ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

CONTROL DEL PROCESO DE RETORNO DE DINERO DE PLANILLAS DE SEGUROS UTILIZANDO UNA HERRAMIENTA BPM LIBRE INTEGRADA AL SISTEMA HOSPITALARIO INNOVATIVA SALUD.

AUTOR: SHWEIZER BONILLA ARIE

DIRECTOR: ING. VICTOR PALIZ

ÍNDICE

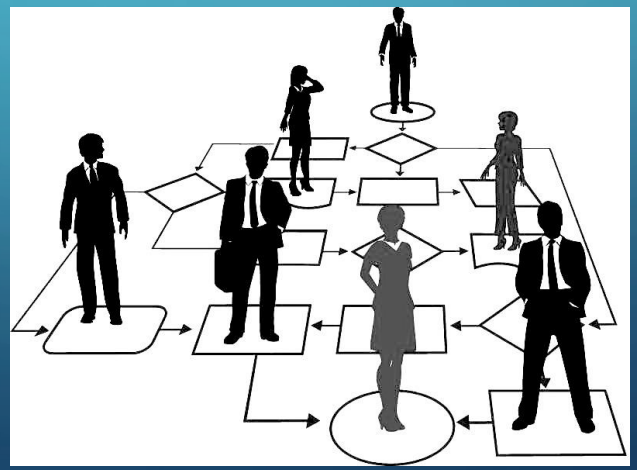
- Problemática
- Propuesta
- Objetivos
- Metodología
- Comparación entre herramientas BPMS
- Análisis y diseño
- Integración y pruebas.
- Conclusiones
- Recomendaciones.

PROBLEMÁTICA.

- Existen desbalances contables por la falta de control en el actual manejo de pagos de seguros.
- Descuido de médicos al llenar la información necesaria en las planillas de seguros.

PROPUESTA

- La solución recomendad por el centro de transferencia tecnológica consiste en la administración por procesos desde un BPMS que se encuentre integrada a Innovativa Salud y permita a los usuarios continuar con sus actividades cotidianas pero controlando desde el BPMS el desarrollo del proceso y permitiendo al administrador realizar un seguimiento del mismo.



OBJETIVO GENERAL

- Controlar el proceso de retorno de dinero de planillas de seguros utilizando una herramienta BPM libre integrada al sistema hospitalario Innovativa Salud.

OBJETIVOS ESPECÍFICOS

- Realizar estudio comparativo entre herramientas BPMS libres.
- Desarrollar el proceso de aprobación de planilla de seguros utilizando un manejador de procesos de negocio.
- Integrar el proceso de aprobación de planillas al Sistema Innovativa Salud.
- Desarrollar reportes gerenciales que permitan monitorear el proceso de aprobación de planillas y brinden información para la toma de decisiones.

METODOLOGÍA

- Ciclo de vida BPM:

COMPARACIÓN ENTRE BMPS LIBRES

BPMS/ CARACTERISCA	Bonita Soft	jBPM	Bizagi	ProcessMaker	Activiti BPM
Administración de perfiles	No en community	Si	Si	Si	Si
Generación de reportes	No en community	Si, como modulo	Si	Si	Si
Simulación	No en community	Si	Si	No	No
Modelamiento de procesos	Si	Si	Si	Si	Si
Soporte para BPMN2	Si	Si	Si	Si	Si
Soporte técnico	Muy bueno	Excelente	Muy bueno	Bueno	Bueno
Nivel de Documentación	Muy bueno	Muy bueno	Muy bueno	Bueno	Bueno
Licencia	LGPL	Apache License 2.0	Modeler Freeware Studio Freeware Studio pagado	Affero GPL	Apache License 2.0
Base de tecnología	Java	Java	Java y .NET	php	Java
Balanceo de carga	No	No	Si	No	No

ANÁLISIS Y DISEÑO

REQUISITOS

Número de requisito	RF1		
Nombre de requisito	Proceso generado en BPMS		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

Número de requisito	RF2		
Nombre de requisito	Integrar proceso a la interfaz de usuario de de Innovativa Salud		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

Número de requisito	RF3		
Nombre de requisito	Utilizar usuarios de Innovativa Salud en BPMS		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

Número de requisito	RF4		
Nombre de requisito	Utilizar postgres como base de datos de BPMS		
Tipo	Requisito	X Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	Alta/Esencial	X Media/Deseado	Baja/ Opcional

Número de requisito	RF5		
Nombre de requisito	Generación de reportes del proceso		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

REQUISITOS

Número de requisito	RF6		
Nombre de requisito	Seguimiento de planillas de manera individual		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

