SEDE CUENCA

CARRERA DE INGENIERÍA DE SISTEMAS

IMPLEMENTACIÓN DEL SISTEMA DE MOVILIZACIÓN EN LA UNIVERSIDAD POLITÉCNICA SALESIANA DE CUENCA UTILIZANDO EL BUSINESS PROCESS MANAGEMENT (BPM) "BONITASOFT OPEN SOLUTIONS"

Tesis previa a la obtención del Título de

Ingeniero de Sistemas.

AUTOR:

Pablo Jairo Angamarca Briceño

DIRECTOR:

Ing. Diego Marcelo Quinde Falconí, MsC.

Cuenca - Ecuador

2015

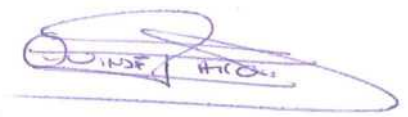
DECLARACIÓN

Yo, Pablo Jairo Angamarca Briceño declaro bajo juramento que el trabajo aquí descrito es de mi autoría, que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través dela presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Politécnica Salesiana – Sede Cuenca, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la Normativa Institucional Vigente.

Pablo Jairo Angamarca Briceño

Yo, Diego Marcelo Quinde Falconí, Secretario Técnico de Tecnologías de la Información y Docente de Carrera de Ingeniería de Sistemas de la Universidad Politécnica Salesiana, CERTIFICO que he leído íntegramente el presente trabajo de grado, comprobando que se cumplen los objetivos planteados en el proyecto. Así mismo declaro que el autor Pablo Jairo Angamarca Briceño es el único responsable de la originalidad del trabajo desarrollado.



Ing. Diego Marcelo Quinde Falconí, MsC.

DEDICATORIA

A mis padres, Sergio G. Angamarca C. y D.Thalía Briceño L., a mis abuelitos Ramón Briceño (+) y Dolores Julieta Ludeña, quienes son fuente de inspiración de amor, lucha y superación en mi vida.

Pablo A.

AGRADECIMIENTOS

A mi DIOS por regalarme la vida y por cada experiencia a su lado.

A mi familia que ha sido, es y será mi principal apoyo y mi mas preciado tesoro en la tierra.

A mis grandes amigos Martín y Sebastián, por su contínuo apoyo y motivación, son una bendición.

A mi bonita inspiración Glorita.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Diagnóstico de la realidad actual en la UPS

En el presente capítulo se busca brindar una idea general sobre la forma en la que se manejan los procesos en la actualidad en la UPS y de las herramientas y recursos empleados en cada actividad.

1.1. Manejo de procesos en la actualidad

El tratamiento de cada uno de los más de 600 procesos, hoy en día en la UPS se viene realizando prácticamente en forma "manual", ya que las herramientas que se emplean en la actualidad para esta actividad, no proporcionan los beneficios requeridos en cuanto al manejo y seguimiento de los procesos de la universidad. En la actualidad los directivos y autoridades de la UPS necesitan invertir una gran cantidad de tiempo y recursos al momento de requerir un reporte ya sea con respecto a los datos estadísticos provenientes de los historiales de los procesos, verificar la eficiencia de los mismos o para identificar los cuellos de botella que existan dentro de un proceso determinado, etc.

1.2. Recursos y esfuerzos empleados en la ejecución de procesos de negocios

Muchos de los trámites (procesos) que se realizan en la UPS, tanto por estudiantes como por el personal docente y administrativo, se los realiza en base de papeles y herramientas no adecuadas para esta actividad. Como un esfuerzo para alcanzar un porcentaje de automatización de los trámites en la UPS se ha implementado el sistema de gestión documental denominado "QUIPUX" una herramienta que si bien es cierto ha aportado mucho en la gestión de documentos mas no así en la gestión propia de los procesos de negocio los cuales necesitan un especial tratamiento y automatización. Otra de las herramientas que se está utilizando en la UPS es el correo electrónico como un medio de comunicación y así evitar el exceso de papeles y agilitar un poco las actividades, sin embargo son herramientas muy básicas que fácilmente se pueden llegar a convertir en "agujeros negros sin retorno" además no brindan los reportes ni el servicio adecuado para una gestión de procesos eficiente. Esta forma en la que se vienen desarrollando los procesos de negocio en la UPS provocan entre otros, los siguientes inconvenientes y desventajas:

Pérdida de tiempo: Al realizar los trámites utilizando papeles y herramientas no adecuadas, la universidad no llega a tener la capacidad de respuesta adecuada a los trámites y solicitudes que a diario realizan los usuarios, causando muchas de las veces retrasos en la respuesta a ciertas solicitudes.

Esfuerzos duplicados: Siempre se verán duplicados los esfuerzos al momento de realizar cada etapa de los procesos de negocio manualmente y aún más cuando se requiera algún reporte, ya que se ocupara más personal, más tiempo y recursos de varios tipos, etc.

Consumo innecesario de recursos monetarios: Uno de los inconvenientes al realizar procesos prácticamente manuales es el gasto innecesario de recursos monetarios, ya que siempre serán necesarios: tiempo, papel, impresoras y el recurso humano, lo cual siempre representará un gasto.

Departamentos "aislados": Algunas de estas complejidades en el desarrollo de los procesos provocan que dentro de la institución existan unidades administrativas y de negocio "aisladas" con flujos de información poco eficientes y muy limitados entre cada departamento y sus empleados o estudiantes que están directamente relacionados con las actividades de determinado proceso.

Falta de conectividad: Las herramientas utilizadas en la actualidad no facilitan una adecuada conexión con otros sistemas que se manejan en la UPS, cuya fusión resultaría de mucha utilidad.

Poca visibilidad: Los procesos no se pueden ver de manera global para identificar su verdadero flujo, eficiencia o deficiencia, cuellos de botella o aplicación excesiva de recursos.

Para dar una eficiente y rápida solución a esta importante tarea la UPS emprende a través de este trabajo de titulación, el inicio de lo que sería la automatización de sus procesos implementando una herramienta que dará una solución flexible, propia de la gestión de procesos de negocio (BPM, por sus siglas en inglés), robusta, con una interfaz gráfica intuitiva, amigable y de fácil uso para cualquier tipo de usuarios finales. Dicha herramienta proporcionará un control completo y sencillo de los procesos de negocio de la UPS.

Capítulo 2

Estudio de la Herramienta

2.1. Conociendo Bonita BPM Open Solution

En este capítulo ilustraremos una idea clara de lo que es Bonita BPM, conoceremos, entre otras cosas acerca de sus características, componentes, funcionalidades, ventajas y bondades que ofrece Bonita BPM.

2.1.1. ¿Qué es Bonita BPM open Solution?

Una de las prioridades presentes en toda organización es el ahorro de recursos y la eficiencia al momento de ejecutar sus procesos los mismos que se encuentran en cada departamento, hoy en día muchas de las organizaciones no disponen de una herramienta que facilite y automatice el manejo de los procesos para lograr ahorrar recursos y hacer más eficientes a cada departamento, en la ejecución "manual" de los procesos las organizaciones invierten muchos recursos y esfuerzos que fácilmente se verían reducidos con la buena gestión y automatización de estos.

Como una muy buena alternativa para dar solución a en este necesidad actual tenemos el concepto de la gestión de procesos de negocio o BPM (Business Process Management), a continuación citaremos algunas definiciones de lo que es

un BPM:

"Una nueva categoría de software empresarial que permite a las empresas modelizar, implementar y ejecutar conjuntos de actividades interrelacionadas es decir, Procesos de cualquier naturaleza, sea dentro de un departamento o permeando la entidad en su conjunto, con extensiones para incluir los clientes, proveedores y otros agentes como participantes en las tareas de los procesos". AuraPortal (2014).

"Es un conjunto de métodos, herramientas y tecnologías utilizados para diseñar, representar, analizar y controlar procesos de negocio operacionales; un enfoque centrado en los procesos para mejorar el rendimiento que combina las tecnologías de información con metodologías de procesos y gestión". (Colabra Procesos, 2012).

"Se puede definir a BPM como una disciplina o enfoque disciplinado orientado a los procesos de negocio, pero realizando un enfoque integral entre procesos, personas y tecnologías de la información. BPM busca identificar, diseñar, ejecutar, documentar, monitorear, controlar y medir los procesos de negocios que una organización implementa. El enfoque contempla tanto procesos manuales como automatizados y no se orienta a una implementación de software. Algo importante a tener presente es que BPM no es una tecnología de software, pero se apoya y hace uso de las mismas para su implementación efectiva." (Sánchez, 2011).

Ahora diremos que Bonita es una potencial suite de BPM de código abierto con licencia GPL (General Public Lincense) que aporta en la búsqueda de integrar eficientemente el modelado de procesos, las tecnologías de la información y el recurso humano.

2.1.2. Orígenes y Versiones

La compañía BonitaSoft ha venido democratizando los conceptos de BPM desde sus principios los cuales tienen origen en el año 2001, cuando es iniciada en los laboratorios del National Institute for Research in Computer Science and Control (Instituto Nacional de Investigación en ciencias de la computación y control).

Ya para el año 2003 el proyecto es acogido y apoyado por la empresa científica francesa Bull. En el año 2008 se lanza al mercado Bonita BPM v4, alcanzando niveles de descargas muy prometedoras (más de 100.000 descargas).

"En junio del 2009 con la ayuda de Miguel Valdés Faura, Charles Souillard y Rodrigue Le Gall , la compañía BonitaSoft es oficialmente fundada en Francia" (BonitaSoft, 2014b, a), la misma que se crea para dar soporte directo a Bonita BPM Open Solution, convirtiéndose en el primer sistema editor y gestor de soluciones BPM bajo la licencia GPL v2 (Licencia Pública General).

En el 2010 el mercado de Bonita se extiende, y se lanza Bonita v5. Durante el año 2010 hasta el 2013 se han realizado varias mejoras para la quinta versión de Bonita, y en este último año se realiza el lanzamiento de Bonita BPM v6, y para mayo del año 2014 se lanza la versión 6.3, y ya para diciembre de 2014 se tiene la versión 6.4, que es la que uzaremos en este proyecto.

Bonita también incursiona en las llamadas soluciones en la nube, y en agosto del 2014, lanza su versión de demostración de la plataforma BonitaCloud, la cual es una versión que en su totalidad está alojada en la nube de Bonita BPM Community.

2.1.3. BonitaSoft en la actualidad

Bonita BPM es considerada la solución BPM de código abierto número uno en el mundo, alcanzando hasta la actualidad las siguientes cifras: alrededor de 2.5 millones de descargas, más de 120.000 miembros activos en la comunidad de bonita, cuentan con más de 1000 clientes que están utilizando las soluciones de BonitaSoft en más de 60 países. Los trabajadores de BonitaSoft ascendieron a un número de 150 empleados distribuidos en sus tres oficinas principales en París, Grenoble y San Francisco.

En el 2013 Bonita ratifica su posición en el mercado al recibir varios premios importantes como por ejemplo: el premio SIAA CODiE que se otorga a la mejor Innovación Open Source, el premio EclipseCon 2011, el premio Bossie de InfoWorld entregada a la mejor aplicación de código abierto, el premio Stevie International Business y el reconocimiento Bossie de InfoWorld que se entrega al mejor software de código abierto. Bonita además tiene un reconocimiento de Gartner la consultora estadounidense quien afirmo, en su informe de BPM de código abierto, que "Bonita BPM es el único producto de código abierto que cumple con la definición de solución BPM de Gartner". (Farrance, 2014).

2.1.4. Componentes y funcionalidades de Bonita BPM open Solution

Bonita BPM en una herramienta BPM muy amigable e intuitiva, esta característica se verá en el desarrollo de los procesos a través de sus componentes: Bonita Estudio, Bonita Execution Engine y Bonita User Experience. Además su implementación es de bajo costo ya que lo único que se requiere es invertir tiempo en conocer la herramienta.

Para comenzar veremos la pantalla de inicio que presenta Bonita Open

Solution, donde se encontraran las secciones de recursos, aprender, y diseño en donde encontraremos nuestro punto de partida para nuestro nuevo proyecto:

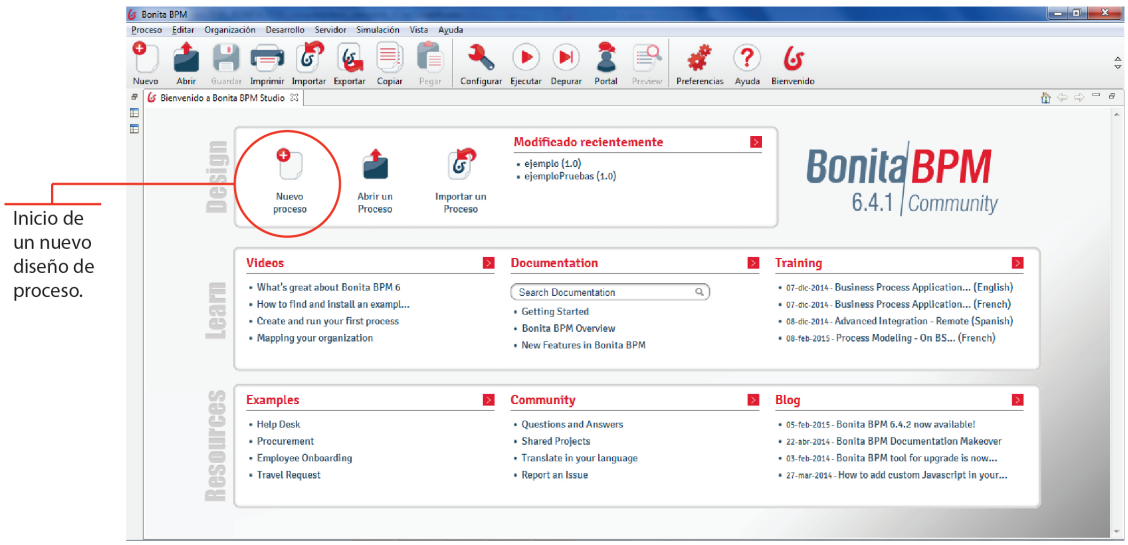


Figura 2.1: Pantalla de inicio de Bonita Open Solution 6.4.1

A continuación revisaremos estos principales componentes que conforman Bonita BMP.

2.1.4.1. Diseño con Bonita Studio

Bonita BMP cuenta con una poderosa herramienta para el diseño de los procesos que se desea automatizar, dentro de un entorno gráfico encontramos elementos de fácil manipulación al ser drag-n-drop. Bonita diseña sus procesos bajo la notación BPMN2 la cual analizaremos más adelante. Bonita Studio consta con dos principales elementos para el diseño de los procesos:

Whiteboard: Es aquí donde se diseñan los procesos y el flujo que estos seguirán, se puede definir las características de cada proceso, el inicio y fin de un proceso, se puede colocar puntos de decisión, y más elementos que veremos más adelante, los mismos que aportan al diseño del proceso. Los procesos son dibujados utilizando la notación BPMN 2.0, y brinda las facilidades de drag-and-drop.

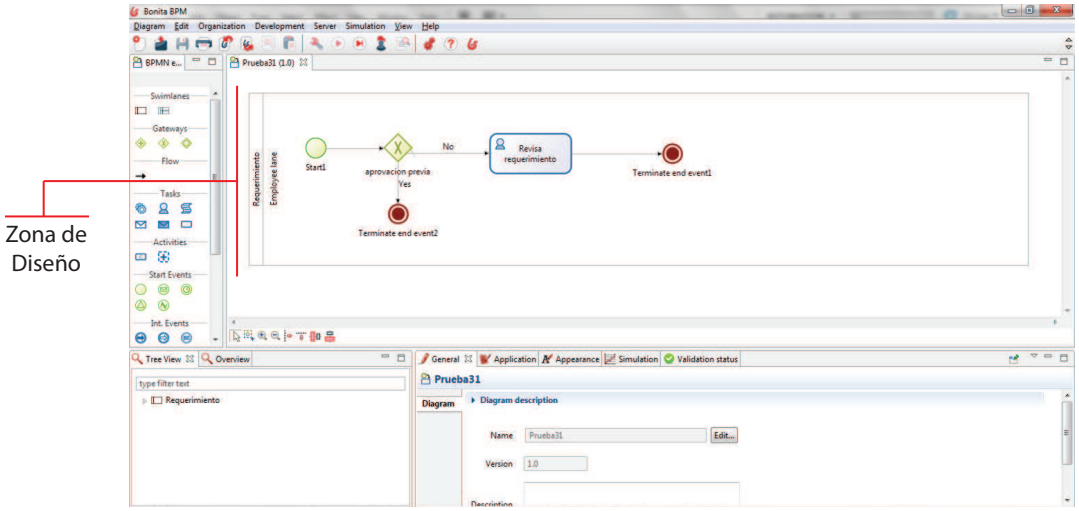


Figura 2.2: Zona para dibujar y editar nuestros diagramas.

Form Builder: Con esta herramienta se pueden crear los diferentes formularios que se requieran para cada usuario final que estará involucrado con el proceso, estas son realmente aplicaciones web, pues en ciertos procesos el usuario necesitara ingresar información a través de ellas.

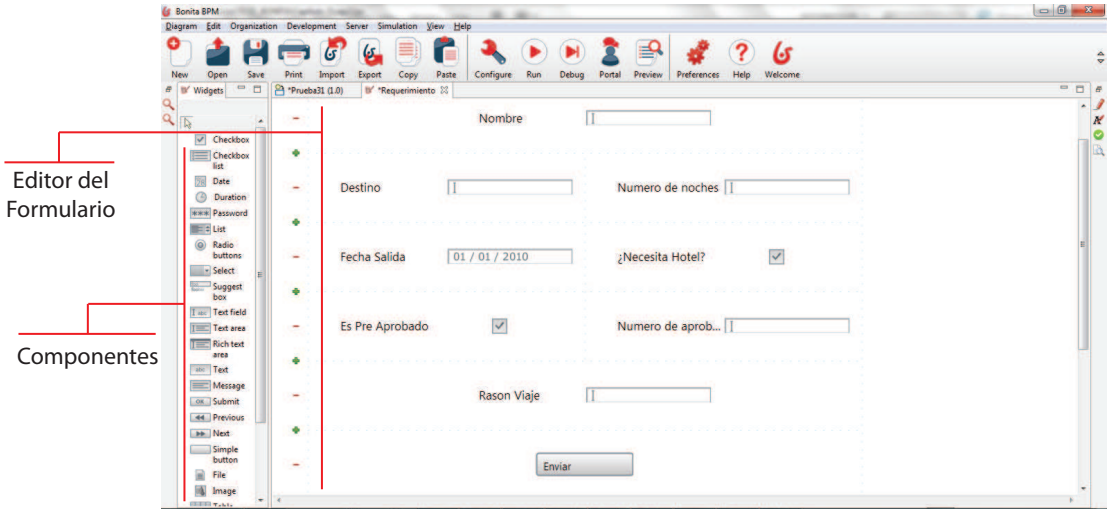


Figura 2.3: Zona donde se edita los formularios para interacción con el usuario final

2.1.4.2. Bonita Execution Engine

Conocido también como BEE por sus siglas en inglés (Bonita Execution Engine) es el poderoso motor de ejecución de Bonita BPM, el cual está diseñado de manera genérica y extensible, debido a la continua evolución de los servicios o

estándares de BPM, a los que continuamente tiene que integrarse dicho motor. El motor de Bonita BPM es muy flexible lo cual facilita la adaptación de cualquier sistema de información esto gracias a que el BEE es completamente configurable manipulando el archivo bonita-server.xml.

El motor de bonita es ejecutado en un segundo plano y es este el encargado de conectar Bonita Studio, los formularios creados con la herramienta form builder, Bonita User Experience y las aplicaciones web personalizadas.

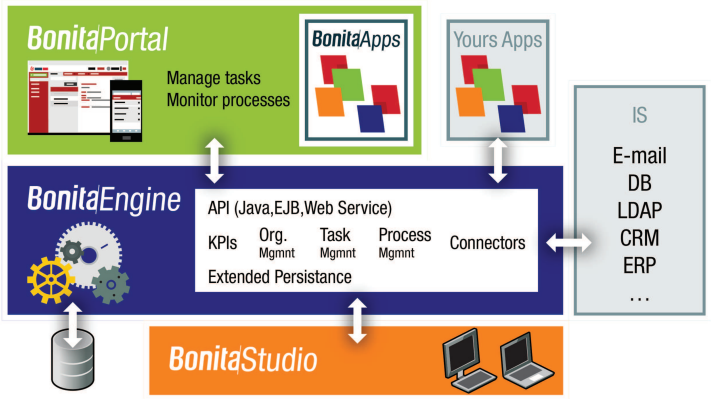


Figura 2.4: Diagrama general del motor de Bonita BPM Open Solution. Fuente: http://community.bonitasoft.com/

2.1.4.3. Corriendo las aplicaciones de Bonita user experience

En este punto Bonita BPM genera sin complicaciones las respectivas aplicaciones según lo necesite cada proceso, ayudando a la interacción entre el usuario y el proceso, esta es una de las facilidades que bonita BPM brinda al usuario haciendo que su experiencia al utilizarla sea fácil, intuitiva y agradable, cabe indicar que esta no es una característica común entre las soluciones BPM. Bonita BPM cuenta con una consola que ayuda al usuario a administrar y monitorear sus procesos, esta consola es similar a la de un correo electrónico por lo que resulta muy intuitiva para quienes han tenido experiencia al utilizar correos electrónicos.

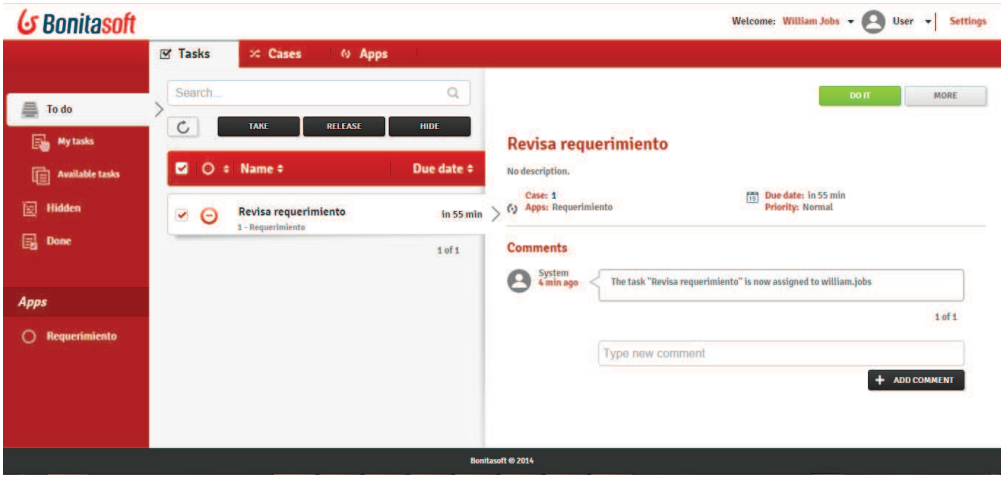


Figura 2.5: Administrador de procesos de Bonita BMP

Esta consola brinda al usuario una perspectiva global de los avances de todos sus procesos, además de permitirle configurar tiempos, prioridades y más características de los mismos como pueden ser: número de minutos a esperar entre las actualizaciones automáticas, número de días en los que un proceso puede entrar en situación de riesgo. También se puede habilitar una vista gráfica de dichos avances.

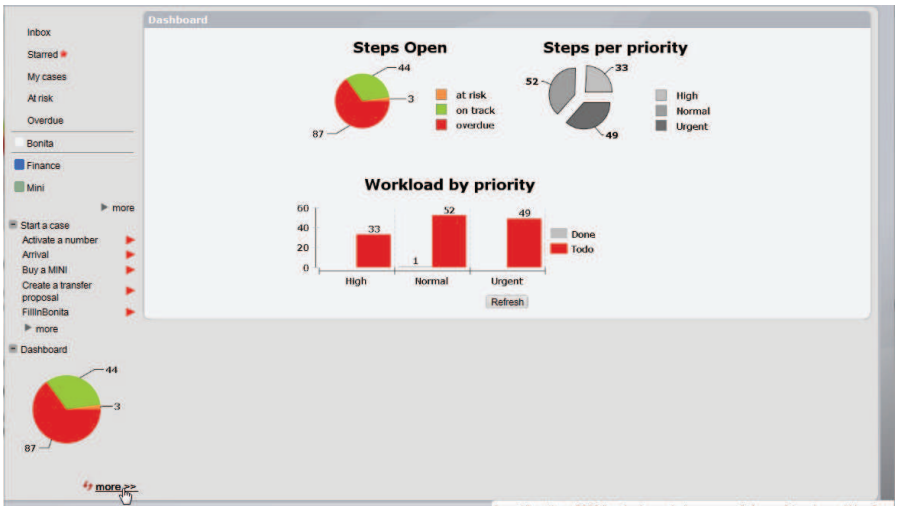


Figura 2.6: Dashboard de reportes en Bonita BPM. Fuente: www.bonitasoft.com

2.1.4.4. Conectividad

Una solución BPM casi siempre se tiene que integrar a plataformas ya existentes en cada empresa o institución como por ejemplo ERP, sistemas de

mensajería, bases de datos, sistemas de gestión de contenidos, aplicaciones de correo electrónico, etc. La conectividad con otros sistemas de información usando Bonita BPM prácticamente está asegurada ya que consta con más de 150 soluciones de conectividad, y esta lista de conectores crece aún más con los aportes de la comunidad. Si un determinado conector a cierto sistema no existe no es un problema ya que es muy sencillo agregar un conector nuevo con la herramienta Creator Conector Bonita Studio, el cual solamente necesita que se establezcan los parámetros requeridos por el sistema al que se quiere conectar y luego hasta se podrá compartir dicho conector.

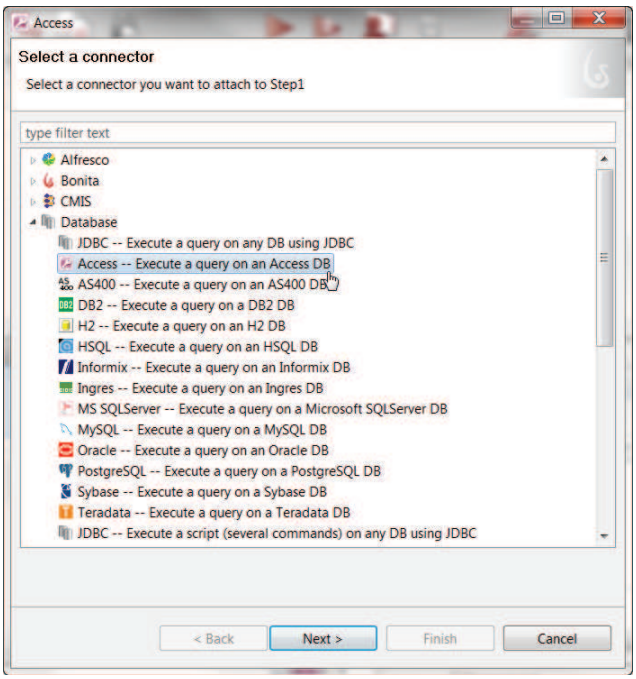


Figura 2.7: Ventana de selección de conectores. Fuente: www.bonitasoft.com

2.1.5. Usuarios de Bonita Open Solution

Básicamente en los entornos o escenarios que brinda Bonita BMP, encontraremos tres tipos de usuarios que estarán siempre relacionados e interactuando entre sí, estos son:

2.1.5.1. Usuario de negocio:

Son los dueños y responsables del éxito de los procesos del departamento al que pertenecen, estos usuarios definen las prioridades de los procesos según su productividad y aporte al buen desempeño de sus áreas de trabajo, descubren cuellos de botellas y tratan de agilizar los procesos, con el enfoque de alcanzar los objetivos de su departamento y empresa en general.

2.1.5.2. Usuario técnico:

Son los encargados de interpretar los requerimientos de los usuarios de negocio para luego poder traducirlos a un lenguaje técnico y de esta manera poder gestionar a través de la herramienta, despliegan las aplicaciones que ayudaran a la ejecución de procesos, se encargan de la conectividad de Bonita con terceros, y vigila el buen rendimiento de las aplicaciones, es el que aporta desde el área técnica para una mayor productividad de la empresa.

