DEFINICIÓN DE PRODUCT BACKLOG (PILA DE PRODUCTOS)

La pila de producto lista todas las características, funcionalidades, requerimientos, mejoras y correcciones que fueron realizadas con el cliente durante el desarrollo del software, esta lista esta ordenada según el valor, riesgo, prioridad y necesidad del cliente. A continuación, se detallan la lista de requerimientos que fueron obtenidas con la colaboración y prioridades del cliente (product Owner).

Nº de Historia	prioridad	Tarea	Descripción	Sprint
001	Muy alta	RT-1	Crear archivo de configuración <b>pom.xml</b>	1
001	Muy alta	RT-2	Crear archivo de configuración <b>.conf</b>	1
002	Muy alta	RT-3	Crear clase de invocación de Driver “BrowserInstance”	7
002	Muy alta	RT-4	Crear clase de configuración de Driver “BrowserSettings”	8
002	Muy alta	RT-5	Crear clase para eventos de espera “WaitTool”	8
002	Muy alta	RT-6	Crear clase abstracta de página “Page”	1
003	Alta	RT-7	Crear clase “wrapper” ExtjsTextBox	1
003	Alta	RT-8	Crear clase “wrapper” ExtjsTextArea	1
003	Alta	RT-9	Crear clase “wrapper” ExtjsDropDown	1
003	Alta	RT-10	Crear clase “wrapper” ExtjsButton	1
003	Alta	RT-11	Crear clase “wrapper” ExtjsGridPanel	1
003	Alta	RT-12	Crear clase “wrapper” ExtjsMessageWindow	2

004	Muy alta	RT-13	Crear clase "wrapper" TextBoxField	2
004	Muy alta	RT-14	Crear clase "wrapper" PasswordField	2
004	Muy alta	RT-15	Crear clase "wrapper" TextAreaField	2
004	Muy alta	RT-16	Crear clase "wrapper" DropDownField	2
004	Muy alta	RT-17	Crear clase "wrapper" CheckGroupField	2
004	Muy alta	RT-18	Crear clase "wrapper" RadioField	2
004	Muy alta	RT-19	Crear clase "wrapper" DateTimeField	2
004	Muy alta	RT-20	Crear clase "wrapper" ContextMenu	3
004	Muy alta	RT-21	Crear clase "wrapper" Button	3
004	Muy alta	RT-22	Crear clase "wrapper" GridPanel	3
004	Muy alta	RT-23	Crear clase "wrapper" TabPanel	3
004	Muy alta	RT-24	Crear clase "wrapper" TreePanel	4
004	Muy alta	RT-25	Crear clase "wrapper" MessageWindow	4
004	Muy alta	RT-26	Crear clase "wrapper" Window	4
005	Muy alta	RT-27	Crear clase ProcessDesignerBPMN	4
005	Muy alta	RT-28	Crear paquete VariableBPMN	4
005	Muy alta	RT-29	Crear paquete MessageTypeBPMN	5
005	Muy alta	RT-30	Crear paquete DynaformBPMN	5
005	Muy alta	RT-31	Crear paquete InputDocBPMN	5
005	Muy alta	RT-32	Crear paquete OutputDocBPMN	5
005	Muy alta	RT-33	Crear paquete TriggerBPMN	5
005	Muy alta	RT-34	Crear paquete ReportTableBPMN	7
005	Muy alta	RT-35	Crear paquete DatabaseConnectionBPMN	6
005	Muy alta	RT-36	Crear paquete TemplateBPMN	6
005	Muy alta	RT-37	Crear paquete PublicFileBPMN	7

005	Muy alta	RT-38	Crear paquete PermissionBPMN	7
005	Muy alta	RT-39	Crear paquete SupervisorBPMN	7
006	Alta	RT-40	Crear clase BPMNDrawElements	7
006	Alta	RT-41	Crear clase BPMNProcessProperties	7
006	Alta	RT-42	Crear clase AssignmentRules	8
006	Alta	RT-43	Crear clase BPMNDynaformManagement	8
006	Alta	RT-44	Crear clase InputDocumentManagement	8
006	Alta	RT-45	Crear clase TriggersManagement	8
006	Alta	RT-46	Crear clase DatabaseConnectionManagement	8
007	Muy alta	RT-47	Instalar Servidor de Integración Continua	8
007	Muy alta	RT-48	Instalar Selenium HUB	8
007	Muy alta	RT-49	Instalar Nodos de Testeo	8
008	Alta	RT-50	Modificar archivos de configuración del Framework para apuntar al servidor de pruebas	8
008	Alta	RT-51	Modificar archivos base y herramientas de ayuda para ejecución correcta	8

Tabla 63 Product Backlog – Requerimientos del Framework de Automatización de Pruebas Funcionales  
Fuente: Elaboración Propia

### 3.2.6 DEFINICIÓN DEL CRONOGRAMA DE TRABAJO

La planificación para el desarrollo de las tareas está regida por el roadmap y la matriz de roadmap, donde se seleccionaron los ítems del product Backlog por prioridades para luego transformarlos en productos entregables. Para tal propósito se desarrolló un diagrama de Gantt (Anexo A).

### 3.2.7 ANÁLISIS DE RIESGOS

La gestión de riesgos se refiere a la reducción de probabilidades e impactos que pudieran afectar en el desarrollo del producto, existen diferentes tipos de riesgos asociados.

Los riesgos del proyecto amenazan el plan del proyecto (roadmap) y pueden alterar los alcances y fechas previstas, estos pueden ser problemas en el presupuesto, personal, recursos y requisitos.

**Los riesgos técnicos**, amenazan en la calidad y actualidad del software, estos pueden ser problemas en diseño de la arquitectura y/o de clases de la librea generadora de formularios, en la implementación sin escalabilidad y sin extensibilidad, afectando el crecimiento futuro de la librería, interfaz siendo crowsbrowsing el aspecto debe ser similar en todos los navegadores, verificación y mantenimiento.

**Los riesgos de negocio**, amenazan con la viabilidad el proyecto del software, este tipo de riesgo se presentan en estrategias, gestión, ventas y presupuestos.

**Valores de impacto (imp.):** Catastrófico (valor 1), Critico (valor 2), Marginal (Valor 3), Despreciable (valor 4) y **RSGR:** Plan de Reducción, Supervisión y Gestión de Riesgo del presente proyecto.

Riesgo	Categoría	Prob.	Imp	RSGR
Cambios constantes en los requerimientos del cliente.	Proyecto	Alta	2	Realizar una revisión o retrospectiva, constante en los requerimientos del cliente.
Los requerimientos de sistema (hardware y software), no se cumplen	Técnica	Alta	2	Solicitar reuniones, para así tener con anticipación los requerimientos del Framework.
No se cumplen con los plazos de entrega del producto.	Proyecto	Media	2	Agilizar los procesos del desarrollo del producto.

Incumplimiento den el cronograma de avance.	Proyecto	Media	3	Replantear fechas en el cronograma.
Reuniones en las que no asista el cliente	Equipo	Baja	3	Planificar la agenda de reuniones con el cliente.

Tabla 64 Análisis de Riesgo  
Fuente: Elaboración Propia

### 3.3 GAME

Durante esta etapa de desarrollo del proyecto se desarrollaron los 7 sprints de acuerdo a la tecnología scrum y roadmap, vista en la parte introductoria de este capítulo se puede observar la presentación de los siguientes puntos.

- **Planificación** de Sprint, donde se definieron las tareas de cada sprint.
- **Desarrollo** del Sprint, es esta etapa se aplicará los criterios de aceptación definidas en las historias y roadmap para seguir un lineamiento acorde a las necesidades.
- **Revisión** del Sprint, donde verificamos si se cumplieron las tareas planificadas en la pila de productos.

Para el desarrollo del proyecto se trabajaron en los días hábiles de la semana y 8 horas diarias.

#### 3.3.1 SPRINT 1: DEFINICIÓN DE ESTRUCTURA DE ARCHIVOS Y CREACIÓN DE CLASES

##### 3.3.1.1 PLANIFICACIÓN DEL SPRINT

La planificación para el desarrollo de los sprints se realizó en la reunión con el Product Owner (QA team Leader).

Durante el primer sprint se desarrolló los requerimientos iniciales que pertenecen a la definición de la estructura de archivos y creación de clases base.

En la siguiente tabla se muestra las tareas programadas para este sprint y que fueron concluidas como se muestra a continuación.

<b>SPRINT 1: ESTRUCTURA DE ARCHIVOS, CREACIÓN DE CLASES BASE Y ABSTRACTAS</b>				
<b>Sprint</b>	<b>Inicio</b>	<b>Fin</b>	<b>Duración horas</b>	<b>Días de Trabajo</b>
<b>1</b>	1 de Febrero 2016	12 de Febrero 2016	80	10 días hábiles
<b>SPRINT BACKLOG</b>				
<b>Backlog-ID</b>	<b>TAREAS</b>		<b>TIPO</b>	<b>ESTADO</b>
<b>RT-1</b>	Crear archivo de configuración <b>pom.xml</b>		Diseño/ Desarrollo	hecho
<b>RT-2</b>	Crear archivo de configuración <b>.conf</b>		Diseño/ Desarrollo	Hecho
<b>RT-3</b>	Crear clase de invocación de Driver "BrowserInstance"		Diseño/ Desarrollo	Hecho
<b>RT-4</b>	Crear clase de configuración de Driver "BrowserSettings"		Diseño/ Desarrollo	Hecho
<b>RT-5</b>	Crear clase para eventos de espera "WaitTool"		Diseño/ Desarrollo	Hecho
<b>RT-6</b>	Crear clase abstracta de página "Page"		Diseño/ Desarrollo	Hecho

Tabla 65 Sprint 1 – Planificación  
Fuente: Elaboración Propia

### 3.3.1.2 DESARROLLO DEL SPRINT

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (Product Owner), y están planificados en el Roadmap del proyecto.

Para la creación del proyecto se desarrolló un archivo de configuración pom.xml (Project Object Model), el cual es la unidad principal del proyecto, contiene información acerca del proyecto, dependencias, estructura de directorios, etc. A partir de este archivo base se generar los demás archivos y clases.

### 3.3.1.3 REVISIÓN DEL SPRINT

En esta etapa se revisó el cumplimiento de las tareas planificadas, como puede observarse en la tabla anterior, el estado de las tareas se muestra como “hecho” lo que significa que estas tareas se concluyeron de forma satisfactoria y sin ningún tipo de observación.

Para verificar el producto entregable de la iteración, se hicieron las siguientes pruebas.

Ejecución del archivo pom.xml

Prueba : 1.1	Operación:
<b>Ejecución del archivo pom.xml</b>	
<b>Precondición:</b>	
Tener un Entorno de desarrollo integrado(IDE) el cual tenga configurado JAVA como Kit De desarrollo de Software.	
<b>Datos/pasos a realizar:</b>	
<ul style="list-style-type: none"><li>- Crear un archivo xml con las especificaciones indicadas por Selenium WebDriver.</li><li>- Fijar todas las dependencias para ejecución de un proyecto MAVEN.</li><li>- Configurar el IDE para ejecutar un proyecto MAVEN.</li><li>- Configurar para que se descarguen los modulos especificados en el documento xml.</li></ul>	
<b>Resultados esperados:</b>	
Se bajan todos los módulos especificados en el documento pom.xml	

**Post condiciones:**

Se genera una ventana de herramientas para ejecución de proyectos MAVEN

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 66 Sprint 1 – Prueba: Ejecución del archivo pom.xml  
Fuente: Elaboración Propia

Invocación de un Navegador mediante el driver:

Prueba : 1.2

Operación:

Crear una clase de prueba para hacer una invocación de un Navegador mediante el driver.

**Precondición:**

Tener un Navegador Mozilla Firefox.

**Datos/pasos a realizar:**

- Crear una clase de ejemplo para invocación del browser.
- Instanciar o crear un objeto de tipo:  
`WebDriver driver = new FirefoxDriver();`

**Resultados esperados:**

Debe poder ejecutarse la clase sin reportar ningún error en compilación.

**Post condiciones:**

Debe iniciar el navegador

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 67 Sprint 1 – Prueba: Invocación de un navegador mediante el driver  
Fuente: Elaboración Propia



### 3.3.2 SPRINT 2: CREACIÓN DE CLASES WRAPPER EXTJS

#### 3.3.2.1 PLANIFICACIÓN DEL SPRINT

La librería Javacript “ExtJS” es parte de la herramienta Processmaker, por lo tanto, necesita que esos elementos sean abstraídos usando clases “wrapper” las cuales nos darán la opción de tratar a estos elementos como objetos dentro del patrón Objeto-Páginas (Page Object Pattern).

SPRINT 2: CREACIÓN DE CLASES WRAPPER EXTJS				
Sprint	Inicio	Fin	Duración horas	Días de Trabajo
2	15 de febrero, 2016	26 de febrero, 2016	80	10 días hábiles
SPRINT BACKLOG				
Backlog-ID	TAREAS		TIPO	ESTADO
RT-7	Crear clase “wrapper” ExtjsTextBox		Diseño/ Desarrollo	Hecho
RT-8	Crear clase “wrapper” ExtjsTextArea		Diseño/ Desarrollo	Hecho
RT-9	Crear clase “wrapper” ExtjsDropDown		Diseño/ Desarrollo	Hecho
RT-10	Crear clase “wrapper” ExtjsButton		Diseño/ Desarrollo	Hecho
RT-11	Crear clase “wrapper” ExtjsGridPanel		Diseño/ Desarrollo	Hecho
RT-12	Crear clase “wrapper” ExtjsMessageWindow		Diseño/ Desarrollo	Hecho

Tabla 68 Sprint 2 – Planificación  
Fuente: Elaboración Propia

### 3.3.2.2 DESARROLLO DEL SPRINT

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (product owner), y están planificados en el roadmap del proyecto.

Para la creación de estas clases “Wrapper” se creó un nuevo paquete dentro de Processmaker el cual alberga a todas las clases que son orientadas a trabajar con la librería JavaScript ExtJS.

### 3.3.2.3 REVISIÓN DEL SPRINT

Las pruebas de funcionalidad según criterios de aceptación que se hicieron para el registro de usuarios se muestran en la siguiente tabla.