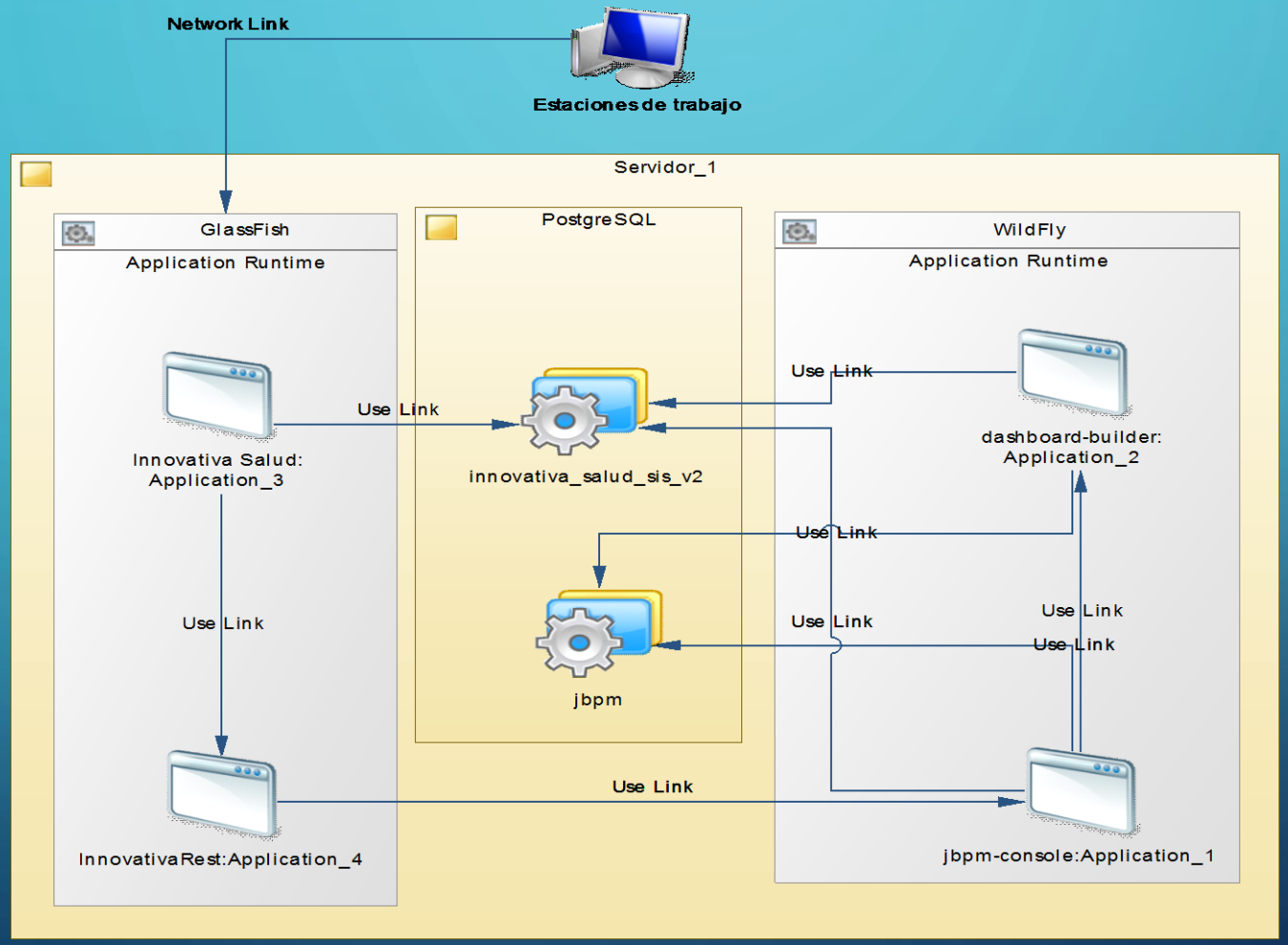
Número de requisito	RF7		
Nombre de requisito	La facturación final se realiza en por grupos de planillas pero el proceso es individual para cada una		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

Número de requisito	RF8		
Nombre de requisito	En caso en que el bpms no pueda ejecutarse, innovativa Salud debe poder seguir operando		
Tipo	Requisito	X Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

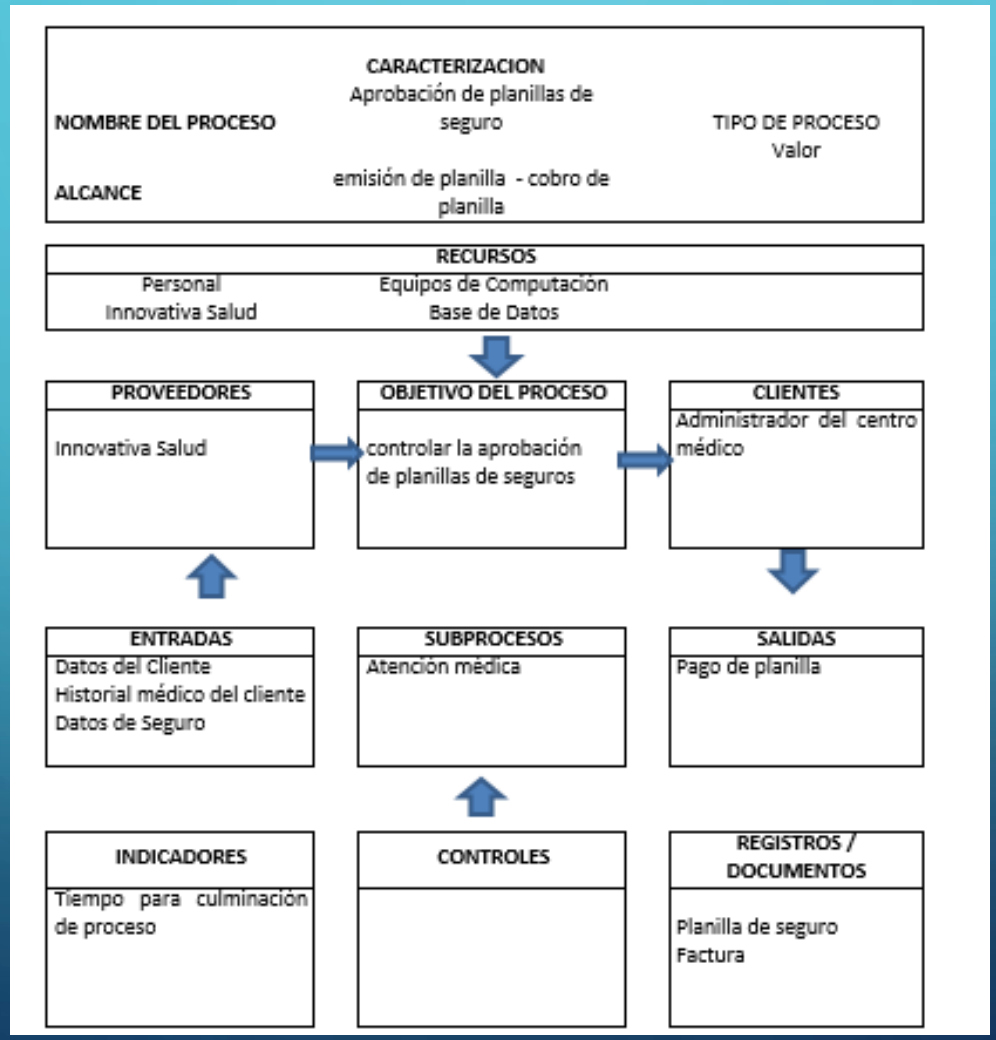
Número de requisito	RF9		
Nombre de requisito	Si se Utilizan Servicios es necesaria la creación de una capa intermedia a modo de bus de servicios		
Tipo	Requisito	X Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	Alta/Esencial	X Media/Deseado	Baja/ Opcional