2.1.5.3. Usuario final:

No se preocupan por definir los procesos ni por la implementación técnica de los mismos, colaboran entre usuario para lograr completar más rápidamente los procesos, son los que interactúan directamente con las aplicaciones, a estos usuarios se les asigna las diferentes tareas según su rol, grupos de usuario y definición de derechos y privilegios dentro de la empresa.

2.1.6. Beneficios de usar Bonita Open Solution

Ya hemos hablado sobre los tres actores principales dentro de una implementación de BPM con Bonita, ahora veremos rápidamente los beneficios que estos actores reciben al usar Bonita BMP.

2.1.6.1. Usuarios de Negocio:

- Estos usuarios encontraran mayor facilidad para alinear rápidamente sus áreas de negocio con las TI.
- Al mismo tiempo se facilita y agiliza el mantenimiento y actualización de sus procesos.
- Otra de las ventajas se presenta al momento de recuperar la inversión lo cual es "inmediato".
- Mejora la colaboración entre usuarios.
- Se puede dar un seguimiento en tiempo real a cada proceso.
- Bonita facilita a estos usuarios examinar sus procesos para descubrir errores o ineficiencias como cuellos de botella, pasos innecesarios, o la falta de algún paso, estos análisis aportarán para realizar cambios con el fin de hacer de cada proceso eficiente y cada vez más productivo.

2.1.6.2. Usuario técnico:

- Los técnicos podrán interpretar rápidamente los requerimientos de los usuarios de negocios.
- Las aplicaciones Web son creadas con facilidad y sobre todo a medida de los requerimientos.
- Las aplicaciones se pueden crear sin necesidad de escribir código.
- Las interfaces pueden ser completamente personalizadas.
- Facilidad de crear procesos escalables.
- La conexión con otros sistemas de información es mucho más fácil y rápida.
- La herramienta ayuda a supervisar el rendimiento de la aplicación para evitar pérdidas en la productividad.

2.1.6.3. Usuario final:

- Las aplicaciones son autónomas e intuitivas.
- Se facilita la administración de las tareas gracias a que su diseño es muy parecido a una bandeja de correo electrónico.
- Se tiene una apreciación más global de todas las tareas de cada proceso.
- El usuario tendrá claridad para dar prioridad a las tareas que necesitan ser atendidas primero.
- Desarrollo de las tareas en menos tiempo.
- Se obtiene una mejor comprensión de los procesos y una clara visión del avance de cada uno.
- Los usuarios involucrados con el proceso mejoran su comunicación.
- Existe mayor colaboración entre usuarios con el objetivo de completar en menos tiempo cada proceso.

2.2. Conocer cómo funciona Bonita Open Solution

A continuación brindaremos una visión general sobre el funcionamiento de Bonita BPM.

Como primer paso para poner en funcionamiento la solución BPM Bonita Open Solution, está el diseñar o modelar los procesos, con la colaboración directa de los usuarios involucrados en el proceso.

Una vez modelado el proceso lo siguiente es desarrollar el proceso y darle funcionamiento mediante la implementación de conectores o y formulario según

la necesidad de cada proceso.

El siguiente paso sería la ejecución del proceso, ahora los usuarios finales solamente tendrán que utilizar los formularios creados en el paso anterior para poner en marcha sus procesos y tareas.

Una vez que los procesos han sido desplegados los usuarios finales puede ejecutar los mismos que hayan asignado a cada uno de ellos según su rol dentro de la institución, además pueden administrar y monitorear cada proceso y avance mediante el módulo de Bonita User Experience.

Figura 2.8: Funcionamiento general de Bonita BPM. Fuente: www.asolif.es

2.2.1. Arquitectura

Como ya hemos descrito anteriormente, Bonita Open Solution cuenta con tres partes básicas entre las que están:

2.2.1.1. Bonita Studio:

Es en donde se diseñan y modelan los procesos que se buscan automatizar los mismos que se cargarán en el motor de Bonita OS.

2.2.1.2. Bonita Engine:

Este es el motor de Bonita OS y es el encargado de la gestión de los procesos, interacción, validación, ejecución, y conexión con otros sistemas de in-

formación, etc.

El motor de Bonita OS también puede interactuar con las aplicaciones del usuario o con las propias aplicaciones de Bonita.

2.2.1.3. Bonita Portal:

Es una interfaz muy intuitiva para la gestión y administración de los procesos. La Arquitectura de Bonita BPM facilita la integración de los procesos de negocio con casi todos los sistemas actuales de TI. A continuación mostramos un diagrama general de la arquitectura básica de Bonita Open Solution:

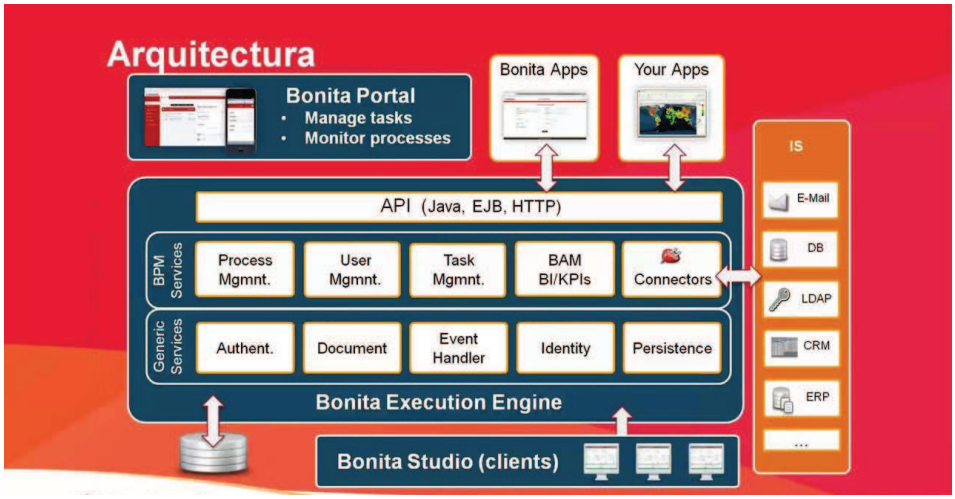


Figura 2.9: Arquitectura de Bonita BPM. Fuente: www.vimeo.com

2.2.2. Escenarios de uso de Bonita Open Solution

Bonita Open Solution se puede aplicar en todas las áreas en donde exista un proceso que se pueda automatizar. A continuación revisaremos brevemente algunos escenarios en los que Bonita Open Solution se aplica con éxito:

2.2.2.1. Gobierno

En muchos paises el área gubernamental siempre ha sido mal vista debido a la ineficiencia en la ejecución de los procesos de la gestión pública y por su alta demora en la respuesta a tramites ciudadanos. Una implementación de Bonita

Open Solution podría aliviar y mejorar de gran manera la capacidad y tiempo de respuesta de dichos procesos gubernamentales y administrativos, reduciendo aun los cotos y mejorando la satisfacción ciudadana. Cabe indicar que a pesar de que cada gobierno es muy diverso en su administración y en la manera como ejecutan sus procesos, la herramienta de automatización se acopla perfectamente dando una solución pertinente a requerimiento.

2.2.2.2. Educación

En las instituciones educativas existen muchos procesos de todo tipo, administrativos, de docencia, de gestión, recursos humanos, etc. los cuales pueden ser automatizados para mejorar el rendimiento del personal y disminuir esfuerzos y recursos, uno de los procesos automatizables podría ser la aprobación de proyectos de tesis o denuncias, las mismas que pueden gestionarse a través de una implementación de un proceso en Bonita Open Solution, la misma que facilitará la interacción del estudiante con las autoridades o secretarías correspondientes, y a su vez se dará un seguimiento y un registro de las versiones del documento, se puede a demás realizar programaciones de las citas que los alumnos tendrán con su tutor, y se podrán registrar las novedades de cada cita. Con la automatización de este tipo de procesos se logra mucha satisfacción en los estudiantes y mucha eficiencia de parte de las instituciones de educación.

2.2.2.3. Bancos

Es muy conocido que en un banco existen una gran cantidad de procesos, los mismos que pueden llegar a ser un dolor de cabeza, tanto para usuarios de los servicios de la institución como para el mismo banco, y además de comprometer un gasto innecesario. Muchos de estos procesos son cruciales y dicen mucho de la institución ya que son procesos que comprometen la atención directa con el cliente como por ejemplo la gestión de “HelpDesk” o “Ayuda de Escritorio” la misma que se automatiza con el fin de buscar mejoras en los tiempos de respuesta.

2.2.2.4. Recursos humanos

Los procesos de la gestión humana en una empresa siempre serán parte importante para tener el personal apropiado en cada departamento de la misma. Procesos como el reclutamiento, selección del personal, gestión de vacaciones, etc. se podrán optimizar y disminuir tiempos y recursos, gracias a la automatización de los mismos con Bonita Open Solution.

2.2.2.5. Manejo de calidad

En una empresa se puede aumentar la calidad de los procesos simplemente aumentando la "gobernabilidad" de los mismos, y esto es justamente lo que nos brinda Bonita Open Solution, y mayor control o gobernabilidad de nuestros procesos, para de esta manera identificar falencias y posibles mejoras.

2.3. Estándares utilizados para el modelamiento de un proceso

2.3.1. La notación BPMN2

2.3.1.1. ¿Qué es la notación BPMN2?

BPMN2 acrónimo de Business Process Model and Notation en su versión 2, nació hace aproximadamente 13 años y se la define como: "Una notación que nos ayuda a representar y detallar gráficamente los procesos de una empresa en un modelo de procesos de negocio." (OMG, 2011). No es propiamente un sistema de información es una norma para el modelo de procesos los mismos que serán interpretados y ejecutados por motores de procesos.

La idea de que BPMN2 es una notación destinada solo a expertos no es del todo cierta ya que es muy intuitiva en cada uno de sus componentes. Se puede decir que es una notación de fácil comprensión y de simple implementación y

capaz de manejar altos niveles de complejidad, hecha para colaboración entre departamentos dentro de la empresa encargados de los procesos de negocios y de las tecnologías de la información.

2.3.1.2. ¿Para qué sirve BPMN?

Los beneficios de esta notación son que permite a las áreas técnicas y las no técnicas (de negocios, comerciales o administrativas) tener una buena y fluida comunicación, y una muy productiva colaboración.

En el desarrollo de un proceso de negocio tendremos que atravesar por algunas etapas las mismas que determinan el flujo y el comportamiento que tendrá en base a diferentes criterios y parámetros, para la representación de este tipo de procesos y acciones es que utilizamos BPMN2, apoyándonos en sus diferentes tipos de símbolos gráficos.

La notación BPMN fue creada para ayudar a los usuarios de negocio y técnicos a plasmar e interpretar complejos procesos sin un mayor esfuerzo y con mucho detalle y precisión manteniendo una comprensión y colaboración mutua y clara.

Con la notación BPMN podemos plasmar nuestros procesos de negocio en diagramas los cuales son perfectos modelos visuales que luego sin mayor inconveniente se podrán traducirlos en programas que se encargarán de la ejecución de nuestros procesos modelados. En este caso Bonita BPM es quien se encarga de dicho proceso de traducción.

2.3.1.3. Elementos de la notación BPMN2

Un estudio de todos los componentes de la notación BPMN2 y sus diferentes combinaciones resulta muy extenso, por lo que vamos a clasificarlos en tres

categorías de enfoque general para facilitar su estudio, identificación y comprensión. Las cuales son:

- Elementos de flujo de trabajo.
- Elementos organizativos.
- Elementos de Legibilidad

Elementos de flujo de trabajo: Son los elementos más básicos en la notación BPMN2 los mismos que le dan el sentido esencial a un diagrama, estos son:

Actividades (Activities): Acciones o tareas que son ejecutadas dentro de un proceso ya sea por un usuario o de forma automática o por subprocesos.

Figura 2.10: Símbolo de una actividad

Existe la posibilidad de crear sub-procesos lo cual le da una gran potencia y profundidad a cada proceso.

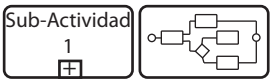


Figura 2.11: Símbolo de un sub-proceso

Eventos (Events): Son elementos que representan algo que genera una acción de inicio, finalización o de un suceso específico durante el desarrollo del proceso que afecta el flujo del mismo, a estos se los clasifica en:

Inicio: Este elemento representa el inicio de un proceso y no es obligatorio ya que el proceso puede ser inicializado por otro tipo de evento.

Intermedios: Son elementos que indican que un evento sucede dentro de un proceso, afectando el flujo del mismo.

Finalización: Señala el fin de un proceso.

Existen varios tipos de eventos entre los que están:

None: En este tipo de eventos no es necesario definir ninguna información del mismo, solamente se sabe que es ahí donde comienza el flujo del proceso.

Figura 2.12: Símbolo de un evento

Message: Este tipo de evento se lanza cuando se recibe un mensaje por otro participante dentro del proceso.

Figura 2.13: Símbolo de un mensaje

Timer: Este evento se dispara cuando se cumple una condición de tiempo, como por ejemplo “cada lunes a las 7:00 am”.

Figura 2.14: Símbolo de un timer

Error: Se lanza cuando se produce un error o se genera un error para capturar.

Figura 2.15: Símbolo de un evento de error

Cancel: Se ejecuta cuando se cancela una transacción o también cancela una transacción.

Figura 2.16: Símbolo de un evento de cancelación

Compensation: Se utiliza para ejecutar eventos que compensan alguna acción que requiera ser cancelada o para generar una actividad de compensación de la actividad en curso o a todas las transacciones que se requiera compensar.

Figura 2.17: Símbolo de un evento de compensación

Conditional: Es un evento que se ejecuta cuando se cumple una determinada condición.

Figura 2.18: Símbolo de un evento condicionado

Link: Con este elemento se representa la conexión entre eventos de diferentes tipos. Puede tener varios orígenes pero solamente un evento de destino.

Figura 2.19: Símbolo de un evento de conexión o link

Signal: Envío y recepción de señales para la comunicación continua durante la ejecución del flujo.

Figura 2.20: Símbolo de un evento de señal

Multiple: Se utiliza cuando existen varias acciones que provocan la ejecución de un evento, y dicho evento también puede tener varias salidas.

Figura 2.21: Símbolo de un evento múltiple

Terminate: Finaliza todas las actividades de un proceso.

Figura 2.22: Símbolo de evento de finalización

Compuertas (Gateways): Son elementos que representan un punto en el que se toma una decisión y se define el flujo que tomará a continuación nuestro proceso. Las compuertas pueden ser:

Exclusive: (Event o Data Based) El flujo después de la compuerta tomará solamente una dirección y dependerá de solo una rama de entrada.

Figura 2.23: Símbolo de una compuerta exclusive

Inclusive: Se pueden considerar una o más ramas de entrada para decidir la dirección del flujo resultante, puede tomar más de una ruta de salida pero al menos tomará una salida.

Figura 2.24: Símbolo de una compuerta inclusive

Complex: Es la combinación de varias compuertas cuando se requiere de condiciones complejas.

Figura 2.25: Símbolo de una compuerta complex

Parallel: Recibe y consume todas las ramas de entrada ejecutando el flujo por todas las salidas, se utiliza cuando se requiere la ejecución de actividades en paralelo.

Figura 2.26: Símbolo de una compuerta parallel

Elementos Conectores: Elementos que proporcionan la estructura del proceso al ser los que conectan a cada objeto dentro del proceso. Los tipos de conectores son:

Secuencia: Flujo de secuencia (Secuence Flow): Para definir la secuencia o el orden en el que se ejecutarán los objetos del proceso.

Figura 2.27: Símbolo de un flujo de secuencia

Mensaje: Flujo de mensaje (Mesage Flow): Se utiliza para representar el intercambio de mensajes que se puede dar entre los participantes del proceso.

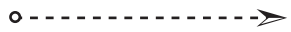


Figura 2.28: Símbolo de un flujo de mensaje

Asociación: (Association): Relaciona los artefactos con los objetos del flujo.

Figura 2.29: Símbolo una asociación

Elementos Organizativos: Para una mejor comprensión de este tipo de elementos diremos que se los puede considerar como los contenedores de los procesos y el flujo de trabajo en general, estos elementos organizativos pueden ser:

Pool: Aquí se puede definir un participante dentro del proceso completo el mismo que estará contenido en un "Pool".

Figura 2.30: Símbolo de elemento pool

Lane: Con el objetivo de tener una mejor organización de nuestras actividades podemos utilizar "Lane" la cual es una partición de un "Pool" ya sea horizontal o vertical.

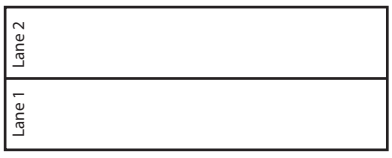


Figura 2.31: Símbolo de un elemento lane

Group: Una vez más tenemos un elemento que nos ayudará a organizar mejor nuestro proceso, agrupando objetos para mejorar la interpretación del proceso sin afectar su flujo.

Figura 2.32: Símbolo un elemento de agrupación

Elementos de Legibilidad: Son elementos que brindan mayor claridad a un proceso haciéndolo más fácil de interpretar y tampoco afecta el flujo del proceso, estos son:

Data Object: Provee información sobre los requerimientos de cada actividad y lo que dichas actividades producirán.

Figura 2.33: Símbolo para representar un data object

Anotaciones: (Text annotation) Nos permite adicionar notas dentro del proceso para facilitar futuras interpretaciones.

Figura 2.34: Símbolo para hacer anotaciones

2.3.2. El lenguaje Groovy

2.3.2.1. ¿Qué es Groovy?

"Groovy nace en el mes de enero de 2007 con su versión 1.0, luego de varias agregaciones finalmente se lanza la versión 1.7 en diciembre del mismo año. Groovy 2.0 fue lanzada en el 2012". (Subramaniam, 2008).

Al hablar de Groovy es inevitable que mencionemos al lenguaje Java, ya que son lenguajes que mantienen una muy estrecha relación.

"EL 99 % del código Java existente puede ser compilado mediante Groovy, y el 100 % del código Groovy es convertido en bytecode Java, y ejecutado en tu JVM de manera natural." (Marco, 2010).

Los programadores Java no tendrán que aprender un nuevo lenguaje ya que Groovy es el lenguaje Java que conocemos con algunas adecuaciones y agregados, de tal manera que también es un lenguaje orientado a objetos.

Se puede decir que Groovy es una forma de mejorar o dinamizar el lenguaje Java ya que se simplifica hasta llegar al punto de escribir solamente código que sea realmente requerido.

Por ejemplo, en Groovy se escribe:

```
println "Hola mundo"
```

en lugar de:

```
public class Saludo
{
    public static void main (String [] args)
    {
        //Se imprimirá nuestro mensaje
        System.out.println("Hola mundo");
    }
}
```

Solo en algunos casos será necesario el uso de los paréntesis y del típico punto y coma ";" al final de cada línea de código, aún el conocido return no será necesario escribir.

Algunas de las características de Groovy que lo hace muy atractivo:

- Su curva de aprendizaje es muy reducida.
- Es el siguiente eslabón en la semántica de Java.

- Otorga mucho dinamismo.
- Extiende la utilización del JDK

2.3.2.2. Groovy y Bonita BPM

Groovy aporta gran potencia a Bonita Open Solution, permitiéndole la interacción con otros sistemas de información y la solución de casi cualquier requerimiento, dentro del proceso, que se pueda solucionar con programación de códigos Groovy.

Como hemos manifestado anteriormente, Bonita BPM consta de un gran número de conectores al momento de requerir una conexión con los sistemas propios de cada institución que va a implementar Bonita BPM o externos al mismo. Sin embargo se puede dar el caso de que no exista un conector definido para un requerimiento específico, este es uno de los casos en donde Bonita BPM utiliza las bondades del lenguaje Groovy, con el que genera los códigos (Groovy scripts) necesarios para entablar la conexión requerida. Sin embargo Groovy es utilizado para muchos requerimientos dentro de un proceso como puede ser cálculos, consultas, expresiones lógicas y más procesos que pueden ser resueltos con scripts de Groovy.

"Bonita Open Solution tiene incorporado un editor de Groovy scripts de modo que no requiere de la utilización de un IDE por separado" (Bhat, 2013), este editor es similar a Eclipse y a otros IDEs, teniendo como una característica común la facilidad que brinda a la escritura de código gracias a la opción que autocompleta nuestros comandos pulsando Ctrl + Espacio.

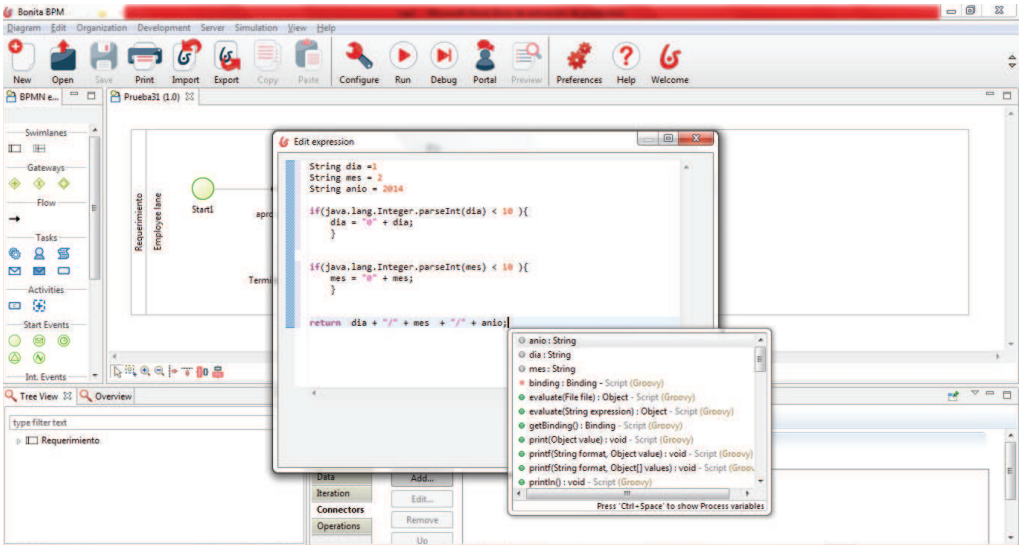


Figura 2.35: Editor de Groovy Scripts

Capítulo 3

Implementación del caso de estudio en BonitaSoft (Sistema de Movilización)

En la Universidad Politécnica Salesiana se ha realizado el levantamiento y diagramación de la mayoría de los procesos que en la actualidad se vienen ejecutando dentro de la institución (mas de 600), entre los que se encuentra el proceso de "Movilización" el mismo que servirá como referencia para el diseño de nuestro diagrama en Bonita BPM Open Solutions.

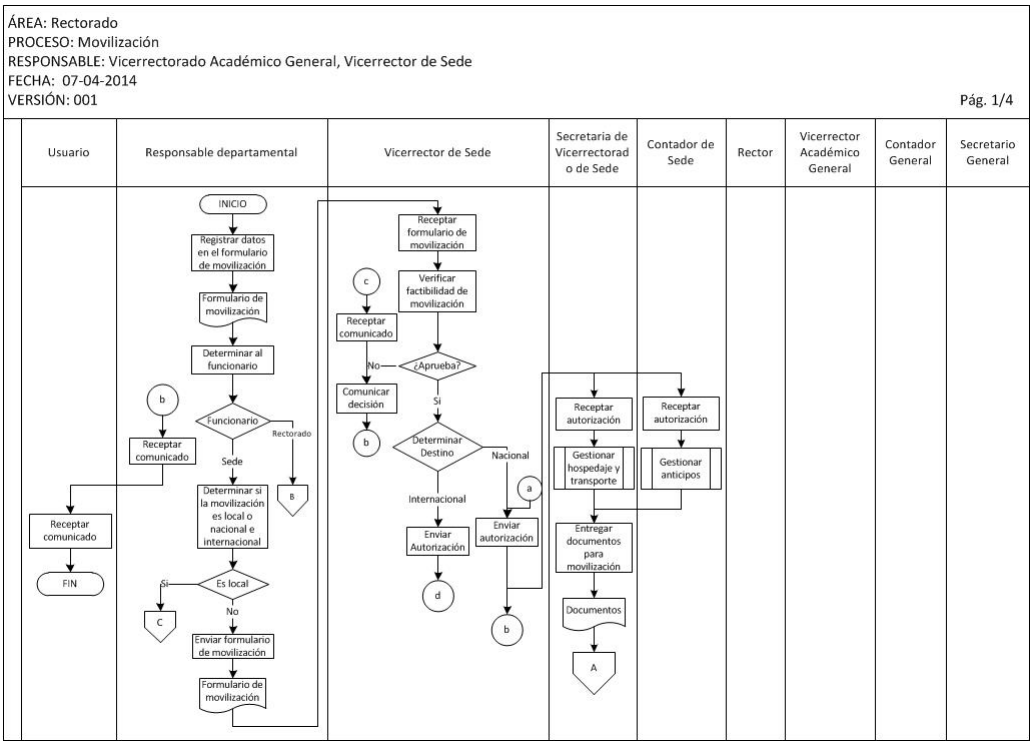


Figura 3.1: Diagrama de Movilización realizado por la UPS, parte 1 de 4. Fuente UPS

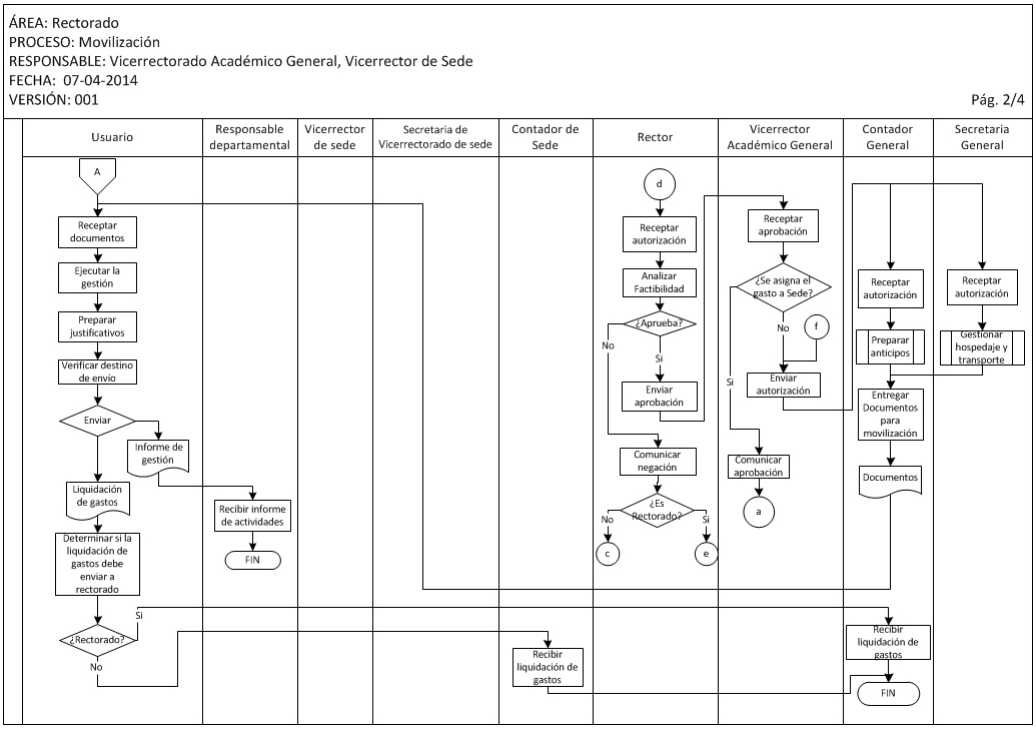


Figura 3.2: Diagrama de Movilización realizado por la UPS, parte 2 de 4. Fuente UPS

