Integración del entorno Bonitasoft con la herramienta de Gestión de Incidencias Jira

TESINA DE

LICENCITARURA EN SISTEMAS

Guillermo Pablo Marcos
Federico Luis Mobrici

UNIVERSIDAD NACIONAL DE LA PLATA

Marzo 2020

Directora

Dra.Patricia Bazán

Dedicatorias

Guillermo Marcos

El presente trabajo está dedicado a mi familia por haber sido mi apoyo a lo largo de toda mi carrera universitaria y a lo largo de mi vida. A todas las personas especiales que me acompañaron en esta etapa, aportando a mi formación tanto profesional y como ser humano.

Federico Mobrici

A mi madre Silvia por darme la vida y creer en mí, por su paciencia, su fortaleza, su gran esfuerzo y consejos que me ayudan a enfrentar la vida. Gracias a ella, tuve la oportunidad de iniciarme en este camino y llegar a su fin.

A mi abuela Nelly, un amor incondicional que está y estará dentro mío, por haber sido un pilar fundamental en mi vida, quien me inculcó enseñanzas y valores que mantendré para siempre.

Agradecimientos

Guillermo Marcos

A mi familia, por haberme dado la oportunidad de formarme en esta prestigiosa universidad y haber sido mi apoyo durante todo este tiempo.

A mi tutora de tesis, por haberme guiado en la elaboración de este trabajo de titulación y haberme brindado el apoyo para desarrollarme profesionalmente.

A la Facultad de informática de la Universidad Nacional de La Plata, por haberme brindado tantas oportunidades y enriquecerme en conocimiento.

Federico Mobrici

Después de un larguísimo tiempo desde la finalización de mis exámenes hasta la ejecución de este trabajo, hoy finalmente puedo agradecerle a Sabrina quien me acompañó y animó hasta conseguir este objetivo tan importante para mí.

También, me gustaría agradecerle a mi directora de tesis, Patricia Bazán, por su paciencia y ayuda aportada durante el tiempo del trabajo.

A la Facultad de Informática de la Universidad Nacional de La Plata, por haberme dado la posibilidad de formarme como profesional.

A Dios, por darme salud, sabiduría y fortaleza que me permitió llegar a este momento en mi vida.

Índice de Contenidos

Índices de Figuras

Índices de Tablas

Introducción

Desde hace tiempo, la informática es una aliada de gran importancia dentro de las organizaciones donde en sus comienzos, los sistemas de software se focalizaban en el rendimiento a nivel operativo, el almacenamiento y posteriormente al acceso de la información.

En la actualidad, es necesario contar con soluciones tecnológicas integradas. El caso que vamos a presentar trata de relacionar un entorno de Gestión de Procesos de Negocio con un Gestor de Incidencias, externo y no relacionado.

El concepto Sistema de Gestión de Procesos de Negocios (*Business Process Management System* - BPMS) es una metodología corporativa que tiene como objetivo mejorar la eficiencia dentro de las organizaciones por medio de la gestión de procesos de negocio que se deben modelar, organizar, documentar y optimizar de forma continua. Este ciclo de vida de los procesos de negocio mejora la toma de decisiones, la planificación y la gestión actual y a futuro de la organización.

Actualmente el negocio y la información van de la mano; por lo tanto, sin la adecuada administración de tareas e incidentes dentro de las organizaciones se pueden generar riesgos que afectan de manera directa tanto a clientes internos como externos de la organización.

Desde nuestro punto de vista, uno de los modos de potenciar un BPMS es relacionarlo con una herramienta de Gestión de Tareas e Incidentes de Servicios brindados por TI (Tecnología de la Información) teniendo como objetivo agilizar el registro de tareas, incidencias y errores relacionados a instancias de la ejecución de procesos de negocio. El correcto manejo de tareas e incidencias ayuda a la disponibilidad y a mantener la calidad de servicio, mejorando así la eficiencia y la duración de los procesos.

En nuestro marco de trabajo utilizaremos como BPMS el entorno Bonitasoft [51] en su versión community. Esta suite está compuesta por un motor de procesos de negocio, un diseñador gráfico para el modelado de procesos de negocio y un portal web que permite a cada usuario gestionar las tareas y procesos en los cuales está involucrado. Es un software de código abierto implementado con tecnología Java y puede ser descargado bajo GPL v2.
Analizando y realizando pruebas de concepto en el producto, pudimos observar que el entorno Bonitasoft tiene como carencia la falta de conectividad (integración) en forma directa o nativa con herramientas externas relacionadas a la gestión de tareas, errores e incidencias que pueden surgir en la ejecución de un proceso desplegado.
A continuación, describiremos algunas desventajas actuales que presenta Bonitasoft por la falta de integración con las herramientas de Gestión de Tareas e Incidentes mencionadas:

- Acceso a herramientas secundarias fuera del entorno de Bonitasoft.
- Los participantes de procesos deben aprender a utilizar en detalle la herramienta de gestión de incidencias para registrar, actualizar o visualizar tickets.
- Se generan demoras en la carga del incidente por parte de los participantes del proceso.