Número de requisito	RF10		
Nombre de requisito	Innovativa salud es concebido como sistema multiempresa por lo que el proceso debe funcionar de igual manera		
Tipo	X Requisito	Restricción	
Fuente del requisito	stakeholder		
Prioridad del requisito	X Alta/Esencial	Media/Deseado	Baja/ Opcional

DIAGRAMA DE IMPLEMENTACIÓN



CARACTERIZACIÓN DEL PROCESO



	CARACTERIZACION	
NOMBRE DEL PROCESO	Aprobación de planillas de seguro	TIPO DE PROCESO Valor
ALCANCE	emisión de planilla - cobro de planilla	

RECURSOS	
Personal Innovativa Salud	Equipos de Computación Base de Datos

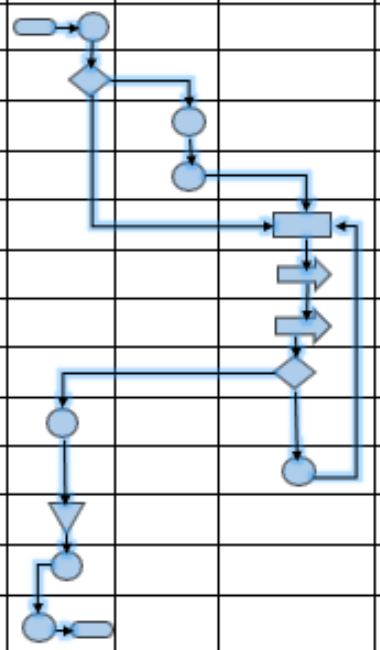
PROVEEDORES	**OBJETIVO DEL PROCESO**	**CLIENTES**
Innovativa Salud	controlar la aprobación de planillas de seguros	Administrador del centro médico

ENTRADAS	**SUBPROCESOS**	**SALIDAS**
Datos del Cliente Historial médico del cliente Datos de Seguro	Atención médica	Pago de planilla

INDICADORES	**CONTROLES**	**REGISTROS / DOCUMENTOS**
Tiempo para culminación de proceso		Planilla de seguro Factura

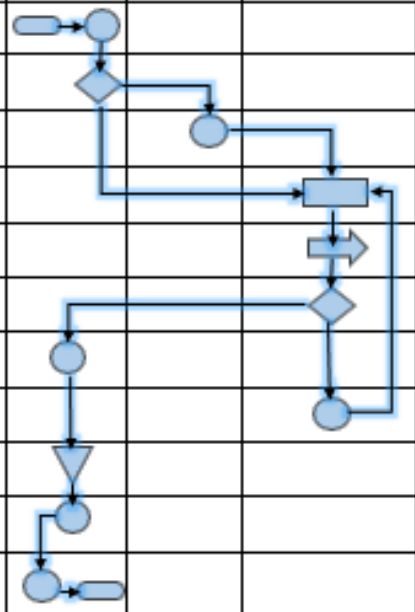
PROCESO ORIGINAL

	PROCESO: Aprobación de planillas de seguro LEVANTAMEINTO DE PROCESOS					
PROCESO	Aprobación de planillas de seguro				CÓDIGO	P001
ENTRADA DEL PROCESO	Datos del cliente, Historial médico del cliente, Datos de Seguro				TIEMPO DE CICLO	170
SALIDA DEL PROCESO	Pago de planilla				EFICIENCIA EN TIEMPO	53.35%
NO.	ACTIVIDADES	DIAGRAMA DE FLUJO				
		Caja	Medico	Administrador	Tiempo Agrega Valor	Tiempo no Agrega Valor
1	Emitir de planilla				5	
2	Decidir servicio					1
3	Atender a paciente				15	
4	Completar planilla				5	
5	Revisar planilla					20
6	Enviar planilla					20
7	Receptar respuesta					15
8	Decidir según respuesta					5
9	Emitir factura				2	
10	Realizar cambios				60	
11	Incluir en cuentas por cobrar					10
12	Verificar pago				2	
13	Retirar de cuentas por cobrar					10
Total					89	81