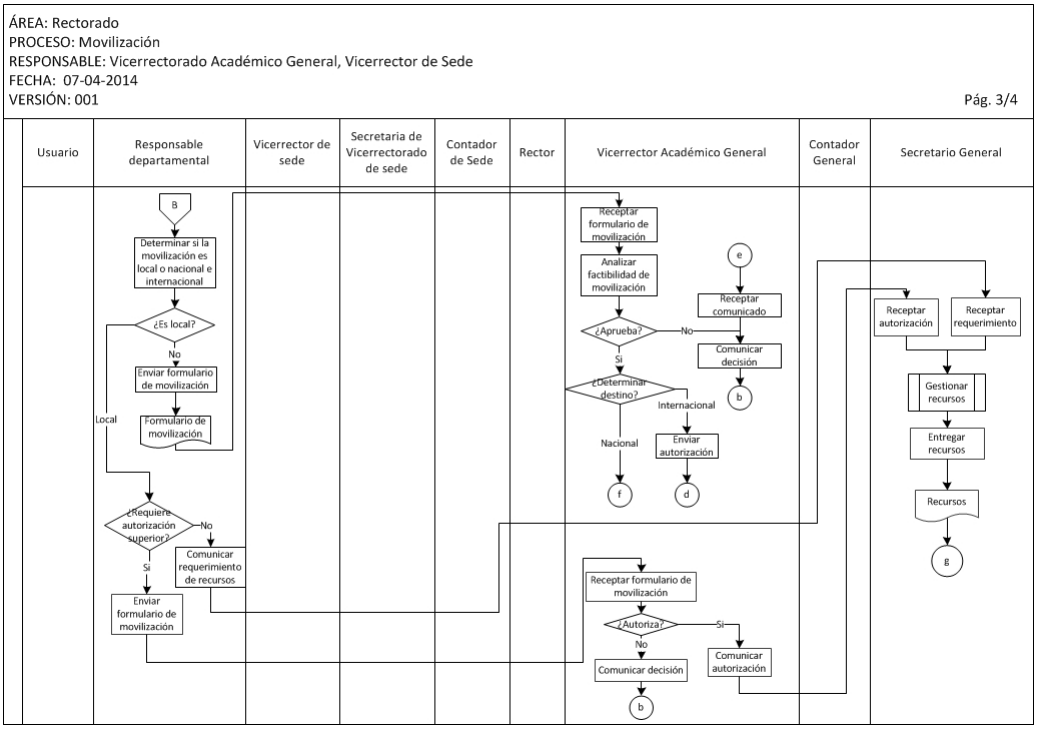


Figura 3.3: Diagrama de Movilización realizado por la UPS, parte 3 de 4. Fuente UPS

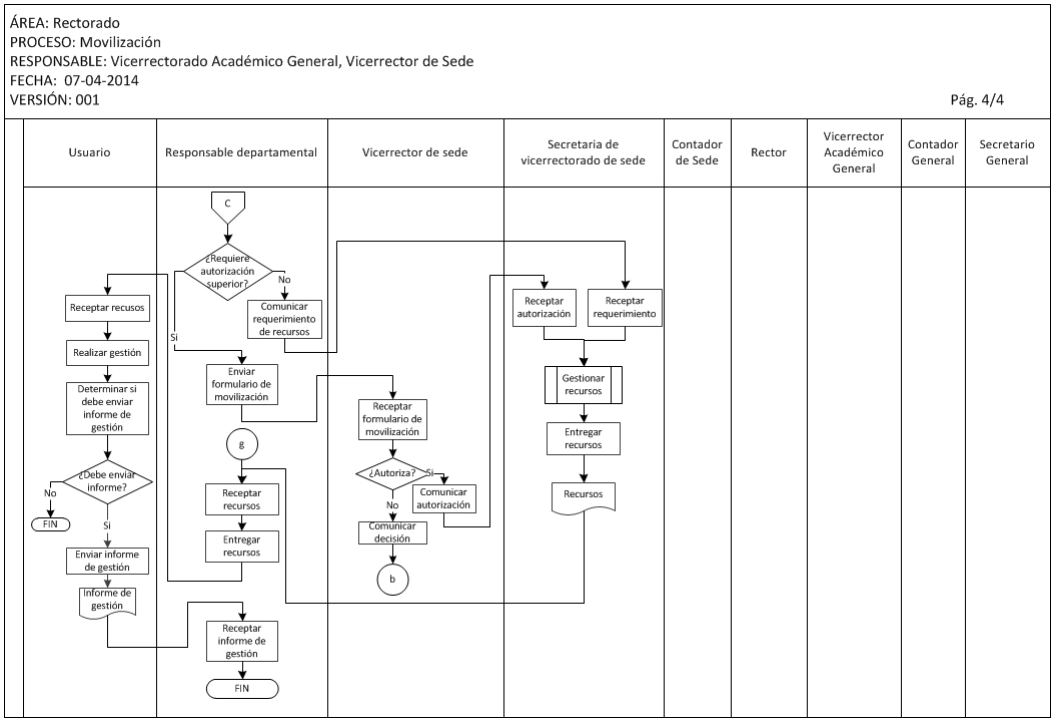


Figura 3.4: Diagrama de Movilización realizado por la UPS, parte 4 de 4. Fuente UPS

En este diagrama podemos observar claramente que en el proceso de movilización están involucradas 9 personas o personajes a las que llamaremos "Ac-

tores", cada uno de los cuales tendrá una senda de acción.

Haciendo una revisión general del diagrama de movilización podremos identificar las siguientes características básicas:

- Son 9 los actores que estan involucrados en el proceso.
- Siempre el "Responsable Departamental" es quien inicia el proceso de movilización.
- Se inicializa el proceso llenando un formulario en donde se ingresan los datos del requerimiento de movilización.
- En base a los datos obtenidos del formulario inicial el flujo puede tener cuatro principales direcciones las cuales se determinan en base al tipo de movilización y al tipo de funcionario que realiza el requerimiento, los cuales pueden ser:

ITERACIÓN	TIPO DE MOVILIZACIÓN	TIPO DE FUNCIONARIO
1	Local	De Rectorado
2		De Sede
3	Nacional o internacional	De Rectorado
4		De Sede

Cuadro 3.1: Valores que determinan el flujo a tomar luego de llenar el formulario de movilización

- Según el valor de estos dos parámetros (Tipo de movilización y tipo de funcionario) se define el flujo que seguirá el proceso, y algunos datos como por ejemplo: si se requiere o no de la autorización superior, quién autoriza el requerimiento, quien gestiona los recursos, etc.

Una vez que hemos revisado y comprendido el comportamiento del proceso que deseamos automatizar, sus actividades, flujos y las personas involucradas, enton-

ces estamos listos para dibujar nuestro diagrama en Bonita BPM.

Como lo hemos manifestado antes el motor de Bonita BPM es una Java API, de tal manera que necesitaremos tener instalado Java en nuestro equipo, lo cual requiere que tengamos configuradas las variables de entorno JAVA_HOME y PATH para que nuestro sistema operativo pueda localizar Java dentro del mismo.

Realizada ya esta configuración procedemos a descargar la última versión de Bonita BPM Open Solution desde la página de BonitaSoft la cual es la empresa encargada en dar soporte a esta herramienta, www.bonitasoft.com.

Figura 3.5: Página web de BonitaSoft en donde se puede descargar Bonita BPM.

Luego de descargar la versión adecuada para nuestro entorno de trabajo entonces procedemos a instalar Bonita BPM e inmediatamente iniciamos con la creación de nuestro diagrama.

Abrimos Bonita BPM y veremos en la pantalla inicial algunas opciones entre las que figurará "Nuevo Proceso" la que elegimos para comenzar a dibujar nuestro diagrama. Bonita BPM mostrará como punto de partida un símbolo de

"Inicio" y una tarea.

Figura 3.6: Opción para crear un nuevo diagrama desde la pagina de bienvenda de Bonita BPM

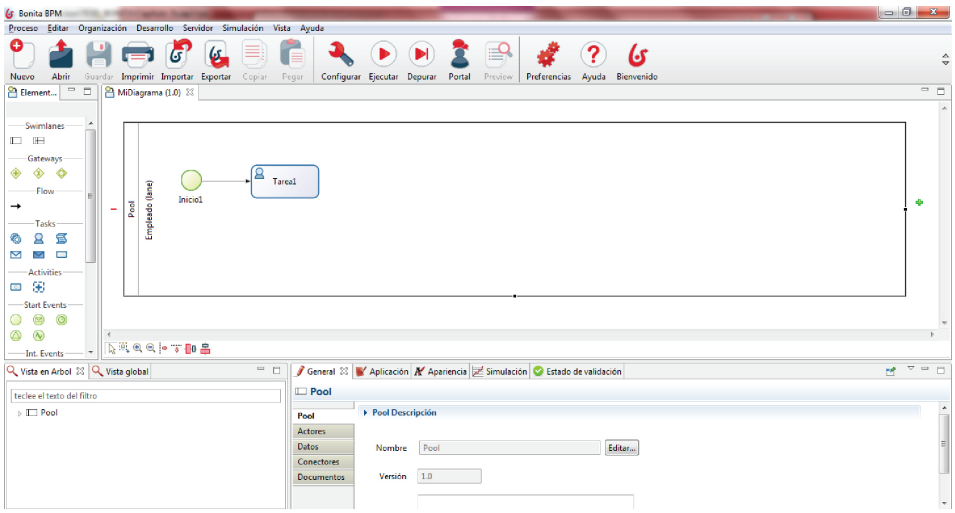


Figura 3.7: Primera pantalla de Bonita con un inicio y una actividad por defecto

3.1. Creación del diagrama de proceso

Como hemos manifestado en el capítulo anterior Bonita BPM utiliza la notación BPMN2 al momento de modelar los diagramas de sus procesos, en esta sección haremos uso cada elemento de BPMN2.

Antes de dibujar nuestro diagrama consideraremos que a los componentes de un proceso dentro de Bonita BPM se los puede clasificar por niveles de complejidad, Básico, Intermedio y Avanzado, como ilustraremos en el siguiente

cuadro:

	BASICO	INTERMEDIO	AVANZADO
ACTIVIDADES	Tareas Abstractas	Tareas Humanas, Tareas de Servicio, Tareas de llamadas	Subprocesos
EVENTOS	Inicio, Fin	En línea y eventos de limitación – ver comportamiento especial.	
GATEWAYS	XOR, AND	Inclusivo	
FLUJO DE SECUENCIAS	Flujo de secuencia	Flujo condicional, Flujo por defecto	
DE COMPORTA-MIENTO ESPECIAL		Mensaje, Temporizador, Error, Señal	Bucle, Multi-Instancia, Transacción, Compensa-ción, Correlación

Cuadro 3.2: Niveles de complejidad de un diagrama BPMN2. Fuente (BonitaSoft, 2014c)

Para la creación de nuestro diagrama trataremos de utilizar esta secuencia de niveles de complejidad, como una guía.

3.1.1. Pool

Como habíamos definido en el capítulo anterior un pool es un componente de organización y contiene un proceso y en nuestro caso definimos un pool al que llamaremos "Movilización". Cambiamos de nombre al Pool en cual viene por defecto como "Pool" y le daremos el nombre de "Movilización". Seleccionamos el Pool y luego en el panel de características seleccionamos la pestaña "General" y luego "Editar" y escribimos el nuevo nombre.

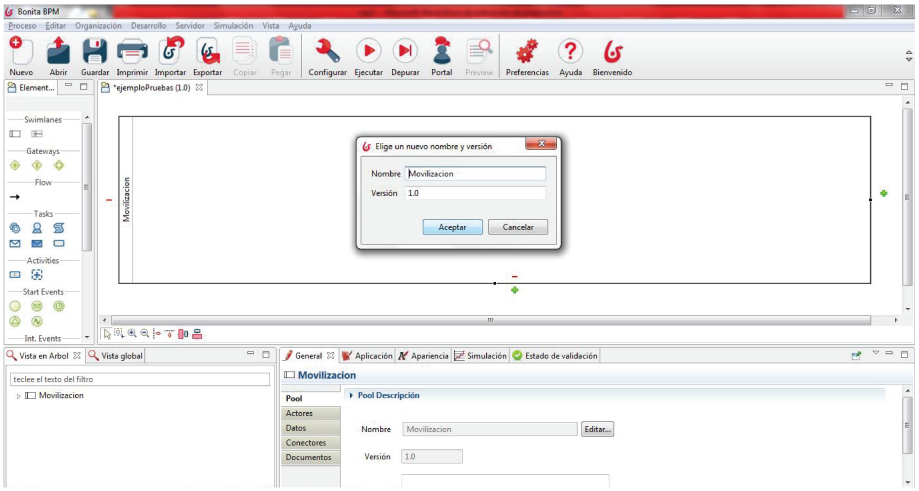


Figura 3.8: Ventana para cambiar de nombre a un Pool

3.1.2. SwimLane, Lane o Sendas

Dentro del pool de "Movilización" definimos cada senda o line que nos servirá para tener más organizado nuestro proceso y para darle más claridad. Las sendas pueden representar un rol, grupo o un usuario en específico. Las sendas que por cada actor declaramos en nuestro caso serían:

- Usuario
- Responsable Departamental
- Vicerrector de Sede
- Secretaría de Vicerrector de Sede
- Contador de Sede
- Rector
- Vicerrector Académico General
- Contador General
- Secretario General

Figura 3.9: Ventana para agregan mas sendas a un Pool

Para crear una senda seleccionamos en la paleta de elementos de la izquierda el símbolo de una senda y arrastramos a nuestro Pool.

De esta manera creamos todas nuestras sendas requeridas teniendo una como resultado una estructura parecida a la siguiente:

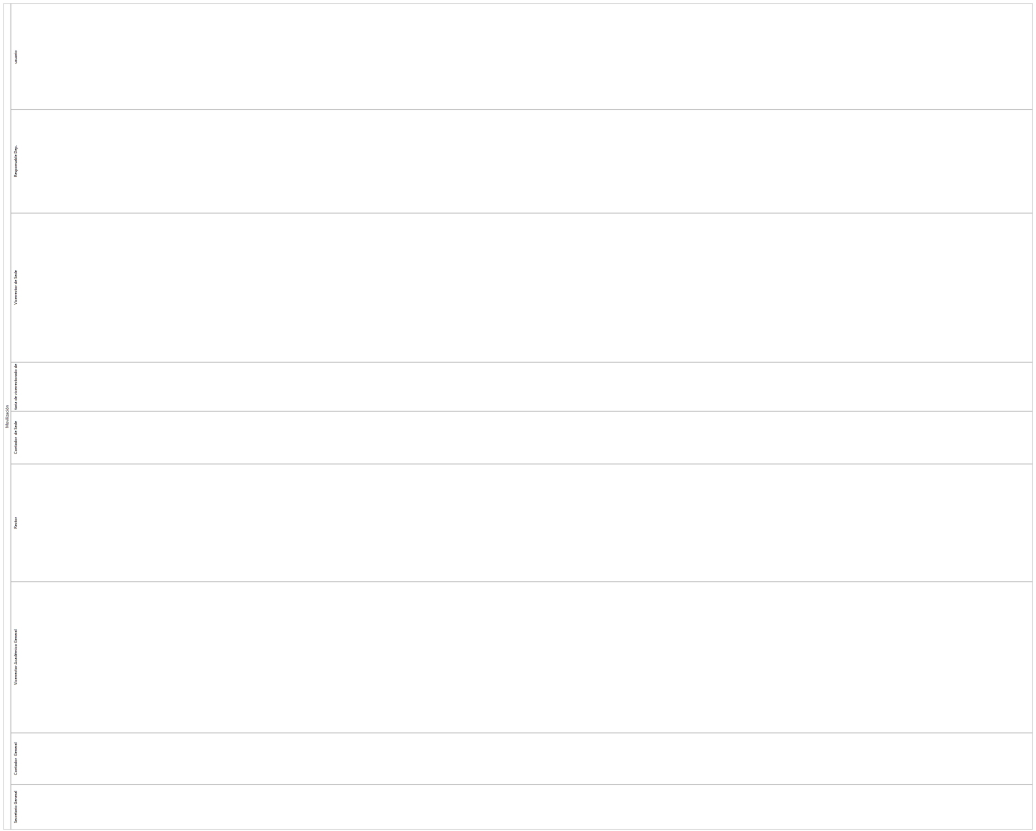


Figura 3.10: Estructura de las sendas definidas para el proceso de Movilización de la UPS.

3.1.3. Actividades

Las actividades del proceso de "Movilización" fueron definidas por la Universidad Politécnica Salesiana por lo que ahora solamente se realizará una depuración del diagrama con el fin de identificar actividades que no son necesarias o que no se puedan automatizar dentro de Bonita BPM, como por ejemplo actividades que las realice una persona en el ámbito físico, sin utilizar Bonita BPM u otro sistema. Para crear una actividad en Bonita Studio basta con arrastrar el icono de actividad o tarea desde la paleta de elementos o desde la esquina superior derecha de uno de los elementos que antecede a nuestra nueva actividad.

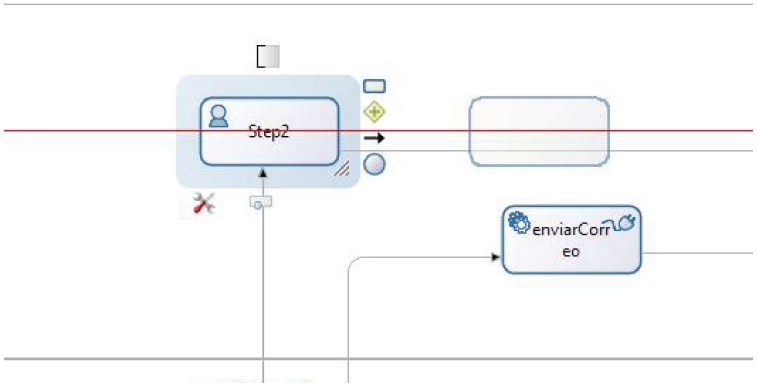


Figura 3.11: Agregando una nueva actividad

3.1.4. Flujos de Secuencia

Los flujos de secuencia son los que nos dirán el camino a seguir a lo largo de todo el proceso. Una vez agregadas las actividades, Gateway, flujos de secuencias, eventos de inicio y de finalización tendremos como resultado un diagrama de nivel básico como el siguiente:

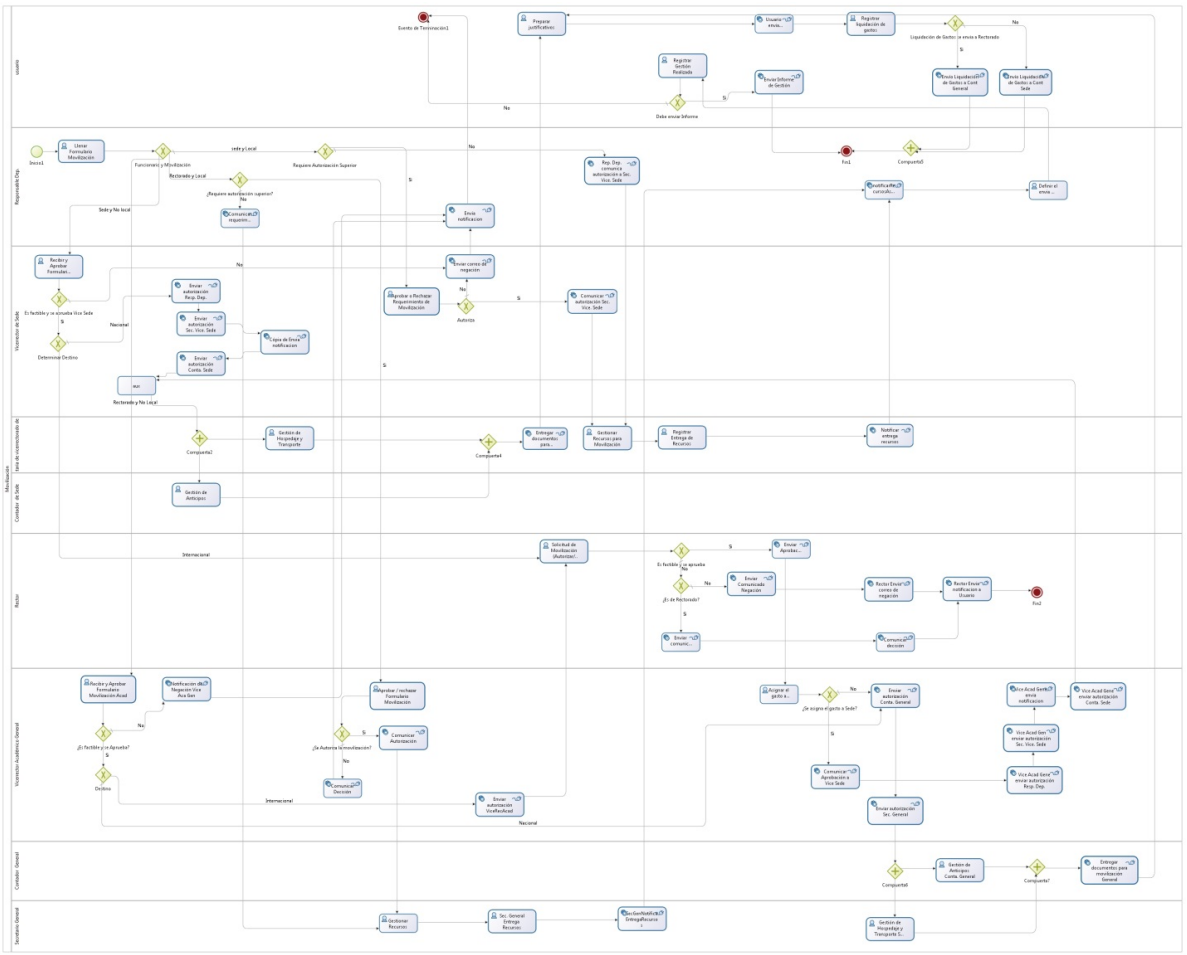


Figura 3.12: Estructura del diagrama de Movilización en Bonita BPM

3.2. Definición de las variables

Dentro de un proceso siempre vamos a solicitar información de parte del usuario o brindar información desde una base de datos o sistema exterior, por lo que es necesario capturar o almacenar esa información dentro de una variable mientras dure la sección de trabajo en Bonita BPM, de esta manera la información podrá viajar a través de cada actividad.

Para definir una variable dentro de Bonita BPM seleccionamos la pestaña "General" en el panel de propiedades de los elementos y luego seleccionar "Datos", luego damos clic en "Agregar" y llenamos los datos solicitados según nuestra variable requerida.

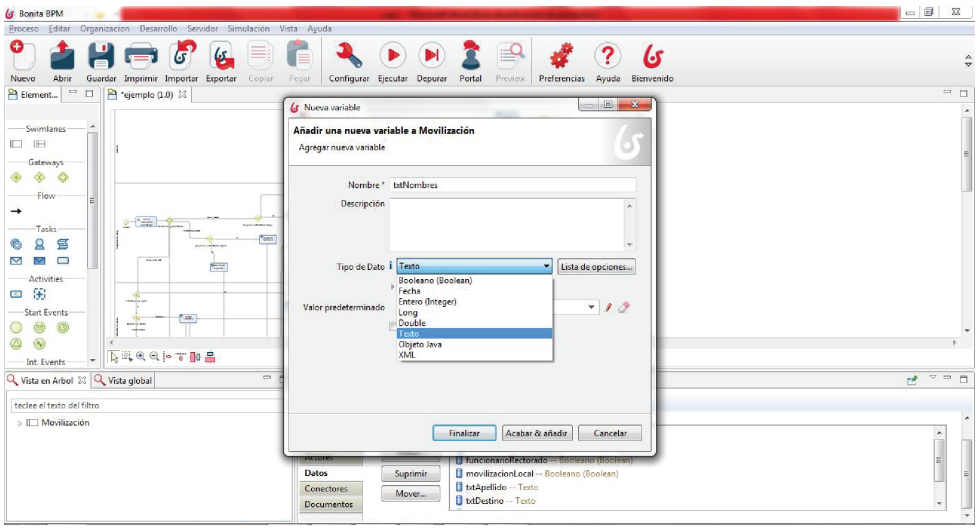


Figura 3.13: Ventana para la creación de variables.

3.2.1. Tipos de variables

Bonita BPM nos brinda la posibilidad de definir los siguientes tipos de variables básicos:

Booleano Con este tipo se pueden definir una variable equivalente a la de tipo Boolean en Java, la misma que la encontramos en java.lang.Boolean, los cual puede tener los valores true o false.

Date En este tipo de variables se pueden almacenar fechas tales como las entradas del widget de tipo fecha de nuestros formularios, en Java encontramos este tipo de dato en java.util.Date que corresponde al tipo Date.

Entero (Integer) Esta es una variable de tipo Integer en Java, pero la encontramos en java.lang.long. Sus valores están entre -2^{31} y 2^{31}-1.

Double Es una variable de tipo decimal la misma que es de tipo Double en Java y la encontramos en java.lang.Double . Sus valores pueden estar entre -2^{63} and 2^{63}-1.

Long Es una variable de tipo decimal pero con una capacidad de almacenar números más grandes que una variable de tipo Double.

Texto Una variable de tipo texto es capaz de recibir cualquier tipo de texto. En java este tipo de variable lo encontramos en java.lang.String la que equivale al tipo String. Con esta variable también se puede definir variables de tipo arreglo que sería un grupo de variables String.

JavaObject Es el tipo de variable más útil dentro de Bonita, ya que nos permite crear un objeto a partir de un archivo .jar de una clase de Java definida anteriormente en cualquier IDE.

XMLSchema (XSD) Para adjuntar archivos, con este tipo de variable podremos representar datos complejos con una estructura XML, bonita nos permite agregar un esquema importando un archivo ASD.

3.2.2. Alcance de las variables

Podemos definir variables de dos tipos con relación a su alcance dentro de un proceso, primeramente tenemos las variables del pool las cuales tienen un alcance equivalente a lo que sería una variable global en un lenguaje de programación como por ejemplo Java.

El otro tipo de variables tienen un alcance parecido a lo que sería una variable local que generalmente se encuentra dentro de un método o procedimiento que serán visibles y accesibles solamente por el método o función local que la contiene.

Para definirla como una variable global o de pool basta con seleccionar el pool al que deseamos agregarle la variable y crear la variable, para definir una variable local bastará con seleccionar la tarea o actividad y crear la variable. Para nuestro caso declararemos las siguientes variables globales.

VARIABLE	TIPO
asignaGastoSede	Booleano
autoriza	Booleano
autorizaRector	Booleano
datosAnticipo	Texto
datosHospedaje	Texto
datosTransporte	Texto
enviaInforme	Booleano
fechaSalida	Fecha
fechaRetorno	Fecha
informeGestion	Texto
justificativo	Texto
liquidacionGastos	Texto
liquidacionGastosEnviaRectorado	Booleano
motivoMovilizacion	Texto
observaciones	Texto
observacionesAcadGen	Texto
observacionesRector	Texto
observacionesVicerrectorSede	Texto
recursosAsignados	Texto
recursosEntregados	Booleano
requiereAutorizacionSup	Booleano
tipofuncionario	tipoFuncionario(“De Rectorado”, “De Sede”)
tipomovilizacion	tipoMovilizacion(“Local”, ”Nacional”, ”Internacional”)
txtNombre	Texto
txtApellido	Texto
txtDestino	Texto
txtEstado	Texto
txtTelefono	Texto

Cuadro 3.3: Variables definidas el el proceso de Movilización

También declaramos variables globales de tipo File o Documento, para utilizarlas al momento de adjuntas archivos en cada proceso y correo de notificación, estas variables se las tiene que declarar en la sección de “Documentos” en el panel de características generales, estas variables son:

VARIABLE	TIPO
logoUPS2	Documento
docInformeGestion	Documento
docJustificativo	Documento
docAnticipos	Documento
docHospedajeTransporte	Documento
docLiquidacionGastos	Documento
docRecursos	Documento

Cuadro 3.4: Variables tipo File declaradas en el proceso de movilización

3.3. Creación de los formularios requeridos

Una vez que hemos definido las variables necesarias para nuestro proceso, entonces podremos, con más facilidad crear los formularios que el usuario final utilizara para desarrollar las tareas humanas. Bonita Open Solution tiene una potente y muy intuitiva herramienta para el diseño de los formularios web, con la habilidad de encargarse de los detalles y con la facilidad de su interfaz drag-and-drop.