En base a lo expuesto, proponemos implementar la integración de la herramienta Atlassian Jira [54] como Gestor de Tareas e Incidentes, la cual nos permita detectar, clasificar, escalar, solucionar y cerrar incidentes con el fin de mejorar los procesos en ejecución y lograr mayor efectividad en el funcionamiento del entorno BPMS.

La elección de la herramienta Atlassian Jira surge de la experiencia con la que contamos y según lo analizado sobre las buenas prácticas de gestión de incidencias contenidas en los modelos establecidos en Cobit 5 (*IT Governance Institute, 2013*) [38], ITIL v3 (*IT Infrastructure Library, 2011*) [37] y CMMI (*Capability Maturity Model Integration*) [41], con el fin de adquirir conocimientos sobre cada uno y evaluar como cada modelo plantea la resolución de incidencias.

Jira es una herramienta desarrollada por la empresa Atlassian, basada en el estándar J2EE. Es una aplicación web para la gestión operativa, gestión de flujos de trabajo (workflows) y para el seguimiento de errores e incidencias de proyectos. Adicionalmente, este software es altamente adaptable y configurable, permitiendo customizar la mayoría de los aspectos tales como tipos de incidencias, campos, estados, resoluciones y flujos de trabajos.

El objetivo de este trabajo consiste en realizar la integración del Sistema de Gestión de Procesos de Negocio Bonitasoft con la herramienta de Administración de Tareas e Incidencias Atlassian Jira para que los participantes que ejecuten procesos dentro del módulo Bonita User Experience puedan registrar, actualizar y visualizar tareas o incidentes asociados a procesos en tiempo de ejecución. A través de lo mencionado, los participantes no necesitarían acceder a dos plataformas distintas, lo cual será validado con un caso de estudio.
Los objetivos específicos del trabajo son:

- Realizar un estudio de la API de Atlassian Jira.
- Realizar un estudio de cómo crear la integración dentro del entorno de Bonitasoft.
- Implementar un módulo de integración que permita relacionar participantes con usuarios y procesos con proyectos.
- Implementar un módulo para crear y modificar tareas o incidentes dentro del ambiente de Bonitasoft.
- Recuperar y visualizar las tareas o incidentes registrados en la herramienta Atlassian Jira, dentro del ambiente de Bonitasoft.
- Realizar un caso práctico de aplicación que demuestre la integración entre ambas herramientas de software.

Capítulo I – Los Procesos de Negocio y su Gestión

Antes de comenzar a tratar el tema de nuestra tesis es necesario definir algunos conceptos claves para su comprensión.

La gestión de procesos de negocio se basa en la idea de que cada producto es el resultado de un conjunto de actividades que se realizan a fin de obtener dicho producto. Por este motivo, la correcta y eficiente gestión de los procesos de negocio es un aspecto importante para la productividad de toda organización, ya que permite identificar las tareas que la organización debe realizar para producir sus productos, el orden de ejecución de las mismas y las personas responsables de realizarlas [53].

En este primer capítulo, para comprender la gestión de los procesos y su ejecución explicaremos los siguientes conceptos básicos:

- En la Sección 1.1 daremos la definición de procesos de negocio.
- En la Sección 1.2 definiremos los tipos de procesos de negocio existentes.
- En la Sección 1.3 enunciaremos los componentes de los procesos de negocio.
- En la Sección 1.4 explicaremos como actúan los procesos dentro de la organización.

En una segunda instancia, haremos mención sobre conceptos relacionados a los Sistemas de Gestión de Procesos de Negocios tomando como base las definiciones especificadas anteriormente. Para esto definimos el siguiente temario:

- En la Sección 1.5 daremos la definición de Sistemas de Gestión de Procesos de Negocio.
- En la Sección 1.6 enunciaremos los módulos principales de un Sistema de Gestión de Procesos de Negocio.
- En la Sección 1.7 daremos la definición de Arquitectura de negocios, procesos y gestión en un Sistema de Gestión de Procesos de Negocio.
- Por último, en la Sección 1.8 se dará una explicación del ciclo de vida de los procesos dentro de un Sistema de Gestión de Procesos de Negocio.

1.1 Definición de proceso de negocio

Un proceso de negocio se puede definir como una secuencia de actividades relacionadas en un orden especifico, con el objetivo de agregar valor a los productos de una organización. Además, puede verse como un conjunto estructurado de tareas relacionadas que contribuyen entre sí para lograr los objetivos de negocio.
Cada proceso de negocio tiene sus entradas, funciones y salidas. Las entradas son requisitos que deben tenerse en cuenta antes de que una función pueda ser aplicada. Cuando una función es aplicada a las entradas de un método, tendremos salidas resultantes.

Un proceso de negocio puede ser parte de un proceso mayor que disponga de éste o bien puede incluir otros procesos de negocio que deban ser incluidos en su función. En este contexto, un proceso de negocio puede ser visto en varios niveles.

Para poder aplicar los procesos se deben tener al conjunto de tareas bien definidas, una estructura jerárquica y una tendencia a la interacción y comunicación vertical. De esta forma, se podrá mejorar el desempeño de los sistemas de trabajos ya que podemos alterar los procesos de negocio eliminando o agregando pasos, o bien cambiar los métodos de cómo se usan estos pasos. Debido a esto último, surgen nuevas necesidades de capturar, modelar, ejecutar y monitorizar los procesos de negocio.