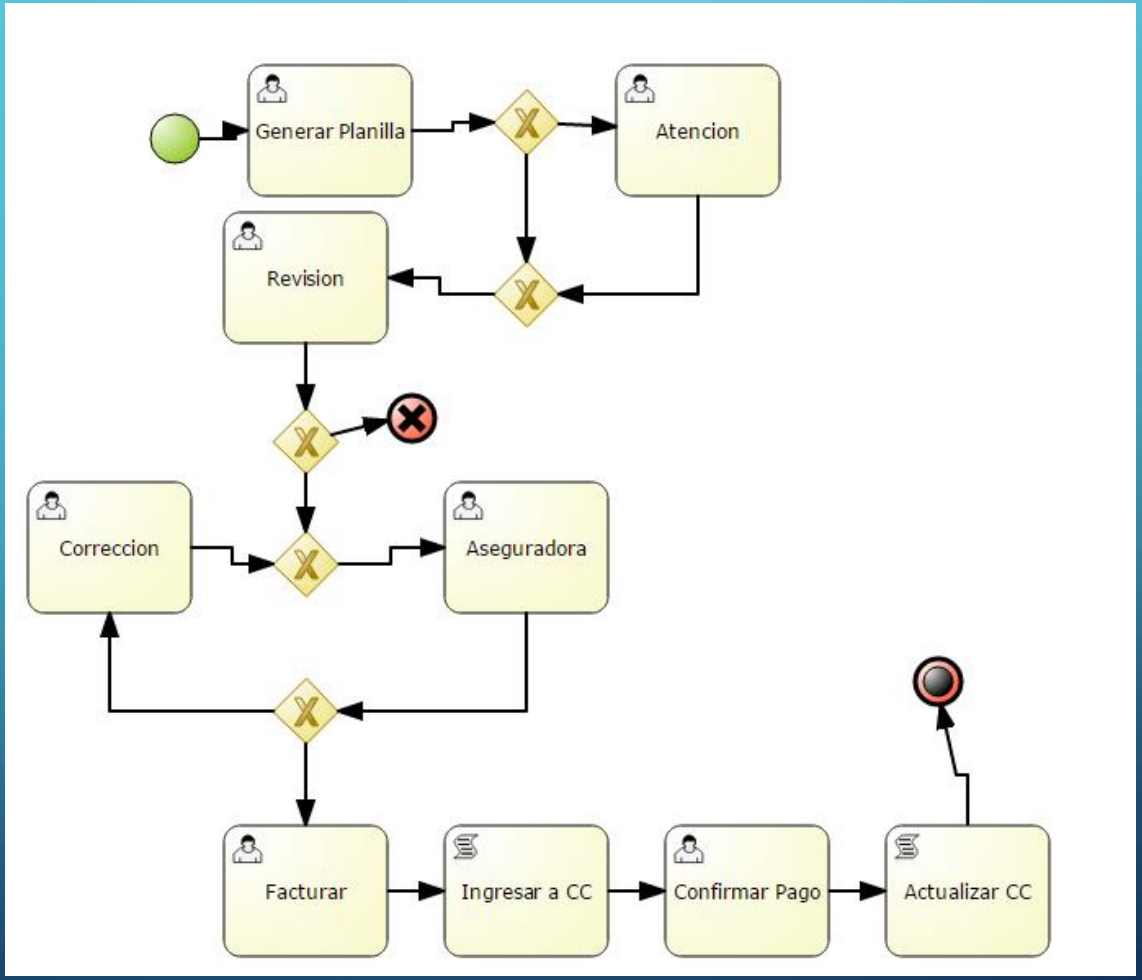


PROCESO RECOMENDADO

	PROCESO: Aprobación de planillas de seguro LEVANTAMEINTO DE PROCESOS					
PROCESO	Aprobación de planillas de seguro				CÓDIGO	P001
ENTRADA DEL PROCESO	Datos del cliente, Historial médico del cliente, Datos de Seguro				TIEMPO DE CICLO	108
SALIDA DEL PROCESO	Pago de planilla				EFICIENCIA EN TIEMPO	77.77%
NO.	ACTIVIDADES	DIAGRAMA DE FLUJO				
		Caja	Medico	Administrador	Tiempo Agrega Valor	Tiempo no Agrega Valor
1	Emitir de planilla				5	
2	Decidir servicio					1
3	Atender a paciente				15	
5	Revisar planilla					20
6	Auditoria Aseguradora					--
8	Decidir según respuesta					1
9	Emitir factura				2	
10	Realizar cambios				60	
11	Incluir en cuentas por cobrar					1
12	Verificar pago				2	
13	Retirar de cuentas por cobrar					1
Total					84	24



PROCESO EN BPMN2



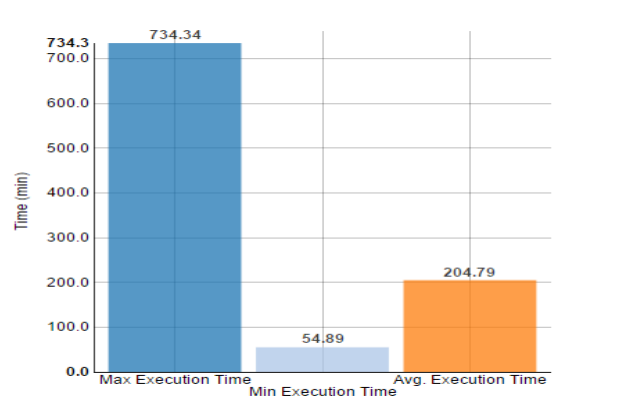
SIMULACIÓN

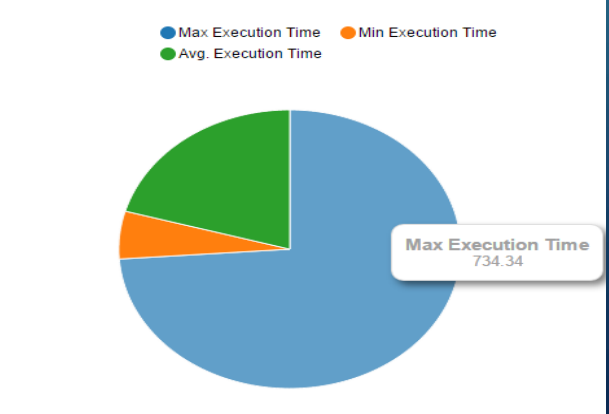
Veces ejecutadas

Max Execution Time (min)	Min Execution Time (min)	Avg. Execution Time (min)
734.34	54.89	204.79

Instancias de actividad

Facturar (#)	Correccion (#)	Atencion (#)	Revision (#)	Aseguradora (#)	Ingresar a CC (#)	Generar Planilla (#)	Confirmar Pago (#)	Actualizar CC (#)
67	50	38	50	67	67	50	67	67