Para agregar un formulario a una tarea tenemos que seleccionar la misma y en el panel inferior de los detalles elegimos la pestaña "Aplicación" y damos clic en "Agregar" y nos mostrara una pantalla en la que podremos digitar el nombre, una descripción y además elegir que variables de las que ya hemos definido antes, formarán parte de nuestro formulario.

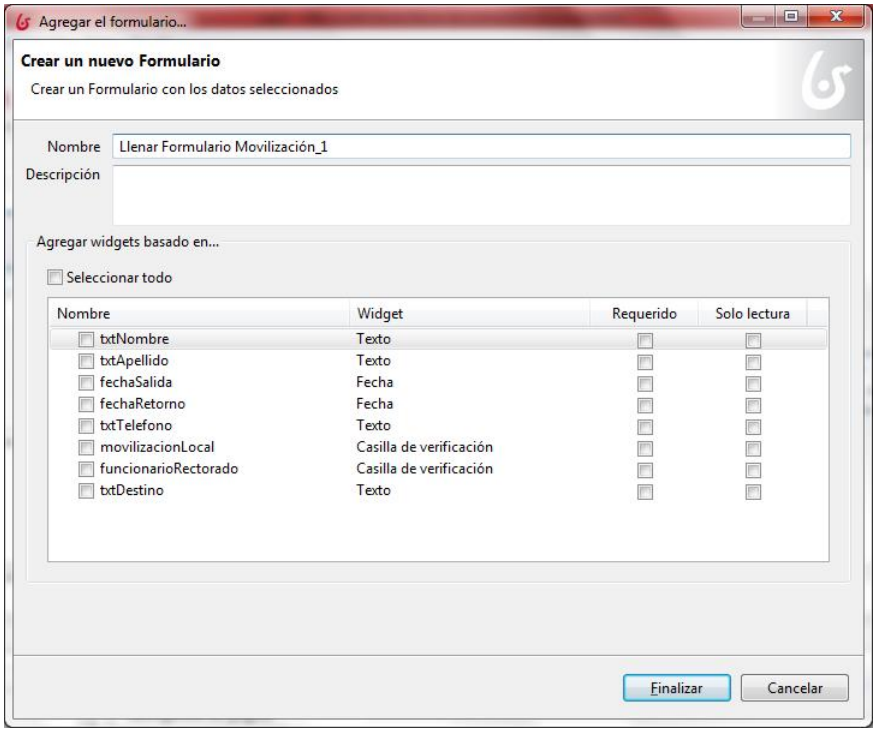


Figura 3.14: Ventana para crear formularios.

Luego de eso podremos ordenar cada elemento de nuestro formulario con facilidad gracias a la característica de drag-and-drop de Bonita.

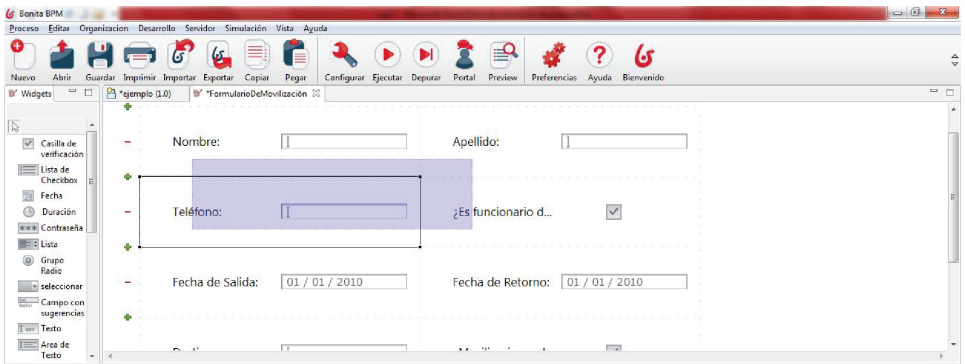


Figura 3.15: Panel drag-and-drop para editar los componentes de un formulario web

Las variables seleccionadas para ser parte del formulario se mapean automáticamente con cada componente del mismo según su tipo.

Algunos de los formularios que se requieren en nuestro proceso son:

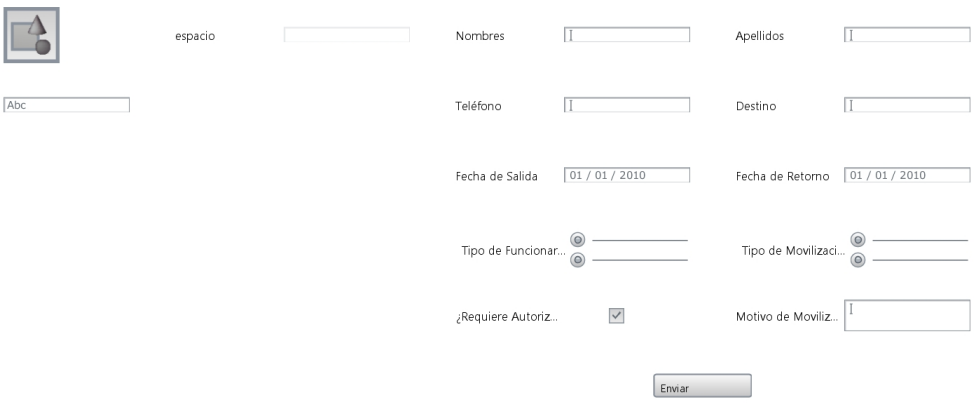


Figura 3.16: Formulario para registrar el requerimiento de Movilización.

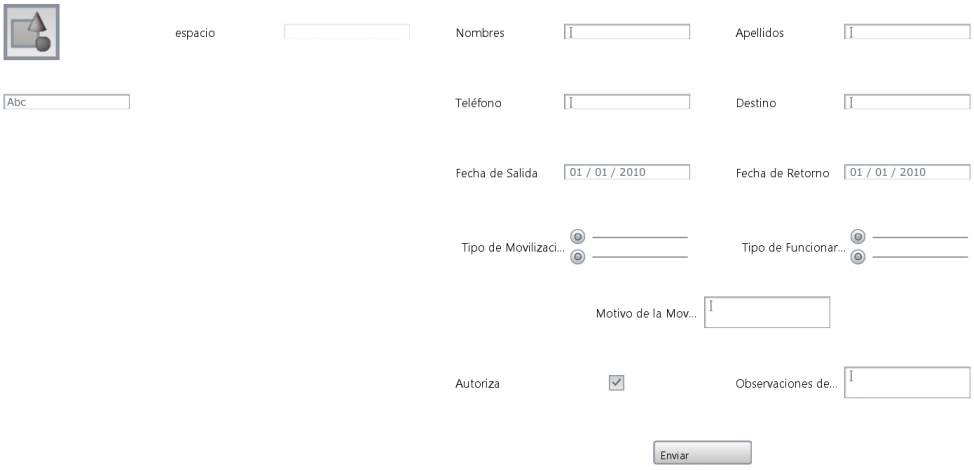


Figura 3.17: Formulario para realizar la contestación al requerimiento de movilización.

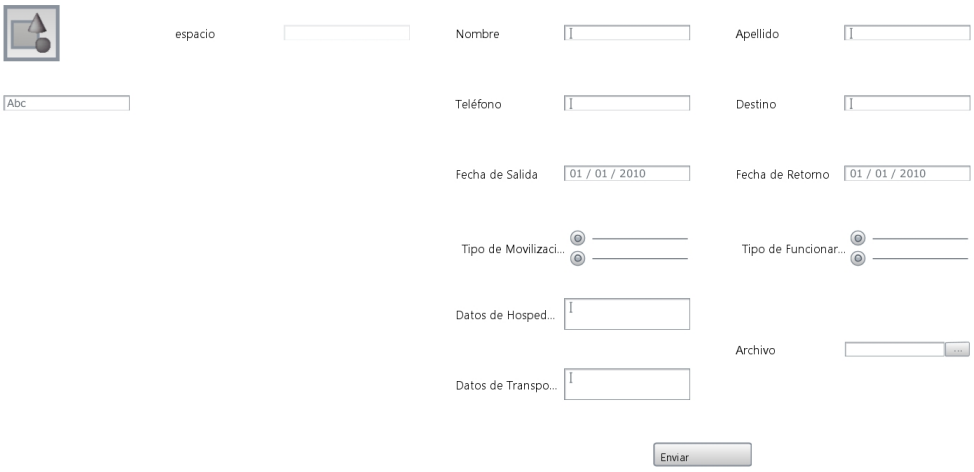


Figura 3.18: Formulario para registrar los resultados de la gestión de Hospedaje y Transporte.

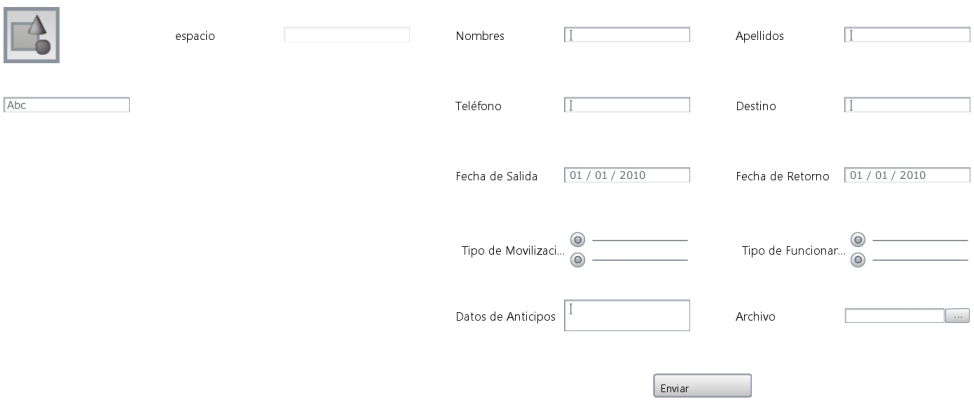


Figura 3.19: Formulario para registrar los resultados de la gestión de Anticipos.

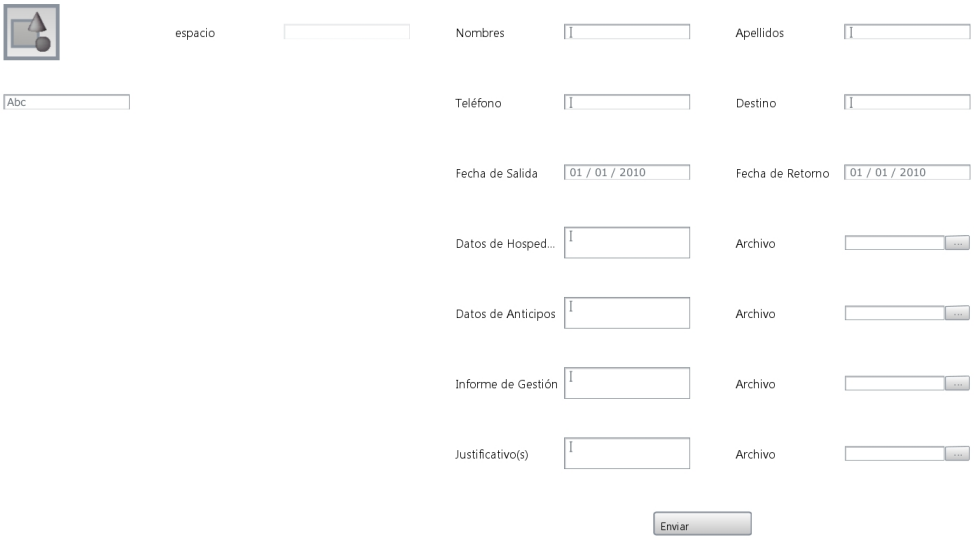


Figura 3.20: Formulario para registrar los resultados de la gestión de movilización y los justificativos.

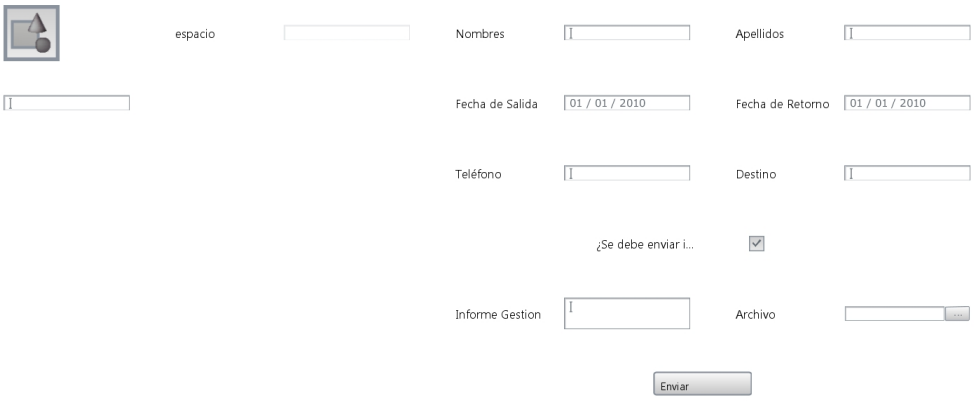


Figura 3.21: Formulario para registrar la liquidación de gastos y especificar destino de los mismos.

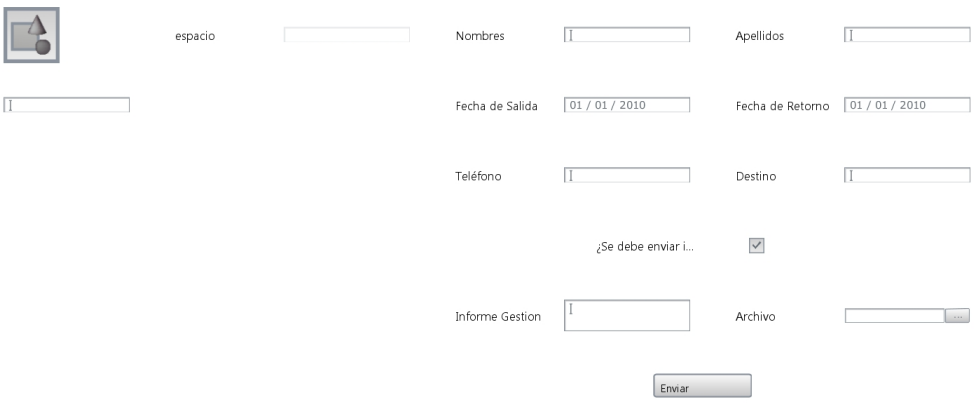


Figura 3.22: Formulario para registrar la Gestión de Movilización realizada.

Figura 3.23: Formulario para registrar los resultados de la gestión de recursos.

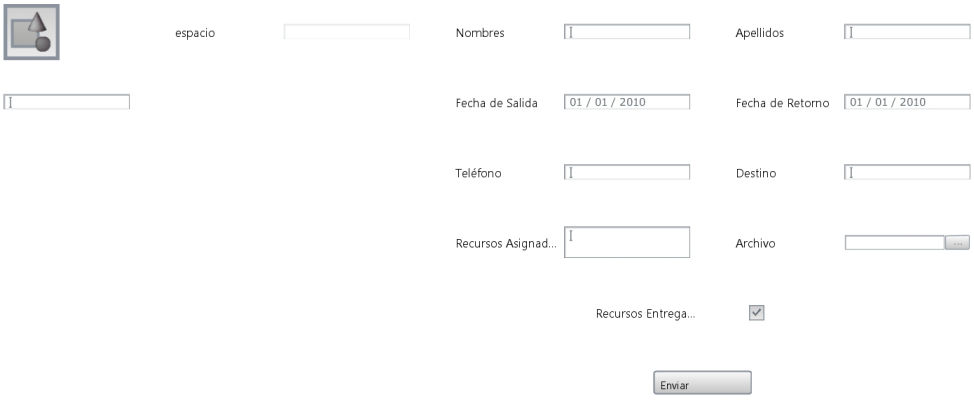


Figura 3.24: Formulario para registrar la Entrega de Recursos.

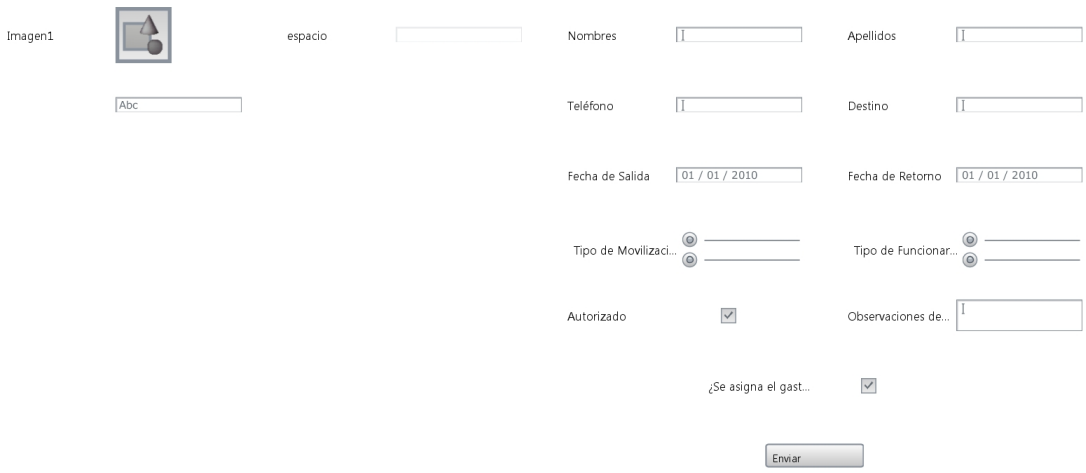


Figura 3.25: Formulario para la Asignación de Gastos

3.4. Especificación de los actores

En Bonita Open Solutions podemos crear lo que se conoce como una "organización" la cual tiene básicamente una estructura como la que ilustramos, en la que podemos definir N número de usuarios, grupos y roles, una membresía se conforma de la unión de un grupo con un rol, y se pueden definir las membresías que sean necesarias.

USUARIO	GRUPO 1	ROL 1	GRUPO 2	ROL 2
	MEMBRESÍA 1		MEMBRESÍA 2	
Usuario 1	X			
Usuario 2	X			
Usuario 3			X	
Usuario 4			X	

Cuadro 3.5: Estructura básica de una Organización en Bonita BPM

Cada usuario puede tener una membresía la misma que se forma de un grupo o subgrupo al que se le asigna un rol. Al momento de realizar un mapeo de los actores y usuarios responsables de cada tarea en Bonita se puede crear un actor el mismo que podrá representar un grupo, rol, membresía o usuario dentro de Bonita.

Para crear una empresa dentro de bonita, en el menú "Organización" seleccionamos la opción "Administrar" y veremos un Wizar que nos guiara en la creación de una empresa para realizar pruebas.

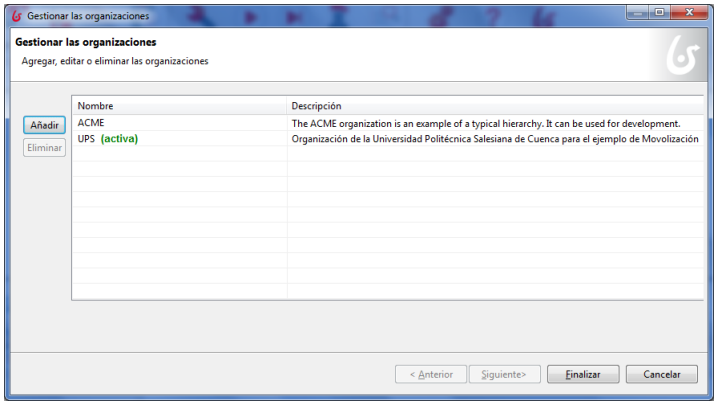


Figura 3.26: Ventana para agregar una nueva organización.

Para definir un grupo y si es necesario sus subgrupos, seleccionamos la organización deseada o creamos una nueva luego damos clic en la siguiente y nos aparecerá la siguiente pantalla:

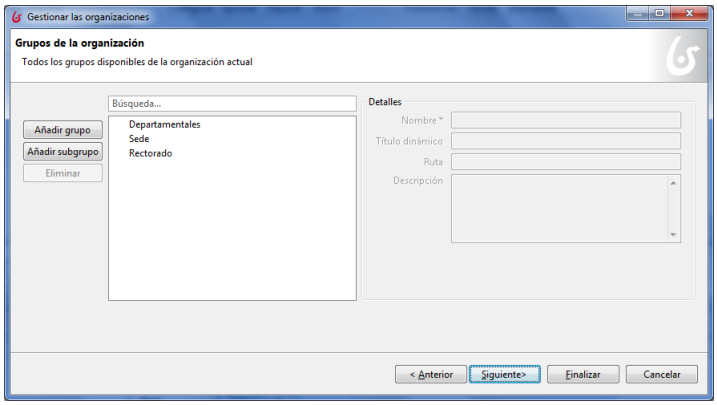


Figura 3.27: Ventana para agregar grupos de trabajo y subgrupos.

Damos clic en siguiente y pasamos a la pantalla donde definiremos los roles que se utilizarán.

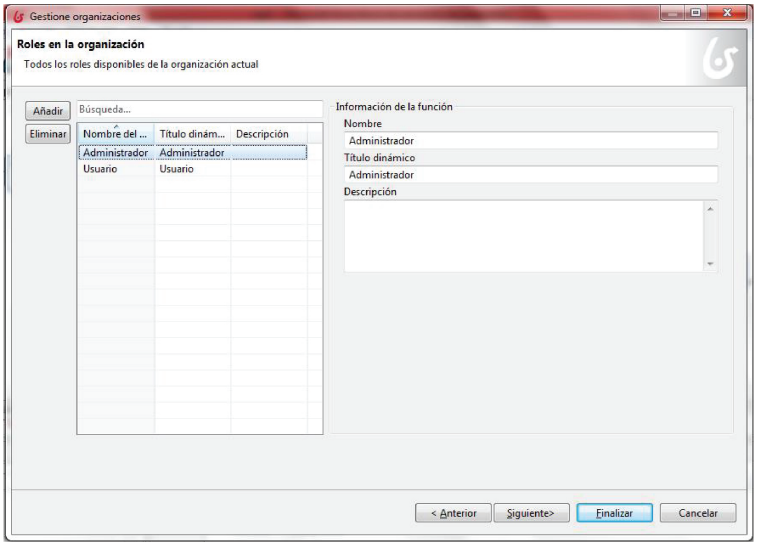


Figura 3.28: Ventana para la creación de roles.

Por último crearemos nuestros usuarios ingresando un nombre de usuario y una clave, en la primera pestaña asignaremos a estos usuarios una membresía al seleccionar un grupo y un rol para cada uno, agregaremos también más información adicional en el resto las pestañas.

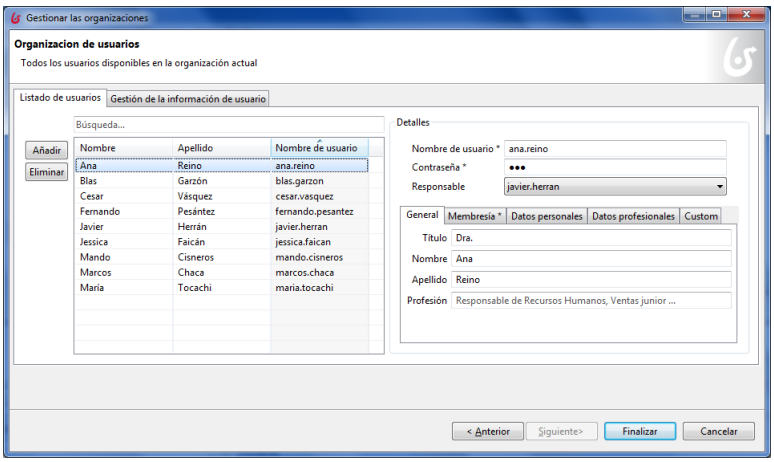


Figura 3.29: Ejemplo de usuarios de la UPS definidos en Bonita BPM.

Podemos definir un actor para toda una senda o para una actividad en particular, Bonita considera de más prioridad a un actor definido en una tarea, de tal manera que el actor que realice dicha tarea será el definido en ella y no el de la senda.

Para nuestro caso del prototipo utilizaremos los usuarios definidos por la UPS:

- mando.cisneros
- blas.garzon
- cesar.vasquez
- maria.tocachi
- jessica.faican
- fernando.pesantez
- ana.reino
- marcos.chaca
- javier.herran

3.5. Configuración del proceso

3.5.1. Mapeo de usuarios

Una vez que el proceso ha sido dibujado en Bonita Studio con sus respectivas actividades dentro de las sendas correspondientes, sus variables definidas y formularios armados procedemos a mapear los actores de las sendas.

Se puede definir para cada actor uno o más grupos, uno o más roles, uno o más membresías, o usuarios, o una combinación de estos. Para nuestro caso asignaremos un usuario por actor. En el menú "Servidor" seleccionamos la opción "Configurar" y procedemos a crear la siguiente estructura:

ACTOR	USUARIO
actorUsuario	mando.cisneros
respDep	blas.garzon
vicerrectorSede	cesar.vasquez
SecretariaVicerrectoradoSede	jessica.faican
vicerrectorAcademico	fernando.pesantez
secretarioGeneral	ana.reino
contadorGeneral	marcos.chaca
contadorSede	maria.tocachi
rector	javier.herran

Cuadro 3.6: Mapeo de los usuarios en el proceso de Movilización

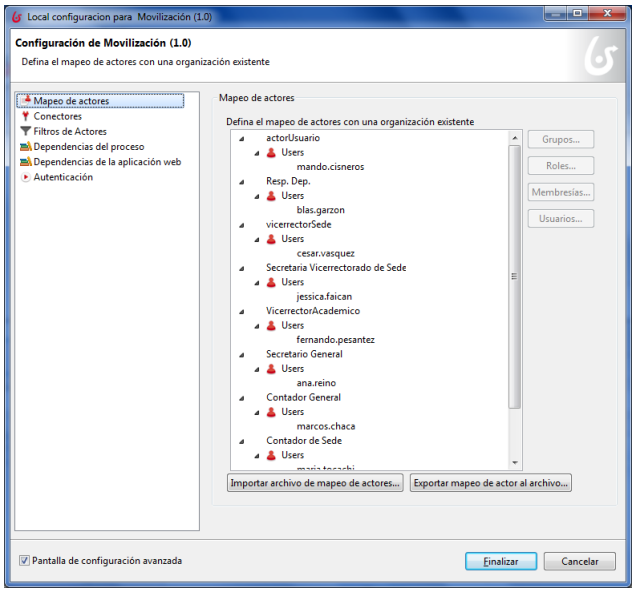


Figura 3.30: Ventana para asignar Grupos, Roles, Membresías o usuarios a cada actor dentro del proceso.

3.5.2. Configurar Gateways

Los Gateways nos guiaran a través del proceso por los flujos correspondientes. El principal y primer gateway que encontraremos en el diagrama de procesos es el que nos canaliza por uno de los cuatro flujos principales del proceso basándose en los datos del formulario de Movilización los que son “tipoFuncionario” y “tipoMovilizacion”.

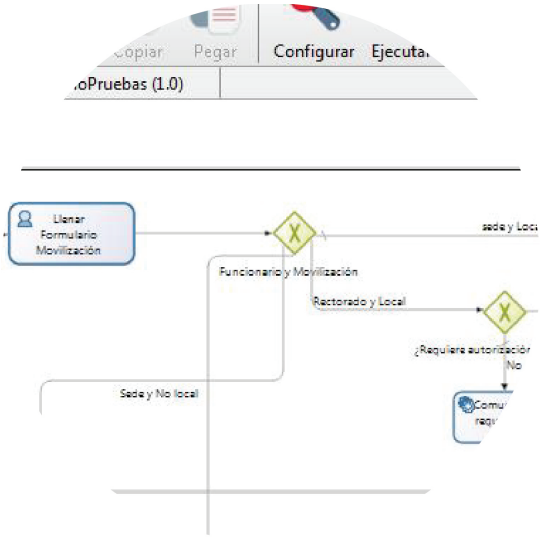


Figura 3.31: Gateway de distribución de los cuatro flujos principales del proceso.