1.2 Componentes de un proceso de negocio

A continuación, se describen los componentes que pueden ser utilizados en la definición de un proceso de negocio:

- Actividades: son las tareas que debe hacer una persona (tareas interactivas, tareas humanas), o debe hacer un sistema (servicios, tareas de sistemas) dentro del proceso de negocio.
- Roles y Usuarios: son los responsables de ejecutar las tareas interactivas.
- Objeto de Negocio: es la información o documento que fluye a través del proceso de negocio.
- Flujos: es la secuencia que se define entre las actividades.
- Decisiones: criterios para la toma de distintas opciones en los procesos.
- Subproceso: otro proceso interno, es parte de un proceso de mayor nivel que tiene su propia meta, propietario, entradas y salidas.

1.3 Tipos de procesos de negocio

- Procesos Estratégicos: estos procesos dan orientación al negocio.
- Procesos Centrales: estos procesos dan el valor al cliente, son la parte principal del negocio.
- Procesos de Soporte: estos procesos dan soporte a los procesos centrales.

El modelado de procesos es usado para capturar, documentar y rediseñar procesos de negocio, facilitando el acercamiento y el acuerdo con los clientes, mejoran la motivación de los empleados y existe una mayor facilidad para responder a cambios en el contexto. Para aplicar los procesos se deben tener claras las tareas, una estructura jerárquica y una tendencia a la interacción y comunicación vertical.

1.4 Los procesos en la organización

Un proceso cruza transversalmente el organigrama de la organización y se orienta al resultado, alineando los objetivos de la organización con las necesidades y expectativas de los clientes. Las actividades de la organización son generalmente horizontales y afectan a varios departamentos y/o funciones.

Por lo tanto, en relación a lo mencionado podemos definir los siguientes conceptos:

- Organización Horizontal: se visualiza como un conjunto de flujos relacionados consiguen el producto y/o servicio final. Estos flujos están constituidos por todas las secuencias de actividades que se producen en la organización.
- Organización Vertical: la Dirección marca objetivos, logros y actividades independientes para cada departamento. La suma de los logros parciales da como resultado el logro de los objetivos globales de la organización.

1.5 Sistemas de Gestión de Procesos de Negocio

Un Sistema de Gestión de Procesos de Negocio (*Bussiness Process Managment System - BPMS*) es un sistema de software genérico que es dirigido por representaciones explícitas de procesos para coordinar la implementación de procesos de negocio. Esta clase de software permite a las organizaciones diseñar soluciones tecnológicas de información centradas en procesos, integrando personas, sistemas, datos y otros recursos empresariales [1].

Los BPMS son considerados como suites de software que están constituidas por un conjunto de tecnologías que trabajan de forma conjunta y coordinada para proporcionar soluciones basadas en la gestión de procesos de negocio [2][3][4].

Estos sistemas manejan el ciclo de vida del proceso a través de sus características funcionales y no funcionales que dan la posibilidad de definir, modelar, implementar y mejorar el proceso durante su operación.

Un BPMS debe tener la capacidad de realizar las siguientes operaciones:

- Modelar procesos de negocio.
- Proveer entornos de desarrollo de aplicaciones para la colaboración entre procesos de negocio.
- Permitir la utilización de arquitecturas SOA. El objetivo principal es poder interactuar con servicios (sean o no servicios web) para minimizar el código y generar rápidamente sistemas integrados.
- Simular procesos de negocio para evaluar su comportamiento en situaciones de cargas exigidas en determinados momentos del proceso.
- Integrar información proveniente de otros sistemas de negocio.
- Analizar los procesos y comportamiento de las operaciones.
- Desplegar aplicaciones que den soporte al proceso en condiciones tales que no se requieran mayor conocimiento y experiencia de un usuario final.
- Automatizar procesos.

- Generar, actualizar y publicar documentación de procesos.

1.6 Módulos principales de un BPMS

La plataforma ideal de un BPMS debería estar compuesta por cuatro módulos tal como se indica a continuación:

I. Modelador Gráfico de Procesos (*Business Modeler*): con este componente podemos modelar procesos de negocio y simular su ejecución. También permite definir métricas para el monitoreo.

II. Ambiente de Integración y Desarrollo (*Integration Developer*): es la herramienta que nos ayuda a implementar procesos, integrar pantallas (utilizadas para la interacción de los participantes), y servicios (interacción con sistemas legacy).

III. Servidor de Procesos de Negocio (*Process Server*): es el motor que ejecuta procesos, aplicaciones compuestas, workflows y la orquestación de servicios. Ademas, se encarga de generar datos relacionadas a métricas y monitoreo. Permite manipular los procesos en tiempo real, balanceo de carga, cambiar flujos de negocio y realizar acciones correctivas.

IV. Monitor de Actividades de Negocio (*Business Activity Monitor* - BAM): es una aplicación de administración que permite gestionar procesos y servicios. De manera gráfica se pueden observar indicadores y Niveles de Servicio a Cumplir (*Service Level Agreements* - SLA). Adicionalmente, permite definir alertas y disparadores de acuerdo a eventos de negocio que sucedan en el proceso. Este monitor ofrece datos reales a los modelos (*Business Modeler*) para ajustar simulaciones y lograr una mejora continua.