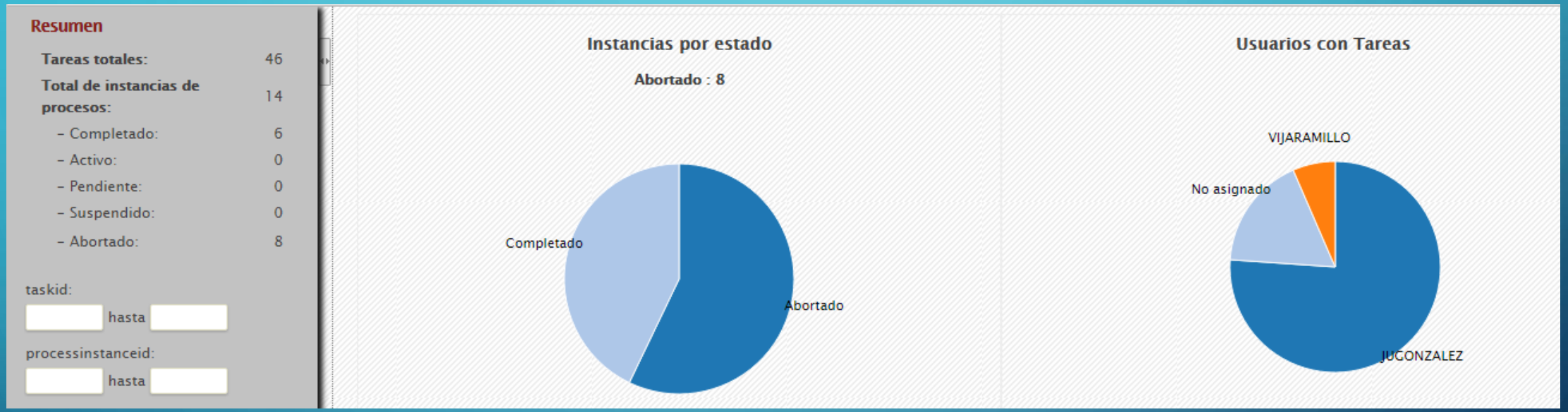




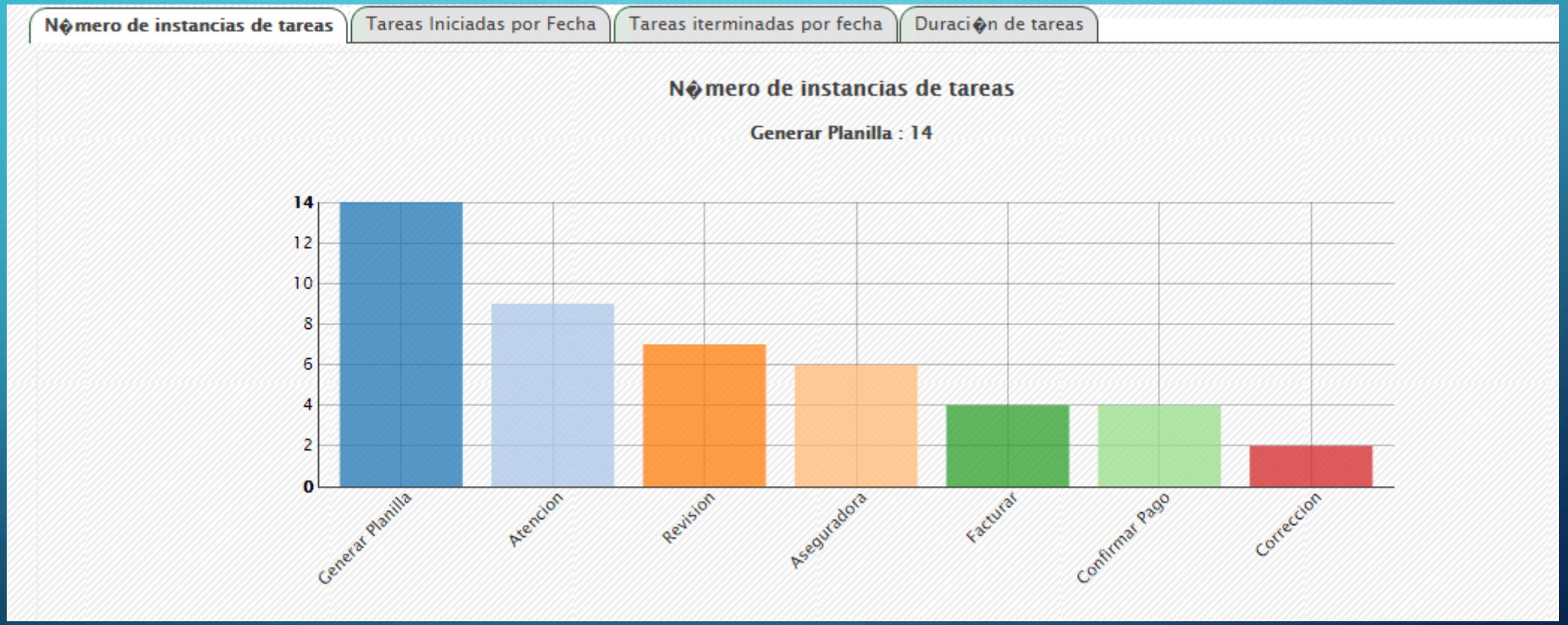
REPORTES

<table>
<tr><td rowspan="3">Resumen</td><td colspan="2">Instancias por estado</td><td colspan="2">Usuarios con tareas</td></tr>
<tr><td>Número de tareas</td><td>Tareas iniciadas</td><td>Tareas terminadas</td><td>Duración tareas</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td rowspan="3">Filtros</td><td colspan="4">Instancias por proceso por estado</td></tr>
<tr><td colspan="2">Procesos iniciados por fecha</td><td colspan="2">Procesos culminados por fecha</td></tr>
<tr><td colspan="4"></td></tr>
</table>