Prueba: 2.1	Operación:  <b>Crear y/o instanciar un wrapper ExtJS de tipo TextBox, TextArea, DropDown</b>
<b>Precondición:</b>  Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.	
<b>Datos/pasos a realizar:</b>  - Instanciar un textBox, TextArea, Dropdown usando el ID:  <pre>ExtJSTextBox extJSTextBox = new ExtJSTextBox(browserInstance, "textbox-1010"); ExtJSTextArea extJSTextArea = new ExtJSTextArea(browserInstance, "textareafield-1017"); ExtJSDropDown extJSDropDown = new ExtJSDropdown(browserInstance, "dropdown-1011");</pre> - Instanciar un textBox, TextArea, Dropdown usando el WebElement:  <pre>WebElement textArea = browserInstance.getInstanceDriver().findElement(By.id("textareafield-1017")); ExtJSTextArea extJSTextArea = new ExtJSTextArea(browserInstance, textArea);  WebElement textBox = browserInstance.getInstanceDriver().findElement(By.id("textbox-1010")); ExtJSTextBox extJSTextBox = new ExtJSTextBox(browserInstance, textBox);  WebElement dropDown = browserInstance.getInstanceDriver().findElement(By.id("dropdown-1011")); ExtJSDropDown extJSDropDown = new ExtJSDropDown(browserInstance, dropDown);</pre>	

<p>- Llamar a métodos para establecer un valor</p> <pre>extJSTextArea.setTextAreaValue("New value Text Area"); extJSTextBox.setTextBoxValue("New value Text Box"); extJSDropDown.setDropDownValue("New value selected");</pre> <p>- Llamar a métodos para retornar el valor establecido:</p> <pre>extJSTextArea.getTextAreaValue(); extJSTextBox.getTextBoxValue(); extJSDropDown.getDropDownValue();</pre>
<p><b>Resultados esperados:</b></p> <p>Se instancian los elementos, se ejecutan correctamente los métodos.</p>
<p><b>Post condiciones:</b></p> <p>Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.</p>
<p><b>Resultados obtenidos:</b></p> <p>Se obtienen los resultados esperados.</p>

Tabla 69 Sprint 2 – Prueba: Creación y/o instanciación de un Wrapper ExtJS de tipo TextBox, TextArea, DropDown  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 2.2	Operación: <b>Crear y/o instanciar un wrapper ExtJS de tipo Button</b>
<b>Precondición:</b>	
Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.	
<b>Datos/pasos a realizar:</b>	

- Obtener un button usando el ID:

```
ExtjsButton extjsButton1 = new ExtjsButton(browserInstance, "buttonContinue");
```

- Obtener un button usando Webdriver:

```
WebElement ButtonExtjs = browserInstance.getInstanceDriver().findElements(By.cssSelector(".x-  
btn.x-noicon.x-btn-noicon.x-btn-default-small-noicon")).get(1);  
ExtjsButton extjsButton1 = new ExtjsButton(browserInstance, ButtonExtjs);
```

- Hacer "click" sobre un boton

```
extjsButton1.click();
```

#### **Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

#### **Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

#### **Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 70 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo Button  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 2.3

Operación:

**Crear y/o instanciar un wrapper ExtJS de tipo GridPanel**

#### **Precondición:**

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.

#### **Datos/pasos a realizar:**

- Obtener un GridPanel usando ID:

```
gridPanelDebug = new ExtJSGrid("gridPanel-001", browser);
```

- Obtener un GridPanel usando WebDriver:

```
gridPanelDebug = new ExtJSGrid(gridPanel,browser);
```

- Seleccionar un item del elemento

```
Int Itemnumber = 5;  
gridPanelDebug.clickOptionsDebugPanel(itemNumber);
```

- Obtener el número de items de un gridPanel:

```
int totalNumber = gridPanelDebug.getNumberOptionsDebugPanel();
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 71 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo GridPanel  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 2.4

Operación:

**Crear y/o instanciar un wrapper ExtJS de tipo MessageWindow**

**Precondición:**

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.

**Datos/pasos a realizar:**

- Obtener un GridPanel usando ID:

```
messWindow = new ExtJSMessageWindow("messWin-006",browser);
```

- Obtener un GridPanel usando WebDriver:

```
messWindow = new ExtJSMessageWindow(window,browser);
```

- Obtener el texto de un elemento MessageWindow:

```
String text1 = messWindow.getBodyText();
```

- Obtener el número de botones del elemento MessageWindow:

```
int numberButtons = messWindow.getButtonsNumber();
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 72 Sprint 2 – Prueba: Creación y/o instanciación de un wrapper ExtJS de tipo MessageWindow  
Fuente: Elaboración Propia

### 3.3.3 SPRINT 3: CREACIÓN DE CLASES WRAPPER PMUI

#### 3.3.3.1 PLANIFICACIÓN DEL SPRINT

La librería Javacript “pmUI” es parte de la herramienta Processmaker, por lo tanto, necesita que esos elementos sean abstraídos usando clases “wrapper” las cuales nos darán la opción de tratar a estos elementos como objetos dentro del patrón Objeto-Páginas (Page Object Pattern).

En la siguiente tabla se muestra las taras programadas para este sprint y que fueron concluidas como se muestra a continuación.

<b>SPRINT 3: CREACIÓN DE CLASES WRAPPER PMUI</b>				
<b>Sprint</b>	<b>Inicio</b>	<b>Fin</b>	<b>Duración horas</b>	<b>Días de Trabajo</b>
<b>3</b>	29 de Febrero, 2016	18 de Marzo, 2016	120	15 días hábiles
<b>SPRINT BACKLOG</b>				
<b>Backlog-ID</b>	<b>TAREAS</b>		<b>TIPO</b>	<b>ESTADO</b>
<b>RT-13</b>	Crear clase “wrapper” TextBoxField		Diseño/ Desarrollo	hecho
<b>RT-14</b>	Crear clase “wrapper” PasswordField		Diseño/ Desarrollo	Hecho
<b>RT-15</b>	Crear clase “wrapper” TextAreaField		Diseño/ Desarrollo	Hecho
<b>RT-16</b>	Crear clase “wrapper” DropDownField		Diseño/ Desarrollo	hecho
<b>RT-17</b>	Crear clase “wrapper” CheckGroupField		Diseño/ Desarrollo	Hecho
<b>RT-18</b>	Crear clase “wrapper” RadioField		Diseño/ Desarrollo	hecho
<b>RT-19</b>	Crear clase “wrapper” DateTimeField		Diseño/ Desarrollo	Hecho
<b>RT-20</b>	Crear clase “wrapper” ContextMenu		Diseño/ Desarrollo	Hecho
<b>RT-21</b>	Crear clase “wrapper” Button		Diseño/ Desarrollo	hecho
<b>RT-22</b>	Crear clase “wrapper” GridPanel		Diseño/ Desarrollo	Hecho

<b>RT-23</b>	Crear clase “wrapper” TabPanel	Diseño/ Desarrollo	Hecho
<b>RT-24</b>	Crear clase “wrapper” TreePanel	Diseño/ Desarrollo	hecho
<b>RT-25</b>	Crear clase “wrapper” MessageWindow	Diseño/ Desarrollo	Hecho
<b>RT-26</b>	Crear clase “wrapper” Window	Diseño/ Desarrollo	Hecho

Tabla 73 Sprint 3 – Planificación  
Fuente: Elaboración Propia

### 3.3.3.2 DESARROLLO DEL SPRINT

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (product owner), y están planificados en el roadmap del proyecto para las fechas de entrega.

Para la creación de estas clases “Wrapper” se creó un nuevo paquete dentro de Processmaker el cual alberga a todas las clases que son orientadas a trabajar con la librería JavaScript pmUI.

### 3.3.3.3 REVISIÓN DEL SPRINT

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 3.1

Operación:

Crear y/o instanciar un wrapper pmUI de tipo TextBoxField, PasswordField, TextAreaField, DropDownField, CheckGroupField, RadioField y DateTimeField

**Precondición:**

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.



**Datos/pasos a realizar:**

- Obtener un elemento usando el ID:

```
PmuiField anyfield = new PmuiField(browserInstance, "field-id");
```

- Obtener un elemento usando WebDriver:

```
WebElement textArea = browserInstance.getInstanceDriver().findElement(By.id("any_field"));  
PmuiField anyfield = new PmuiField(browserInstance, textArea);
```

- Establecer un valor en el campo:

```
String valueText = "Hello World";  
anyfield.setValue(valueText);
```

- Obtener el valor establecido en el campo:

```
String value1 = anyfield.getValue(valueText);
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 74 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI para los campos del formulario  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 3.2

Operación:

Crear y/o instanciar un wrapper pmUI de tipo ContextMenu

Precondición:

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.

**Datos/pasos a realizar:**

- Obtener un elemento usando el ID:

```
PmuiContextMenu contextMenu = new PmuiContextMenu(browserInstance, "field-id");
```

- Obtener un elemento usando WebDriver:

```
WebElement context = browserInstance.getInstanceDriver().findElement(By.id("menuOptions1"));  
PmuiContextMenu contextMenu = new PmuiContextMenu(browserInstance, context);
```

- Seleccionar un Item del menu:

```
contextMenu.click(item);
```

- Seleccionar el texto de un determinado item del menu:

```
String text1 = contextMenu.getText(item);
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 75 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo ContextMenu  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 3.3

Operación:

Crear y/o instanciar un wrapper pmUI de tipo Button

Precondición:

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales para ejecución.

**Datos/pasos a realizar:**

- Obtener un button usando el ID:

```
PmuiButton Button1 = new PmuiButton(browserInstance, "buttonContinue");
```

- Obtener un button usando Webdriver:

```
WebElement ButtonpmUI = browserInstance.getInstanceDriver().findElements(By.cssSelector(".x-  
btn.x-noicon.x-btn-noicon.x-btn-default-small-noicon")).get(1);  
PmuiButton Button1 = new PmuiButton(browserInstance, ButtonpmUI);
```

- Hacer "click" sobre un boton

```
Button1.click();
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 76 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo Button  
Fuente: Elaboración Propia

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 3.4

Operación:

**Crear y/o instanciar un wrapper pmUI de tipo Panel**

**Precondición:**

Tener el proyecto funcional y con las clases base creadas además de las configuraciones iniciales

para ejecución.

**Datos/pasos a realizar:**

- Obtener un Panel usando ID:

```
PmuiPanel Panel1 = new PmuiPanel("panel-001",browser);
```

- Obtener un GridPanel usando WebDriver:

```
PmuiPanel Panel1 = new PmuiPanel(gridPanel,browser);
```

- Seleccionar un item del elemento

```
Int Itemnumber = 3;  
Panel1.clickOptionPanel(itemNumber);
```

- Obtener el número de items de un gridPanel:

```
int totalNumber = Panel1.getNumberOptionsPanel();
```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Los wrappers están listos para su uso, y por consiguiente los métodos pueden ser usados en las páginas.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 77 Sprint 3 – Prueba: Creación y/o instanciación de un wrapper pmUI de tipo Panel  
Fuente: Elaboración Propia

### 3.3.4 SPRINT 4: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE I)

#### 3.3.4.1 PLANIFICACIÓN DEL SPRINT

La planificación para el desarrollo de los sprints se realizó en la reunión con los clientes (Product Owner).

Durante este sprint se desarrollaron las páginas usando el patrón de diseño Page Object Pattern el cual hace uso de todas las clases “Wrapper”, y genera páginas abstrayendo todos los elementos de cada una de las páginas de la herramienta Processmaker.

<b>SPRINT 4: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE I)</b>				
<b>Sprint</b>	<b>Inicio</b>	<b>Fin</b>	<b>Duración horas</b>	<b>Días de Trabajo</b>
<b>4</b>	21 de Marzo, 2016	8 de Abril, 2016	120	15 días hábiles
<b>SPRINT BACKLOG</b>				
<b>Backlog-ID</b>	<b>TAREAS</b>		<b>TIPO</b>	<b>ESTADO</b>
<b>RT-27</b>	Crear clase ProcessDesignerBPMN		Diseño/ Desarrollo	hecho
<b>RT-28</b>	Crear paquete VariableBPMN		Diseño/ Desarrollo	Hecho
<b>RT-29</b>	Crear paquete MessageTypeBPMN		Diseño/ Desarrollo	Hecho
<b>RT-30</b>	Crear paquete DynaformBPMN		Diseño/ Desarrollo	Hecho

Tabla 78 Sprint 4 – Planificación  
Fuente: Elaboración Propia

### **3.3.4.2 DESARROLLO DEL SPRINT**

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (Product Owner), y están planificados en el Roadmap del proyecto. Las Clases ProcessDesignerBPMN y DynaformBPMN son extensas por lo que para este Sprint se concentra el objetivo en concluir ambas.

### 3.3.4.3 REVISIÓN DEL SPRINT

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba: 4.1 Operación:  
Crear y/o instanciar una página ProcessDesignerBPMN, VariableBPMN y DynaformBPMN

#### Precondición:

Tener el proyecto funcional y con las clases base creadas, wrappers, además de las configuraciones iniciales para ejecución.

#### Datos/pasos a realizar:

- Instanciar un objeto-página de tipo ProcessDesignerBPMN y sus métodos representativos:

```
ProcessDesignerBPMN processDesignerBPMN =  
    designerProcessList.createNewBPMNProcess("testNewBPMN", "Test New BPMN Project");  
//verificamos los elementos de página  
processDesignerBPMN.verifyPage();
```

```
//dibujamos un elemento en el diseñador  
int x = 120;  
int y = 240;  
task1Id = processDesignerBPMN.drawNewTask(x,y);
```

```
//guardamos el proceso  
processDesignerBPMN.saveDesign();
```

- Instanciar un objeto-página de tipo VariablesList y sus métodos representativos:

```
VariablesList variablesList = processDesignerBPMN.openVariables();  
CreateVariables createVariables = variablesList.clickButtonCreateVariable();  
//verificamos los elementos de página  
createVariables.verifyPage();  
//creamos una variable de tipo String  
createVariables.setNameVariable("variable1");  
createVariables.setFieldTypeVariable("String");  
//guardamos los cambios  
createVariables.clickSave();
```

<p>- Instanciar un objeto-página de tipo DynaformListBPMN y sus métodos representativos:</p> <pre> DynaformListBPMN dynaformListBPMN1 = new DynaformListBPMN(browserInstance); //verificamos los elementos de página dynaformBlankBPMN.verifyPage(); //creamos un dynaform y guardamos los cambios dynaformBlankBPMN.setMasterTitle(title); dynaformBlankBPMN.clickSaveButton(); </pre>
<p><b>Resultados esperados:</b></p> <p>Se instancian los elementos, se ejecutan correctamente los métodos.</p>
<p><b>Post condiciones:</b></p> <p>Las páginas están listas para su uso, y por consiguiente los métodos pueden ser usados en los Scripts de testeo.</p>
<p><b>Resultados obtenidos:</b></p> <p>Se obtienen los resultados esperados.</p>

Tabla 79 Sprint 4 – Prueba: Creación y/o instanciación de páginas de Sprint 4  
Fuente: Elaboración Propia

### 3.3.5 SPRINT 5: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE II)

#### 3.3.5.1 PLANIFICACIÓN DEL SPRINT

La planificación para el desarrollo del sprint se realizó en la reunión con los clientes (Product Owner).

Durante este sprint se desarrollaron las páginas usando el patrón de diseño Page Object Pattern el cual hace uso de todas las clases “Wrapper”, y genera páginas abstrayendo todos los elementos de cada una de las páginas de la herramienta Processmaker.

En la siguiente tabla se muestra las taras programadas para este sprint y que fueron concluidas como se muestra a continuación.

### SPRINT 5: CONSTRUCCIÓN DE PAGE-OBJECTS (PARTE II)

<b>Sprint</b>	<b>Inicio</b>	<b>Fin</b>	<b>Duración horas</b>	<b>Días de Trabajo</b>
<b>5</b>	11 de Abril, 2016	29 de Abril, 2016	120	15 días hábiles
<b>SPRINT BACKLOG</b>				
<b>Backlog-ID</b>	<b>TAREAS</b>		<b>TIPO</b>	<b>ESTADO</b>
<b>RT-31</b>	Crear paquete InputDocBPMN		Diseño/ Desarrollo	hecho
<b>RT-32</b>	Crear paquete OutputDocBPMN		Diseño/ Desarrollo	Hecho
<b>RT-33</b>	Crear paquete TriggerBPMN		Diseño/ Desarrollo	Hecho
<b>RT-34</b>	Crear paquete ReportTableBPMN		Diseño/ Desarrollo	Hecho
<b>RT-35</b>	Crear paquete DatabaseConnectionBPMN		Diseño/ Desarrollo	hecho
<b>RT-36</b>	Crear paquete TemplateBPMN		Diseño/ Desarrollo	Hecho
<b>RT-37</b>	Crear paquete PublicFileBPMN		Diseño/ Desarrollo	Hecho
<b>RT-38</b>	Crear paquete PermissionBPMN		Diseño/ Desarrollo	Hecho
<b>RT-39</b>	Crear paquete SupervisorBPMN		Diseño/ Desarrollo	Hecho

Tabla 80 Sprint 5 – Planificación  
Fuente: Elaboración Propia

#### 3.3.5.2 DESARROLLO DEL SPRINT

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (Product Owner), y están



planificados en el Roadmap del proyecto. En esta segunda parte se concluyó con las Páginas pendientes del diseñador de procesos.

### 3.3.5.3 REVISIÓN DEL SPRINT

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

<b>Prueba:5.1</b>	<b>Operación:</b>  Crear y/o instanciar una página InputDocumentBPMN , OutputDocumentBPMN, CreateTriggerTypeBPMN, ReportTables, DataBaseConnectionBPMN, ProcessFilesTemplateBPMN, ProcessFilesPublicBPMN, ProcessPermissionsBPMN y SupervisorStepsBPMN.
<b>Precondición:</b>  Tener el proyecto funcional y con las clases base creadas, wrappers, además de las configuraciones iniciales para ejecución.	
<b>Datos/pasos a realizar:</b>  - Instanciar un objeto-página de tipo InputDocumentBPMN y sus métodos representativos:  <pre>InputDocumentBPMN inputDocumentBPMN2= processDesignerBPMN.openInputDocuments(); //verificamos los elementos de página inputDocumentBPMN2.verifyPage();</pre> - Instanciar un objeto-página de tipo OutputDocumentBPMN y sus métodos representativos:  <pre>OutputDocumentBPMN outputDocumentBPMN = processDesignerBPMN.openOutputDocument(); //verificamos los elementos de página outputDocumentBPMN.verifyPage();</pre> - Instanciar un objeto-página de tipo CreateTriggerTypeBPMN y sus métodos representativos:  <pre>CreateTriggerTypeBPMN triggerNew = triggersListBPMN.openNewTrigger(); CustomTriggerBPMN custom = triggerNew.clickOnCustomTrigger(); //verificamos los elementos de página custom.verifyPage();</pre>	

- Instanciar un objeto-página de tipo ReportTables y sus métodos representativos:

```
ReportTables reportTables = processDesignerBPMN.openReportTables();  
//verificamos los elementos de página  
reportTables.verifyPage();
```

- Instanciar un objeto-página de tipo DataBaseConnectionBPMN y sus métodos representativos:

```
DataBaseConnectionListBPMN dataBaseConnectionListBPMN =  
    processDesignerBPMN.openDataBaseConnection();  
DataBaseConnectionBPMN dataBaseConnectionBPMN =  
    dataBaseConnectionListBPMN.openNewDataBaseConnectionWindow();  
//verificamos los elementos de página  
dataBaseConnectionBPMN.verifyPage();
```

- Instanciar un objeto-página de tipo ProcessFilesTemplateBPMN y sus métodos representativos:

```
ProcessFilesListBPMN processFilesListBPMN = processDesignerBPMN.processFiles();  
ProcessFilesTemplateBPMN template1 = processFilesListBPMN.clickOnTemplate();  
//verificamos los elementos de página  
template1.verifyPage();
```

- Instanciar un objeto-página de tipo ProcessFilesPublicBPMN y sus metodos representativos:

```
ProcessFilesListBPMN processFilesListBPMN = processDesignerBPMN.processFiles();  
ProcessFilesPublicBPMN public1 = processFilesListBPMN.clickOnPublic();  
//verificamos los elementos de página  
public1.verifyPage();
```

- Instanciar un objeto-página de tipo ProcessPermissionsBPMN y sus metodos representativos:

```
ProcessPermissionsListBPMN processPermissionsListBPMN =processDesignerBPMN.processPermissions();  
ProcessPermissionsBPMN processPermissionsBPMN = processPermissionsListBPMN.createPermissions();  
//verificamos los elementos de página  
processPermissionsBPMN.verifyPage();
```

- Instanciar un objeto-página de tipo SupervisorStepsBPMN y sus metodos representativos:

```

SupervisorStepsBPMN assignUsersAndGroupsAsSupervisors =
    processDesignerBPMN.processSupervisorsSteps();
//verificamos los elementos de página
assignUsersAndGroupsAsSupervisors.verifyPage();

```

**Resultados esperados:**

Se instancian los elementos, se ejecutan correctamente los métodos.

**Post condiciones:**

Las páginas están listas para su uso, y por consiguiente los métodos pueden ser usados en los Scripts de testeo.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 81 Sprint 5 – Prueba: Creación y/o instanciación de páginas de Sprint 5  
Fuente: Elaboración Propia

### 3.3.6 SPRINT 6: DISEÑO DE TESTING SCRIPTS

#### 3.3.6.1 PLANIFICACIÓN DEL SPRINT

Los Testing Scripts, son clases desarrolladas con el propósito de aplicar herramientas del Framework para ejecutar pruebas funcionales.

A continuación, se detallan los aspectos de la planificación.

SPRINT 6: DISEÑO DE TESTING SCRIPTS				
Sprint	Inicio	Fin	Duración horas	Días de Trabajo
6	2 de Mayo, 2016	13 de Mayo, 2016	80	10 días hábiles
<b>SPRINT BACKLOG</b>				

Backlog-ID	TAREAS	TIPO	ESTADO
RT-40	Crear clase BPMNDrawElements	Diseño/ Desarrollo	Hecho
RT-41	Crear clase BPMNProcessProperties	Diseño/ Desarrollo	Hecho
RT-42	Crear clase AssignmentRules	Diseño/ Desarrollo	Hecho
RT-43	Crear clase BPMNDynaformManagement	Diseño/ Desarrollo	Hecho
RT-44	Crear clase InputDocumentManagement	Diseño/ Desarrollo	Hecho
RT-45	Crear clase TriggersManagement	Diseño/ Desarrollo	Hecho

Tabla 82 Sprint 6 – Planificación  
Fuente: Elaboración Propia

### 3.3.6.2 DESARROLLO DEL SPRINT

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (Product Owner), y están planificados en el Roadmap del proyecto. Los Testing Scripts son desarrollados bajo estándares especificados más adelante.

### 3.3.6.3 REVISIÓN DEL SPRINT

Se realizaron los siguientes pasos y se introdujeron algunos datos para probar la funcionalidad desarrollada en este Sprint:

Prueba : 6.1

Operación:

Crear un Script de testeo de ejemplo

**Precondición:**

Tener el proyecto funcional y con las clases base creadas, wrappers y páginas, además de las

configuraciones iniciales para ejecución.

**Datos/pasos a realizar:**

- Ejecutar el script de testeo:

```
public class testingScriptExample extends Test {
    public testingScriptExample() throws Exception {

    }
    @Before
    public void setup(){
    }

    @After
    public void cleanup(){
        browserInstance.quit();
    }

    @org.junit.Test
    public void testProcess() throws FileNotFoundException, IOException, Exception{
        //STEP 0:login and searching the process: TestSupervisors
        pages.gotoDefaultUrl();
        pages.Login().loginUser("admin", "admin", "testing", "English");
        DesignerProcessList designerProcessList = pages.Main().goDesigner();
        ProcessDesignerBPMN processDesignerBPMN = designerProcessList.openBPMNProcess("new
        Process Testing");
        //end of STEP 0

        //Testing Asserts
        Assert.assertTrue("Description1", Condition1);
        Assert.assertTrue("Description2", Condition2);

    }
}
```

**Resultados esperados:**

Se ejecuta el script de testeo de ejemplo

**Post condiciones:**

Los scripts de testeo ya cuentan con una plantilla.

**Resultados obtenidos:**

Se obtienen los resultados esperados.

Tabla 83 Sprint 6 – Prueba: Creación de Script de Testeo  
Fuente: Elaboración Propia

### 3.3.7 SPRINT 7: ARMADO DE INFRAESTRUCTURA Y ESTABILIZACIÓN DEL FRAMEWORK

#### 3.3.7.1 PLANIFICACIÓN DEL SPRINT

La planificación para el desarrollo de los sprints se realizó en la reunión con los clientes (Product Owner).

Durante este sprint se realizó el armado de infraestructura sobre la cual el Framework se ejecutará, además de estabilizar las herramientas creadas en anteriores sprints.

SPRINT 7: ARMADO DE INFRAESTRUCTURA Y ESTABILIZACIÓN DEL FRAMEWORK				
Sprint	Inicio	Fin	Duración horas	Días de Trabajo
7	16 de Mayo, 2016	27 de Mayo, 2016	80	10 días hábiles
SPRINT BACKLOG				
Backlog-ID	TAREAS		TIPO	ESTADO
RT-47	Instalar Servidor de Integración Continua		Diseño/ Implementación	hecho
RT-48	Instalar Selenium HUB		Diseño/ Implementación	Hecho
RT-49	Instalar Nodos de Testeo		Diseño/ Implementación	Hecho
RT-50	Modificar archivos de configuración del Framework para apuntar al servidor de pruebas		Revisión/ Desarrollo	Hecho
RT-51	Modificar archivos base y herramientas de ayuda para ejecución correcta		Revisión/ Desarrollo	Hecho

Tabla 84 Sprint 7 – Planificación  
Fuente: Elaboración Propia

### **3.3.7.2 DESARROLLO DEL SPRINT**

El desarrollo del sprint se realizó verificando los criterios de aceptación de las historias de usuario, que son los requerimientos del cliente (product Owner), y están planificados en el Roadmap del proyecto, los detalles del desarrollo, se encuentran descritos más adelante.

### **3.3.7.3 REVISIÓN DEL SPRINT**

La revisión de este sprint sobre la implementación se detalla en capítulo posterior.

### **3.3.8 DESARROLLO DE LOS SPRINTS**

En esta sección se muestra y detalla el desarrollo de los sprints, por consiguiente, toda la documentación generada y entregables como ser diagramas de clase, implementación de patrones de diseño y todos los resultados obtenidos en cada una de las tareas detalladas en los sprints.

También se detalla el armado y diagrama de la infraestructura aplicada a las pruebas funcionales usando el Framework de Automatización y la integración continua usando la herramienta Jenkins.

#### **3.3.8.1 DISEÑO DE DIAGRAMA DE CLASES**

A continuación, se muestra el modelo de diagrama de clases donde se puede observar la creación de todas las clases que dependen de la clase base PMPage y conforman todas páginas del Framework en cuestión:

a) Diagrama de clases (Page-objects)

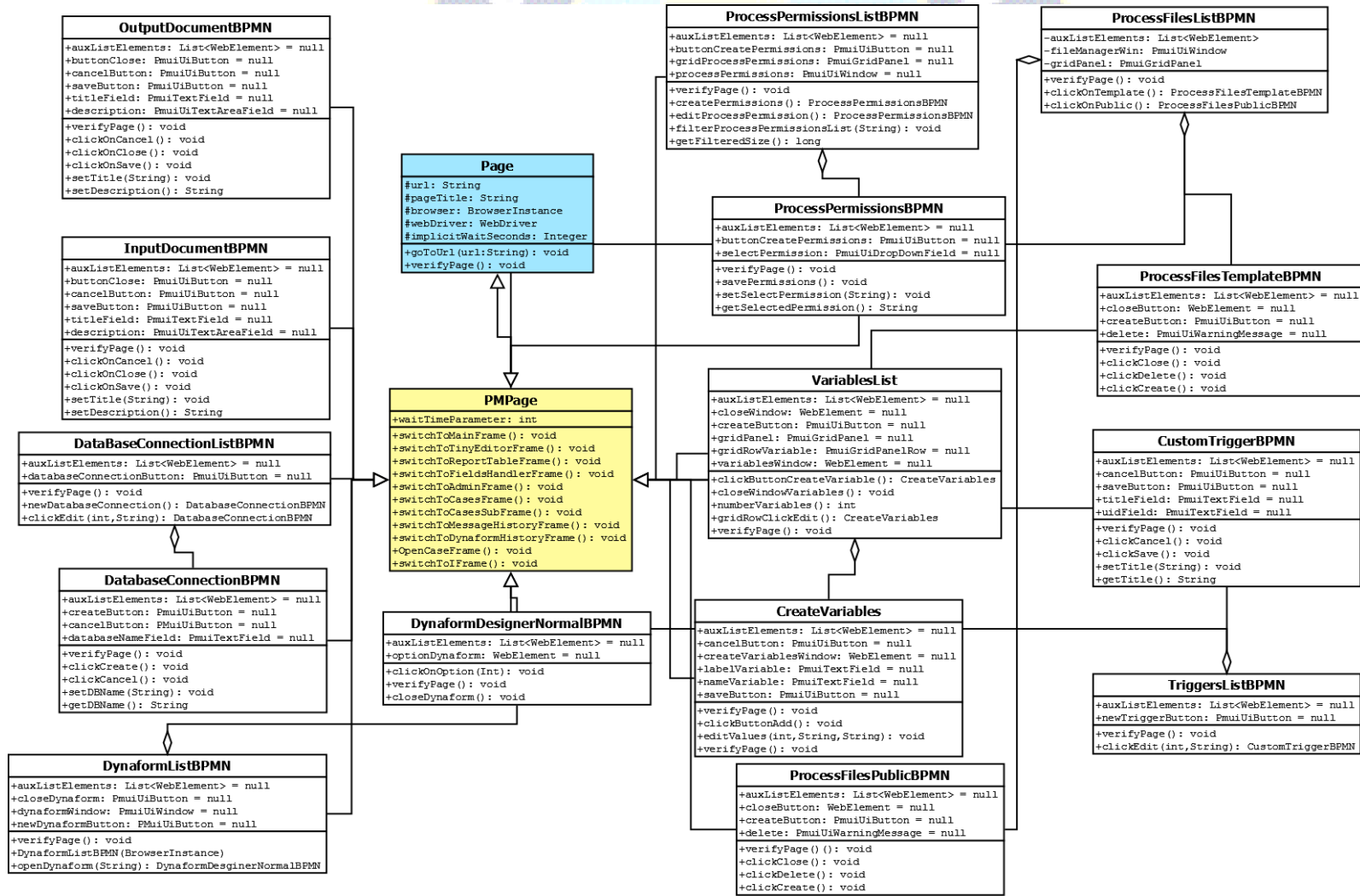


Figura 18.4 Diagrama de clases (Page-objects)  
Fuente: Elaboración Propia



b) Diagrama de clases (Testing Cases)

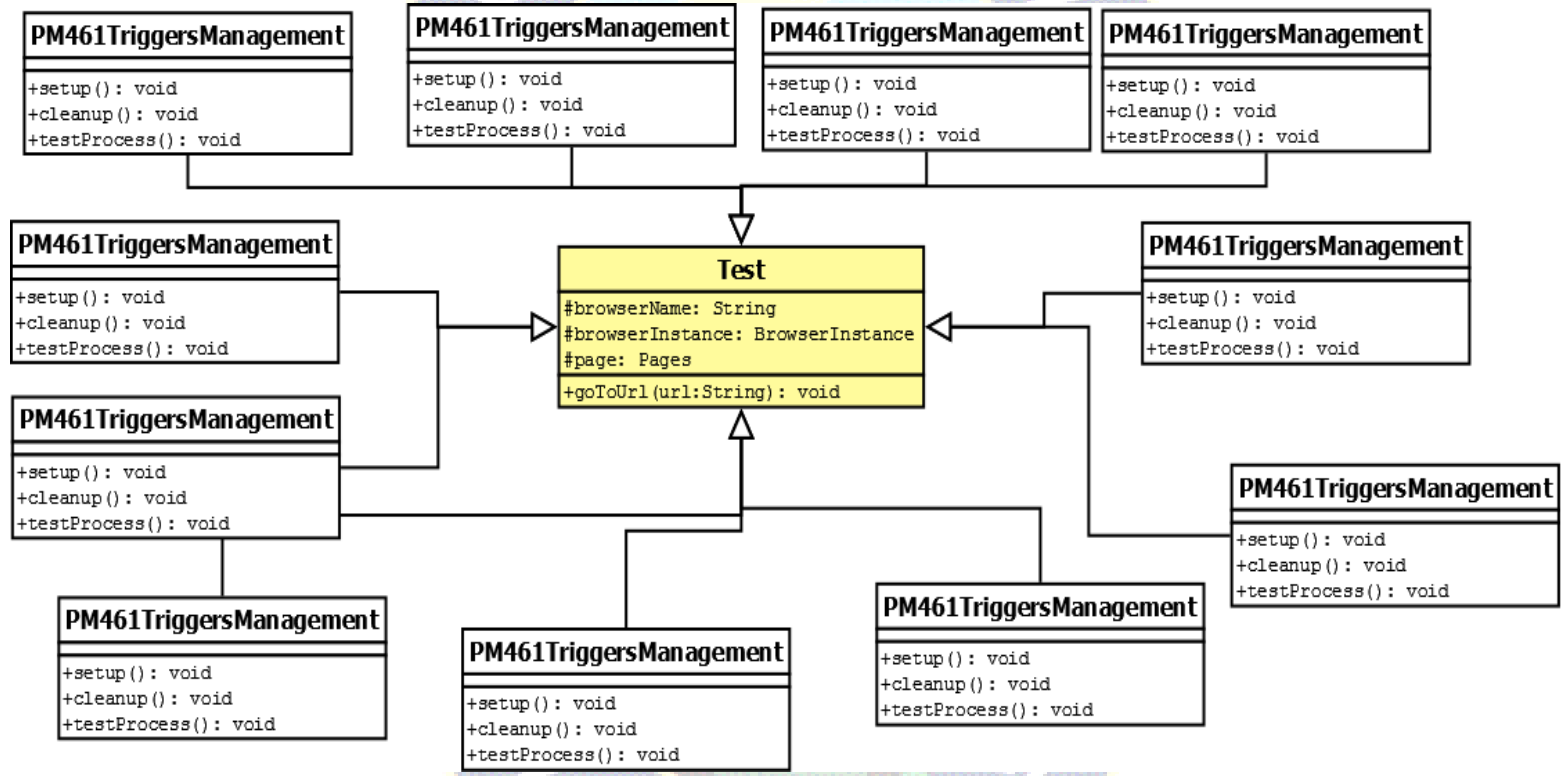


Figura 19.5 Diagrama de clases (Testing Cases)  
Fuente: Elaboración Propia

### 3.3.8.2 CREACIÓN DE CLASES WRAPPER

Java es un lenguaje de programación orientado a objetos. Un programa Java debe contener objetos y operaciones entre ellos. En ocasiones es muy conveniente poder tratar los datos primitivos (int, boolean, etc.) como objetos. Por ejemplo, los contenedores definidos por el API en el package java.util (Arrays dinámicos, listas enlazadas, colecciones, conjuntos, etc.) utilizan como unidad de almacenamiento la clase Object. Dado que Object es la raíz de toda la jerarquía de objetos en Java, estos contenedores pueden almacenar cualquier tipo de objetos. Pero los datos primitivos no son objetos, con lo que quedan en principio excluidos de estas posibilidades. Para resolver esta situación el API de Java incorpora las clases envoltorio (wrapper class), que no son más que dotar a los datos primitivos con un envoltorio que permita tratarlos como objetos. Por ejemplo, podríamos definir una clase envoltorio para los enteros, de forma bastante sencilla, con:

```
public class Entero {
    private int valor;

    Entero(int valor) {
        this.valor = valor;
    }

    int intValue() {
        return valor;
    }
}
```

Figura 20.6 Clase Envoltorio para un Entero  
Fuente: Elaboración Propia

Las clases envoltura se usan como cualquier otra:

```
Entero i = new Entero(5);
int x = i.intValue();
```

Figura 21.7 Uso de la clase Envoltorio  
Fuente: Elaboración Propia

De la misma forma se usa este tipo de clases para poder envolver a los elementos que componen el diseñador de Processmaker y tratarlos como objetos.

El diseñador de Processmaker ha sido desarrollado usando dos bibliotecas de desarrollo web, ExtJS y pmUI, desde un simple Botón hasta una Ventana completa. Todos los campos que componen esta funcionalidad necesitan ser tratados como objetos, ya que al momento de instanciarlos en las páginas que se desarrollarán más adelante se necesitan ejecutar algunas operaciones.

### 3.3.8.2.1 CLASES WRAPPER EXTJS

Los elementos Extjs que necesitan una clase de tipo “wrapper” son las siguientes:

- TextBox
- TextArea
- DropDown
- Button
- GridPanel
- MessageWindow

Todos éstos elementos forman parte del diseñador de procesos, el cual necesita una página que pueda abstraer los mismos para poder aplicar operaciones y obtener información de los mismos.

La imagen adjunta es un ejemplo completo de todo lo que se necesita implementar como clase de tipo “wrapper”, se pueden apreciar todos los elementos que intervienen en la creación de un proceso de tipo BPMN, el cual contiene un título, una descripción del proceso, la categoría a la cual pertenece, los botones de creación, todos éstos contenidos dentro de un MessageWindow y en el fondo se puede notar el GridPanel que contiene la lista de todos los procesos previamente creados.



Figura 22.8 Elementos de creación de un proceso BPMN (Processmaker)  
Fuente: Elaboración Propia

En este documento se mencionan algunas de las clases de tipo wrapper para ExtJS:

- ExtjsTextBox. El wrapper correspondiente para un TextBox es el siguiente:

Title:

```
<input type="text" size="100" autocomplete="off" maxlength="100" id="PRO_TITLE" name="PRO_TITLE" class="x-form-text x-form-field" style="width: 252px;">
```

Figura 23.9 TextBox y su respectivo elemento HTML  
Fuente: Elaboración Propia

Según el elemento HTML podemos diferenciar a este TextBox usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```
public class ExtJSTextBox {  
    BrowserInstance browserInstance = null;  
    WebDriver webDriver = null;  
  
    public ExtJSTextBox(BrowserInstance browserInstance, String idTextBox) throws Exception {  
    }  
  
    public void setTextBoxValue(String text) {  
    }  
  
    public String getTextBoxValue() {  
    }  
}
```

Figura 24.10 ExtJSTextBox wrapper  
Fuente: Elaboración Propia

b) ExtjsTextArea. El wrapper correspondiente para un TextArea es el siguiente:

Description:

```
<textarea style="width: 252px; height: 60px;" autocomplete="off" id="PRO_DESCRIPTION" name="PRO_DESCRIPTION" class="x-form-textarea x-form-field"></textarea>
```

Figura 25.11 TextArea y su respectivo elemento HTML  
Fuente: Elaboración Propia

Según el elemento HTML podemos diferenciar a este TextArea usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```
public class ExtJSTextArea {
    BrowserInstance browserInstance = null;
    WebDriver webDriver = null;

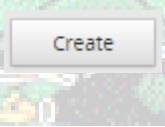
    public ExtJSTextArea(BrowserInstance browserInstance, String idTextArea) throws Exception {
    }

    public void setTextAreaValue(String text){
    }

    public String getTextAreaValue(){
    }
}
}
```

Figura 26.12 ExtJSTextArea wrapper  
Fuente: Elaboración Propia

c) ExtjsButton. El wrapper correspondiente para un Button es el siguiente:



```
<button type="button" id="ext-gen107" class=" x-btn-text">Create</button>
```

Figura 27.13 Button y su respectivo elemento HTML  
Fuente: Elaboración Propia

Según el elemento HTML podemos diferenciar a este Button usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```
public class ExtjsButton {
    BrowserInstance browserInstance = null;
    WebDriver webDriver = null;

    public ExtjsButton(BrowserInstance browserInstance, String id) throws Exception {
    }

    public void click(){
    }
}
```

Figura 28.14 ExtJSButton wrapper  
Fuente: Elaboración Propia

### 3.3.8.2.2 CLASES WRAPPER PMUI

Los elementos pmUI que necesitan una clase de tipo “wrapper” son las siguientes:

- TextBoxField
- PasswordField
- TextAreaField
- DropDownField
- RadioField
- DateTimeField
- ContextMenu
- Button
- GridPanel
- TabPanel
- TreePanel
- MessageWindow

- Window

Todos éstos elementos forman parte del diseñador de procesos, el cual necesita una página que pueda abstraer los mismos para poder aplicar operaciones y obtener información de los mismos, a diferencia de ExtJS, pmUI posee más elementos dentro del diseñador de procesos de Processmaker.

La imagen adjunta es un ejemplo casi completo de todo lo que se necesita implementar como clase de tipo “wrapper”, se pueden apreciar algunos de los elementos que intervienen en la configuración de un proceso BPMN:

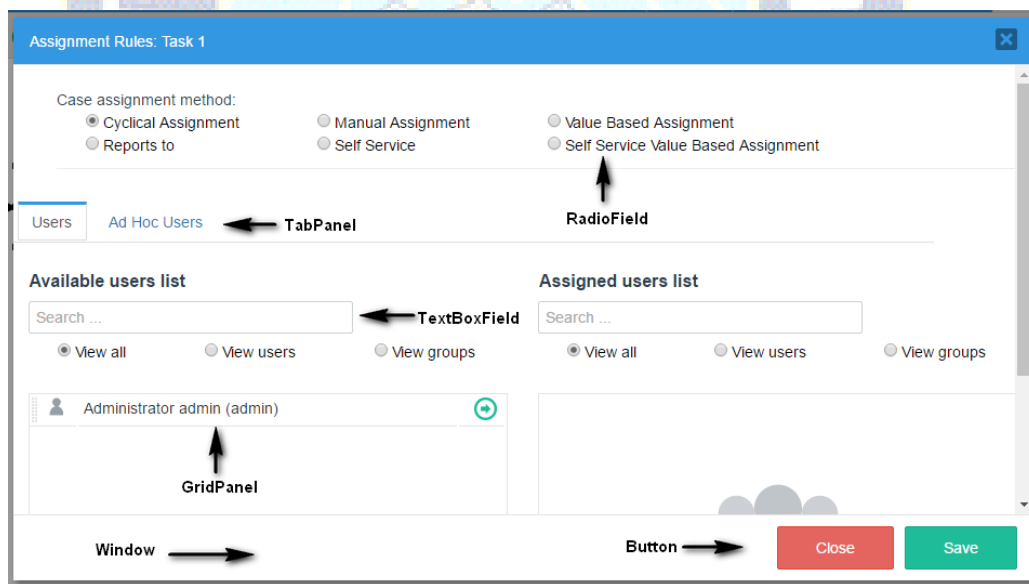


Figura 29.15 Elementos de configuración de un proceso BPMN (Processmaker)

Fuente: Elaboración Propia

En este documento se mencionan algunas de las clases de tipo wrapper para pmUI:

- a) RadioField. El wrapper correspondiente para un RadioField es el siguiente:



Case assignment method:

- Cyclical Assignment       Manual Assignment       Value Based Assignment  
 Reports to       Self Service       Self Service Value Based Assignment

```
<div id="formTasAssignType" class="pmui pmui-radiobuttongroupfield pmui-field" style="left: 0px; top: 0px; width: 890px; display: block; position: relative; z-index: auto;">
```

Figura 30.16 RadioField y su respectivo elemento HTML

Fuente: Elaboración Propia

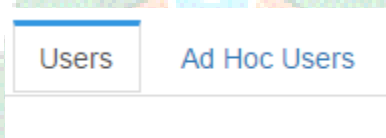
Según el elemento HTML podemos diferenciar a este RadioField usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```
public class PmuiUIRadioButtonGroupField {  
    WebElement radioButtonGroupField = null;  
    BrowserInstance browserInstance = null;  
  
    public PmuiUIRadioButtonGroupField(BrowserInstance browserInstance, String id) throws Exception{  
        /*CODE HERE*/  
    }  
  
    public void selectRadioOptionByValue(String valor){  
        /*CODE HERE*/  
    }  
  
    public String getRadioOptionByValue(){  
        /*CODE HERE*/  
    }  
  
    public void click(int index){  
        /*CODE HERE*/  
    }  
}
```

Figura 31.17 PmuiUIRadioButtonGroupField wrapper

Fuente: Elaboración Propia

b) TabPanel. El wrapper correspondiente para un TabPanel es el siguiente:



```

▼<div class="pmui-tabpanel pmui-tabpanel-top pmui pmui-empty" id=
"tabPanelAssignmentRules" style="left: 0px; top: 0px; width: 100%; height:
auto; position: relative; z-index: auto; box-sizing: border-box;">

```

Figura 32.18 TabPanel y su respectivo elemento HTML  
Fuente: Elaboración Propia

Según el elemento HTML podemos diferenciar a este TabPanel usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```

public class PmuiUITabPanel {
    BrowserInstance browserInstance = null;
    WebDriver webDriver = null;

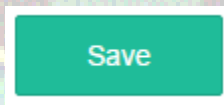
    public PmuiUITabPanel(BrowserInstance browserInstance,String id) throws Exception {
        /*CODE HERE*/
    }

    public void clicktabPanel(int index){
        /*CODE HERE*/
    }
}

```

Figura 33.19 PmuiUITabPanel wrapper  
Fuente: Elaboración Propia

c) Button. El wrapper correspondiente para un Button es el siguiente:



```

▼<a id="windowPropertiesButtonSave" class="pmui pmui-button pmui-success"
href="#" title style="left: 0px; top: 0px; width: auto; line-height: normal;
position: relative; height: auto; z-index: auto;">

```

Figura 34.20 Button pmUI y su respectivo elemento HTML  
Fuente: Elaboración Propia

Según el elemento HTML podemos diferenciar a este Button usando su “id” o “className”, en la siguiente imagen podemos ver el desarrollo de la clase “wrapper” usando como identificador el id del elemento:

```

public class PmuiUIButton {
    BrowserInstance browserInstance = null;
    WebDriver webDriver = null;

    public PmuiUIButton(BrowserInstance browserInstance, String id) throws Exception {
        /*CODE HERE*/
    }

    public WebElement getWebElement(){
        /*CODE HERE*/
    }

    public void click(){
        /*CODE HERE*/
    }
}

```

Figura 35.21 Button pmUI wrapper  
Fuente: Elaboración Propia

### 3.3.8.3 USO DE PATRÓN DE DISEÑO PAGE-OBJECT

Una vez creadas las clases de tipo “wrapper”, se debe tomar el siguiente paso del proyecto, el cual se define como la creación de páginas usando el patrón de diseño Page Object Pattern, que no es otra cosa que la mejor opción para implementar y una de las mejores prácticas de Selenium WebDriver.

Todas las clases “wrapper” serán usadas en las páginas de testeo usando la siguiente estructura detallada en la siguiente sección.