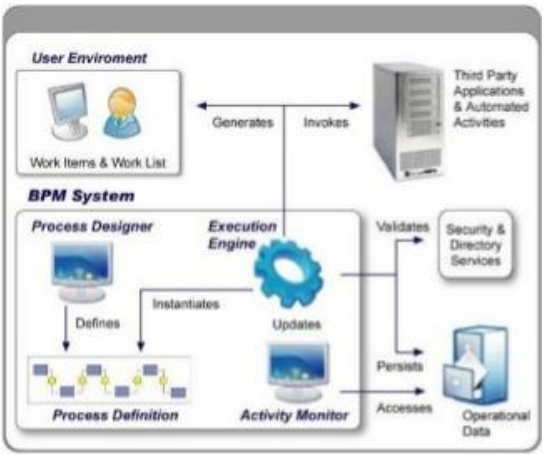

Figura 1 – Módulos ideales en la definición de un BPMS

1.7 Arquitecturas de Negocio, Procesos y Gestión en un BPMS

Una de las características más destacadas de un BPMS es que ofrece a todos los niveles de la organización un marco de trabajo definido y claro, donde todos los actores están considerados, y donde cada nivel se centra en los procesos de acuerdo a las estrategias y metas del negocio.
Un sistema BPM se encargará de describir cómo van a interactuar los sistemas y las personas para llevar adelante una actividad; es decir que cuentan con una arquitectura organizacional completa donde tiene la capacidad de desarrollar, implementar y modificar procesos de negocio rápidamente.
Para poder resolver inconvenientes que puedan surgir en cortos periodos de tiempo deberemos planificar la arquitectura del negocio, de los procesos y de la gestión.

1.7.1 Arquitectura de Negocio

La arquitectura de negocio es la representación en diseño de cómo una organización se define a sí misma en términos de su misión, visión y propósito; es decir, cómo define la forma en la que crea valor sobre los mismos. Cada organización define sus objetivos organizacionales de alto nivel y creando una estructura, incluyendo una descomposición funcional en unidades de operaciones como estructura básica para cumplir sus objetivos. La organización desarrolla relaciones a través de esas unidades, y determina cómo se relacionará con sus clientes, accionistas y propietarios [5].

Un BPMS integra de forma natural personas, procesos y tecnología mediante un cierto enfoque. El objetivo es incluir de forma directa los procesos de negocio que crean valor. Una organización dirigida por procesos trata a éstos como activos corporativos.
Un punto importante a tener en cuenta es el flujo global de las personas, los sistemas y las tecnologías que trabajan juntos para crear un valor en relación al cliente, así como el rol del director, arquitecto, propietario, ingeniero, analista y el actor del proceso. Cada uno es responsable de respaldar los negocios centrados en los procesos que administran, siendo conscientes de su posición y de cómo su propio trabajo afecta a las personas que están delante o detrás de cada uno.

En la Arquitectura de Negocio hay un conjunto de elementos de infraestructura que permiten adaptar, optimizar y facilitar el comportamiento y rendimiento de los procesos que son provistos por el equipo directivo y los miembros de la organización, asegurando de ese modo el rendimiento de la misma.

1.7.2 Arquitectura de Proceso

La arquitectura de procesos es la representación escrita o mediante diagramas de cadenas de valor que operan en toda la organización, describiendo de forma clara dónde se crea valor y cómo se relacionan y alinean los procesos operacionales con las estrategias y objetivos de la organización [5].

Este tipo de arquitectura representa procesos orquestados y estructurados que son repetibles y están automatizados, caracterizados, medidos y analizados. Un punto a tener en cuenta es que muchas de las actividades de una organización son ad-hoc, por lo tanto, ocurren fuera de los límites de orquestación.

En cada organización existen muchos procesos de negocios, los cuales definen tareas, reglas, personas y aplicaciones que conjuntamente permiten generar productos, servicios o información, ya sea para el uso interno o para clientes.
Los entornos de procesos de negocios pueden ser:

- Fundamentales (directos): incluyen procesos de flujo de valor para clientes como la presentación de nuevos productos.
- Habilitadores (indirectos): incluyen el reclutamiento de empleados y gestión de recursos.

En conjunto, ambos entornos conforman la arquitectura de procesos e incluyen todos los procesos utilizados dentro de una organización.
Para construir entornos de procesos y alinearlos con la arquitectura se deberá seguir una metodología de procesos que sirve para caracterizar y optimizar procesos de negocio dando un enfoque completo para asociar el rendimiento de personas, procesos y la tecnología con la creación de valor.

A lo largo del ciclo de vida del sistema, los procesos de negocio requieren cambios de diseño. Los cambios de estados por los que pasa un proceso desde una condición de rendimiento a otra se conocen como ciclo de vida de los procesos.

Un BPMS dentro del ciclo de procesos, abarca cuatro etapas:

I. Modelización: A través del modelado de las actividades y de los procesos se logra un mejor entendimiento del negocio y muchas veces esto presenta la forma de mejorarlos.