Utilizamos una línea de código de Groovy para realizar la comparación de las variables de tipo de movilización y tipo de funcionario:

La condición para cada caso sería:

Condición para el Flujo 1:

```
(tipoFuncionario == "De Rectorado" && tipoMovilizacion != "
  Local")
```

Condición para el Flujo 2:

```
(tipoFuncionario == "De Sede" && tipoMovilizacion != "Local")
```

Condición para el Flujo 3:

```
(tipoFuncionario == "De Rectorado" && tipoMovilizacion == "
  Local")
```

Condición para el Flujo 4:

```
(tipoFuncionario == "De Sede" && tipoMovilizacion == "Local")
```

Estas cuatro condiciones determinarán el flujo que seguirá nuestro proceso.

3.5.3. Definir actor para la actividad

Luego establecemos una tarea como "Actividad Humana", entonces tenemos que definir un actor para que realice dicha actividad. Lo que haremos en nuestro caso es establecer que la actividad sea hecha por el actor que está definido en la senda y no crearemos otro actor más, esto lo haremos seleccionando la "Actividad Humana" a la que deseamos asignarle un actor seleccionamos la opción "Use el actor definido en la senda" entre paréntesis veremos el actor al que hace referencia. Este trato daremos a todas las actividades.

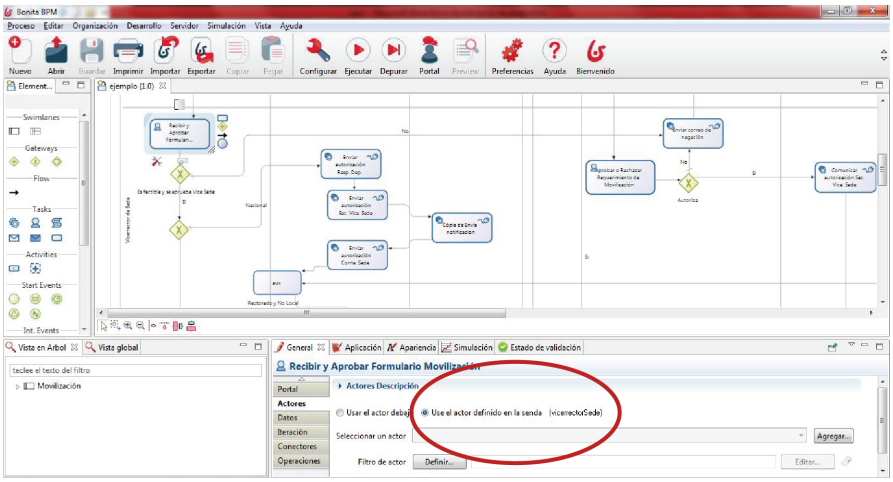


Figura 3.32: Definiendo como actor al definido en la senda para que realice la actividad.

3.5.4. Control de variables

En la gran mayoría de los formularios se solicitará el ingreso de "Observaciones" con respecto a las decisiones que se registran en el mismo, internamente en la sección de "Operaciones" del panel de características generales de la tarea, realizamos una acción que nos ayuda a mantener una única variable de "observaciones" la misma que será utilizada para enviar las notificaciones a los correos, actividades o a los formularios contiguos.

Una de operaciones que también realizamos consiste en asignarle un valor de tipo texto a la variable "txtEstado" de manera que podamos utilizarla en

las notificaciones a través de los correos. Una vez seleccionada la actividad a la que deseamos agregarle una operación, definimos este tipo de operaciones de la siguiente manera:

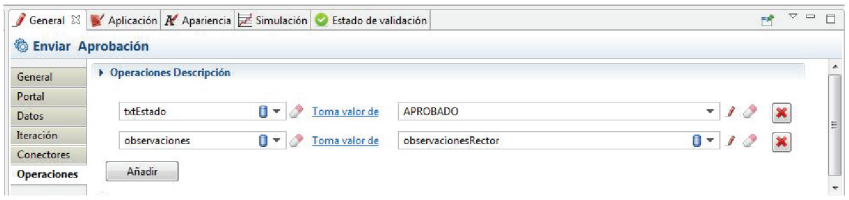


Figura 3.33: Operaciones que se realizan para tener datos consistentes en las variables.

3.5.5. Definir conectores

Los conectores le dan a Bonita BPM una potencia extra, al facilitarle la conexión con casi cualquier tipo de sistema exterior, en nuestro caso haremos uso de estos conectores para realizar el envío de correos electrónicos los cuales nos servirán como notificaciones de las actividades correspondientes.

Para definir nuestro conector seleccionamos la tarea en cuestión y luego en la pestaña "Conectores" damos clic en "Agregar" y nos mostrara la siguiente ventana en donde seleccionaremos el conector deseado, en nuestro caso seleccionamos "Mensajería" y luego "Correo electrónico (SMTP)" y siguiente.

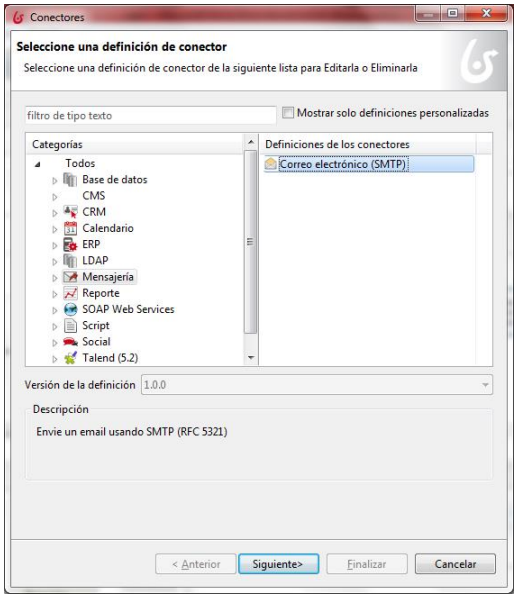


Figura 3.34: Ventana para seleccionar un tipo de conector.

En la siguiente pantalla le damos un nombre y decidimos si el conector se ejecutara al iniciar la actividad o al concluirla, nosotros ejecutaremos nuestros conectores siempre al final, de esta forma permitimos que las operaciones que puedan estar definidas en la tarea puedan ejecutarse primer y luego utilizarse en el conector.

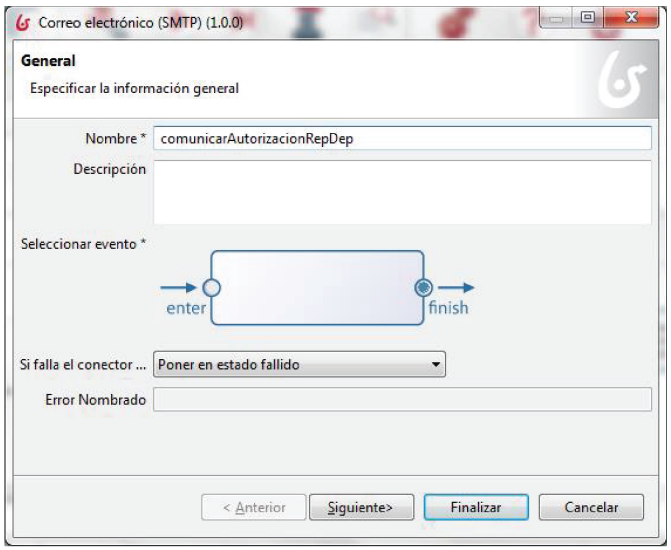


Figura 3.35: Ventana para definir un nombre y tiempo de ejecución de la actividad.

Luego especificamos los parámetros necesarios de nuestro servidor de co-

rreos y los demás datos requeridos.

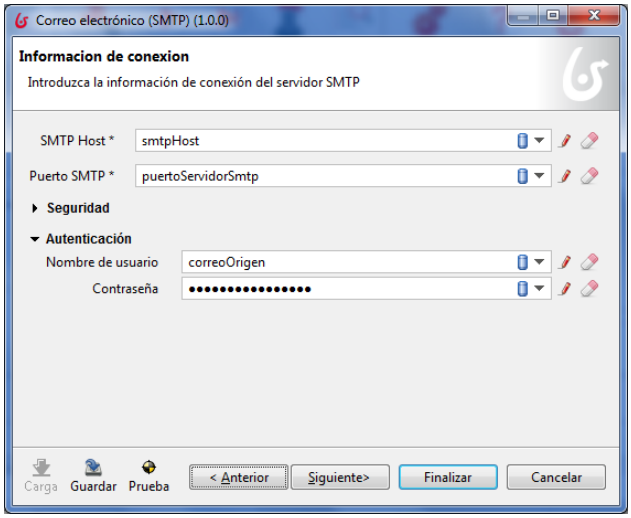


Figura 3.36: Ventana para especificar datos del servidor de correos.

En la siguiente ventana especificamos el correo de origen y el de destino. Para realizar nuestro ejemplo utilizaremos el correo bonita@ups.edu.ec.

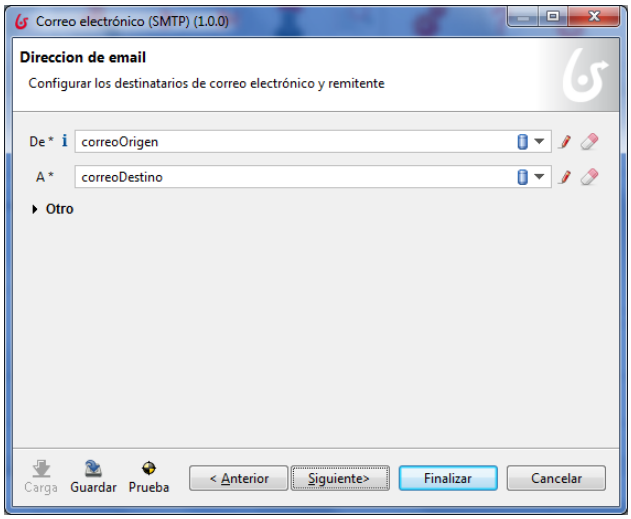


Figura 3.37: Ventana para definir el correo de origen y el de destino.

Al dar clic en siguiente nos aparecerá una ventana en donde escribiremos el mensaje que deseamos que llegue al correo de destino, al activar la casilla "Usar HTML" el Wizar nos permitirá ingresar código HTML para mejorar la presentación de nuestras notificaciones.

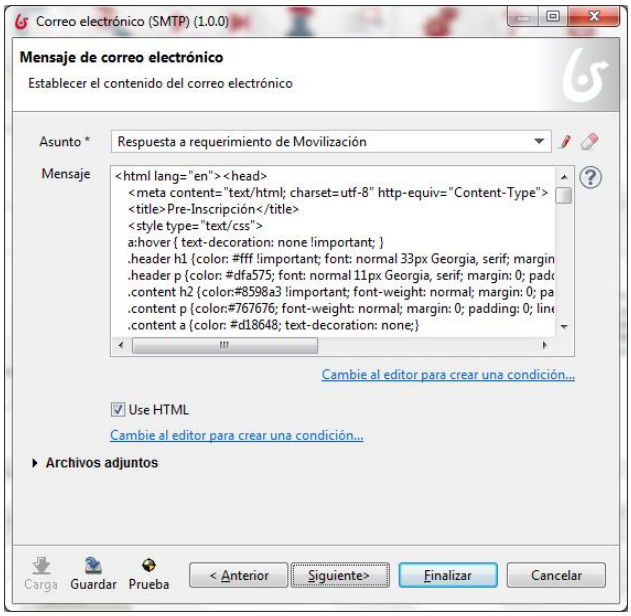


Figura 3.38: Ventana donde se define el mensaje que se enviará

El código HTML utilizado para generar el mensaje que se enviará como notificacion a través de nuestro conector de mensaje electrónico es el que siguie, el cual contiene variables que explicaremos mas adelante:

```
<html lang="en">
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type">
<title>Notificación</title>
<style type="text/css">  a:hover { text-decoration: none !important; }  .header h1 {color
   : #fff !important; font: normal 33px Georgia, serif; margin: 0; padding: 0; line-height:
    33px;}  .header p {color: #dfa575; font: normal 11px Georgia, serif; margin: 0; padding
   : 0; line-height: 11px; letter-spacing: 2px}  .content h2 {color:#8598a3 !important;
   font-weight: normal; margin: 0; padding: 0; font-style: italic; line-height: 30px; font-
   size: 30px;font-family: Georgia, serif; }  .content p {color:#767676; font-weight:
   normal; margin: 0; padding: 0; line-height: 20px; font-size: 12px;font-family: Georgia,
   serif;}  .content a {color: #d18648; text-decoration: none;}  .footer p {padding: 0;
   font-size: 11px; color:#fff; margin: 0; font-family: Georgia, serif;}  .footer a {color:
    #f7a766; text-decoration: none;}
</style>
</head>
<body style="margin: 0; padding: 0; background: #bccdd9;">
<table cellpadding="0" cellspacing="0" border="0" align="center" width="100%">
<tr>
<td align="center" style="margin: 0; padding: 0; background:#bccdd9 url('images/bg_email.
   png');padding: 35px 0">
<table cellpadding="0" cellspacing="0" border="0" align="center" width="650" style="font-
   family: Georgia, serif;" class="header">
<tr>
<td bgcolor="#698291" height="115" align="center">
```

```
<h1 style="color: #fff; font: normal 30px Georgia, serif; margin: 0; padding: 0; line-height: 33px;">Universidad Politécnica Salesiana</h1>
<p style="color: #dfa575; font: normal 14px Georgia, serif; margin: 0; padding: 0; line-height: 11px;">SEDE MATRIZ CUENCA</p>
</td>
</tr>
<tr>
<td style="font-size: 1px; height: 5px; line-height: 1px;" height="5"> </td>
</tr>
</table>

<table cellpadding="0" cellspacing="0" border="0" align="center" width="650" style="font-family: Georgia, serif; background: #fff;" bgcolor="#ffffff">
<tr>

<td width="14" style="font-size: 0px;" bgcolor="#ffffff"> 
</td>
<td width="620" valign="top" align="left" bgcolor="#ffffff"style="font-family: Georgia, serif; background: #fff;">
<table cellpadding="0" cellspacing="0" border="0"  style="color: #717171; font: normal 11px Georgia, serif; margin: 0; padding: 0;" width="620" class="content">

<tr align="center">
<td style="padding: 17px 0 5px; "  valign="middle" align="left">

<img src="https://fbcdn-profile-a.akamaihd.net/hprofile-ak-xfa1/v/t1.0-1/c5.0.160.160/p160x160/10431504_10153298230548906_8028313822133829579_n.png?oh=60e976fbc206757b13b8cb01a74fd284&oe=5533AF14&__gda__=1428431522_a5d6b43545354873ba5a560f362d145d">
</td>
<td style="padding: 25px 0 0;" valign="top" align="left">
<h2 style="color:#8598a3; font-weight: normal; margin: 0; padding: 0; font-style: italic; border-bottom: 3px solid #d2b49b;font-family: Georgia, serif; line-height: 30px; font-size: 22px;font-family: Georgia, serif; ">
Respuesta a requerimiento de Movilización<br>
</h2>
<h3 style="font-size: 14px;font-family: Georgia, serif;margin: 0; padding: 0;">
<p>Estimado(a),</p>
</h3>

<h3 style="font-size: 19px;">
<b><i>Responsable Departamental</i></b>
</h3>

<p style="color:#767676; font-weight: normal; margin: 0; padding: 0; border-bottom: 1px solid #d2b49b; line-height: 20px; font-size: 12px;font-family: Georgia, serif; "align="justify"> El requerimiento de movilización de <b>${txtNombre} ${txtApellido}</b>, con destino a
<b>${txtDestino}</b>, con el objeto de <b>${motivoMovilizacion}</b>, a sido <b>${txtEstado}</b>, con las siguientes observaciones:<br>
<b>${observaciones}</b>.<br><br>
</p>
<h2 style="color:#8598a3; font-weight: normal; margin: 0; padding: 0; font-style: italic; line-height: 30px; font-size: 22px;font-family: Georgia, serif; "><br>
</h2>
<p style="color:#767676; font-weight: normal; margin: 0; padding: 0; line-height: 20px; font-size: 12px;font-family: Georgia, serif; "align="justify">Le comunicamos que se han realizado las notificaciones correspondientes para que el flujo del requerimiento continúe.<br><br>
</p>
```

```
57 <p style="color:#767676; font-weight: normal; margin: 0; padding: 0; line-height: 20px;
   font-size: 14px;font-family: Georgia, serif; "align="justify">
58 Atentamente,<br>
59 <h3 style="font-size: 19px;">
60 <b><i>Vicerrector de Sede</i></b><br>
61 </h3>
62 </p>
63 </td>
64 </tr>
65 </table>
66 </td>
67 <td width="16" bgcolor="#ffffff" style="font-size: 0px;font-family: Georgia, serif;
   background: #fff;">   
68 </td>
69 </tr>
70 </table>
71 <table cellpadding="0" cellspacing="0" border="0" align="center" width="650" style="font-
   family: Georgia, serif; line-height: 10px;" bgcolor="#698291" class="footer">
72 <tr>
73 <td bgcolor="#698291"  align="center" style="padding: 15px 0 10px; font-size: 11px; color:#
   fff; margin: 0; line-height: 1.2;font-family: Georgia, serif;" valign="top">
74 <p style="padding: 0; font-size: 11px; color:#fff; margin: 0; font-family: Georgia, serif;"
   > Este correo ha sido enviado de forma automática desde Bonita Open Solutions, por favor
    no contestar el mismo.
75 </p>
76 </td>
77 </tr>
78 </table>
79 </td>
80 </tr>
81 </table>
82 </body>
83 </html>
```

En este código insertaremos las variables que le darán dinamismo a nuestro mensaje de correo, las cuales son básicamente:

- ${txtNombre}
- ${txtApellido}
- ${txtDestino}
- ${motivoMovilizacion}
- ${txtEstado}
- ${observaciones}

Las mismas que, como puede apreciarse, están entre llaves y antecedidas del signo "$" de esta forma son reconocidas como variables en el entorno. Este será el

formato que se usará para todas las notificaciones a través de correo electrónico.

3.5.6. Adjuntar archivos a las actividades y correos de notificación

Dentro del proceso existen actividades que requieren del envío de un correo electrónico de notificación pero con datos adjuntos, se realizará el mismo procedimiento anterior con un paso adicional, en donde agregaremos los datos adjuntos que previamente son cargados en anteriores actividades.

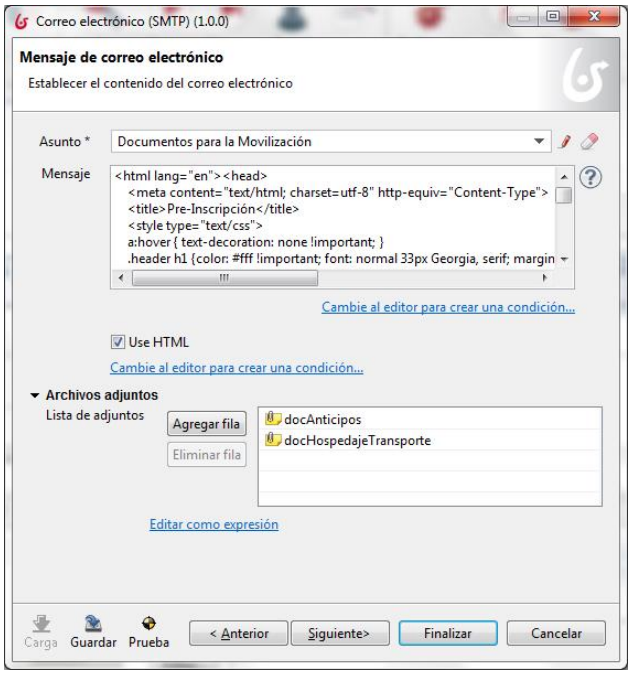


Figura 3.39: Ventana donde se adjuntan dos archivos al correo de notificación (docAnticipos y docHospedajeTransporte)

3.5.7. Ejecutando tareas paralelas

En algunas ocasiones se requiere de la ejecución de dos o más actividades en forma paralela, y luego continuar con el flujo normal, para lograr este comportamiento en nuestro diagrama agregamos la compuerta paralela o conocida también como AND (Y), la cual actúa como agente divergente permitiéndonos dividir el flujo por los dos o más sentidos de salida que tenga la compuerta. Sin embargo la actividad que está a continuación de esta acción paralela corre el

riesgo de ejecutarse dos o más veces según vayan llegando los flujos a ella, para corregir este inconveniente utilizamos otra compuerta paralela la cual actúa como agente convergente, el mismo que espera que todos los flujos se ejecuten para continuar con la siguiente actividad, la misma que se ejecutara una sola vez.

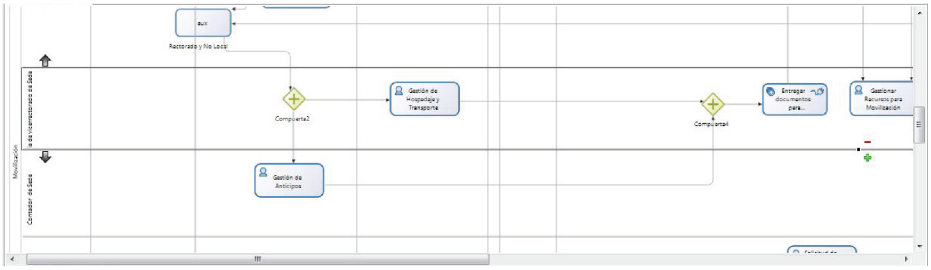


Figura 3.40: Compuerta de divergencia y convergencia.

Esta es la solución que aplicaremos a cada caso similar dentro de nuestro proceso.

3.6. Ejecución del proceso

Para ejecutar nuestro proceso bastará con dar clic en el botón de ejecución de la barra de herramientas.

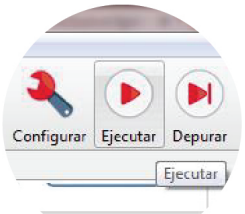


Figura 3.41: Botón de ejecución

Vamos ejecutar nuestro proceso y realizar un caso de ejemplo para verificar el flujo de las actividades y el comportamiento de cada elemento.

Para realizar esta ejecución de prueba necesitamos seleccionar un usuario que será quien por defecto ingrese al portal de Bonita BPM. Esto lo realizamos

en la siguiente ventana, en nuestro caso seleccionaremos al usuario “blas.garzon”, cuyo actor dentro del proceso es “Resp. Dep.” el cual es el único que puede iniciar el proceso de movilización.

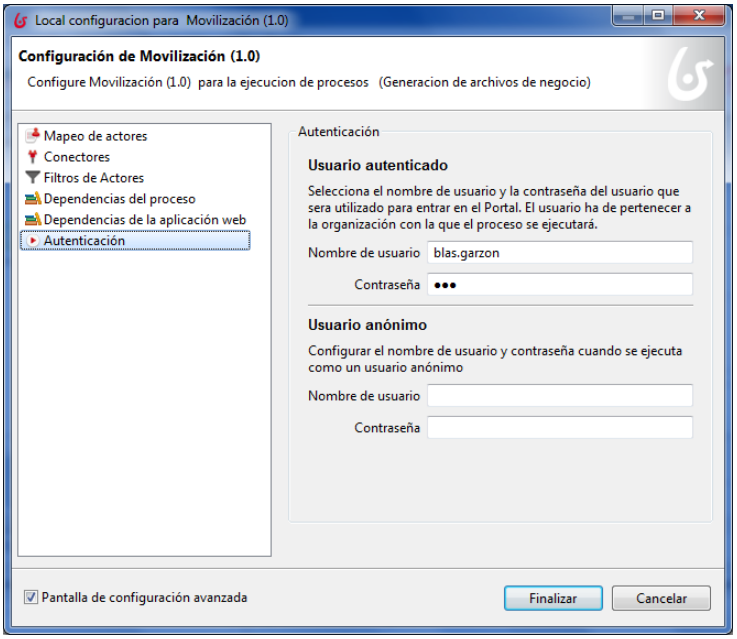


Figura 3.42: Ventana para definir el usuario que ingresará al portal de Bonita BPM por defecto.

Ventana para definir el usuario que ingresará al portal de Bonita BPM por defecto. A continuación veremos la ventana de “Login” con la que cada usuario podrá identificarse e ingresar al sistema Bonita BPM.

Figura 3.43: Ventada de Login de Bonita BPM

Una vez que el usuario “blas.garzon” haya ingresado al sistema, podrá verificar sus tareas pendientes por hacer y en la pestaña “Procesos” podrá verificar

la los que se le han asignado para poder iniciar, en nuestro caso está disponible para blas.garzon el proceso de "Movilización".

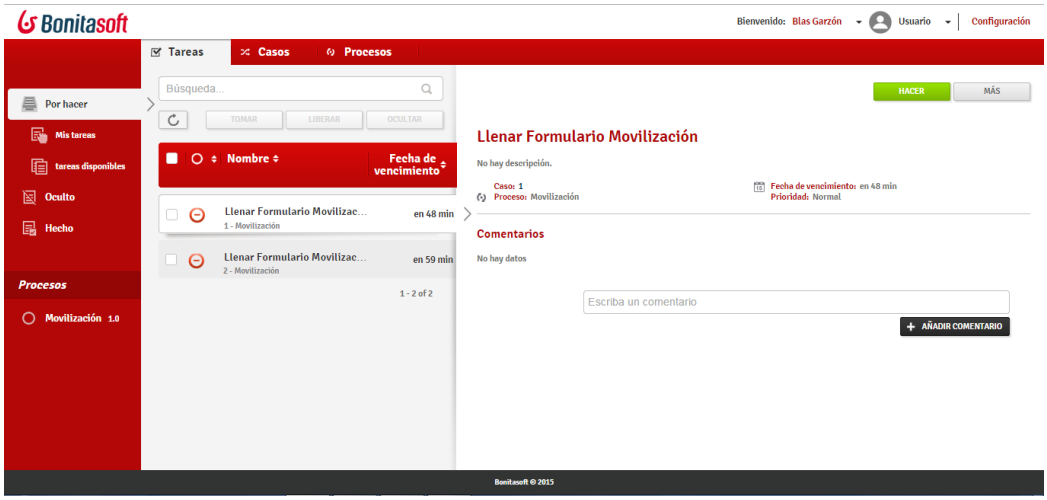


Figura 3.44: Ventana de procesos disponibles para los usuarios

Damos clic en el botón "INICIO" para dar inicio a un requerimiento de movilización. Y se tendrá que llenar el primer formulario. Para este ejemplo a las variables "tipoMovilizacion" y "tipoFuncionario", daremos los valores "Nacional" y "De Sede" respectivamente.

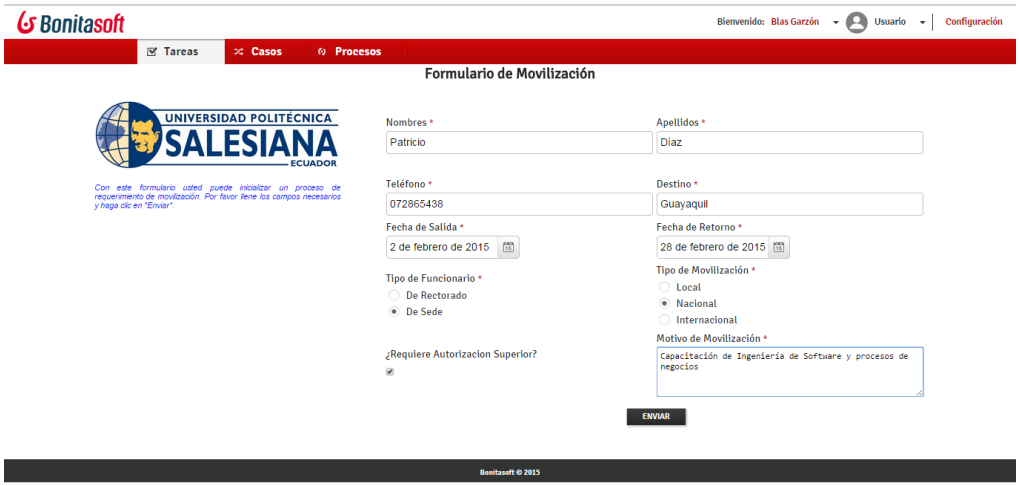


Figura 3.45: Formulario para realizar un requerimiento de Movilización

Siguiendo el flujo del proceso nos encontramos que dicho requerimiento debe ser aprobado por el "Vicerrector de Sede" el cual es "cesar.vasquez". Al in-

gresar con este usuario se verifica que se le ha asignado la tarea de aprobación o negación del requerimiento.