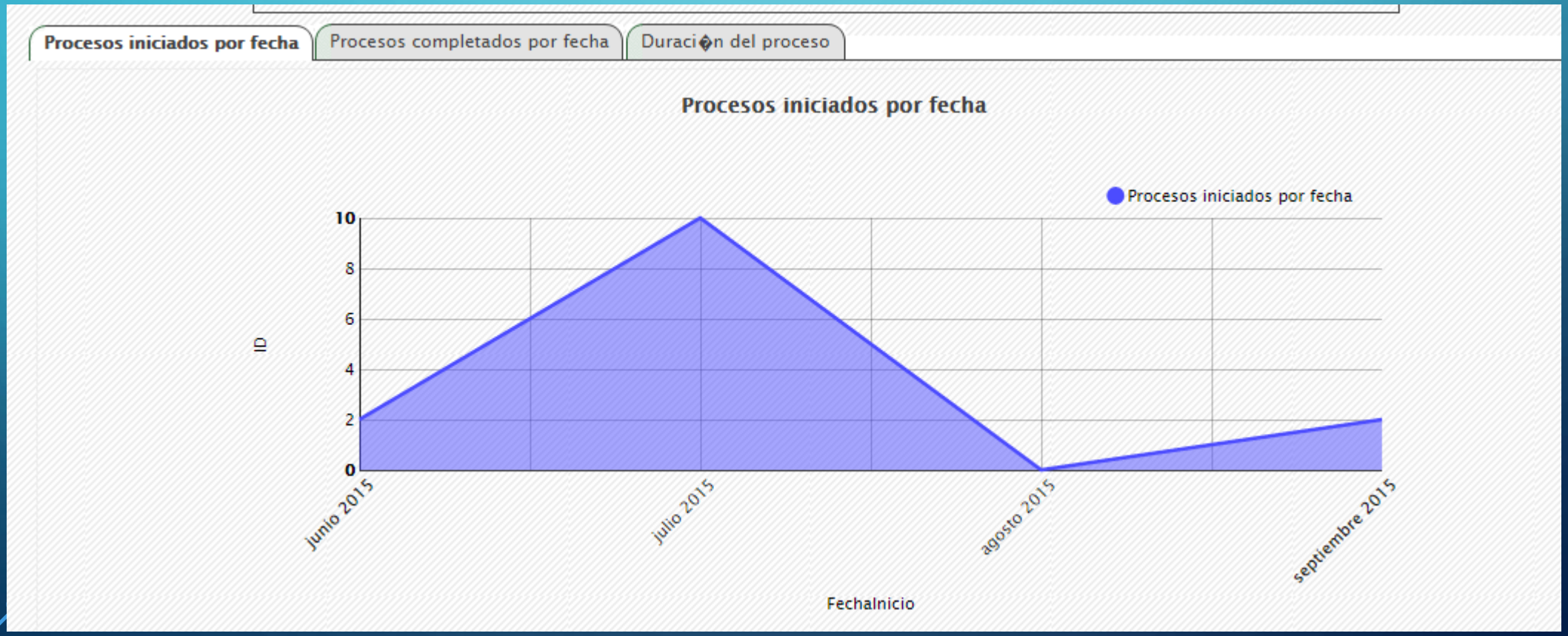
REPORTES


Resumen
Tareas totales: 46
Total de instancias de procesos: 14
- Completado: 6
- Activo: 0
- Pendiente: 0
- Suspendido: 0
- Abortado: 8
taskid:
hasta
processinstanceid:
hasta
Instancias por estado
Abortado : 8
Completado
Abortado
Usuarios con Tareas
VIJARAMILLO
No asignado
JUGONZALEZ

REPORTES

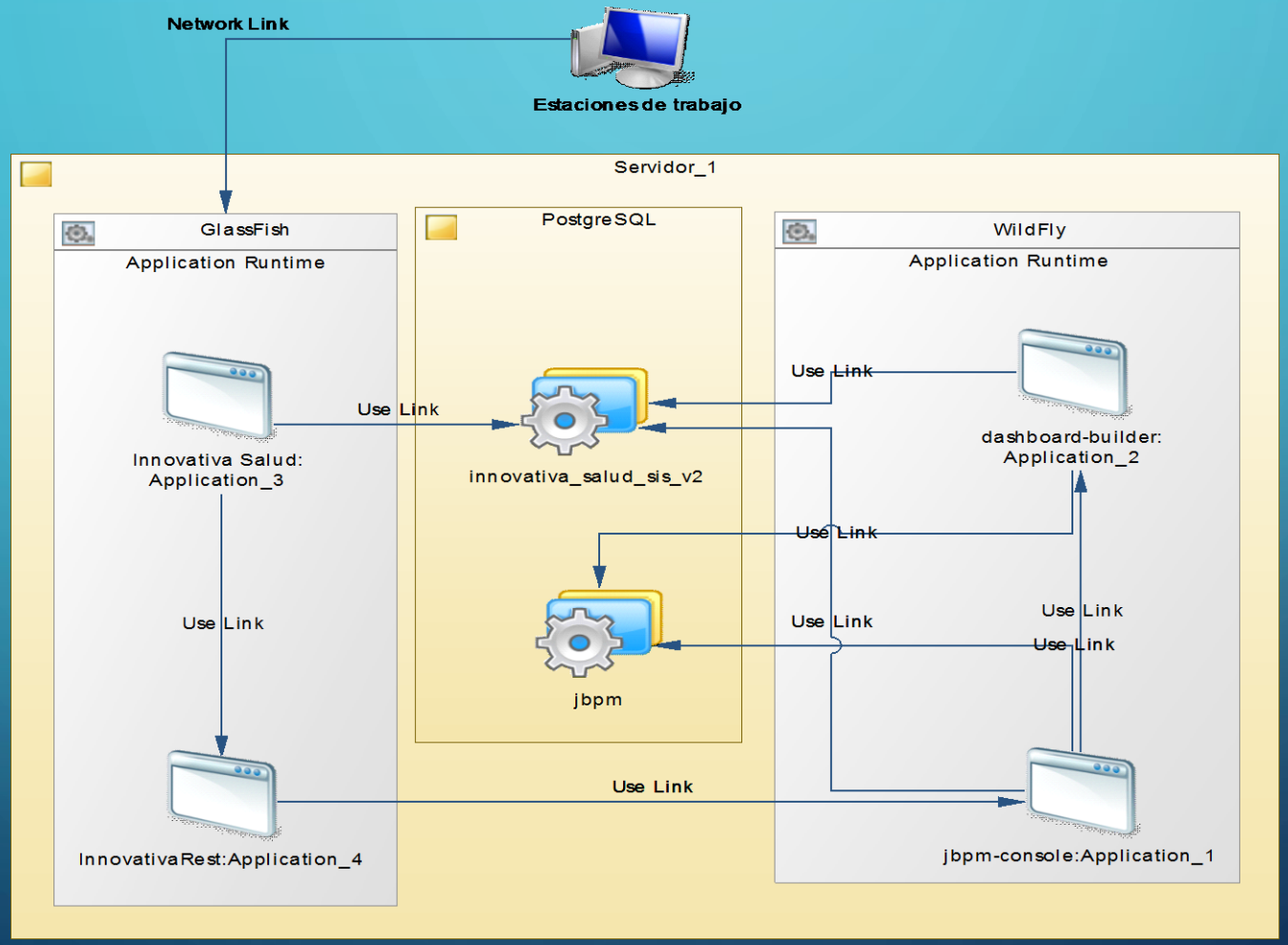

N◆mero de instancias de tareas
Tareas Iniciadas por Fecha
Tareas iterminadas por fecha
Duraci◆n de tareas
N◆mero de instancias de tareas
Generar Planilla : 14
14
12
10
8
6
4
2
0
Generar Planilla
Atencion
Revision
Aseguradora
Facturar
Confirmar Pago
Correccion

REPORTES


Procesos iniciados por fecha
Procesos completados por fecha
Duraci&#65533;n del proceso
Procesos iniciados por fecha
Procesos iniciados por fecha
10
8
6
4
2
0
ID
junio 2015
julio 2015
agosto 2015
septiembre 2015
FechaInicio

INTEGRACIÓN Y PRUEBAS

DIAGRAMA DE IMPLEMENTACIÓN



INNOVATIVAREST.