II. Ejecución: La automatización de procesos reduce errores, asegurando que los mismos se ejecuten siempre de la misma manera y dando elementos que permitan visualizar el estado de los mismos.

III. Monitorización: La monitorización de procesos permite asegurarnos de que los mismos estén ejecutándose eficientemente y nos da la posibilidad de obtener información que luego puede ser usada para mejorarlos.

IV. Optimización: Por medio de la información que se obtiene de la ejecución diaria de los procesos se puede identificar posibles ineficiencias en los mismos y de esta forma poder optimizarlos.

Figura 2. Ciclo de vida de un Proceso de Negocio

1.7.3 Arquitectura de Gestión

En la arquitectura organizacional el rol de la gestión es dirigir las acciones, el comportamiento de las personas y de los sistemas; así como el flujo de información a lo largo del tiempo y todo lo relacionado con la utilización y ajustes de procesos para alcanzar los objetivos organizacionales.

La arquitectura de gestión tiene en cuenta:

I. Gestión de proyectos

Un proyecto BPM es un nuevo tipo de proyecto organizacional que toma como base parte al proceso y parte a la tecnología.
A diferencia de un proyecto de desarrollo de software típico, se implementan de forma frecuente en ciclos reducidos de tiempo.

Como todo proyecto, se debe poder realizar una planificación en relación a los procesos de negocios que se van a implementar, en qué iteración serán tratados cada proceso, los requisitos cubiertos de cada proceso, etc.

En la fase de análisis y diseño, los proyectos BPM comienzan caracterizando la línea base del proceso en sí. Se mide y se valida el estado actual del proceso, y una vez conocido el alcance se crean las condiciones de las líneas base contra las que se van a comparar los progresos y mejoras.

Luego se diseña e implementa no en miras al estado ideal, sino al siguiente estado futuro. Este enfoque constituye una distinción crítica y una desviación del desarrollo clásico que busca construir el estado ideal. De esta forma, se obtiene agilidad y hace posible la mejora continua. El desarrollo de procesos de negocio requiere de la composición de servicios que realizan las funciones y simulan las acciones que van a llevar a cabo las personas y los sistemas en función al modelo de procesos.

Para optimizar el diseño, se puede tener en cuenta el análisis de los modelos de procesos mediante la simulación. [5]

II. Gestión de procesos

Una vez que un proceso se implementa de acuerdo a las especificaciones, el objetivo es mantenerlo indefinidamente (hasta que la siguiente mejora quede justificada).
Luego el modelo de proceso se orquesta mediante un motor en tiempo de ejecución que facilita la ejecución de los servicios y proporciona la transformación de valor añadido de entradas e información en salidas y resultados.
El rendimiento del proceso se mide en tiempo real y el proceso implementado es objeto de supervisión para ver si se ajusta a las especificaciones. Se da seguimiento al mismo y se registran el volúmen, la velocidad y los errores. [5]

III. Mejora de procesos

Se debe establecer un mecanismo que permita identificar y determinar cuáles son los procesos necesarios a mejorar o corregir en cuanto a su efectividad debido a que los procesos sufren cambios, variaciones y deterioros en el tiempo por diferentes causas.
Con el paso del tiempo, pueden surgir nuevas necesidades organizacionales o nuevas tecnologías, el reemplazo de desarrollos propios por funciones estándares o el uso de nuevas funciones. Con la aplicación de metodologías de mejora en procesos, se pueden corregir defectos en los procesos y al mismo tiempo mejorar su efectividad.
Los métodos de Mejora Continua de Procesos (*Continuous Process Improvement* - CPI) es una parte esencial y de gran importancia en el ámbito BPM. [5]

1.8 Ciclo de vida de los procesos de negocio en un BPMS

La mayoría de las metodologías de Ingeniería de Software definen una secuencia de actividades similares, tales como el análisis, diseño, implementación y despliegue. Hoy día, estos métodos han evolucionado y se aplican de forma iterativa realizando cambios incrementales. El ciclo de vida para la gestión de un proceso de negocio es similar a las metodologías de desarrollo tradicionales, pero toman un enfoque particular en el análisis y supervisión, ya que está orientado a ciclos iterativos de mejora continua [6].

Las fases del ciclo de vida BPM están relacionadas entre sí y se encuentran organizadas en una estructura cíclica con dependencias lógicas. Un punto a tener en cuenta, es que cada fase no tiene definido un orden temporal estricto en cuanto a su ejecución. Por ejemplo, muchas actividades de análisis y diseño se llevan a cabo en cada una de las fases [1][6], y son actividades comunes en varias fases para los enfoques incrementales y evolutivos.

La clasificación y las actividades que se desarrollan en cada una de las fases del ciclo de vida varían en función de los autores que las definen, aunque existe un consenso en el que todos los autores coinciden en la definición de un conjunto de las actividades fundamentales que se deben ejecutar a lo largo del ciclo de vida de los procesos de negocio.

A continuación, se presentan los ciclos de vida de procesos de negocio definidos por Panagacos, Smith & Fingar y Weske.