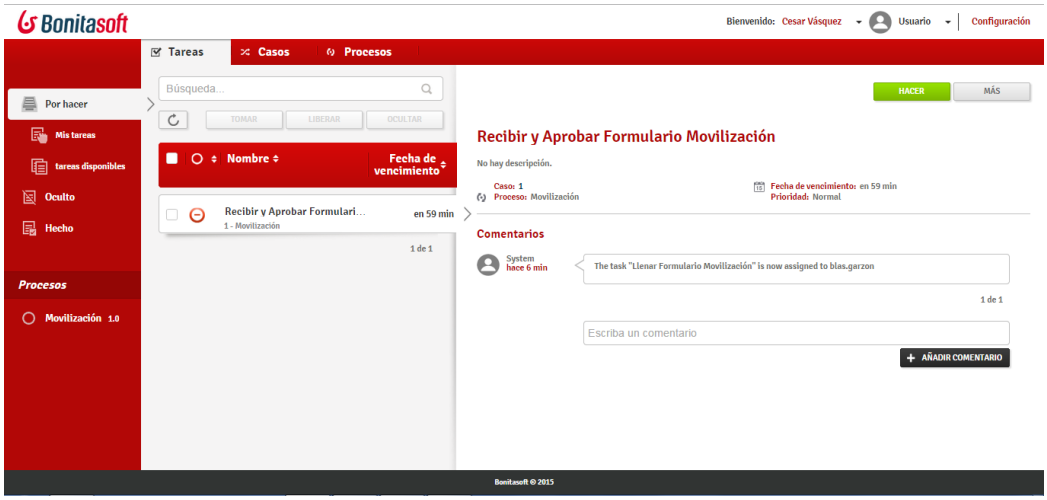


Figura 3.46: Tarea asignada a “cesar.vasquez"

Damos clic en “HACER” y tendremos el siguiente formulario con la opción de “Aprobar” o no el requerimiento y con el campo “observación” el cual es obligatorio para de esta manera poder notificar al usuario la aprobación o la negación de su requerimiento con su respectiva razón.

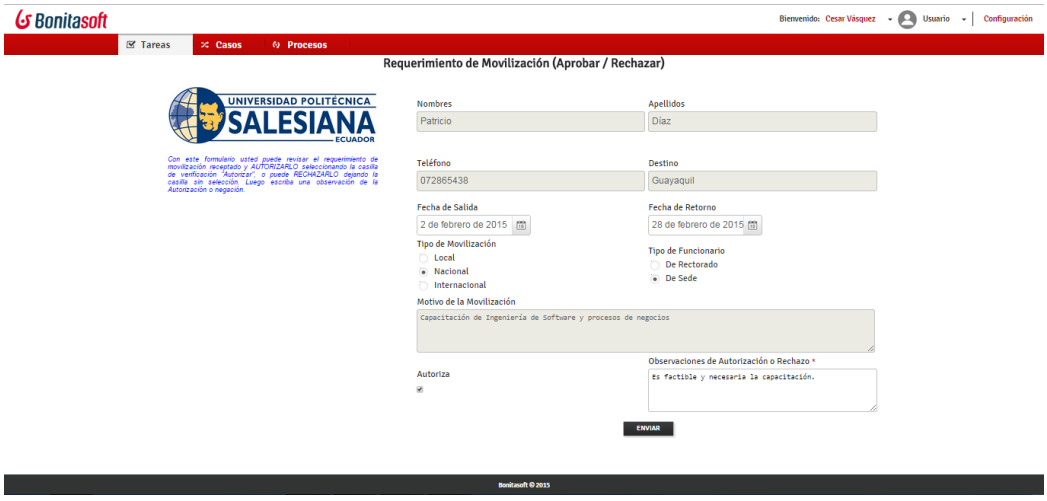


Figura 3.47: Formulario para Aprobar o Negar un requerimiento de movilización.

Si el requerimiento es “Negado” se realizara las notificaciones correspondientes al Responsable Departamental y al Usuario mediante un correo elec-

trónico. Si por el contrario el requerimiento es "Aprobado", el sistema también realizará las respectivas notificaciones de Aprobación a través de correos electrónicos, también informa las responsabilidades generadas a cada funcionario a raíz de la aprobación de dicho requerimiento. Estas notificaciones son las siguientes:

Figura 3.48: Notificación de aprobación para el Responsable Departamental.

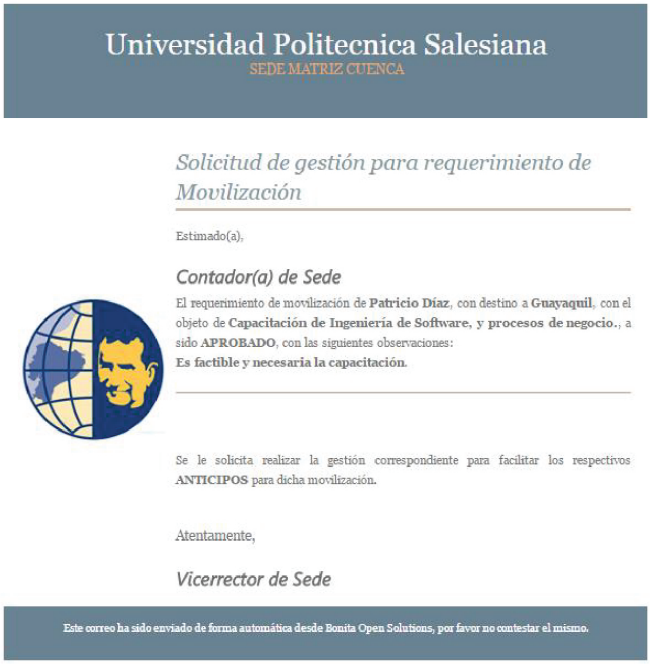


Figura 3.50: Notificación de aprobación para el Contador(a) de Sede.

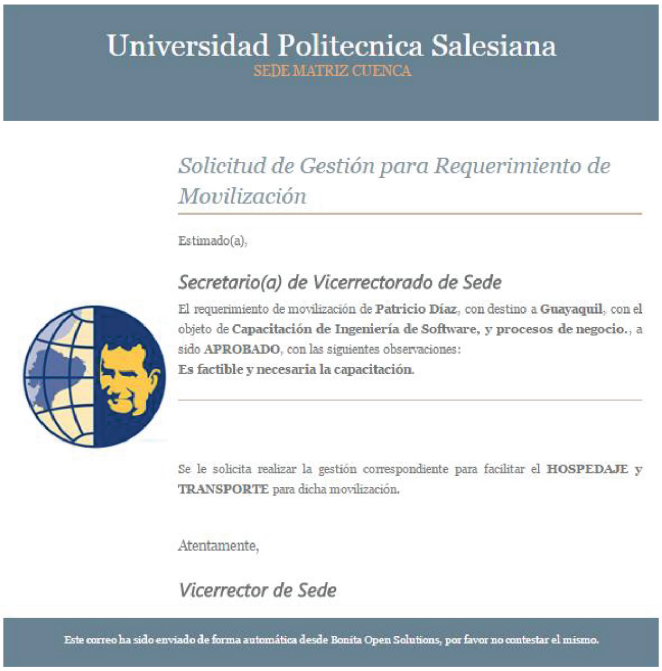


Figura 3.49: Notificación de aprobación para el Secretario(a) de Vicerrectorado de Sede

Figura 3.51: Notificación de aprobación para el usuario que realizó el requerimiento.

Una vez realizadas las notificaciones respectivas, cada funcionario procede a gestionar los recursos según sus propias funciones y el tipo de movilización requerido.

Continuando con el flujo del proceso vemos que se asignan las siguientes tareas: La de "Gestión de Hospedaje y Transporte" al "Secretario(a) de Vicerrectorado de Sede" y la tarea de "Gestión de Anticipos" al "Contador de Sede" de forma PARALELA. Por lo que procedemos a ingresar al sistema con esos usuarios que serían "jessica.faican" y "maria.tocachi" respectivamente.

Verificamos la asignación de la tarea para el usuario "jessica.faican" en la siguiente página:

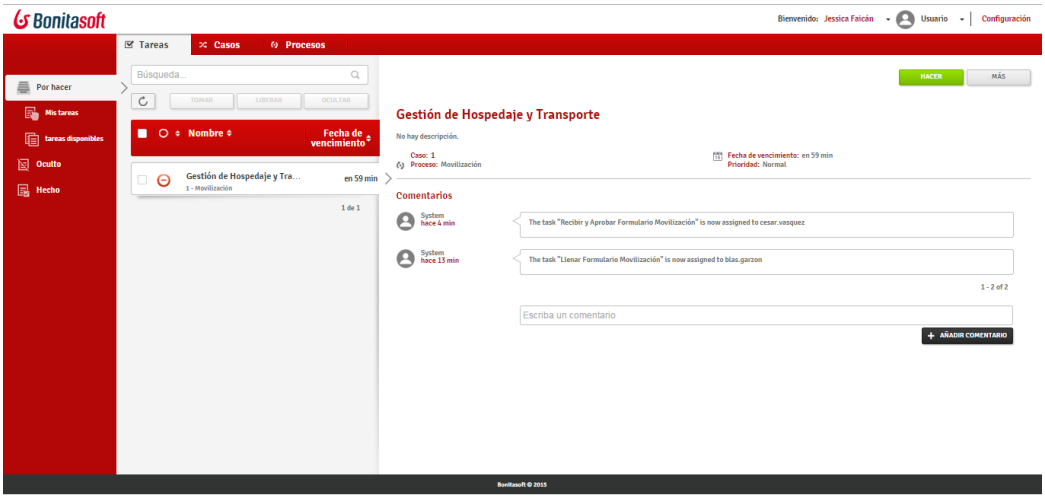


Figura 3.52: Tarea asignada a "jessica.faican"

Damos clic en "HACER" lo cual nos abrirá el formulario donde registraremos los datos de la gestión de "Hospedaje y Transporte" realizada. Cabe indicar que aquí ya tenemos la opción de adjuntar archivos resultantes de la gestión.

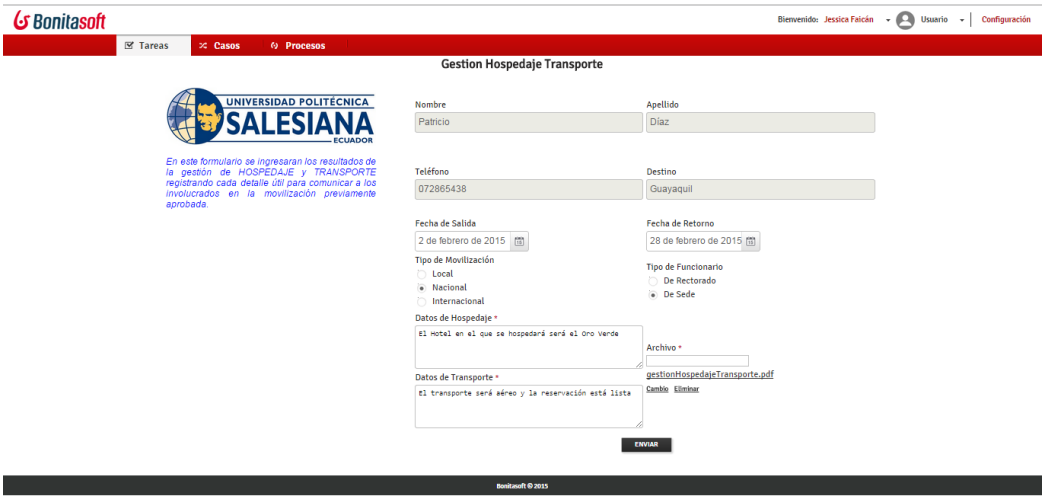


Figura 3.53: Formulario para registrar datos de la Gestión de Hospedaje y Transporte.

La asignación de las tareas de gestión se las hizo utilizando una compuerta paralela ya que se requiere que las dos actividades de Gestión se realicen en paralelo. Por lo tanto, la siguiente tarea inmediata que es la notificación de los recursos a través de un correo electrónico, tendrá que esperar hasta que el usuario "maria.tocachi" realice la tarea de "Gestión de Anticipos" que se le fue asignada.

Verificamos la asignación de la tarea al usuario “maria.tocachi”:

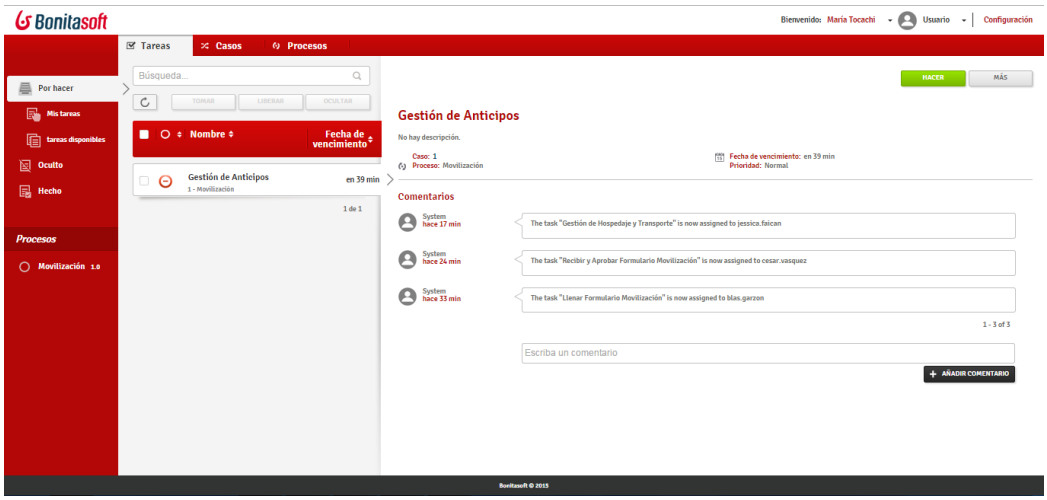


Figura 3.54: Tarea asignada a “maria.tocachi”

Una vez que se ha realizado la gestión de Anticipos por parte de usuario “maria.tocachi” procedemos a registrarla en el siguiente formulario dando clic en “HACER”. También este formulario nos permite adjuntar archivos correspondientes a la gestión:

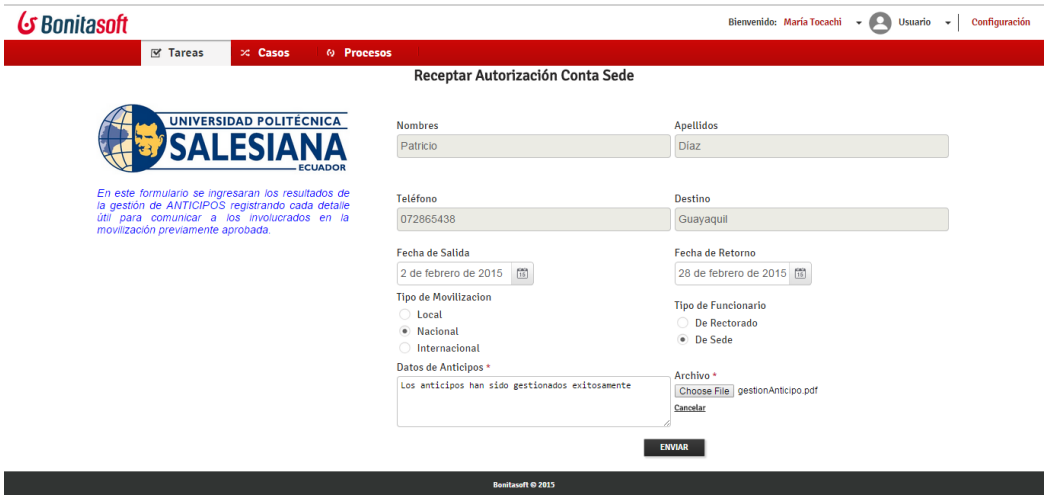


Figura 3.55: Formulario para registrar datos de la Gestión de Anticipos

Ahora que el usuario “maria.tocachi” ha realizado la tarea de gestión de anticipos, completando así el proceso paralelo, se podrá continuar con la siguiente

actividad la cual es la notificación de la asignación de recursos, la cual envía el siguiente mensaje a través de un correo electrónico:

Figura 3.56: Notificación de asignación de recursos con datos adjuntos.

Una vez que el usuario final "mando.cisneros" ha recibido la notificación de asignación de recursos procede a realizar la movilización y una vez terminada tendrá que realizar los respectivos justificativos tal como se registra en el diagrama original del proceso de movilización facilitado por la Universidad Politécnica Salesiana.

Ingresamos con el usuario "mando.cisneros" para verificar la asignación de la tarea de "Preparar Justificativos":

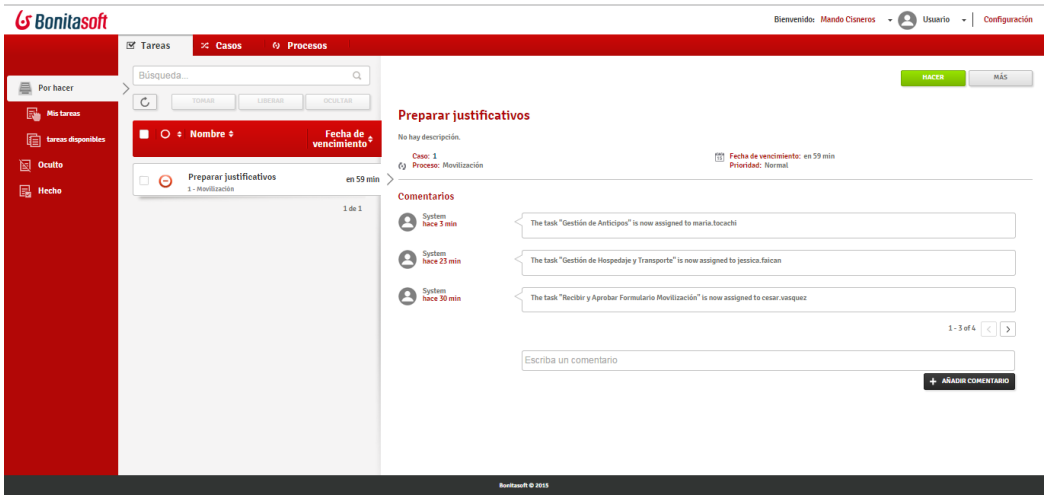


Figura 3.57: Tarea asignada a usuario final “mando.cisneros”

Damos clic en “HACER” para proceder a registrar los justificativos de la movilización y el respectivo informe de Gestión.

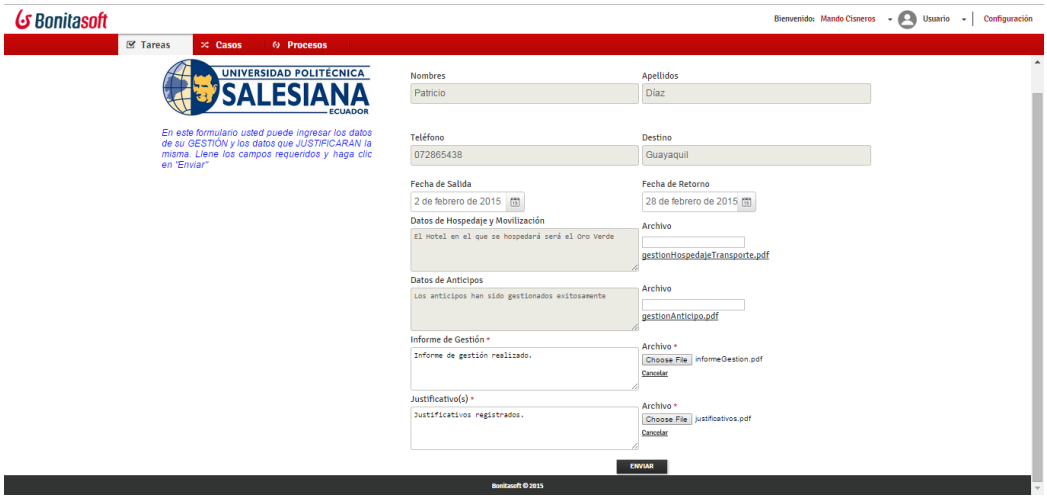


Figura 3.58: Formulario para registrar los justificativos y el informe de gestión.

Estos datos y documentos adjuntos llegaran al Responsable Departamental a través de un correo enviado de igual manera automáticamente.

Al usuario final “mando.cisneros” se le asigna una tarea más la cual es “Registrar la liquidación de Gastos”. Nos vamos a la bandeja de tareas en donde verificamos la asignación de la nueva tarea.

Figura 3.59: Informe de gestión enviado como archivo Adjunto.

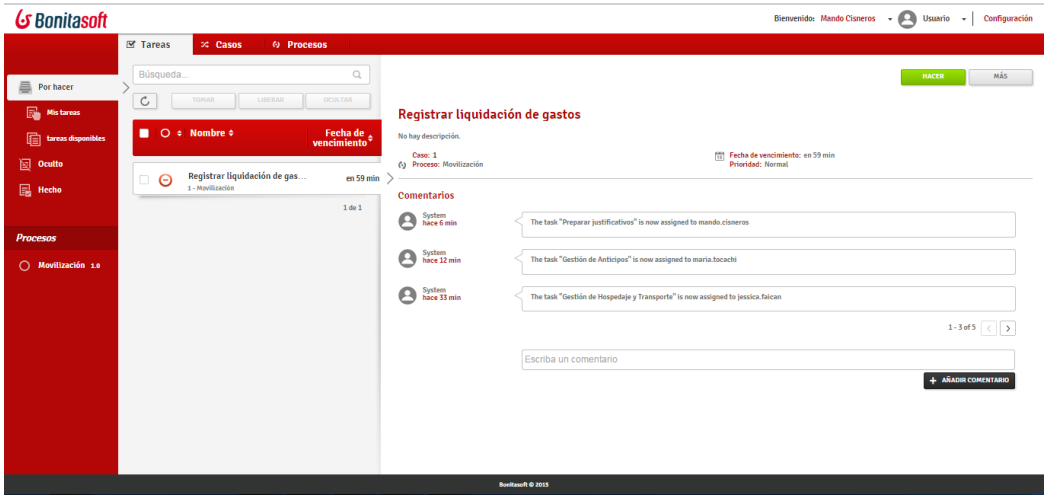


Figura 3.60: Tarea “Registrar liquidación de gastos” asignada a “mando.cisneros”

Damos clic en “HACER” y procedemos a realizar la tarea en donde registraremos los datos de la liquidación de gastos y adjuntaremos el archivo necesario. En esta tarea también se determinará el destino de la liquidación que puede ser a Rectorado o a Sede, para nuestro caso se enviará a Rectorado:

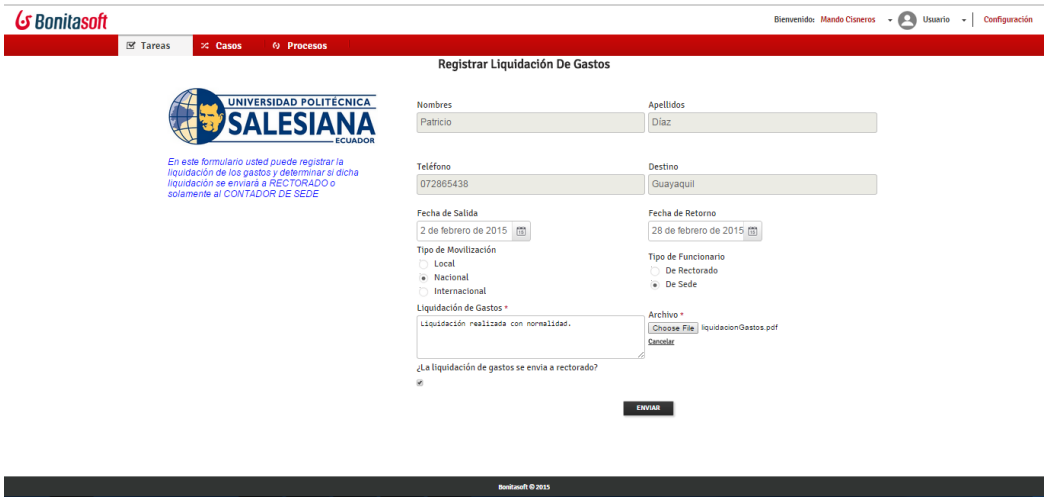


Figura 3.61: Formulario para registrar la Liquidación de Gastos y enviar a quien corresponda

Al elegir que el gasto se enviará a rectorado, el sistema enviará la notificación al Contador General quien podrá verificar los datos y el archivo adjunto.

Figura 3.62: Notificación de liquidación de gastos recibida por el Contador general.

Con esto se da por terminada la Gestión de Movilización. Cabe indicar que lo que hemos realizado es un ejemplo siguiendo uno de los cuatro flujos del proceso ya que nos servirá de ilustración para comprender el comportamiento del proceso completo.

Capítulo 4

Fase de despliegue, implementación y monitorización de procesos con BonitaSoft

4.1. Preparación del entorno para la instalación de BonitaSoft Open Source

4.1.1. Componentes de bonita

Bonita BPM consta de dos componentes importantes que se los podría categorizar en un componente de Desarrollo y otro de Producción, los cuales son Bonita BPM Studio y Bonita BPM Plataform respectivamente.

Bonita BPM Studio: Es la herramienta de diseño con facilidades drag-and-drop y notación BPMN2 para dibujar y configurar los diagramas de nuestros procesos haciendo el trabajo mucho más intuitivo y práctico a la hora de diseñar. Este componente contiene el servidor de servlets Tomcat y para almacenar su información utiliza la base de datos h2 y la aplicación de Bonita la cual incluye el denominado motor de Bonita o Bonita BPM Engine.

Bonita es distribuido en varias versiones, incluida la versión Comunity, utilizada en el desarrollo de este proyecto, la cual no necesita licenciamiento.

Bonita BPM Platform: Es la herramienta de despliegue de Bonita BPM la necesita de un servidor Java, este puede ser un contenedor de servlets como Tomcat o el servidor de aplicaciones como JBoss el cual usaremos en nuestro caso.

Tanto en Bonita Studio como en el componente Plataform se puede configurar en BPM Engine las bases de datos Oracle o PostgresSQL.

4.1.2. Requisitos

4.1.2.1. Instalación de Bonita BPM Studio

Para poder utilizar el componente de diseño Bonita BPM Studio necesitaremos de los siguientes requisitos.

Hardware: Los requisitos de mínimos y recomendados de acuerdo a la página de BonitaSoft, para el eficiente desempeño de la aplicación, se muestra a continuación:

TIPO	MÍNIMO	RECOMENDADO
Procesador	2 CPU cores	4 CPU cores o superior
Memoria (RAM)	4 GB	6 GB o superior
Espacio en disco	10 GB	10 GB o superior depende del uso

Cuadro 4.1: Requisitos mínimos y recomendados de Hardware para Bonita BPM Studio. Fuente: (BonitaSoft, 2015a)

Sistema operativo: La aplicación Bonita BPM Studio es multiplataforma lo que quiere decir que puede ser utilizada en cualquier sistema operativo, ya sea Linux, Windows o Mac en cualquiera de sus publicaciones de 32 o 64 bits, el

único requisito para usar estos sistemas operativos es que tengan soporte para las siguientes versiones de java: Oracle Java SE JRE 7 u OpenJDK 7.

4.1.2.2. Instalación Bonita BMP Platform

A continuación veremos los requisitos recomendados para dejar operativo nuestro componente de producción Bonita BPM Platform.