#### 3.3.8.3.1 ESTRUCTURA DE UNA PÁGINA

Para que un proyecto de software adquiera una categoría que todos los interesados desean, se necesitan estandarizar varios aspectos y definir una forma correcta de trabajo, esto, para que la herramienta en cuestión pueda ser revisada y extendida por otras personas ajenas al proyecto. Para lo cual se definió una estructura básica de una página:

```

class Page1{
    WebElement nameField;
    WebElement mailField;
    ...
    verifyPage(){
    }
    setName(){
    }
    ...
}

```

← Elements Definition

← Verify page and WebElements Assignment

← all methods

Figura 36.22 Estructura de una página bajo el patrón Page Object Pattern  
Fuente: Elaboración Propia

Consta de tres partes:

- **Definición de los elementos.** En esta sección se definen todos los elementos que van a interactuar con nuestros test, en palabras más sencillas, se define los nombres de todos los objetos que invocaran a las clases wrapper.
- **Verificación de página.** En esta sección se hace una verificación de todos los elementos existentes, es la primera prueba que se ejecuta sobre cada una de las páginas, si un elemento que fue recientemente removido, esta sección lo detectará y lanzará el primer error en la automatización, indicando que el elemento ya no está presente. Una vez encontrados todos los elementos, se hace la asignación de los objetos definidos en la anterior sección a sus respectivos wrappers.
- **Creación de Métodos.** Una vez asignados los objetos a sus respectivos wrappers, se realiza la creación de todos los métodos que realizará la página, es decir, todas las acciones que la página podrá efectuar al momento de ejecutar los test.

Se tiene la página de Creación de variables las cuales, dentro de Processmaker juegan un papel importante al ser las portadoras de datos de los procesos. Se muestra a continuación la página con todos sus componentes:

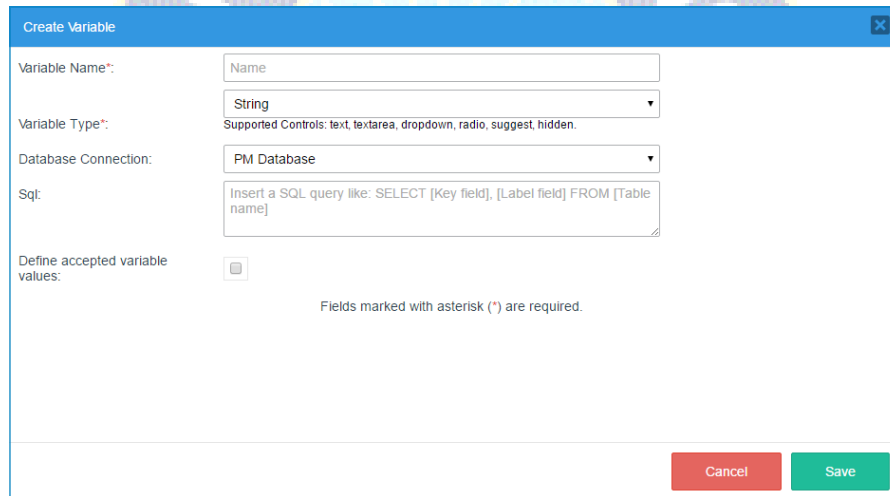


Figura 37.23 Página de Creación de Variable  
Fuente: Elaboración Propia

La estructura de la página será la siguiente:

```
public class ProcessVariables extends Page {

    /*ALL DEFINITIONS HERE*/

    public ProcessVariables(BrowserInstance browser) throws Exception {
        super(browser);
        verifyPage();
    }

    @Override
    public void verifyPage() throws Exception {
        /*CODE HERE*/
    }

    /*ALL METHODS SECTION HERE*/

}
```

Figura 38.24 Creación de Variable bajo el patrón Page Object Pattern  
Fuente: Elaboración Propia

La primera sección de Definición de los elementos se muestra de la siguiente forma:

```
public class ProcessVariables extends Page {
    //elementos auxiliares
    List<WebElement> auxListElements = null;
    //Variable name
    PmuiTextBoxField variableName = null;
    //Variable Type
    PmuiUiDropDownField variableType = null;
    //Database Connection
    PmuiUiDropDownField dbConnection = null;
    //SQL
    PmuiTextAreaField sqlQuery = null;
    //Define accepted variable values
    PmuiCheckGrooupField acceptedValues = null;
    //CANCEL Button
    PmuiUIButton cancelButton = null;
    //SAVE Button
    PmuiUIButton saveButton = null;
}
```

Figura 39.25 Primera sección de Creación de Variables bajo el patrón Page Object Pattern  
Fuente: Elaboración Propia

En esta primera sección se hace la definición de todos los objetos que intervendrán e interactuarán con los test, como se puede ver todos los objetos son de tipo wrapper y se detallan cada uno de los objetos con comentarios indicando a que objeto de la página pertenecen.

La segunda sección de Verificación de página se muestra de la siguiente forma:

```
@Override
public void verifyPage() throws Exception {
    /*CODE HERE*/
}
```

Figura 40.26 Segunda sección de Creación de Variables bajo el patrón Page Object Pattern  
Fuente: Elaboración Propia

Dentro de esta sección se realiza la implementación de métodos Wrapper en Objetos de página, se detalla a continuación cada uno de ellos:

- a) Variable Name. El nombre de la variable es lo que se usará en todo el proceso para citar a la misma, a continuación, se ve el campo y su respectivo elemento HTML:

Variable Name\*:

```
><div id="variableName" class="pmui pmui-textfield pmui-field" style="left: 0px; top: 0px; width: auto; display: block; position: relative; z-index: auto;">...</div>
```

Figura 41.27 Campo Variable Name  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```
auxListElements = webDriver.findElements(By.id("variableName"));  
if(auxListElements.size()>0) {  
    variableName = new PmuiTextBoxField(browser, "variableName");  
    Logger.addLog("Variable Name Found");  
}  
else{  
    throw new Exception("Variable Name not found");  
}
```

Figura 42.28 Asignación de Variable Name a su respectivo Objeto  
Fuente: Elaboración Propia

- b) Variable Type: Se refiere al tipo de variable. Una variable puede ser tipo “String”, Integer, DateTime, etc. a continuación, se ve el campo y su respectivo elemento HTML:

Variable Type\*:  Supported Controls: text, textarea, dropdown, radio, suggest, hidden.

```
▶<div id="varType" class="pmui pmui-dropdownlistfield pmui-field" style="left: 0px; top: 0px; width: auto; display: block; position: relative; z-index: auto;">...</div>
```

Figura 43.29 Campo Variable Type  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```
auxListElements = webDriver.findElements(By.id("varType"));  
if(auxListElements.size()>0){  
    variableType = new PmuiUiDropDownField(browser, "varType");  
    Logger.addLog("Variable Type Found");  
}  
else{  
    throw new Exception("Variable Type not found");  
}
```

Figura 44.30 Asignación de Variable Type a su respectivo Objeto  
Fuente: Elaboración Propia

- c) Database Connection. Se refiere a la conexión a base de datos ligada a la variable. A continuación, se ve el campo y su respectivo elemento HTML:

Database Connection:

```
▶<div id="varConnection" class="pmui pmui-dropdownlistfield pmui-field" style="left: 0px; top: 0px; width: auto; display: block; position: relative; z-index: auto;">...</div>
```

Figura 45.31 Campo Database Connection  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.



```

auxListElements = webDriver.findElement(By.id("varConnection"));
    if(auxListElements.size()>0){
        dbConnection = new PmuiUiDropDownField(browser, "varConnection");
        Logger.addLog("Database Connection Found");
    }
    else{
        throw new Exception("Database Connection not found");
    }
}

```

Figura 46.32 Asignación de Database Connection a su respectivo Objeto  
Fuente: Elaboración Propia

- d) SQL. Se refiere a la consulta SQL que la variable creada contendrá. A continuación, se ve el campo y su respectivo elemento HTML:

Sql:

Insert a SQL query like: SELECT [Key field], [Label field] FROM [Table name]

```

▶<div id="varSql" class="pmui pmui-textareafield mafe-textarea-resize pmui-field" style="left: 0px; top: 0px; width: auto; display: block; position: relative; z-index: auto;">...</div>

```

Figura 47.33 Campo SQL  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```

auxListElements = webDriver.findElement(By.id("varSql"));
    if(auxListElements.size()>0){
        sqlQuery = new PmuiTextAreaField(browser, "varSql");
        Logger.addLog("Sql field Found");
    }
    else{
        throw new Exception("Sql field not found");
    }
}

```

Figura 48.34 Asignación de SQL field a su respectivo Objeto  
Fuente: Elaboración Propia

- e) Define Accepted Variable Values. Indica si la variable a crear tendrá valores por defecto. A continuación, se ve el campo y su respectivo elemento HTML:

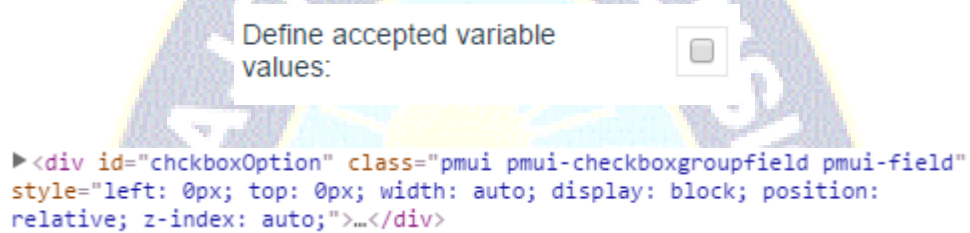


Figura 49.35 Campo Define Accepted variable values  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```
auxListElements = webDriver.findElements(By.id("chckboxOption"));  
if(auxListElements.size()>0){  
    acceptedValues = new PmuiCheckGrooupField(browser, "chckboxOption");  
    Logger.addLog("Accepted Values Found");  
}  
else{  
    throw new Exception("Accepted Values not found");  
}
```

Figura 50.36 Asignación de Accepted Variable Values a su respectivo Objeto  
Fuente: Elaboración Propia

- f) Cancel Button. Cancela la creación de la variable. A continuación, se ve el botón y su respectivo elemento HTML:

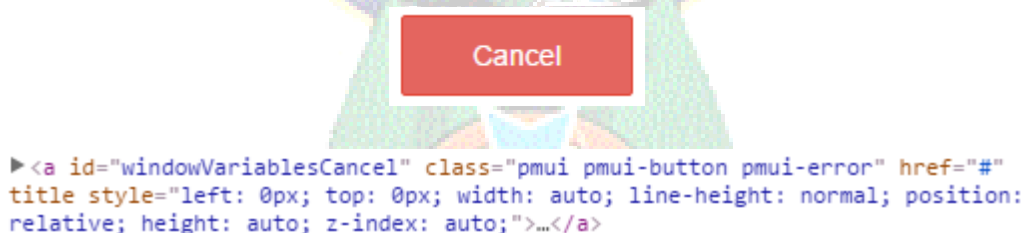


Figura 51.37 Cancel Button  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```
auxListElements = webDriver.findElements(By.id("windowVariablesCancel"));  
if(auxListElements.size()>0){  
    cancelButton = new PmuiUIButton(browser, "windowVariablesCancel");  
    Logger.addLog("Cancel Button Found");  
}  
else{  
    throw new Exception("Cancel Button not found");  
}
```

Figura 52.38 Asignación de Cancel Button a su respectivo Objeto  
Fuente: Elaboración Propia

- g) Save Button. Guarda y confirma la creación de la variable. A continuación, se ve el botón y su respectivo elemento HTML



```
><a id="windowVariablesSave" class="pmui pmui-button pmui-success" href="#"  
title style="left: 0px; top: 0px; width: auto; line-height: normal; position:  
relative; height: auto; z-index: auto;">...</a>
```

Figura 53.39 Save Button  
Fuente: Elaboración Propia

Como se puede ver el campo tiene un id, el cual puede ser usado como identificador al momento de asignar este campo a su respectivo objeto.

```
auxListElements = webDriver.findElements(By.id("windowVariablesSave"));  
if(auxListElements.size()>0){  
    saveButton = new PmuiUIButton(browser, "windowVariablesSave");  
    Logger.addLog("Save Button Found");  
}  
else{  
    throw new Exception("Save Button not found");  
}
```

Figura 54.40 Asignación de Save Button a su respectivo Objeto  
Fuente: Elaboración Propia

Finalmente, la tercera sección de Creación de Métodos se muestra de la siguiente forma:

- a) Establecer valores(“Setters”). El siguiente método se encarga de establecer valores:

```
public void setVariableValues(String nVar, String tVar, String dbVar, String sqlVar, boolean acVar){
    variableName.setValue(nVar);
    variableType.selectByValue(tVar);
    dbConnection.selectByValue(dbVar);
    sqlQuery.setValue(sqlVar);
    acceptedValues.setValue(acVar);
}
```

Figura 55.41 Método para establecer valores  
Fuente: Elaboración Propia

- b) Recuperar valores(“Getters”). Los siguientes métodos son los que se encargan de obtener los valores previamente establecidos, esto, con el objetivo de usar en aserciones al momento de ejecutar las pruebas.

```
public String getVariableName () {
    variableName.getValue ();
}
public String getVariableType () {
    variableType.getSelectedValue ();
}
public String getDbConnection () {
    dbConnection.getSelectedValue ();
}
public String getSqlQuery () {
    sqlQuery.getValue ();
}
public boolean getAcceptedValue () {
    acceptedValues.getValue ();
}
```

Figura 56.42 Métodos para recuperar valores  
Fuente: Elaboración Propia

- c) Acciones de los botones. Los siguientes métodos se encargan de ejecutar acciones sobre los botones de la página:

```
public void clickOnCancel () {  
    cancelButton.click ();  
}  
  
public void clickOnSave () {  
    saveButton.click ();  
}
```

Figura 57.43 Métodos para ejecutar acciones sobre los botones  
Fuente: Elaboración Propia

### 3.3.8.4 SCRIPTS DE TESTEO

Un Script de testeo es la secuencia de comandos de prueba que se ejecuta sobre un sistema para comprobar que funciona correctamente.

En nuestro caso, una vez terminada la creación de clases wrapper y el diseño de Páginas usando el patrón Page Object Pattern, se procede a la creación de los scripts de testeo que son la razón principal del presente proyecto.

Un Script de testeo debe tener una estructura estándar, con el objetivo de que todos los usuarios encargados de desarrollar los scripts sigan esta tendencia, para que el código pueda ser fácilmente entendido y si es posible pueda ser extensible. A continuación, se define la estructura de un Script de testeo.

#### 3.3.8.4.1 ESTRUCTURA DE UN SCRIPT DE TESTEO

Un Script de testeo básicamente se conforma de dos secciones, detalladas a continuación:

```

SCRIPT1{
    login();
    goToDesigner();
    setName("process1");
    setDescription("testing only")
    ...
    assert.assertTrue(getName == "process1")
    ...
}

```

← Commands

← Asserts

Figura 58.44 Estructura de un Script de Testeo  
Fuente: Elaboración Propia

Consta de dos partes:

- **Comandos.** En esta sección se hace una invocación de la página que se desea testear, por consiguiente, tenemos todos los objetos con los que el Script de testeo interactuará y además los métodos que son ejecutados para seguir cada uno de los pasos de un caso de prueba.
- **Asserts.** Esta es la sección más importante ya que se configuran las condiciones que harán que el Script devuelva un error en caso de que haya un bug en la funcionalidad de Processmaker o en un sentido positivo, continúe sin reportar ningún error. Un assert tiene la siguiente estructura:

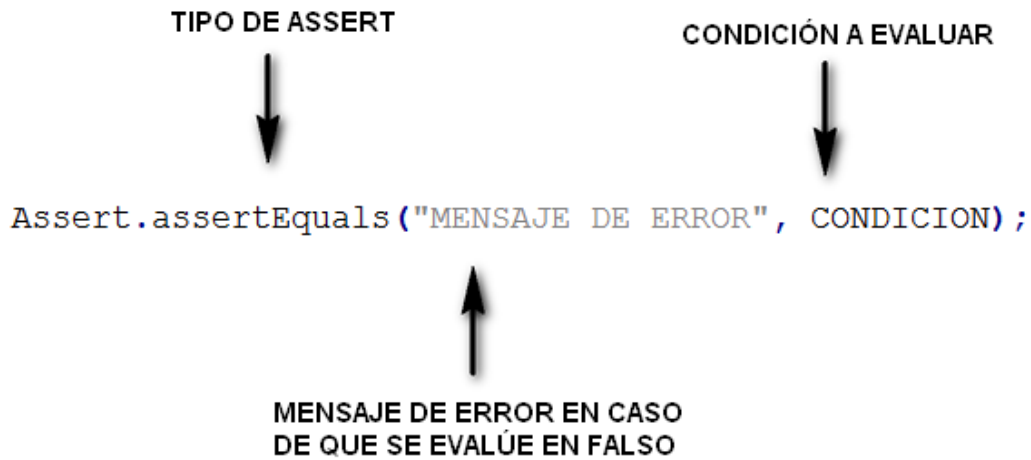


Figura 59.45 Estructura de un Assert  
Fuente: Elaboración Propia

A continuación, se muestra el armazón de un Script de testeo que prueba la funcionalidad de Creación de Variables:

```
public class PM605VariablesManagement extends com.colosa.qa.automatization.tests.common.Test{
    public PM605VariablesManagement() throws Exception {
    }

    @Before
    public void setup(){
    }

    @After
    public void cleanup(){
        browserInstance.quit();
    }

    @Test
    public void testExampleDynaform() throws FileNotFoundException, IOException, Exception{
        /*COMMANDS HERE*/

        /*ASSERTS HERE*/
    }
}
```

Figura 60.46 Script de testeo para Creación de Variables  
Fuente: Elaboración Propia

Los Comandos para la creación de una variable se detallan a continuación:

```

//Abre la ventana "Create Variable"
ProcessVariables createVariables = processDesignerBPMN.openVariables();
//Establece los valores
createVariables.setVariableValues("variable1"
    , "String"
    , "PMDatabase"
    , "SELECT * FROM USERS"
    , true);
//Realiza un click sobre el botón SAVE
createVariables.clickOnSave();

```

Figura 61.47 Comandos para la creación de variables  
Fuente: Elaboración Propia

Los Asserts para verificar la funcionalidad se detallan a continuación, cabe mencionar que al ejecutar los Asserts, el Framework nos devuelve un valor, en este caso “True” o “False”:

```

//Abre la ventana "Create Variable"
ProcessVariables createVariables = processDesignerBPMN.openVariables();
//Recupera los valores y ademas comprueba que sean los correctos
Assert.assertEquals("ERROR: No es el mismo Nombre", createVariables.getVariableName(), "variable1");
Assert.assertEquals("ERROR: No es el mismo tipo", createVariables.getVariableType(), "String");
Assert.assertEquals("ERROR: No es la misma DB Connection", createVariables.getDbConnection(), "PMDatabase");
Assert.assertEquals("ERROR: No es el mismo Query", createVariables.getSqlQuery(), "SELECT * FROM USERS");
Assert.assertEquals("ERROR: No esta seleccionada esta opcion", createVariables.getAcceptedValue());

```

Figura 62.48 Asserts para la verificación de creación de variables  
Fuente: Elaboración Propia

### 3.3.8.5 SELENIUM GRID

Una vez completado el Framework de Automatización de Pruebas Funcionales, se tiene una configuración tradicional para ejecución simple de pruebas, tal como muestra la imagen adjunta:



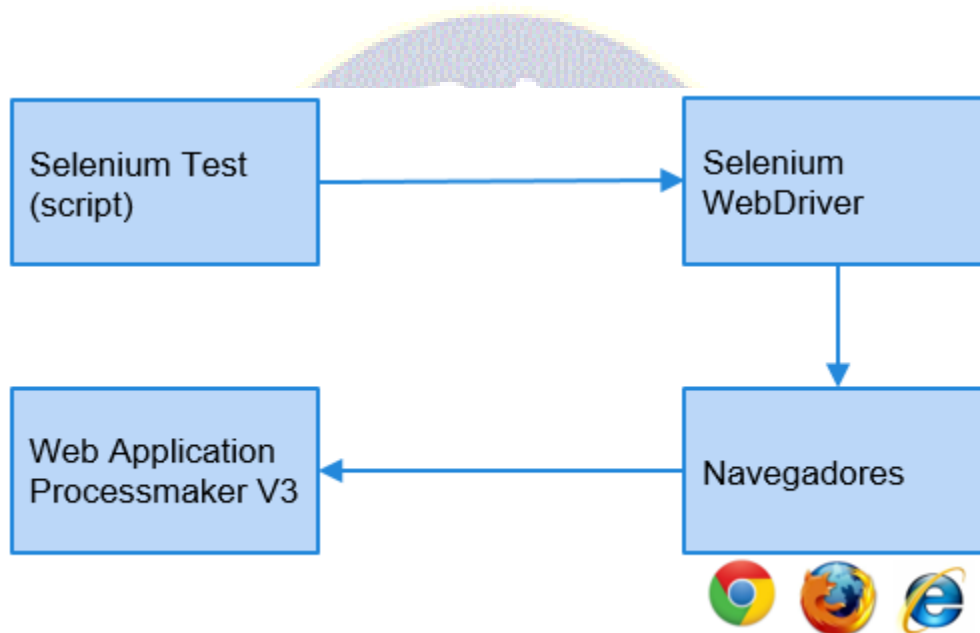


Figura 63.49 Configuración simple para ejecución de Pruebas  
Fuente: Elaboración Propia

Pero para realizar un Testing completo de tipo “crossbrowsing”, se necesita levantar una infraestructura de testeo. A esta infraestructura se denomina Selenium Grid el cual consta de un hub (Selenium Hub) y Nodos de testeo.

### 3.3.8.5.1 SELENIUM HUB

Selenium Hub es el centro de mando principal para administrar las máquinas (físicas o virtuales) en las que se ejecutará las pruebas de Selenium. Aquí es donde se puede ver la lista de todos los nodos registrados disponibles actualmente en ejecución y en espera de recibir órdenes de ejecución.

Cuando se ejecuta un test se pasan algunos parámetros a través de un Script de testeo, esto va hacia el Hub y su trabajo es encontrar nodos disponibles, si se encuentra uno, este se encarga de ejecutar las pruebas.

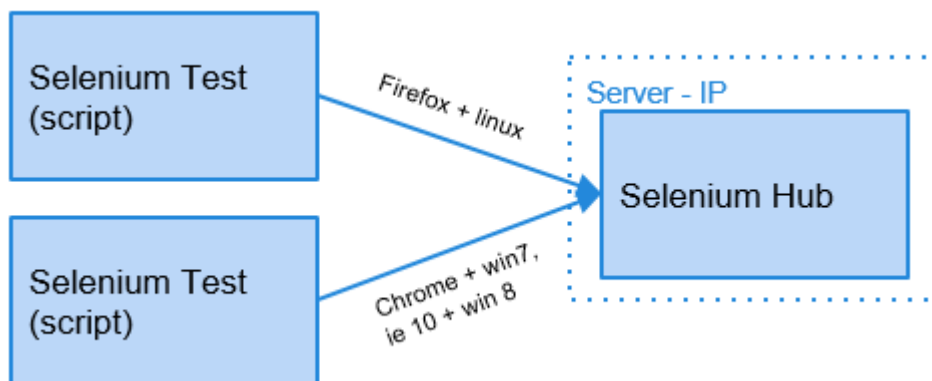


Figura 64.50 Envío de Scripts de testeo a un Hub  
Fuente: Elaboración Propia

La forma de lanzar la ejecución del servidor es usando el siguiente comando en la terminal del servidor donde se ubica el Selenium Hub:

```
java -jar selenium-server-standalone-2.42.2.jar -port 4444
```

Figura 65.51 Comando para iniciar el Hub en el servidor  
Fuente: Elaboración Propia

### 3.3.8.5.2 NODOS DE TESTEO

Un nodo es una máquina (física o virtual) que se registra en el Hub. Una vez registrado, el Hub sabe que existe el nodo y éste está disponible para recibir órdenes. Se necesita ejecutar un comando para levantar el servidor como nodo y registrar en el hub:

```
java -jar selenium-server-standalone-2.53.0.jar -role node -hub http://<ip_hub>:<puerto_hub>  
-host <ip_máquina_virtual> -port <puerto_máquina_virtual>
```

Figura 66.52 Comando para iniciar un nodo y registrar en el hub  
Fuente: Elaboración Propia

De esta forma se tendrán varios nodos funcionando al mismo tiempo y a la espera de los test que mande el Hub, tal como indica el gráfico siguiente:

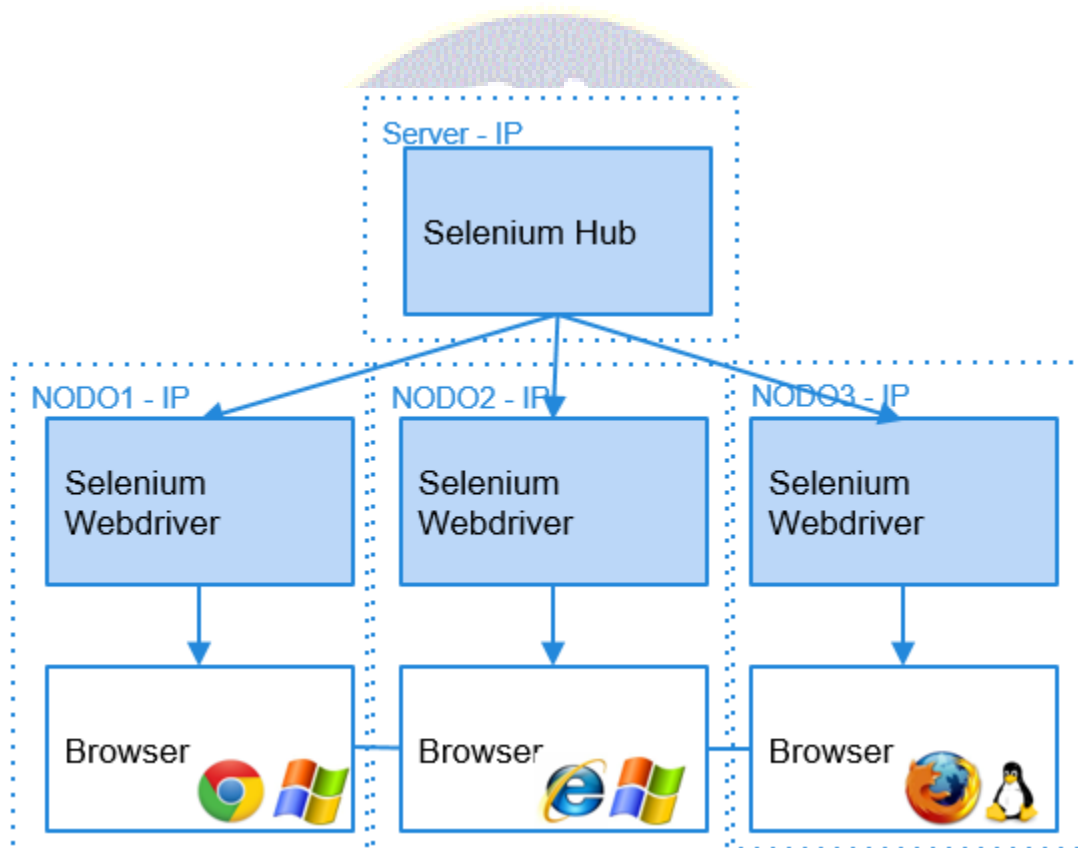


Figura 67.53 Envío de Scripts de testeo de un Hub a los nodos  
Fuente: Elaboración Propia

### 3.3.8.6 SERVIDOR DE INTEGRACIÓN CONTINUA

Como se mencionó en el capítulo 2, para completar el proyecto se necesita **La Integración Continua**, la cual no es más que tener todos los procesos automatizados, y ejecutándose periódicamente o en una secuencia definida por el interesado, de esta forma se puede garantizar el testeo constante de la herramienta.

#### 3.3.8.6.1 INSTALACION DE JENKINS

La herramienta de integración continua **Jenkins**, permite al proceso de desarrollo automatizar tareas, tales como la descarga automática de código y la ejecución de todos los test cases usando al Hub como mediador entre el servidor de integración continua y los nodos de testeo.

Usando tecnologías tales como GIT (software de control de versiones) y la herramienta Maven mencionada en el capítulo 2 se tiene un Servidor de integración continua en producción y trabajando a favor del proceso de desarrollo.

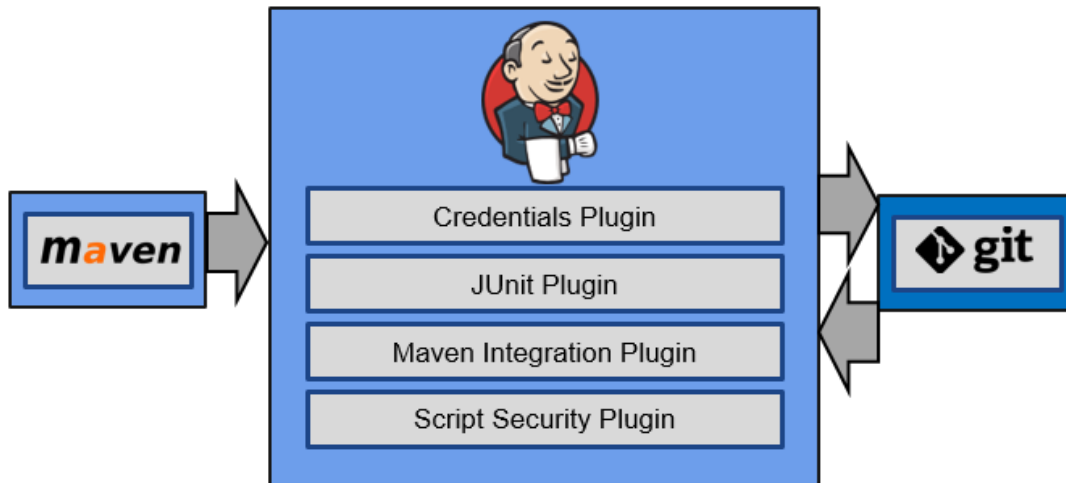


Figura 68.54 Jenkins y plugins necesarios para automatización  
Fuente: Elaboración Propia

El producto final se puede ver reflejado en el siguiente gráfico, mostrando los componentes del Framework de Automatización de pruebas funcionales:

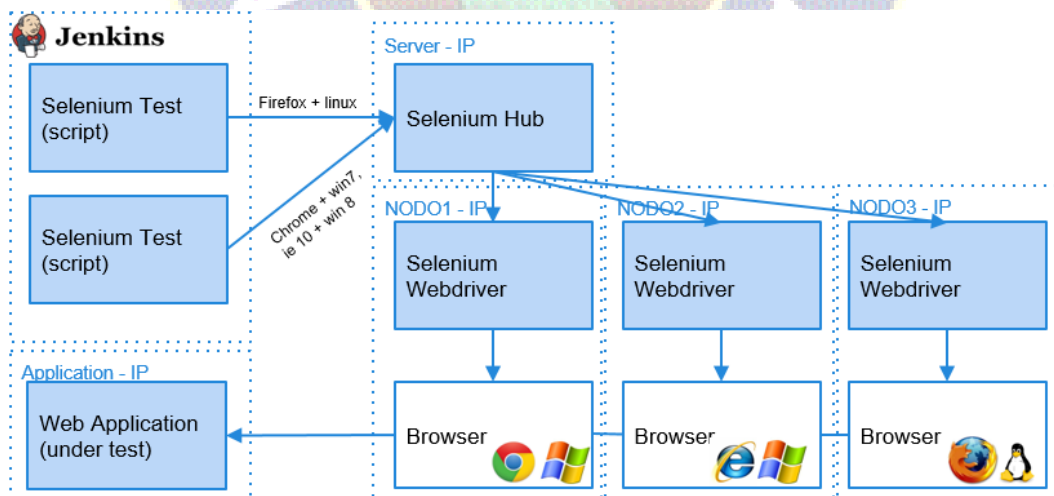


Figura 69.55 Componentes del Framework de Automatización  
Fuente: Elaboración Propia

### 3.4 POST – GAME

Todas las especificaciones definidas en la fase GAME fueron completadas y se tiene un Framework de Automatización listo para poner a producción.

En la imagen adjunta se puede ver el inicio de la compilación de los tests:



The screenshot shows the Jenkins web interface for a build job. The job name is "Compilar #19 (31-oct-2016 15:52:47)". The progress bar is partially filled, and the status is "Comenzó hace 2,7 Seg" (Started 2.7 seconds ago). The console output is currently empty, showing "Sin determinar" (Undetermined). The modules section shows "RoboTest (No se ejecutó)" (Not executed).

Figura 70.56 Compilación de tests para iniciar el testeo  
Fuente: Elaboración Propia

Al ejecutar los tests se puede ir verificar el estado de los mismo mediante la salida en consola, ésta puede informar que todo esta por buen camino y que el objetivo esta en curso:



The screenshot shows the Jenkins web interface for the same build job, but now the console output is visible. The output shows the execution of Maven commands to fetch and checkout the test code. The status is "Ejecución previa" (Previous execution).