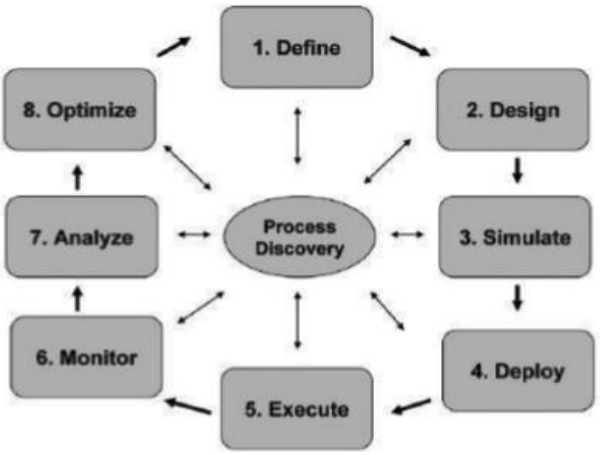

Figura 3. Ciclo de vida BPM [Panagacos 2012]

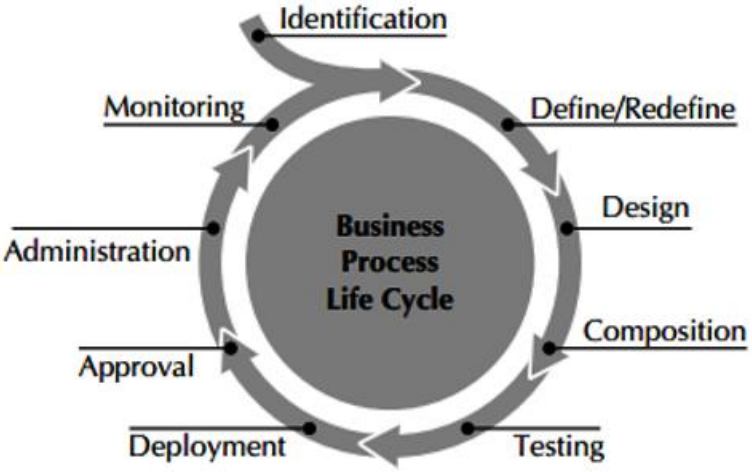

Figura 4. Ciclo de vida BPM [Smith, Fingar 2006]

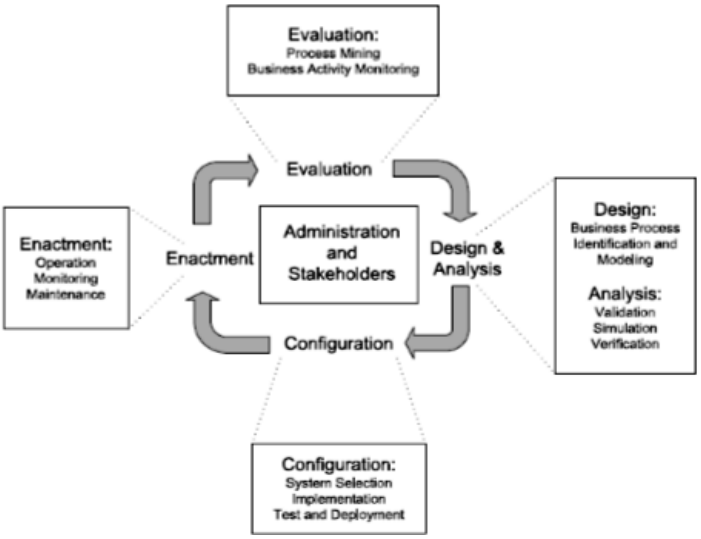

Figura 5. Ciclo de vida BPM [Weske 2012]

En general, el ciclo de vida comienza con la identificación de los procesos de negocio que aparecen al momento de su definición. A partir del modelo generado se deberá identificar y definir los objetivos estratégicos de la organización. Cuando el proceso de negocio es implementado, el modelo del proceso debe hacerse ejecutable, por lo que debe existir un entorno de ejecución. El entorno está formado por un motor de ejecución que es capaz de ejecutar los modelos de los procesos ejecutables y controlar la interacción con los recursos de la organización. A partir de esto, los usuarios pueden interactuar con el proceso en ejecución, y los administradores pueden monitorizar y controlar la ejecución de los mismos.
Los procesos en ejecución o ya ejecutados, pueden ser analizados para encontrar mejoras a aplicar, creando así un bucle de mejora continua y permitiendo el alineamiento de los procesos de negocio con los objetivos estratégicos de la organización.

El ciclo de vida debe soportar la interacción colaborativa de todos los interesados para aportar beneficios y superar las deficiencias de las metodologías BPM tradicionales, mediante la adopción de tecnologías sociales dentro de la organización [7].
Para definir las actividades que se llevan a cabo durante el ciclo de vida de los procesos de negocio, proponemos las siguientes fases del ciclo de vida BPM, tal como se describen a continuación.

1.8.1 Identificación

En esta fase se realizan dos actividades:

I. Análisis de la estrategia de la organización

En esta actividad, se trata de identificar los objetivos estratégicos de la organización. Estos deben ser definidos y documentados incluyendo también las personas interesadas en el proceso de negocio, los objetivos del proceso, los factores críticos de éxito, los requerimientos, las métricas de rendimiento, etc. [7][8].