Hardware: El hardware requerido por Bonita BPM Platform, depende de la cantidad de procesos que se van a administrar y de la complejidad de los mismos, además de los usuarios involucrados e instancias a manejar. La tabla a continuación muestra los requerimientos básicos:

TIPO	MÍNIMO	RECOMENDADO
Procesador	4 CPU cores	4 CPU cores o superior
Memoria (RAM)	4 GB	6 GB o superior
Espacio en disco	10 GB	Depende del uso

Cuadro 4.2: Requisitos mínimos y recomendados de Hardware para Bonita BPM Platform. Fuente: (BonitaSoft, 2015c)

Stack Recomendado: Componentes Open Source: Estas son las recomendaciones de BonitaSoft con respecto a los componentes de Software libre:

- "Ubuntu 14.04 LTS 64 bits
- OpenJDK 7
- Tomcat 7
- PostgreSQL 9.3
- Firefox"

(BonitaSoft, 2015c)

Stack Recomendado: Componentes Propietarios:

- "Red Hat Enterprise Linux 6 or Windows Server 2012 R2
- Oracle Java SE JRE 7
- Oracle WebLogic Server 12c
- Oracle database 11gR2
- Chrome"

(BonitaSoft, 2015c)

Operating Systems: De la misma manera que Bonita BPM Studio, así mismo Bonita BPM Platform es multiplataforma de tal manera que se puede utilizar cualquier sistema operativo, ya sea Linux, Windows o Mac en cualquiera de sus publicaciones de 32 o 64 bits siempre y cuando que soporte JVMs.

Java Virtual Machines: Las versiones de Java que el sistema operativo tiene que soportar son las siguientes:

- Oracle Java SE JRE 7
- OpenJDK 7

Cabe indicar que la versión 8 de Java aún no es soportada.

Servidor de Aplicaciones Java: Se puede utilizar cualquier servidor de aplicaciones que sea compatible con las especificaciones de Java Enterprise Edition (JEE 6), especialmente tienen que existir compatibilidad con los siguientes componentes: servlets de Java, JSP, y los servicios de JTA. Como por ejemplo:

- Tomcat 7 (con adición de Bitronix para JTA)

- JBoss 7.1.1
- Oracle WebLogic Server 12c

Relational Data Base management System (RDBMS)o Bases de datos: Las bases de datos recomendadas para desplegar el esquema de Bonita BPM son:

- PostgreSQL 9.3
- Oracle Database 11gR2
- Microsoft SQL Server 2012 R2
- MySQL 5.5

Una de las recomendaciones que encontramos en la página de BonitaSoft es que configuremos nuestra base de datos con el set de caracteres UTF-8.

Web Browsers: Los navegadores que se recomiendan utilizar son:

- Internet Explorer 9 o superior
- Mozilla Firefox
- Google Chrome

4.2. Instalación y configuración de BonitaSoft Open Source en Servidor

4.2.1. JBoss Bundle

JBoss es un servidor de aplicaciones Java Enterprise Edition (JEE), el mismo viene empaquetado con la aplicación Bonita BPM. Bonita requiere de un directorio de instalación denominado "Bonita Home", el cual viene empaquetado con el Bundle, en este directorio se almacenan los archivos de configuración y

archivos temporales de Bonita. Adicionalmente el Servidor de aplicaciones necesita un directorio "JBOSS_HOME", el cual se encontrará ubicado en la siguiente ruta:

Windows: C:\BonitaBPM6

Linux: /opt/BonitaBPM6

Para nuestro caso utilizaremos un sistema operativo Linux por lo que debemos asegurarnos que el usuario de Linux que va a ejecutar JBoss, sea el propietario de los archivos y carpetas que vamos a configurar. Finalmente el directorio final de bonita será:

<JBOSS_HOME>/bonita

4.2.2. Instalación del sistema operativo CentOS 7.0

CentOS es uno de los sistemas operativos Linux más utilizados para hacer el trabajo empresariales de servidor, por ser un sistema fácil de instalar y utilizar a la vez que es robusto y muy estable. A continuación realizaremos la instalación de este Sistema Operativo:

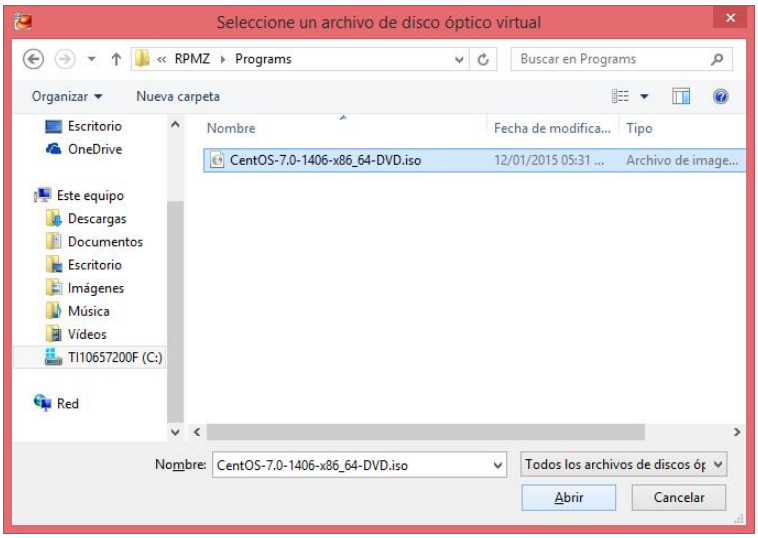


Figura 4.1: imagen ISO para instalación de CentoOS 7.0

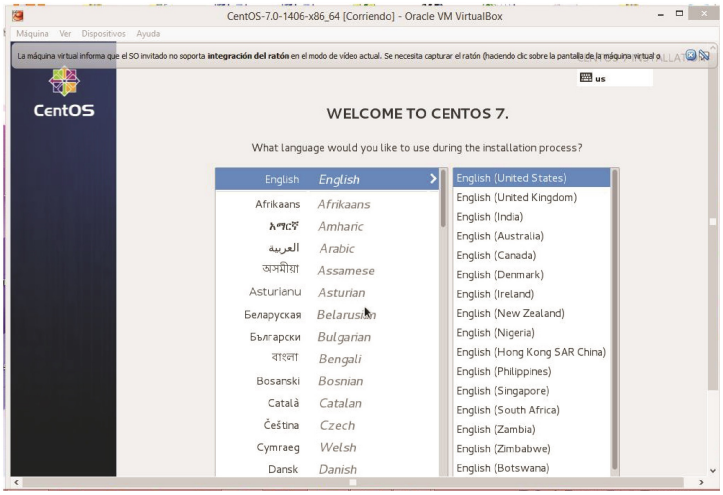


Figura 4.2: Ventana para seleccionar el idioma para la instalación de CentOS.

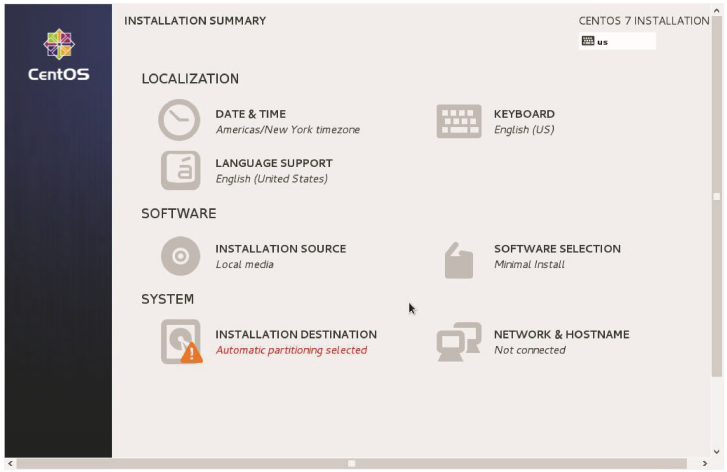


Figura 4.3: Configuraciones de la instalación de CentOS

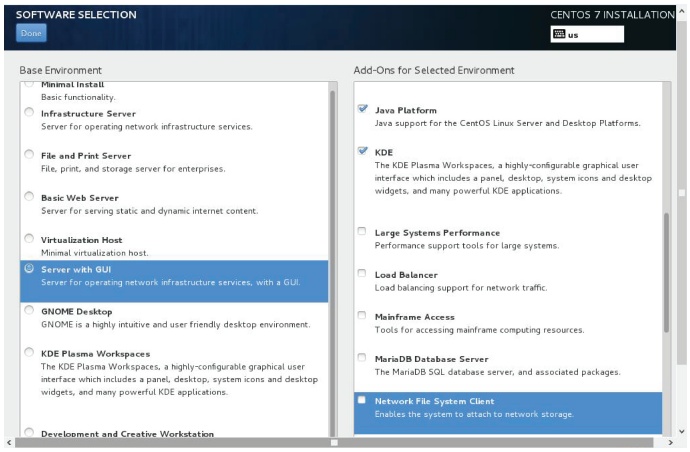


Figura 4.4: Selección de tipo de Software Servidor

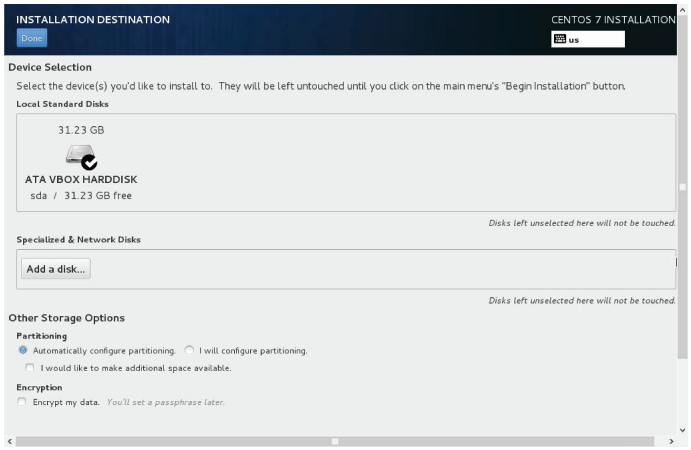


Figura 4.5: Selección del destino de la instalación

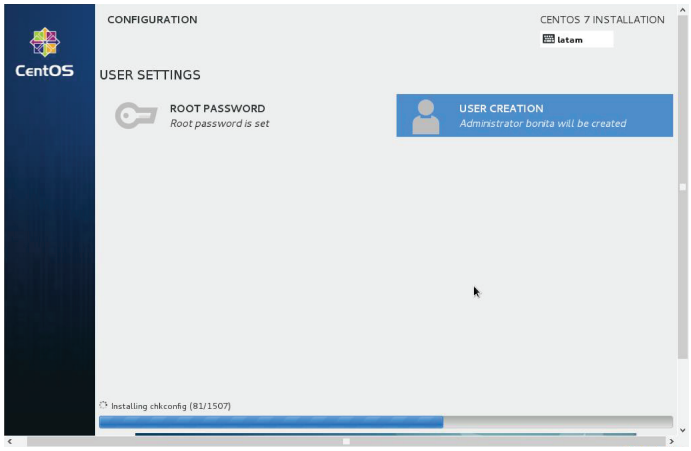


Figura 4.6: Usuario bonita creado con perfil de administrador.

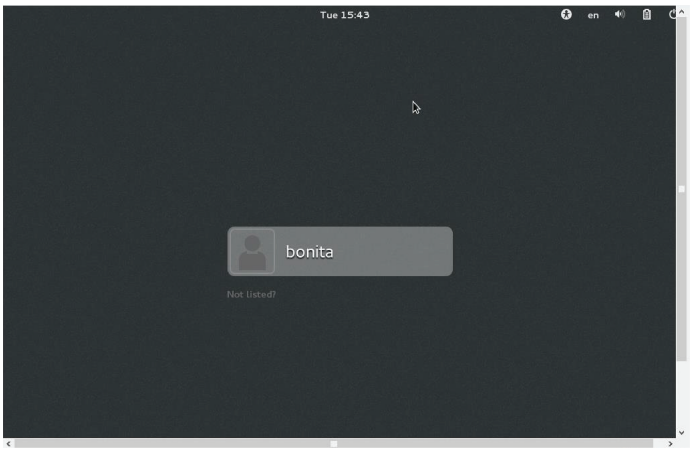


Figura 4.7: Instalación de CentOS lista.

Una vez que tenemos instalado nuestro sistema operativo CentOS, estamos listos para realizar las instalaciones y configuraciones de Bonita BPM.

4.2.3. Conexión remota a servidor Linux

Para poder realizar la transferencia del servidor de aplicaciones y de Bonita BPM de forma segura hacia nuestro sistema servidor CentOS vamos a utilizar la aplicación WinSCP la cual también es una herramienta de Software libre, la cual emplea los protocolos SCP y SSH.

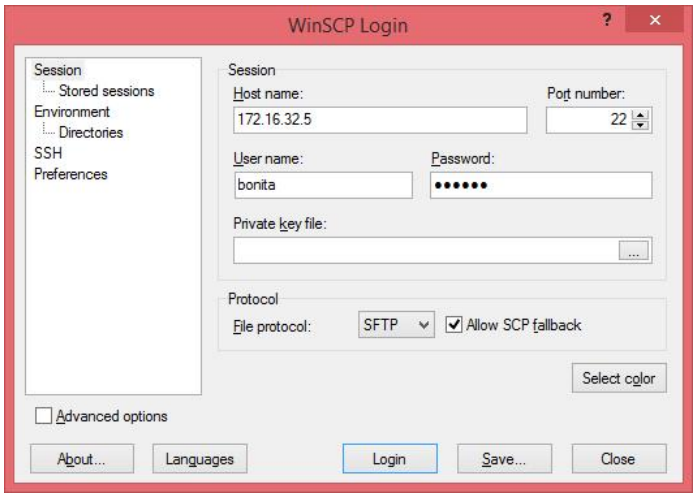


Figura 4.8: Programa WinSCP para la transmisión remota de archivos al servidor.

Una vez que hemos establecido la conexión remota veremos que al lado izquierdo se encuentra el listado de directorio del equipo local, al lado derecho se observa el listado de directorio del sistema operativo del equipo servidor CentOS.

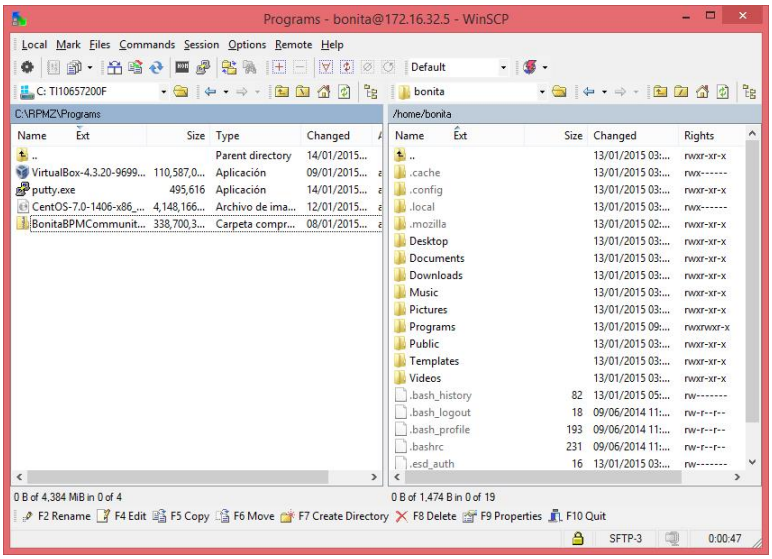


Figura 4.9: Ventana para gestión de transferencia de archivos en WinSCP

Realizamos la copia del servidor de aplicaciones y bonita BPM Portal

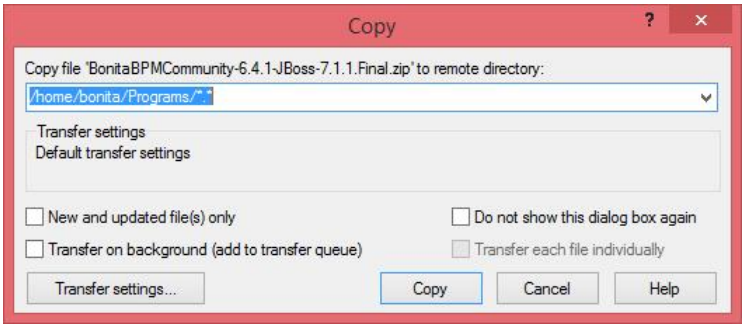


Figura 4.10: Transferencia de Archivos

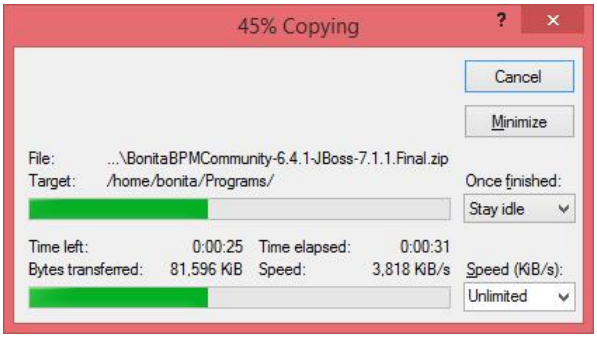


Figura 4.11: Copiando archivos de JBoss y Bonita.

Para administrar el servidor CentOS 7.0, utilizaremos el protocolo Secure Shell (SSH), para el acceso remoto encriptado al servidor, mediante el uso de la herramienta Putty, que también es una herramienta de software libre y código abierto que implementa el protocolo SSH.

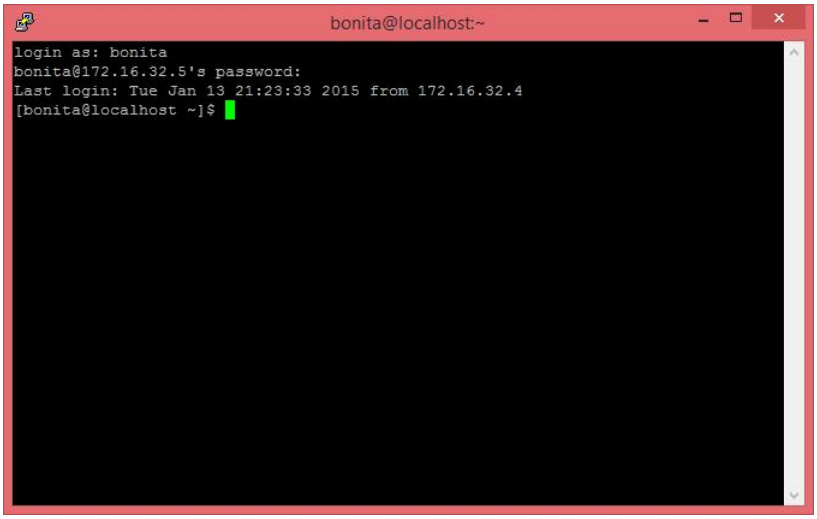


Figura 4.12: Autenticación en Putty para la administración del Servidor.

Mediante el uso de la herramienta verificamos que el archivo que contiene el servidor de aplicaciones JBoss y bonita BMP portal se encuentre en el servidor.

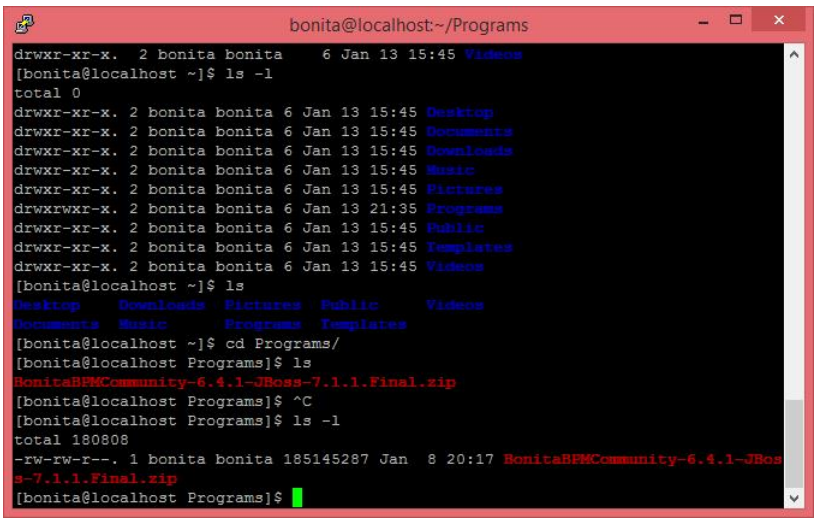


Figura 4.13: Verificación del la correcta transferencia de los archivos del servidor y Bonita BPM.

4.2.4. Instalación de Bonita BPM

Una vez que nos hemos asegurado que nuestros archivos se encuentran en la ruta deseada procedemos a descomprimir Bonita BPM utilizando la siguiente línea de comando:

```
1 [bonita@localhost Programs]$ unzip BonitaBPMCommunity-6.4.1-
    JBoss-7.1.1.Final.zip
```

Cambiamos el nombre, por BonitaBPM6, el cual va a ser el Directorio Home <JBOSS_HOME>, recomendado para instalación de bonita.

```
1 [bonita@localhost Programs]$ mv BonitaBPMCommunity-6.4.1
    -JBoss-7.1.1.Final BonitaBPM6
```

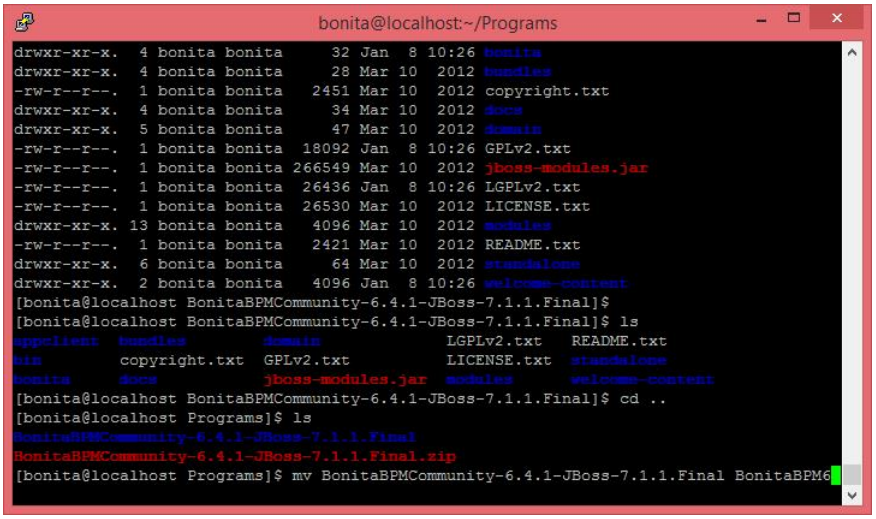


Figura 4.14: Cambiando de nombre desde Putty

Luego copiamos al directorio recomendado del servidor en donde se debe encontrar <JBOSS_HOME>, para lo cual debemos usar el usuario root de Linux.

```
[bonita@localhost Programs]$ su
Password:
[root@localhost Programs]#
```

Procedemos a mover el directorio BonitaBPM6 al directorio sugerido:

```
[root@localhost Programs]# mv BonitaBPM6 /opt/
```

Verificamos que el usuario del sistema operativo que va a ejecutar JBoss sea el propietario de los archivos y carpetas del directorio Home BonitaBPM6.

```
[root@localhost bonita]# ls -l /opt/
```

Obtenemos como resultado:

```
drwxr-xr-x. 11 bonita bonita 4096 Jan  8 10:26 BonitaBPM6
drwxr-xr-x.  2 root   root      6 Jun  9  2014 rh
```

En el resultado de la consulta observamos que el usuario “bonita” es el propietario del directorio Home BonitaBPM6 y dispone de permisos de lectura (r), escritura (w), y ejecución (x) y de todos sus subdirectorio y archivos. Adicionalmente podemos observar que dentro del directorio BonitaBPM6, se encuentra

el directorio “bonita”, que a su vez corresponde al directorio home de Bonita BPM, en el cual se almacena; la configuración, trabajos y archivos temporales.

```
[root@localhost bonita]# ls -l /opt/
total 4
drwxr-xr-x. 11 bonita bonita 4096 Jan  8 10:26 BonitaBPM6
drwxr-xr-x.  2 root   root      6 Jun  9  2014 rh
[root@localhost bonita]# ls -l /opt/BonitaBPM6/
total 360
drwxr-xr-x.  3 bonita bonita     26 Mar 10  2012 appclient
drwxr-xr-x.  4 bonita bonita   4096 Jan  8 10:26 bin
drwxr-xr-x.  4 bonita bonita     32 Jan  8 10:26 bonita
drwxr-xr-x.  4 bonita bonita     28 Mar 10  2012 bundles
-rw-r--r--.  1 bonita bonita   2451 Mar 10  2012 copyright.txt
drwxr-xr-x.  4 bonita bonita     34 Mar 10  2012 docs
drwxr-xr-x.  5 bonita bonita     47 Mar 10  2012 domain
-rw-r--r--.  1 bonita bonita  18092 Jan  8 10:26 GPLv2.txt
-rw-r--r--.  1 bonita bonita 266549 Mar 10  2012 jboss-modules.
  jar
-rw-r--r--.  1 bonita bonita  26436 Jan  8 10:26 LGPLv2.txt
-rw-r--r--.  1 bonita bonita  26530 Mar 10  2012 LICENSE.txt
drwxr-xr-x. 13 bonita bonita   4096 Mar 10  2012 modules
-rw-r--r--.  1 bonita bonita   2421 Mar 10  2012 README.txt
drwxr-xr-x.  6 bonita bonita     64 Mar 10  2012 standalone
drwxr-xr-x.  2 bonita bonita   4096 Jan  8 10:26 welcome-
  content
[root@localhost bonita]#
```

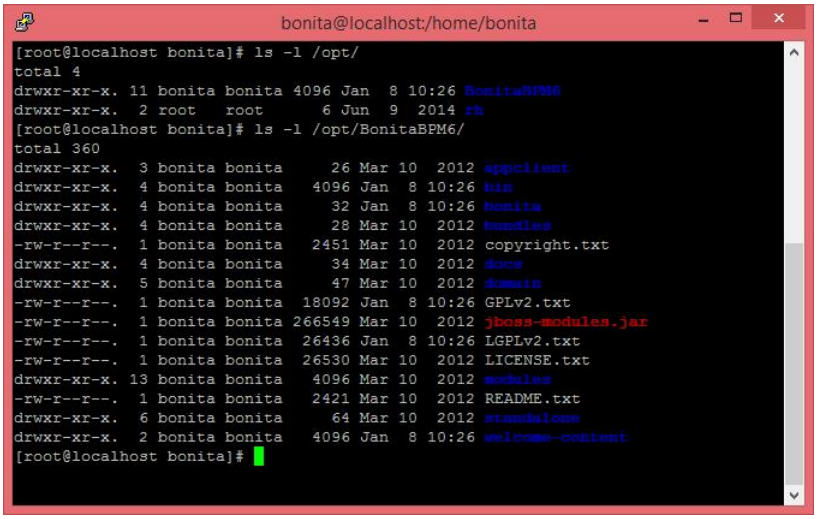


Figura 4.15: Verificación del directorio Bonita BPM6

4.3. Conexión con el Servidor de la UPS

4.3.1. Configuración de la base de datos

"El paquete de JBoss, viene configurado con la base de datos h2" (BonitaSoft, 2015b), la cual vamos a cambiar por la base de datos a utilizar en el entorno de producción que será Oracle en su versión 11gR2. Bonita BPM Engine, necesita de un Sistema Relacional de Base de Datos (RDBMS, por sus siglas en ingles), en el cual almacena información de los procesos, configuración, ejecución, etc.

"Bonita BPM Engine utiliza la herramienta Hibernate, la cual es una solución de mapeo Objeto/Relacional para ambientes Java. El mapeo Objeto/-Relacional hace referencia a una técnica para mapear datos de una representación de modelo de objetos a un modelo relacional de datos y viceversa". (JBoss, 2015). De esta manera bonita interactúa con el RDBMS.

Creando la nueva estructura de base de datos para Oracle 11gR2: La base de datos que se utiliza en la Universidad Politécnica Salesiana es Oracle

11gR2 de alta disponibilidad, por lo que utilizaremos la misma base de datos para realizar el despliegue de nuestra aplicación. Primeramente vamos a crear la estructura de base de datos necesaria para que Bonita BPM pueda desplegar su esquema.