- ActividadActual
- Aprobacion.
- Generar.
- IniciarProceso.
- Ping.
- RealizarTarea
- Revisión

COM.JVC.MEDISYS.BEAN.CLIENTEREST

```
  /// Incio del proceso de Planilla
  /*  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
   */
  ProcesoBean pb = new ProcesoBean();
  SeussUsrsistem usr = ((MenuBarBean) FacesUtils.getManagedBean("menuBar")).getUser();
  TripleDES des = new TripleDES();
   pb.IniciarProceso(usr.getUssIdentifica(), des.decrypt(usr.getUssPassword()),""+usr.getCodigoHospital().getCodigo());
      if(pb.isNumeric(pb.getID()))
       {planilla.setId_proceso(Integer.parseInt(pb.getID()));}
      else{
          planilla.setId_proceso(0);}

      boolean atencion=false;
/*xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
*/
```

```
  }
      //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
     // proceso de generar factura con o sin atencion medica

if(!pb.GenerarPlanilla(""+planilla.getId_proceso(), ""+atencion,usr.getUssIdentifica(),des.decrypt(usr.getUssPassword())))
{
    planilla.setId_proceso(0);

}
 //xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

CONFIGURACIÓN PARA INTEGRACIÓN

- Cambio de puerto WildFly.
- Registro de Driver dataSource PostgreSQL.
- Agregar Roles.
- Agregar Permisos.
- Realizar cambios en base de datos de Innovativa Salud.

PRUEBAS

Caso de Prueba 003 Proceso generado en BPMS	
Código de Identificación:	CP003
Descripción:	Ejecución sin atención médica, aprobado por administrador y no fue aprobado por aseguradora.
Variables de Entrada (Inputs):	Atención médica, Continuar, Aprobado.
Flujo normal del proceso	1. Crear nueva instancia de proceso. 2. Emitir planilla 3. Revisión. 4. Aseguradora. 5. Correccion. 6. Aseguradora. 7. Facturar. 8. Ingresar a cuentas por cobrar 9. Confirmar pago. 10. Actualizar cuentas por cobrar.
Resultado esperado:	Proceso culminado y almacenado en base de datos.
Flujo alterno	Atención médica, cancelar el proceso o no ser aprobado por aseguradora.
Resultado alternativo esperado:	No se puede continuar con el proceso.
Errores encontrados	N/A
Sugerencias de corrección	N/A

PRUEBAS

Caso de Prueba 004 Proceso generado en BPMS	
Código de Identificación:	CP004
Descripción	Ejecución con atención médica cancelado por administrador
Variables de Entrada (Inputs):	Atención médica, Continuar, Aprobado.
Flujo normal del proceso	1. Crear nueva instancia de proceso. 2. Emitir planilla 3. Revisión.
Resultado esperado:	Proceso culminado y almacenado en base de datos.
Flujo alterno	Omitir atención médica, continuar con proceso.
Resultado alternativo esperado:	Continúa el proceso
Errores encontrados	N/A
Sugerencias de corrección	N/A

CONCLUSIONES

- Existe variedad entre los BPMS libres en el mercado, sin embargo por las características de lenguaje, documentación, soporte e integración jBPM se seleccionó como la mejor opción para este proyecto.
- La interfaz gráfica de jBPM facilita el desarrollo de procesos y el mantenimiento de los mismos, además la notación BPMN proporciona a los usuarios no técnicos una idea clara de como opera el proceso.
- La metodología para procesos no posee un enfoque de Ingeniería de Software incluso en BPMS, debido a que procesos es una herramienta administrativa y los BPMS heredan el ciclo de vida de estos como metodología de mejora continúa.
- jBPM ofrece varias opciones para integrar el administrador de procesos a otros proyectos, se determinó que la manera apropiada de realizar esta tarea es mediante servicios en este caso utilizando la tecnología REST que requiere menores recursos para su ejecución.
- DashBoard proyecto generado por JBoss puede ser utilizado fácilmente como complemento de jBPM facilitando la presentación de informes en pantalla y permite la aplicación de filtros en estos informes. Para utilizar otros proyectos como Innovativa Salud es necesario realizar configuraciones de seguridad en su código.

RECOMENDACIONES

- Se recomienda la utilización de la versión 6.1.0 de jBPM.
- Existe un bug dentro del proyecto en su versión 6.1.0 , por lo que se recomienda utilizar el plugin de Eclipse para realizar de manera adecuado el mapeo de variables.
- Debido a la gran cantidad de librerías necesarias para la correcta integración del sistema se recomienda utilizar maven para facilitar la descarga de estos recursos.
- Durante la integración, se presentaron problemas con las versiones de ciertas librerías descargadas por maven, por lo que se recomienda utilizar las versiones manejadas en el archivo pom.xml de proyecto InnovativaRest.
- Se recomienda la utilización de bpms para la gestión de procesos debido a que el ciclo de vida con estas herramientas posee una etapa de simulación que permite analizar si un determinado cambio al proceso tendrá un impacto favorable en la producción.

¿Preguntas ?

CONFIGURACIÓN:

- Cambio de puerto

```
    <socket-binding-group name="standard-sockets" default-
interface="public" port-offset="${jboss.socket.binding.port-
offset:0}">
        <socket-binding name="management-http"
interface="management"
port="${jboss.management.http.port:9990}"/>
        <socket-binding name="management-https"
interface="management"
port="${jboss.management.https.port:9993}"/>
        <socket-binding name="ajp"
port="${jboss.ajp.port:8009}"/>
        <socket-binding name="http"
port="${jboss.http.port:8088}"/>
        <socket-binding name="https"
port="${jboss.https.port:8443}"/>
```

CONFIGURACIÓN:

- Drivers y Data Source:

module add --name=org.postgres --resources=tmp/postgresql-9.3-1101.jdbc41.jar --dependencies=javax.api,javax.transaction.api

/subsystem=datasources/jdbc-driver=postgres:add(driver-name="postgres",driver-module-name="org.postgres",driver-class-name=org.postgresql.Driver)

Registra el driver, creando previamente una carpeta tmp dentro la cual se tiene el driver descargado.

data-source add --jndi-name=java:jboss/datasources/jBPMPostgresDS --name=PostgresDS1 --connection-url=jdbc:postgresql://localhost:5432/jBPM --driver-name=postgres --user-name=jBPM --password=jBPM

CONFIGURACIÓN:

- Roles:

```
    <security-constraint>
      <web-resource-collection>
        <web-resource-name>console</web-resource-name>
        <url-pattern>/org.kie.workbench.KIEWebapp/*</url-pattern>
        <url-pattern>*.erraiBus</url-pattern>
      </web-resource-collection>
      <auth-constraint>
        <role-name>admin</role-name>
        <role-name>analyst</role-name>
        <role-name>developer</role-name>
        <role-name>user</role-name>
<role-name>user1</role-name>

<role-name>user5</role-name>
```

CONFIGURACIÓN:

- Permisos:

```
# Granted roles per feature
# Users with a given role will only be able to access those features specified.
#
# NOTES:
# - If a group feature is granted that also implies granting all its children features.
# - Features left out of the list are granted to all roles by default.
# - A role can be denied by adding the prefix  '!'.

roles.wb_everything=admin
roles.wb_for_developers=developer
roles.wb_for_business_analysts=analyst
roles.wb_for_business_users=user,user1,user5
roles.wb_for_managers=manager
```

CONFIGURACIÓN:

- Base de datos Innovativa:

```
ALTER TABLE postgres.seuss_usrsistem ADD COLUMN rol_jbpm character varying(50);
ALTER TABLE postgres.seuss_usrsistem ALTER COLUMN rol_jbpm SET DEFAULT 'user'::character varying;

update postgres.seuss_usrsistem set rol_jbpm = 'user';

insert into postgres.aemed_medico(tes_codigo, med_cedula, med_apellidos, med_nombres, med_direccion)
values ('1', '0000000001', 'USR JBPM', 'ADMIN', '');
```

COM.JVC.MEDISYS.BEAN.CLIENTEREST

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 // Doy por realizada la atencion en el proceso
 int ID = facturasServicio.listarPorServicio(this.notaEvo.getCodigoServicio().getCodigo_servicio()).getPlanilla().getId_proceso();
 ProcesoBean pb = new ProcesoBean();
 SeussUsrsistem usr = ((MenuBarBean) FacesUtils.getManagedBean("menuBar")).getUser();
 TripleDES des = new TripleDES();
 if (!pb.Atencion("" + ID, usr.getUssIdentifica(), des.decrypt(usr.getUssPassword()))) {
     Planilla pl = facturasServicio.listarPorServicio(this.notaEvo.getCodigoServicio().getCodigo_servicio()).getPlanilla();
     pl.setId_proceso(0);

 }
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
public String enviarAseguradora() throws Exception
{
      SeussUsrsistem usr = ((MenuBarBean) FacesUtils.getManagedBean("menuBar")).getUser();
         TripleDES des = new TripleDES();
     this.currentPlanilla.setEstadoPlanilla("P");
     this.currentPlanilla= planillasServicio.editar(currentPlanilla);
     ProcesoBean pb= new ProcesoBean();
     if(pb.ActividadActual("Revision", ""+currentPlanilla.getId_proceso(), usr.getUssIdentifica(),des.decrypt(usr.getUssPassword())))
     { if(!pb.RevisarPlanilla(""+currentPlanilla.getId_proceso(),"true",usr.getUssIdentifica(),des.decrypt(usr.getUssPassword())))
         {
             currentPlanilla.setId_proceso(0);
             planillasServicio.editar(currentPlanilla);
         }
     }else
     {
         if(!pb.Atencion( ""+currentPlanilla.getId_proceso(), usr.getUssIdentifica(), des.decrypt(usr.getUssPassword())))
         {
              currentPlanilla.setId_proceso(0);
             planillasServicio.editar(currentPlanilla);
         }
     }
     FacesUtils.addInfoMessage("La Planilla lista para envio.");
      actualizarBotones();
     return null;
}
```

COM.JVC.MEDISYS.BEAN.CLIENTEREST

```
public String RespuestaAseguradoraF() throws Exception

ProcesoBean pb= new ProcesoBean();
if(!pb.Aseguradora(""+currentPlanilla.getId_proceso(),"false",usr.getUssIdentifica(), des.decrypt(usr.getUssPassword())))
{
     currentPlanilla.setId_proceso(0);
         planillasServicio.editar(currentPlanilla);
}

   this.currentPlanilla.setEstadoPlanilla("K");
this.currentPlanilla= planillasServicio.editar(currentPlanilla);
 actualizarBotones();
return null;
```

```
}
if (detalle.getDetalle().startsWith("Planilla")) {
    ProcesoBean pb = new ProcesoBean();
    SeussUsrsistem usr = ((MenuBarBean) FacesUtils.getManagedBean("menuBar")).getUser();
    TripleDES des = new TripleDES();
    if (!pb.Atencion("" + detalle.getPlanilla().getId_proceso(), usr.getUssIdentifica(), des.decrypt(usr.getUssPassword())))
    {
        detalle.getPlanilla().setId_proceso(0);
        planillasServicio.editar(detalle.getPlanilla());
    }
}
```

COM.JVC.MEDISYS.BEAN.CLIENTEREST

```
        for( DetalleFactura df : cabecera.getDetalleFacturaList())
        {
            int num= this.obtenerNumPlanilla(df.getDetalle());
            int id=planillasServicio.buscarCodigo(num).getId_proceso();


         ProcesoBean pb = new ProcesoBean();
         SeussUsrsistem usr = ((MenuBarBean) FacesUtils.getManagedBean("menuBar")).getUser();
TripleDES des = new TripleDES();
 if(!pb.Atencion(""+id,usr.getUssIdentifica(), des.decrypt(usr.getUssPassword())))
 {
     cabecera.setId_proceso(0);
     facturasServicio.editar(cabecera);
 }

        }
        facturasServicio.editar(cabecera);

    }
```