```
Lanzada por el usuario jenkins
Ejecutando en el espacio de trabajo /var/lib/jenkins/workspace/testing
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url git@bitbucket.org:rcgongora/robotest.git # timeout=10
Fetching upstream changes from git@bitbucket.org:rcgongora/robotest.git
> git --version # timeout=10
> git fetch --tags --progress git@bitbucket.org:rcgongora/robotest.git +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision fd1a58b99540227ecbb47c6864844c2cf02ee3db (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f fd1a58b99540227ecbb47c6864844c2cf02ee3db
> git rev-list fd1a58b99540227ecbb47c6864844c2cf02ee3db # timeout=10
Parsing POMs
Established TCP socket on 38571
[testing] $ /usr/lib/jvm/java-8-oracle/bin/java -cp /var/lib/jenkins/plugins/maven-plugin/WEB-INF/lib/maven3-agent-1.7.jar:/usr/share/maven/boot/plexus-classworlds-2.x.jar org.jvnet.hudson.maven3.agent.Maven3Main /usr/share/maven/var/cache/jenkins/war/WEB-INF/lib/remoting-2.62.jar /var/lib/jenkins/plugins/maven-plugin/WEB-INF/lib/maven3-interceptor-1.7.jar /var/lib/jenkins/plugins/maven-plugin/WEB-INF/lib/maven3-interceptor-commons-1.7.jar 38571
<====[JENKINS REMOTING CAPACITY]====>channel started
Executing Maven: -B -f /var/lib/jenkins/workspace/testing/pom.xml test
```

Figura 71.57 Salida de consola  
Fuente: Elaboración Propia

Finalmente, los resultados finales se ven reflejados en el siguiente gráfico, indicando el estado de las pruebas según la última compilación realizada:

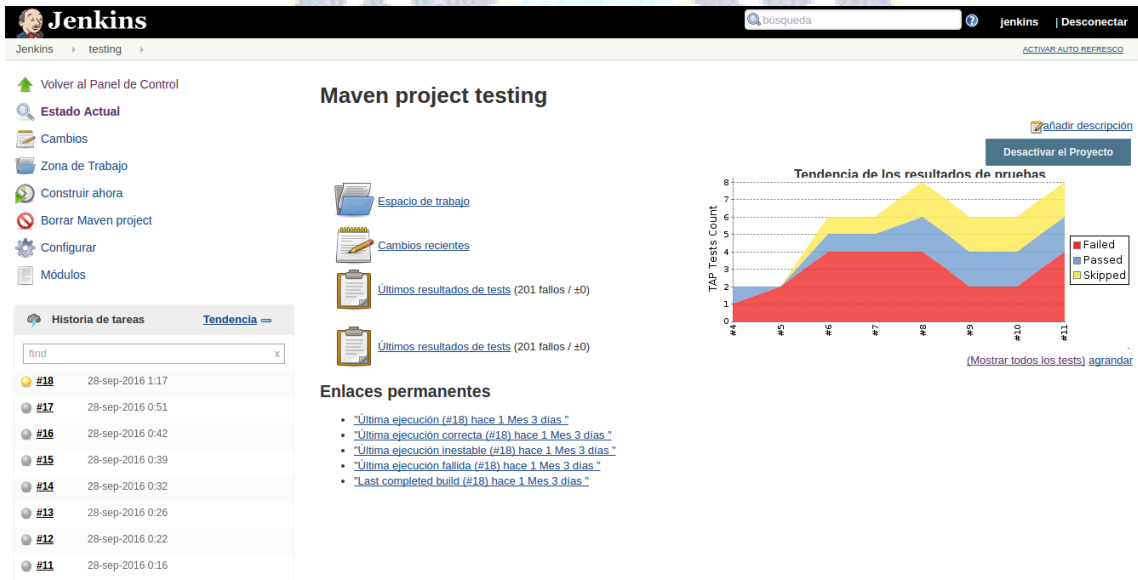


Figura 72.58 Resultados de pruebas  
Fuente: Elaboración Propia

## CAPÍTULO 4 CALIDAD DE SOFTWARE

### 4.1 INTRODUCCIÓN

La medición de la calidad de un determinado software es una de las tareas más difíciles que se presenta en el desarrollo de un sistema. Pero gracias a esta necesidad se fueron creando diferentes formas de medición de las mismas. A éstas las llamamos métricas y entre algunas podemos mencionar las siguientes:

- Modelo de McCall.
- Modelo de Boehm
- Modelo ISO 9126

Para el presente proyecto se usará el modelo de calidad de la ISO 9126.

### 4.2 ISO 9126

La norma ISO 9126 nos ayudará a medir la calidad de nuestro sistema siguiendo los criterios definidos para este modelo.

#### 4.2.1 FUNCIONALIDAD

La funcionalidad del sistema está dada por el dominio de la información al cual esta asociado un valor de complejidad. Los dominios de información son:

- **Número de Entradas de usuario.** Se cuenta cada entrada de usuario que proporciona al software diferentes datos orientados a la aplicación. Las entradas deben ser restringidas de las peticiones que se contabilizan por

separado. En este caso se contabilizará todos los casos de pruebas introducidos por el usuario.

- **Número de Salidas de usuario.** La Salida de Usuario se refiere a informes, pantallas, mensajes de error, etc. En este caso se contabilizará todos los casos de prueba introducidos por el usuario y que espera un resultado del test.
- **Número de peticiones de usuarios.** Una petición está definida como una entrada interactiva que resulta de la generación de algún tipo de respuesta en forma de salida interactiva. De la misma forma al tratarse de un framework sujeto a modificaciones e introducción de nuevos casos de prueba, se contabiliza todos los casos de prueba básicos.
- **Número de archivos.** Se cuenta cada archivo maestro lógico. En otras palabras, las tablas existentes en la base de datos.
- **Numero de interfaces externas.** Se cuenta todas las interfaces legibles por el ordenador que son utilizados para transmitir información a otro sistema. Debemos tomar en cuenta la salida en consola de la ejecución de pruebas, los resultados de los test cases y finalmente la gráfica generada en el sistema de integración.

PARÁMETRO DE MEDICIÓN	Factor de ponderación				
	CUENTA	SIMPLE	MEDIO	COMPLEJO	TOTAL
Nro de entradas de usuario	28	3	4	6	112
Nro. de salidas de usuario	28	4	5	7	140
Nro. de peticiones de usuarios	28	3	4	6	112
Nro. de archivos o tabla	1	7	10	15	10
Nro. de Interfaces	3	5	7	10	21
<b>Cuenta total</b>					<b>395</b>

Tabla 85 Factor de Ponderación para la funcionalidad  
Fuente: Elaboración Propia



Para el presente proyecto se tomará en todos los casos un factor de ponderación medio. Al tratarse de un sistema de complejidad media.

A continuación, para ajustar la complejidad de los puntos de función según el factor de ajuste se asignan pesos de acuerdo a los siguientes casos como se ve en la siguiente tabla:

FACTOR	VALOR
Sin importancia	0
Incidental	1
Moderado	2
Medio	3
Significativo	4
Esencial	5

Tabla 86 Pesos de los puntos función  
Fuente: Pressman, 2002

Los ajustes de complejidad con sus respectivos pesos se ven a continuación:

FACTOR	PESO
¿Requiere el sistema copias de seguridad?	5
¿Se requiere comunicación de datos?	4
¿Existe funciones de procesamiento distribuido?	4
¿Es crítico el rendimiento?	4
¿Se ejecutara el sistema en entorno operativo existente y utilizado?	3
¿Se requiere entrada de datos?	5

¿Requiere la entrada de datos que las transiciones de entrada hagan sobre múltiples pantallas u operaciones?	5
¿Se utilizan los archivos maestros de forma interactiva?	3
¿Son complejos las entradas, las salidas, los archivos o las peticiones?	5
¿Es complejo el procesamiento interno?	5
¿Se ha diseñado el código para ser reutilizable?	5
¿Están incluidas en el diseño la conversión y la instalación?	4
¿Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones?	4
¿Se ha diseñado la aplicación para facilitar los cambios y ser fácilmente utilizada por el usuario?	4
<b>TOTAL</b>	<b>60</b>

Tabla 87 Factores de evaluación  
Fuente: Elaboración propia

Utilizando los resultados obtenidos hasta el momento y reemplazándolos en la ecuación tenemos:

$$PF = 395 * 0.65 + 0.01 * 60 = 493,75$$

Veamos la siguiente tabla:

ESCALA	OBSERVACIÓN
PF > 300 🏆	Óptimo
200 < PF < 300	Bueno
100 < PF < 200	Suficiente
PF < 100	Deficiente

Tabla 88 Escala de Punto Función  
Fuente: Pressman, 2002

Se puede observar que el presente sistema tiene una funcionalidad óptima, ya que los puntos de función encontrados tienen el valor de 493.75.

A continuación, calculamos el ajuste, éste, se obtiene del uso de la ecuación anterior, pero aplicando los factores de evaluación con sus pesos máximos. De ahí obtenemos:

$$PF_{ajuste} = \text{Cuenta total} * (\text{Valor máximo de complejidad})$$

$$PF_{ajuste} = 395 * 1,35 = 533,25$$

Con estos resultados se puede calcular la funcionalidad del sistema, esto los vemos a continuación.

$$F = PF / PF_{ajuste}$$

$$F = 493,75 / 533,25 = 0,9259$$

Donde:

$$F = \text{Funcionalidad del sistema}$$

Si expresamos  $F$  en porcentajes obtendremos una funcionalidad del 93%, lo que quiere decir que tomando en cuenta todos los parámetros de medición expresados en la tabla 4.1 nuestro sistema tiene una funcionalidad aceptable.

#### 4.2.2 FIABILIDAD

La fiabilidad es el tiempo medio entre fallos(TMEF) y la fórmula es:

$$TMEF = TMDF + TMDR$$

Donde:

$$TMDF = \text{Tiempo Medio de fallo}$$

$$TMDR = \text{Tiempo Medio de reparación}$$

Para calcular TMDR hacemos uso de la siguiente fórmula:

$$TMDR = TMAC + TMIC + TMPC + TMDC$$

Donde:

$$TMAC = \text{Tiempo medio de analizar los cambios}$$

$$TMIC = \text{Tiempo medio de implementar los cambios}$$

$$TMPC = \text{Tiempo medio de probar los cambios}$$

$$TMDC = \text{Tiempo medio de distribuir los cambios}$$

Tomando una muestra de 10 mediciones por día y en base a la media aritmética se obtuvo la siguiente tabla:

TMAC	TMIC	TMPC	TMDC
<i>ti n</i>	<i>ti n</i>	<i>ti n</i>	<i>ti n</i>
15 10	10 10	20 10	10 10
1.5	1.5	2	1

Tabla 89 Mediciones de fiabilidad  
Fuente: Elaboración propia

De los datos obtenidos en la tabla 4.5 obtenemos el TMDR:

$$TMDR = TMAC + TMIC + TMPC + TMDC$$

$$TMDR = 6$$

De las 10 muestras que se hicieron se estima una falla aproximadamente cada 30 horas de uso continuas. Esto lo expresamos como sigue:

$$TMDF = 30 / 10 = 3$$

Ya contando con TMDF y TMDR procedemos a calcular TMEF:

$$TMEF = 3 + 6 = 9$$

Si se multiplica el resultado TMEF por 10 obtendremos 90. Este resultado se lo puede expresar de la siguiente manera: De cada 100 veces que se realice una ejecución, 90 de las veces éste correrá de forma correcta. Esto representa un 90% de Fiabilidad, lo que quiere decir que nuestro sistema está dentro del rango de aceptabilidad.

Además de medir la fiabilidad también tenemos que medir la Disponibilidad, que es la probabilidad de que un sistema funcione de acuerdo con los requisitos exigidos en un momento dado. La Disponibilidad se la puede obtener a través de la siguiente fórmula:

$$D = \frac{TMDR}{TMDF + TMDR} * 100\%$$

Donde:

$D =$  disponibilidad del sistema

Reemplazando los datos obtenidos anteriormente tenemos:

$$D = \frac{6}{3 + 6} * 100$$

$$D = 66,66 = 67\%$$

Por lo tanto, la probabilidad de que el sistema se encuentre disponible para su uso en un momento dado es del 67%. Esta medida está dentro de una calificación de “satisfactoria”, pero se debe procurar mejorarla, aunque este no sea un sistema de tipo militar o algo parecido.

La medida de disponibilidad es algo más sensible al tiempo promedio de respuesta, por lo tanto, es una medida indirecta de la facilidad de mantenimiento del software.

### 4.2.3 USABILIDAD

La Usabilidad representa facilidad de uso que el usuario final percibirá del sistema. Esta métrica nos muestra el esfuerzo necesario para aprender a manipular el sistema. Se lo obtiene a través de los datos obtenidos de una pequeña encuesta que se la realiza directamente a un grupo determinado de usuarios finales.

La tabla de preguntas para la encuesta será la siguiente:

N°	PREGUNTA	VALOR
1	¿El sistema satisface los requerimientos de manejo de información?	
2	¿Las salidas del sistema están de acuerdo a sus requerimientos?	
3	¿Cómo considera el ingreso de datos al sistema?	
4	¿Cómo considera los reportes que elabora el sistema?	
5	¿El sistema facilita el trabajo que realiza?	
<b>TOTAL</b>		

Tabla 90 Preguntas de facilidad de uso  
Fuente: Elaboración propia

Y la escala de ponderación es la siguiente:

DESCRIPCIÓN	VALOR
Pésimo	1
Malo	2
Regular	3
Bueno	4
Muy bueno	5

Tabla 91 Escala de Evaluación para la usabilidad  
Fuente: Elaboración propia

La encuesta se la realizó a 5 usuarios finales y los datos obtenidos los expresamos en la siguiente tabla:

N°	Usuario1	Usuario2	Usuario3	Usuario4	Usuario5	Promedio
1	4	3	5	5	4	4,2
2	4	4	4	5	4	4,2
3	4	4	3	4	5	4
4	5	4	3	4	3	3,8
5	4	3	4	4	4	3,8
TOTAL						20

Tabla 92 Resultados de la encuesta de usabilidad  
Fuente: Elaboración propia

Con el resultado obtenido y utilizando la siguiente ecuación tendremos:

$$U = \frac{\sum x_i}{n} * 100$$

Donde:

$$U = \text{usabilidad}$$

Entonces, tenemos:

$$U = \frac{\frac{20}{5} * 100}{5}$$

$$U = 80\%$$

Por lo tanto, concluimos que la facilidad de uso del sistema es del 80% y que está dentro del rango de aceptabilidad de la norma ISO 9126. También podemos interpretar este resultado indicando que de cada 10 personas que utilicen el sistema 8 encontrarán al mismo fácil de usar. Cabe mencionar que todos los usuarios del sistema deben tener habilidades de desarrollo para introducir los casos de prueba.

#### 4.2.4 MANTENIBILIDAD

El estándar [IEE94] sugiere un índice de Madurez del Software (IMS) que proporciona un indicador de la estabilidad de un producto, basándose en los cambios que ocurren en cada versión del producto. Cuando el IMS se aproxima a 1 se dice que el sistema es más estable.

Se determina con la siguiente fórmula:

$$IMS = \frac{Mt - Fa + Fc + Fd}{Mt}$$

Donde:

*Mt = número de módulos en la versión actual*

*Fc = Número de módulos en la versión actual que han cambiado*

*Fa = Número de módulos en la versión actual que se han añadido*

*Fd = Número de módulos en la versión anterior que se han borrado*



A partir del sistema se obtuvieron los siguientes valores necesarios para el cálculo del IMS. Tomamos en cuenta los cambios que se realizarón en la fase de mantenimiento del sistema, después de la puesta en prueba del mismo. Se los expresa en la siguiente tabla:

VARIABLE	VALOR
Mt	28
Fc	4
Fa	0
Fd	0

Tabla 93 Valores IMS  
Fuente: Elaboración propia

Reemplazando los valores obtenidos en la tabla anterior, tenemos lo siguiente:

$$IMS = \frac{28 - 4 + 0 + 0}{28} = 0,8571$$

A medida que el IMS se aproxima a 1 el producto empezará a estabilizarse.

Además, podemos expresar la Mantenibilidad con la siguiente ecuación:

$$M = IMS * 100$$

$$M = 85\%$$

Donde:

$$M = \text{Mantenibilidad del sistema}$$

Por lo tanto, podemos concluir que nuestro sistema cuenta con una Mantenibilidad del 85%.

También podemos interpretar este resultado indicando que de cada 10 casos en los que sean necesarios un mantenimiento, en 8.5 de las veces se podrá realizar el mismo de forma sencilla y práctica, para el resto de las veces se necesitará un análisis más complejo.

#### 4.2.5 PORTABILIDAD

La portabilidad mide la facilidad con la que un sistema puede moverse de un entorno a otro. El framework en su totalidad puede instalarse en todas las plataformas y de la misma forma cada uno de sus colaboradores pueden ejecutar pruebas e instalar los WebDrivers en cada uno de los sistemas operativos más comunes.

La ecuación para medir la portabilidad se detalla a continuación:

$$P = \frac{IE}{n}$$

Dónde:

*P = Portabilidad del sistema*

*IE = Instalaciones Exitosas*

*n = Número de instalaciones totales*

El resultado obtenido es el siguiente:

$$P = 1$$

Esto nos indica que la portabilidad del sistema se dá al 100% por ser multiplataforma y ejecutable desde cualquier terminal.

### 4.3 CALIDAD GLOBAL

Para poder obtener la calidad global de nuestro sistema, sacaremos la media de todas medidas expresadas en porcentajes hasta el momento. Esto lo expresaremos en la siguiente tabla:

CRITERIOS	RESULTADO
Funcionalidad	93
Fiabilidad	90
Usabilidad	80
Mantenibilidad	85
Portabilidad	100
<b>Calidad global</b>	<b>89,6</b>

Tabla 94 Calidad global  
Fuente: Elaboración propia

Con el resultado obtenido en la tabla 4.10 se puede afirmar que el framework tiene una calidad del 89.6%, en otras palabras, tiene una calidad más que aceptable.

## CAPÍTULO 5 ANÁLISIS DE COSTO BENEFICIO

### 5.1 INTRODUCCIÓN

El análisis de costo beneficio apunta a demostrar que la implementación del proyecto en cuestión dara más beneficios evaluando su rentabilidad obteniendo mayores y mejores resultados al menor esfuerzo invertido, tanto por eficiencia técnica como por motivación humana.

### 5.1 COCOMO II

En todo proyecto es importe tener la planificación o estimación de costo del proyecto, tanto de los costos en tiempo, y esfuerzo. Uno de los métodos para realizar el COCOMO II (COConstructive COSt MOdel) orientado a los puntos función. Para estimar el costo total del sistema se tomarán en cuenta los siguientes costos: Costo de la elaboración del proyecto, costos del software desarrollado, costos de implementación del sistema.

#### 5.1.1 COSTOS DEL SOFTWARE DESARROLLADO

Se utilizara el punto función encontrado en el acápite 4.2 donde el punto  $PF = 1225,08$  ,a continuación se realizara la conversión de un punto función a miles de líneas de código mediante la siguiente tabla.

Lenguaje	Factor LDC/PF
<b>JavaScript</b>	54
<b>Assembler</b>	320
<b>C</b>	150
<b>Pascal</b>	91
<b>C++</b>	64
<b>Visual Basic</b>	32
<b>Java</b>	55

Tabla 95 Factor LCD/PF del lenguaje de programación  
Fuente: COCOMO, Marvin Romero, 2013

$$LDC = PF * Factor \quad LDC/PF$$

$$LDC = 454,25 * 55 = 24983,75$$

$$KLDC = 24,98$$

Se aplica las formulas básicas de esfuerzo, tiempo calendario y personal requerido.

Las fórmulas de COCOMO son las siguientes:

$$E = a_b KLDC^{b_b} * FAE$$

$$D = c_b E^{d_b}$$

Dónde:

E: Esfuerzo aplicado en personas por mes.

D: Tiempo de desarrollo en mes.

KLDC: Número estimado de líneas de código distribuidas (en miles).

FAE: Factor de ajuste del esfuerzo.

En la siguiente tabla se muestran los tipos de proyectos de software.

Proyecto de software	$a_b$	$b_b$	$c_b$	$d_b$
<b>Orgánico</b>	2.4	1.05	2.5	0.38
<b>Semiacoplado</b>	3.0	1.12	2.5	0.35

<b>Empotrado</b>	3.6	1.2	2.5	0.32
------------------	-----	-----	-----	------

Tabla 96 Tipos de proyectos de Software  
Fuente: COCOMO, Marvin Romero, 2013

- Orgánico: relativamente sencillo y pequeños, en los que trabajan equipos pequeños con experiencia, sobre conjunto de datos poco dirigidos.
- Semiacoplado: proyectos intermedios (en tamaño y complejidad) en los que participan equipos con varios niveles de experiencia, deben satisfacer requisitos medio rígidos.
- Empotrados: proyectos que deben ser desarrollados en un conjunto de hardware, software y restricciones operativas muy restringido.

El proyecto se adecua a un proyecto semi-acoplado y el cálculo realizado para el factor de ajuste del esfuerzo  $FAE = 0,82$

Por lo tanto, se reemplaza en las formulas mencionadas:

$$E = a_b KLDC^{b_b} * FAE$$

$$E = 3 \cdot 24,98^{1,12} * 0,82 = 90,41[\text{persona/mes}]$$

$$D = 2,5 \cdot 90,41^{0,35} = 12,09[\text{meses}]$$

El personal requerido para el desarrollo del proyecto se obtiene de la siguiente formula:

$$\text{Numero de programadores} = \frac{E}{D} = \frac{90,41}{12,09} = 7$$

Por lo tanto, se necesita 7 programadores para el desarrollo del proyecto. El costo de salario por programados es de 300\$us mensual. Entonces con este dato se calculará la estimación del costo del software:

*Costo de software = # de programadores \* salario del programador \* meses de trabajo*

$$\text{costo de software} = 7 * 300 * 12 = 25200\$us$$

### 5.1.2 COSTOS DE LA IMPLEMENTACIÓN DEL SISTEMA

La siguiente tabla muestra los años de mantenimiento con respecto al costo de egresos e ingresos; el costo del año uno corresponde al costo obtenido con COCOMO II.

	Año 0	Año 1	Año 2	Año 3	Año 4	Año 5
<b>Egresos</b>	25200	12600	10000	6000	4000	2000
<b>Ingresos</b>	-25200	20000	40000	70000	100000	150000

Tabla 97 Estimación de rendimiento  
Fuente: Elaboración propia

El valor actual neto (VAN), se calcula por medio de los flujos de inversión, cuyo resultado refleja si la inversión en el proyecto genera beneficios si el resultado que se obtiene es favorable, lo anterior se calcula con la siguiente formula:

$$VAN = \sum_{t=1}^n \frac{V_t}{1 + k^t} - I_0$$

Dónde:

$V_t$  =Flujos de caja en cada periodo t.

$I_0$  = Valor del desembolso inicial de la inversión.

N = Numero de periodos considerados.

$k$  =Interés.

Teniendo en cuenta el valor de interés del 10% y reemplazando datos tenemos:

$$VAN = 71868$$

Como el valor del VAN > 0, indica que la inversión genera ganancias por encima de la rentabilidad exigida.

La fórmula para el cálculo del TIR es:

$$VAN = \sum_{t=1}^n \frac{V_{ft}}{1 + TIR^t} - I_0 = 0$$

Donde:

$V_{ft}$  = Flujo de caja en el periodo t.

Y se obtiene  $TIR = 23,5\%$ , y como es un valor positivo lo que nos indica que se acepta el desarrollo del proyecto.

A continuación se muestra el análisis costo beneficio donde la tasa de descuentos se calcula de la siguiente manera:  $1 / (1 + \text{tasa de descuento}^{\text{indice\_año}})$ . En este caso la tasa de descuento tomada es de 10%.

Año de operación	Costos totales(\$us)	Beneficios totales(\$us)	Factor de actualización (10%)	Costos actualizados(\$us)	Beneficios actualizados(\$us)
1	12600	12000	0,91	13104	12740
2	10000	18000	0,83	8300	14940
3	6000	20000	0,75	4500	15000
4	4000	24000	0,68	2720	16320
5	2000	28000	0,62	1240	17360
<b>Total</b>				29864	76360

Tabla 98 Análisis de costo  
Fuente: Elaboración propia



La relación de beneficio/costo es como sigue:

$$\frac{B}{C} = \frac{\text{Beneficios actualizado}}{\text{Costos actualizados}} = \frac{76360}{29864} = 2.5$$

Lo que indica que por cada 2 dólares invertidos se obtiene un rendimiento de 0.5 \$us, por ello el proyecto es rentable.



## CAPÍTULO 6 CONCLUSIONES Y RECOMENDACIONES

### 6.1 CONCLUSIONES

El proyecto de grado “Framework de Automatización de Pruebas Funcionales para Diseñador de Procesos de Processmaker v3” desarrollado e implementado en la empresa COLSER SRL. demostró dar solución a los problemas planteados en el capítulo 1 aplicando las metodologías especificadas en el presente texto además de demostrar el beneficio que trae su implementación. Para una mejor comprensión se detalla lo siguiente:

- Se pudo desarrollar una solución efectiva y funcional al problema del equipo de Quality Assurance (QA) de COLSER SRL.
- Se pudo implementar un sistema de testeo crossbrowsing el cual garantiza una correcta ejecución de la herramienta Processmaker bajo distintos ambientes y plataformas de desarrollo y producción.
- Se elaboró un sistema de testeo organizado, esto con la intención de valorar cada una de las funcionalidades de la herramienta Processmaker, específicamente del diseñador de procesos.
- Se aplicó la integración continua la cual no es solamente una tarea del equipo de Quality Assurance, sino también de todo el departamento de Ingeniería, demostrando que la integración de tareas mejora el proceso de desarrollo.
- Se instaló un Framework completo con miras a un futuro crecimiento y adaptable a los cambios que la herramienta de diseño de procesos Processmaker realizará en su constante mejora y desarrollo.

## 6.2 RECOMENDACIONES

Según la experiencia que se adquirió desarrollando el presente proyecto se recomienda lo siguiente:

- Las pruebas automáticas son de gran ayuda, pero nunca pueden compararse a las pruebas realizadas de forma manual, ya que el razonamiento humano permite aplicar casos de prueba de forma mucho mas completa y enfocando los problemas de una manera mas precisa. Es por eso que se recomienda automatizar pruebas que sean completamente repetitivas.
- Usar el mismo patrón de diseño con el que se inició el proyecto, de esta manera se evita la duplicidad de código y éste podrá ser entendible por terceras personas.
- Aplicar la infraestructura usando máquinas físicas y no así virtuales, de esta manera se garantiza una correcta ejecución de pruebas, al orientar todos los recursos a una sola actividad, en este caso, la automatización de pruebas.
- Ampliar el alcance del framework introduciendo más combinaciones de plataformas, sistemas operativos y navegadores, de esta forma se podrá garantizar el correcto funcionamiento de la herramienta Processmaker.

## BIBLIOGRAFIA

- [1] Freund-Ruecker-Hitpass, Jakob Freund, Bernd Rucker, Bernhard Hitpass. (2013), *BPMN 2.0 Manual de Referencia y Guía Práctica* (4<sup>ta</sup> eds) Dimacofi - Santiago de Chile.
- [2] Brian Underdahl. (2011). *Business Process Management for Dummies®*, IBM Limited Edition. Indianapolis, Indiana, Wiley Publishing
- [3] Aron Gustafson, Jeffrey Zeldaman. (2011). *Adaptative Web Design, Crafting Rich Experiences with Progressive Enhancement*. Chattanooga, Tennessee USA
- [4] Douglas Crockford. (2014). *JavaScript: The Good Parts*. United States of America. O'Reilly Media.
- [5] Métricas de itreacion (2009), *scrum sprint metrics*, recuperado de <http://proyectosagiles.org/2009/11/08/metricas-iteracion-scrum-sprint-metrics/>
- [6] Marcelo MEJIA. 3<sup>ra</sup> edición (2006). *Sistemas, Cibernética E Informática, Automatización de Procesos de Negocio utilizando un BPMS*. 1,2 México, D.F.
- [7] Roger PRESSMAN. 5<sup>ra</sup> edición (2002). *Ingeniería de Software, Un enfoque práctico*. Mc. Graw Hill, Madrid
- [8] Ian SOMMERVILLE. 9<sup>ma</sup> edición (2006). *Ingenieria de software*. Prentice Hall. Madrid.
- [8] G. LARMAN. 2<sup>da</sup> edición (2002). *UML y patrones, Introducción al Análisis y Diseño Orientado a Objetos*. Prentice Hall. Madrid.

**ANEXOS**





## Diagrama de Gantt

Completado Pendiente

Proyecto	Fecha inicio prevista	Días trabajado	Fecha final prevista	Situación	Días para el final
Sprint 1	1-feb-16	11	12-feb-16	Terminado	0
Sprint 2	15-feb-16	11	26-feb-16	Terminado	0
Sprint 3	29-feb-16	18	18-mar-16	Terminado	0
Sprint 4	21-mar-16	18	8-abr-16	Terminado	0
Sprint 5	11-abr-16	18	29-abr-16	Terminado	0
Sprint 6	2-may-16	11	13-may-16	Terminado	0
Sprint 7	16-may-16	11	27-may-16	Terminado	0