II. Análisis de los procesos de negocio de alto nivel

En esta actividad, se realiza el descubrimiento de los procesos de negocio de alto nivel y sus dependencias [1] mediante la visualización de los patrones de trabajo y aplicaciones de la organización. En general, los procesos de negocio están implícitos dentro de los sistemas de software verticales [9][10] y en el conocimiento de las personas pertenecientes a la organización. Estos procesos de negocio son representados con una notación gráfica.

Los objetivos de la organización se pueden capturar, por ejemplo, mediante el uso de un cuadro de mando integral y los procesos de negocio de alto nivel se pueden mostrar mediante un diagrama de cadena de valor agregado [9].

Los resultados de esta fase los podemos obtener desde un Modelo de Proceso Empresarial (*Enterprise Process Model* - EPM) que es una representación gráfica de los procesos de la empresa (procesos principales y habilitantes) que especifica cómo se conectan entre sí las entradas y salidas [12] pasando por los modelos de cadena de valor añadido [9], hasta la arquitectura de procesos que es un inventario de los procesos de

negocio identificados de alto nivel especificando las relaciones con otros procesos de negocio, prioridades, personas interesadas, métricas, etc. [6][11].
Otro modelo que puede ser tenido en cuenta para la representación de las decisiones de la organización y la conexión entre la organización y las operaciones [13] es el Modelo de Motivación de Negocio (*Business Motivation Model* - BMM) que proporciona un esquema para desarrollar, comunicar y gestionar planes de negocio de manera organizada, independientemente de la metodología aplicada [14].

En la siguiente fase, el conjunto de los objetivos estratégicos y de los procesos de negocio de alto nivel son la base de los objetivos individuales que deben alcanzar los procesos de negocio, ya que a partir de estos se definen los Indicadores Clave de Rendimiento (*Key Performance Indicator* - KPI). Los KPI permiten medir si se alcanzan los objetivos definidos [11], asegurando que los procesos de negocio están alineados con los objetivos de la organización.

1.8.2 Definición

En la fase Definición, se analiza en detalle el proceso de negocio para describirlo con una notación gráfica usando técnicas de modelado.
Los modelos de alto nivel descritos en la fase anterior se hacen explícitos mediante un lenguaje de modelado formal, como por ejemplo el Modelo y Notación de Procesos de Negocio (*Business Process Model and Notation* - BPMN) [15][16].

El objetivo principal de BPMN es proporcionar una notación que pueda ser comprendida por todos los usuarios de la organización, partiendo desde los analistas de negocio que crean el proyecto inicial de los procesos, los desarrolladores técnicos que son los responsables de la implementación de la tecnología que va a soportar el desempeño de los procesos, y las personas del negocio que gestionan y supervisan los procesos [17].
En esta fase se debe realizar un análisis en profundidad, así como el modelado de tareas que son parte de segmentos predefinidos dentro de un caso determinado para permitir la Gestión Dinámica de Casos (*Dynamic Case Management* - DCM) [18]. Estos casos pueden ser definidos mediante un lenguaje de modelado formal como el Modelo de Gestión de Casos y Notación (*Case Management Model and Notation* - CMMN) [19].

Estos modelos de procesos integran varios dominios de modelado, tales como el modelado de funciones, de datos, de organización, de recursos, de reglas de negocio, KPIs, y el de integración con los sistemas de información [20]. Si bien los subdominios mencionados son los más importantes, se pueden definir subdominios adicionales en caso de que sean relevantes [1][8]. Los modelos de procesos deben proveer una imagen completa de los procesos de negocio.

En esta fase se produce la primera iteración de mejoras, aplicando técnicas de validación, simulación y verificación [6].

Con las técnicas de simulación se comprueba la validez, ya que ciertas secuencias de ejecución no deseadas pueden ser simuladas para detectar deficiencias en el modelo del proceso. Estas técnicas también ayudan a los interesados a comprobar si el proceso expone el comportamiento deseado [1][17].
En iteraciones posteriores de refinamiento y optimización, se puede utilizar la información analítica proporcionada por la fase de seguimiento y monitorización de los

procesos previamente desplegados, entrando así al ciclo de mejora continua de los procesos de negocio.

1.8.3 Diseño

Una vez que el modelo de proceso de negocio está definido y verificado, el proceso de negocio debe ser implementado.

La implementación puede realizarse de 2 maneras diferentes:

I. El modelo de proceso puede implementarse a través de un conjunto de políticas y procedimientos que los usuarios deben cumplir; por lo tanto, se puede realizar sin ningún tipo de soporte por parte de un sistema dedicado de gestión de procesos de negocio.

II. Si en la implementación se utiliza un sistema de software para el desarrollo del proceso de negocio se debe incluir en el modelo información técnica para facilitar el despliegue y ejecución en el sistema de gestión de procesos de negocio. Para lograr esto, se realiza una actividad técnica en la que se determina la viabilidad de la implementación del modelo de proceso como un proceso ejecutable [6]. El modelo debe configurarse de acuerdo al entorno de la organización y con el resto de los procesos que deban ejecutarse y controlarse. Esta configuración incluye las interacciones de los usuarios con el sistema, o sea, las interfaces de usuario, la integración con motores de reglas de negocio [21], la integración de los sistemas existentes con el sistema de gestión de procesos de negocio, el manejo de excepciones y aspectos transaccionales [1][8][9].
   Luego de implementar el proceso de negocio con la configuración técnica necesaria para su ejecución, éste debe ser probado. Las técnicas de pruebas son las mismas que se utilizan en Ingeniería del Software. El objetivo de las pruebas es comprobar si el proceso expone a través del sistema el comportamiento esperado. A nivel de procesos, las pruebas de integración y rendimiento son importantes para detectar posibles problemas de tiempo de ejecución.