Abrimos la aplicación SQL Command Line y nos autenticamos con el usuario "sysdba":

```
SQL>connect as sysdba
Enter user-name: sys
Enter password: oracle
Connected.
SQL>
```

Creación de la base de datos: Creamos la base de datos en donde vamos a almacenar el esquema relacional de bonita, el cual debe los siguientes privilegios:

- CREATE TABLE
- CREATE INDEX
- SELECT, INSERT, UPDATE, DELETE on created TABLE

Script de creación del usuario "bonita": Con el siguiente Script SQL creamos el usuario "bonita" al que le asignamos diferentes roles y privilegios:

```
-- USER SQL
CREATE USER bonita IDENTIFIED BY bonita;
-- ROLES GRANT "RESOURCE" TO bonita;
GRANT "CONNECT" TO bonita;
-- SYSTEM PRIVILEGES
GRANT CREATE ANY INDEX TO bonita;
GRANT CREATE ANY TABLE TO bonita;
```

4.3.2. Especificación de la Base de Datos

Bonita BPM Engine necesita conocer el tipo de base de datos que va a utilizar, para lo cual tenemos que configurar el parámetro de la máquina virtual sysprop.bonita.db.vendor, para el caso de JBoss, se encuentra en la siguiente path:

<JBOSS_HOME>/standalone/configuration/standalone.xml

Adicionar el driver jdbc al servidor de aplicaciones: El servidor de aplicaciones JBoss AS7, maneja los controladores a manera de módulos por lo que es necesario crear el directorio en donde vamos a ubicar el controlador jdbc para la conexión con Oracle. Para lo cual debemos asegurarnos de contar con la siguiente ruta, en la que colocaremos el controlador:

"modules/com/oracle/main"

En el mismo directorio debemos crear un archivo xml, el cual es el descriptor del controlador, y contendrá lo siguiente:

```
<module xmlns="urn:jboss:module:1.0" name="com.oracle">
<resources>
<resource-root path="ojdbc6.jar"/>
</resources>
<dependencies>
<module name="javax.api"/>
<module name="javax.transaction.api"/>
</dependencies>
</module>
```

Configuración de la fuente de datos: En la configuración de la fuente de datos especificamos el sistema relacional de base de datos (RDBMS) al cual se va a conectar Bonita BPM.

Se debe realizar la configuración de ciertos parámetros, esto se hará en la sección de datasources del archivo:

<JBOSS_HOME>/standalone/configuration/standalone.xml

A Continuación se muestran los parámetros necesarios de configurar:

- La dirección del servidor RDBMS
- El puerto del servidor RDBMS
- El nombre del esquema de la base de datos
- El usuario y la contraseña para la conexión con la Base de Datos

Eliminando la base de pruebas h2: Tenemos que eliminar la base de datos h2 que trae por defecto Bonita BPM, para esto es necesario eliminar el siguiente directorio:

$JBOSS_HOME/standalone/deployments/h2.sar

Configuraciones específicas de la base de datos: Para asegurar el correcto funcionamiento de la base de datos tenemos que asegurarnos que cumpla con las siguientes configuraciones:

- La base de datos debe utilizar el juego de caracteres AL32UTF8.
- Validar si la base de datos cumplen con las condiciones especificadas en la siguiente tabla:

Comp_name	Version	Status
Oracle Database Catalog Views	11.2.0.1.0	VALID
Oracle Database Packages and Types	11.2.0.1.0	VALID
JServer JAVA Virtual Machine	11.2.0.1.0	VALID
Oracle XDK	11.2.0.1.0	VALID
Oracle Database Java Packages	11.2.0.1.0	VALID
OLAP Analytic Workspace	11.2.0.1.0	VALID
Oracle OLAP API	11.2.0.1.0	VALID

Cuadro 4.3: Especificaciones requeridas para nuestra base de datos.

Para realizar la validación de la base de datos utilizaremos la siguiente consulta:

```
1 select comp_name, version, status from dba_registry;
```

Esta consulta tiene como resultado lo siguiente:

Comp_name	Version	Status	Requerimiento
OWB	11.2.0.1.0	VALID	
Oracle Application Express	3.2.1.00.10	VALID	
Oracle Enterprise Manager	11.2.0.1.0	VALID	
OLAP Catalog	11.2.0.1.0	VALID	
Spatial	11.2.0.1.0	VALID	
Oracle Multimedia	11.2.0.1.0	VALID	
Oracle XML Database	11.2.0.1.0	VALID	
Oracle Text	11.2.0.1.0	VALID	
Oracle Expression Filter	11.2.0.1.0	VALID	
Oracle Rules Manager	11.2.0.1.0	VALID	
Oracle Workspace Manager	11.2.0.1.0	VALID	
Oracle Database Catalog Views	11.2.0.1.0	VALID	OK
Oracle Database Packages and Types	11.2.0.1.0	VALID	OK
JServer JAVA Virtual Machine	11.2.0.1.0	VALID	OK
Oracle XDK	11.2.0.1.0	VALID	OK
Oracle Database Java Packages	11.2.0.1.0	VALID	OK
OLAP Analytic Workspace	11.2.0.1.0	VALID	OK
Oracle OLAP API	11.2.0.1.0	VALID	OK

Cuadro 4.4: Resultado de la consulta, para verificar las especificaciones de la base de datos.

En la tabla hemos colocado una columna adicional para marcar con "OK" los requisitos cumplidos por la base de datos con respecto a la tabla de especificaciones requeridas anteriormente mencionada, así podemos apreciar que nuestra base de datos si cumple con las condiciones establecidas.

- Si en el servidor de la base de datos a la que vamos a conectarnos no tiene activado el servicio de transacciones distribuidas XA(eXtended Architecture), tenemos que hacerlo ejecutando un script que lo encontraremos en la siguiente dirección:

  /product/11.2.0/dbhome_1/javavm/install/initxa.sql

- Agregar los siguiente permisos al esquema que va a ser utilizado por bonita

```
GRANT select ON sys.dba_pending_transactions TO bonita;
GRANT select ON sys.pending_trans$ TO bonita;
GRANT select ON sys.dba_2pc_pending TO bonita;
GRANT execute ON sys.dbms_system TO bonita;
GRANT select ON sys.v$xatrans$ TO bonita;
```

4.4. Despliegue de la aplicación

4.4.1. Arrancar servidor de aplicaciones

Start and shut down JBoss

JBoss start script

Para levantar o inicializar nuestro servidor de aplicaciones JBoss ejecutaremos el siguiente comando:

```
sh <JBOSS_HOME>/bin/standalone.sh
```

En donde <JBOSS_HOME> para nuestro caso sería: /opt/BonitaBPM6/

4.4.2. Configuración inicial de la Aplicación Bonita Platform

Verificando la instalación: Para realizar la verificación de que nuestra instalación ha sido exitosa y nuestra base de datos ha sido configurada correctamente tenemos que realizar una conexión con Bonita BPM Portal.

En la barra de direcciones de nuestro navegador de preferencia escribimos la siguiente dirección local:

http://localhost:8080/bonita

El resultado de esta dirección debe ser la página de autentificación para ingresar usuario y contraseña, de no ser el caso podemos intentarlo de nuevo ac-

tualizando la página después de haber borrado la cache de nuestro navegador.

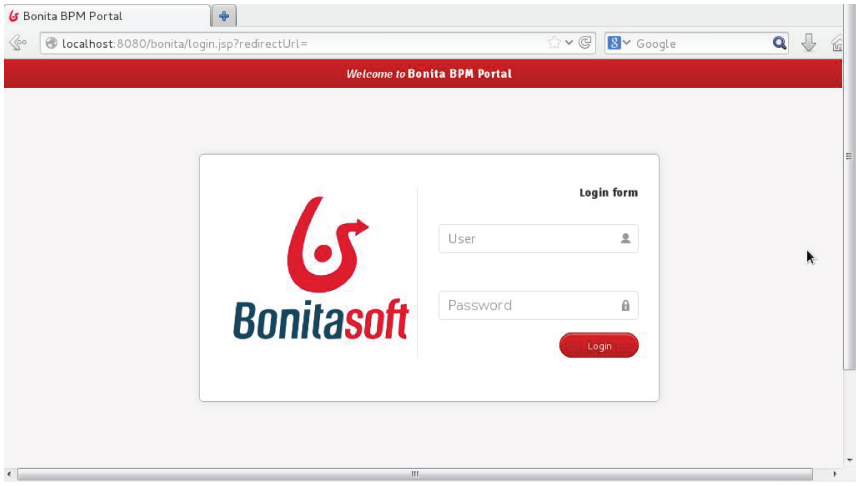


Figura 4.16: Primera ventana de autenticación de Bonita desde el Servidor

Al momento no existe ninguna información de organización por lo que solamente existe un usuario técnico.

Tenant Administrator: Un administrador tenant es conocido también como un usuario técnico. Al momento de la creación de la plataforma los valores por defecto de un usuario técnico administrador son creados en Bonita Home_code_replace/server/platform/tenant-template/conf/bonita-server.properties, con las siguientes credenciales:

Usuario: install

Contraseña: install

Cuando creamos un usuario administrador este adquiere credenciales por defecto que si deseamos podríamos cambiar en Bonita: Home_code_replace/server/tenants/tenant_id/ server.properties. (BonitaSoft, 2014b).

4.4.3. Creación del usuario administrador de la plataforma bonita BPM

A continuación vamos a enumerar los pasos para crear un usuario que tendrá un perfil de administrador, este usuario será creado por el usuario técnico y con sus propias credenciales:

1. "Primeramente tenemos que entrar al portal de Bonita BPM con nuestro usuario técnico.
2. Luego creamos un usuario con perfil estándar.
3. Una vez creado el usuario nos vamos al menú “Configuración” y elegimos “Privilegios”.
4. En la lista de perfiles seleccionamos “Administrador” y damos clic en el botón “Más”, ubicado en la parte superior derecha de la pantalla.
5. Veremos una pantalla en la que estará una tabla con todos los usuarios con perfil de administrador, debajo de esta tabla hacer clic en el botón “Agregar un usuario”.
6. Nos mostrará una ventana en donde elegiremos nuestro usuario creado en el paso 2 y damos clic en “Añadir”.
7. Este usuario podrá crear los usuarios necesarios con perfil estándar." (BonitaSoft, 2014a).

4.4.4. Creando el archivo .bar del proceso de “Movilización” para el despliegue en el Portal de Bonita BPM

Una vez que nuestro proceso está completo y listo para ser instalado en el Portal de Bonita BPM para ser utilizado, necesitamos generar el archivo que instalaremos el cual tendrá una extensión .bar, este archivo lo creamos desde el

mismo Bonita BPM Studio en donde diseñamos y configuramos nuestro proceso. Para ello haremos lo siguiente: .

1. "Nos vamos al menú “Servidor” y elegimos la opción “Compilar”.

2. En la ventana que nos aparece seleccionamos el proceso que deseamos compilar y la ruta en donde queremos que se cree nuestro archivo .bar. Se exportará toda la información que tenga el proceso incluido los conectores y dependencias.

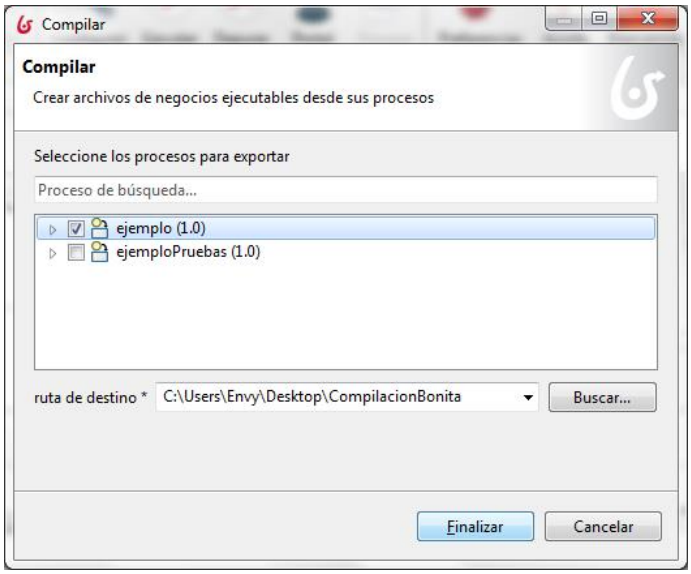


Figura 4.17: Ventana para construir un archivo desplegable .bar

3. El archivo se creará con el mismo nombre del proceso, en esta etapa no es recomendable cambiar el nombre del mismo.

4. Finalizar"(BonitaSoft, 2014a)

4.4.5. Exportar una organización

Además del archivo .bar, en nuestro Bonita BPM Portal, también necesitaremos de la organización que va a utilizar el proceso que hemos creado para ello tendremos que exportar nuestra organización igualmente desde Bonita BPM Studio, de la siguiente forma:

1. "Nos dirigimos al menú “Organización” y elegimos la opción “Exportar”.

2. Luego elegimos la organización y la ruta en donde queremos guardar el archivo resultante de la exportación el cual tiene una extensión .xml.

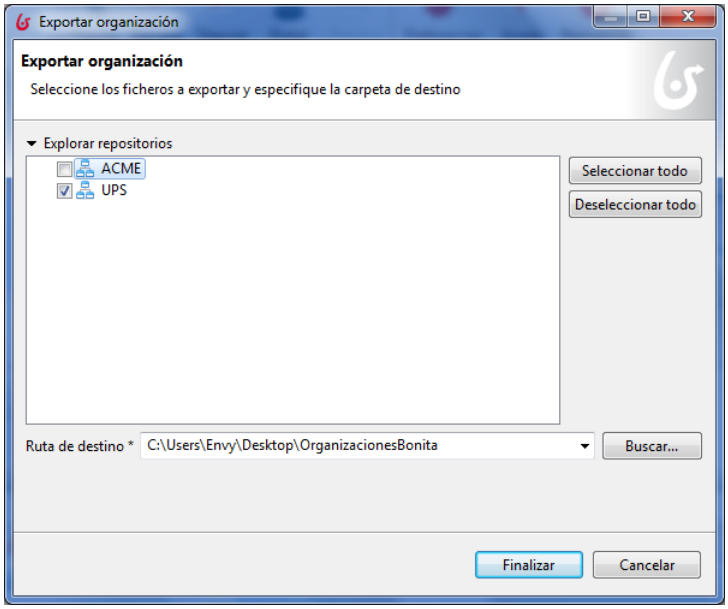


Figura 4.18: Ventana para exportar la organización UPS desde Bonita BPM Studio.

3. Finalizar." (BonitaSoft, 2014a)

4.4.6. Importando una organización

Una vez que tenemos listos nuestros archivos .bar (proceso) y .xml (organización) el siguiente paso será importar la organización para lo cual realizamos lo siguiente:

1. "Nos dirigimos al menú “Organización” y seleccionamos la opción “Importar / exportar”.

2. Luego seleccionaremos nuestro archivo .xml desde siguiente pantalla, en la sección “Importar una organización existente”:

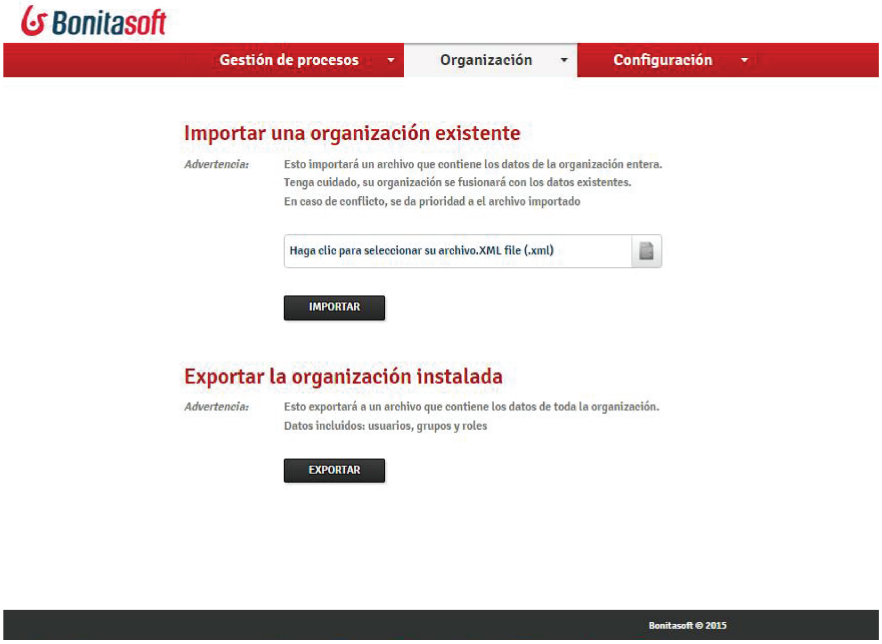


Figura 4.19: Ventana para importar una organización

3. Luego damos clic en Importar." (BonitaSoft, 2015d).

4.4.7. Instalar un proceso en Bonita BPM Portal

Para realizar la instalación de un proceso en el Bonita BPM Portal, realizamos lo siguiente:

1. "Nos vamos al menú “Gestión de procesos” y elegimos la opción “Procesos”

2. Elegimos nuestro archivo .bar que es nuestro proceso a instalar.

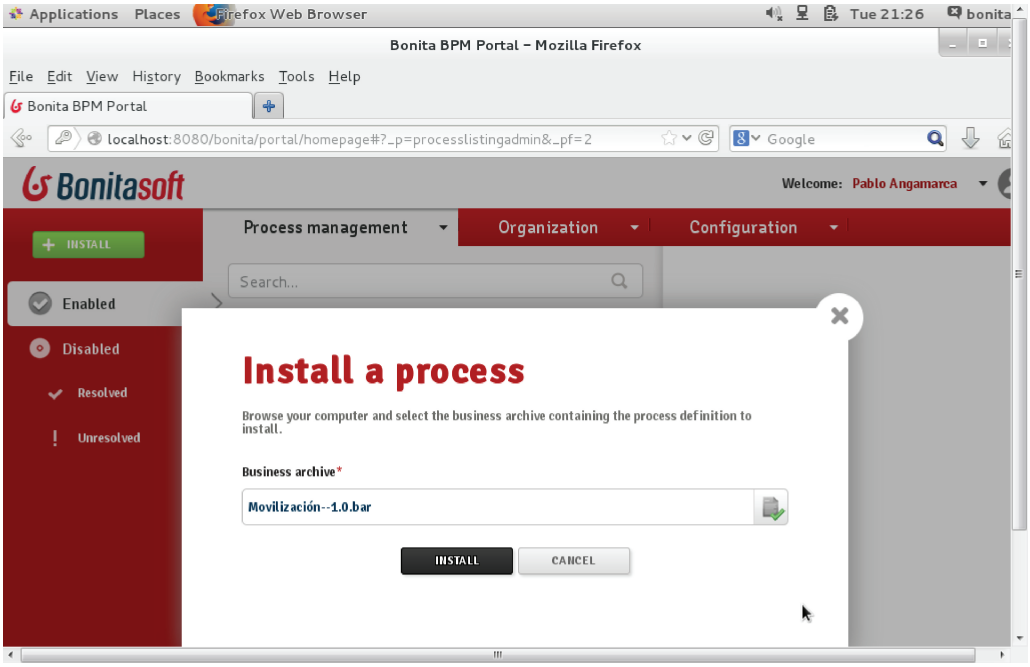


Figura 4.20: Ventana para elegir e instalar un proceso

3. Damos clic en el botón “Instalar”". (BonitaSoft, 2014c).

Cada proceso tiene definido sus actores, los mismos que ejecutarán y realizarán las tareas que se les sea asignada. De igual manera una organización tiene sus grupos, roles, membresías y usuarios definidos. Una vez que tengamos nuestra organización importada y proceso instalado podremos realizar un “mapeo” de los usuarios y actores de nuestro proceso.

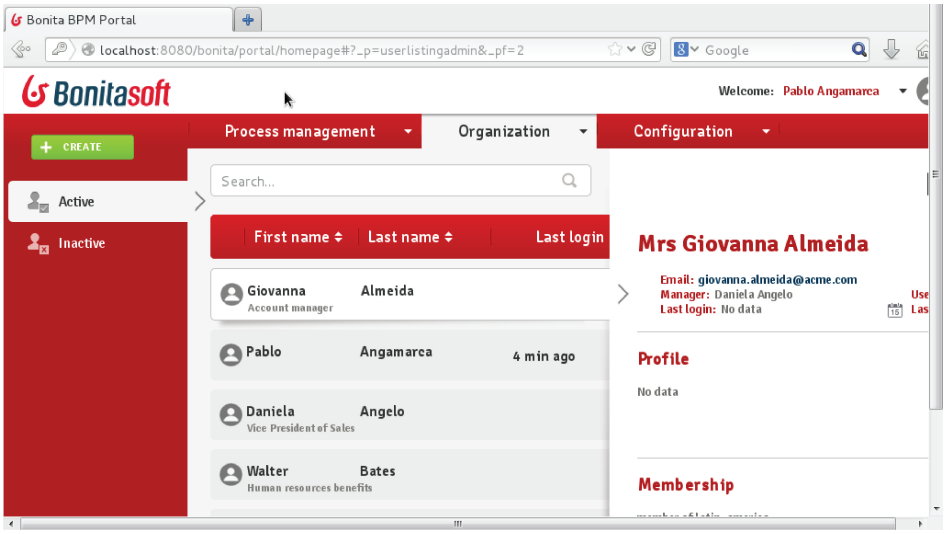


Figura 4.21: Organización importada.

Ahora solamente tenemos que habilitar nuestro proceso, dando clic en el botón color verde ubicado en la esquina superior derecha de la pantalla y así poder dar por finalizada la configuración de nuestro proceso y organización en Bonita BPM Portal.

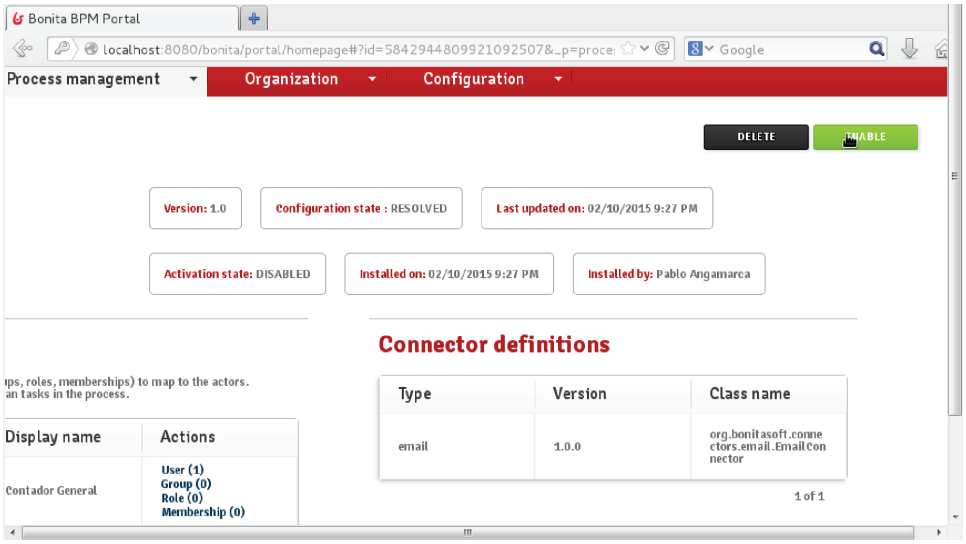


Figura 4.22: Venana para habilitar nuestro proceso.

4.5. Capacitación del personal

En conjunto con el Ing. Diego Quinde - Secretario Técnico de Tecnologías de la Información de la UPS se ejecutaron todos los pasos descritos en este documento, se realizó la implementación en un servidor propio de la institución.

Como Anexo a esta monografía se adjunta una carta que certifica lo dicho.

Capítulo 5

Conclusiones y recomendaciones

5.1. Conclusiones

La escasa e ineficiente gestión que hoy en día se da al manejo de los procesos de negocios en muchas instituciones, está provocando una gran pérdida de recursos y retrasos innecesarios en los mismos y en la gestión en general de las instituciones. La implementación de una solución de BPM puede mejorar considerablemente el control y eficiencia de los procesos, a la vez que colabora en la reducción de pérdida de recursos y contratiempos al momento de ejecutar los procesos de negocio en las instituciones.

Bonita es una herramienta que ha venido madurando hasta convertirse en una de las más importantes soluciones de BPM en la actualidad. Bonita BPM, al ser una solución "Open Source" y de gran potencia gracias a sus conectores que brindan la facilidad de integración a cualquier sistema externo, se convierte en la mejor opción para la implementación de un sistema BPM en las instituciones, Bonita BPM tiene una curva de aprendizaje relativamente corta, y ayuda en la reducción de costos al no requerir de licencias al ser libre solamente requiere de invertir tiempo para poder estudiar la herramienta.

Como se ha demostrado en el presente trabajo, la implementación de la automatización del proceso de "Movilización", el cual es uno de los muchos procesos que se manejan en la Universidad Politécnica salesiana, es totalmente factible y viable. Considerando que muchos de los procesos están levantados y plasmados en diagramas, bien podrían facilitar la automatización de estos. Los costos de la implementación pueden reducirse al utilizar software libre en toda la estructura de automatización, que está totalmente disponible para estos casos.

La automatización de procesos causará una gran satisfacción en la institución como en los usuarios finales y personas externas, las cuales necesitan realizar un determinado proceso dentro de la institución.

Con el presente trabajo hemos constatado lo importante y factible que es para una empresa la eficiente gestión de los procesos de negocio, es decir, la automatización de los mismos. Para así adquirir un excelente prestigio y beneficios al incrementar su eficiencia en los procesos que ejecuta.

Hoy, en un mundo invadido de tecnología se concibe vital la automatización de los procesos de negocio en las instituciones de todo tipo y mucho más en las de educación superior que siempre suelen ir a la par con la tecnología.

5.2. Recomendaciones

La Universidad Politécnica Salesiana ha realizado el levantamiento de sus más de 600 procesos lo cual ha significado una gran inversión de tiempo y recursos, de tal manera que se considera oportuna la recomendación de continuar con tan importante proceso de automatización de procesos en la universidad. La implementación de una herramienta BPM como Bonita Open Solution sería un buen camino a seguir para lograr esta automatización.

Luego de la automatización de procesos la localización de cuellos de botellas, pasos o eventos innecesarios, se podrá realizar sin mayor esfuerzo, de la misma manera la búsqueda de mejoras resultará mucho más fácil de realizar. La universidad podrá avanzar en estas etapas posteriores a la automatización con los mismos estudiantes que requieran de un proyecto de fin de carrera u otros, logrando así ir mucho más allá de la automatización de procesos hasta llegar a la completa mejora y rendimiento de los mismos.

Bibliografía

AuraPortal (2014). ¿Qué es un BPM?

Bhat, R. (2013). *Bonita Open Solution 5.x essentials model simple-to-complex business workflow processes with Bonita Open Solution.* Birmingham, UK: Packt Pub.

BonitaSoft (2014a). Build a process for deployment | Bonita Documentation.

BonitaSoft (2014b). Empresa | Bonitasoft | Open Source Workflow & BPM software.

BonitaSoft (2014c). La Guía Definitiva de BPMN2.

BonitaSoft (2015a). Bonita BPM Studio installation | Bonita Documentation.

BonitaSoft (2015b). Database configuration | Bonita Documentation.

BonitaSoft (2015c). Hardware and software requirements | Bonita Documentation.

BonitaSoft (2015d). Import/export an organization | Bonita Documentation.

Colabra Procesos (2012). BPM.

Farrance, M. (2014). Bonitasoft continúa su impresionante crecimiento en 2013 | Bonitasoft | Open Source Workflow & BPM software.

JBoss (2015). HIBERNATE - Relational Persistence for Idiomatic Java.

Marco, D. (2010). Introducción a Groovy (I).

OMG, O. M. G. (2011). *Business Process Model and Notation (BPMN), Version 2.0*.

Sánchez, D. (2011). Introducción a Business Process Management (BPM).

Subramaniam, V. (2008). *Programming Groovy : dynamic productivity for the Java developer*. Raleigh, N.C: Pragmatic Bookshelf.

ANEXOS