1.8.4 Despliegue

Es esta fase, el modelo de proceso de negocio ejecutable se despliega sobre la infraestructura tecnológica de la organización teniendo como objetivo ponerlos a disposición de los usuarios correspondientes.
Las aplicaciones, los recursos y los servicios se asignan; mientras que los modelos de procesos de negocio ejecutables se envían al motor de ejecución de procesos.
Los procesos de negocio se pueden desplegar en 2 tipos de entornos:

I. En entornos de ejecución distribuidos, donde el proceso de negocio se ejecuta en múltiples motores de ejecución de procesos.

II. En entornos distribuidos, donde el motor de ejecución de procesos de negocio se ejecuta en un clúster de servidores.

Un punto importante a tener en cuenta en la fase de despliegue, es que realizar esta tarea, cause el mínimo de interrupciones al resto de los procesos que estén ejecutándose. Al finalizar el despliegue del proceso dentro del entorno de ejecución del BPM se debe informar a todos los participantes involucrados que dicho proceso está disponible para su uso [22].

1.8.5 Ejecución

En esta fase, cada vez que se ejecutan los modelos de procesos de negocio se crean instancias de los procesos para cumplir con los objetivos de la organización. Normalmente, el inicio de una instancia de un proceso de negocio se produce por la activación de un evento predefinido [1][2].

El objetivo principal de esta etapa es controlar la ejecución de las instancias de los procesos, verificando una orquestación correcta, lo que garantiza que las actividades de los procesos se ejecutan de acuerdo a las características de ejecución especificadas en el modelo del proceso de negocio [1][8]. Además, en el transcurso de la ejecución se trata de asegurar de no perder información y que las actividades de las instancias sean ejecutadas por el usuario, la aplicación o el servicio correcto.

En algunas ocasiones de ejecución, las instancias de los procesos pueden generar fallos, por lo tanto, su resolución debe poder resolverse automáticamente en esta fase.
En ciertos casos se necesita la intervención humana para manejar o resolver una excepción, por lo que ésta será tratada en la fase de Administración. Los usuarios deberán ser informados para luego poder interactuar con el sistema de gestión de procesos de negocio, el cual es capaz de ejecutar y mostrar las interfaces de usuario implementadas en la fase de diseño.

En los casos de actividades automáticas, el sistema debe ser capaz de integrarse con otros sistemas externos de la organización, teniendo como objetivo adquirir o proporcionar información durante la ejecución de las instancias de los procesos de negocio. Durante la ejecución de los procesos, y utilizando técnicas de monitorización se recolectan datos sobre los eventos ocurridos durante la ejecución, los cuales se almacenarán en algún tipo de registro. Los registros están formados por conjuntos ordenados de entradas que representan eventos tales como el usuario que inició el proceso, el inicio y fin de una actividad, el usuario que ejecuta una actividad, etc.

Otro punto de importancia en esta fase es que se puede asegurar que los modelos son ejecutados eficientemente, ya que el motor de ejecución se encarga del cálculo de los KPI’s definidos en los modelos de proceso.

Toda la información registrada en esta fase es usada como base para el análisis que permite la mejora de procesos mediante su optimización en la fase de Definición y/o Diseño.

1.8.6 Administración

La fase de Administración se encarga de la gestión operativa relacionada con la Tecnología de la Información y OA&M (*Operation, Administration and Management*), pero requiere tener conocimiento del negocio [6].

El objetivo principal es tratar de asegurar que los procesos de negocios desplegados estén funcionando como fueron definidos y diseñados en las fases correspondientes y que todos los pasos de control sean correctos [23]. La información de los procesos es presentada a los usuarios de negocios mediante cuadros de mando [24].

La fase de Administración está relaciona con la fase de Monitorización, donde el punto de unión radica en cubrir la supervisión sobre el comportamiento de los procesos, responder a alertas y a condiciones de excepción que puedan producirse en ejecución. Durante la ejecución de procesos de negocio pueden ocurrir excepciones no controladas en el flujo del proceso debido a fallas en la definición, en el diseño o excepciones debidas a circunstancias complejas que se deban manejar mediante la intervención humana. Cualquier otro tipo de excepciones o errores producidos en el sistema deben ser separados de las excepciones de negocio, por lo tanto, deberán gestionarse por los Administradores de TI.

Los procesos de negocio pueden gestionarse de mejor manera en tiempo de ejecución si la monitorización presenta una respuesta eficaz para alertar sobre las condiciones excepcionales. Esto permite a los usuarios de negocio detectar proactivamente las excepciones e iniciar los procedimientos para su resolución, colaborando también a los administradores con la identificación de patrones erróneos recurrentes [25].

Otro objetivo que tiene esta fase es brindar la posibilidad de que el sistema cambie dinámicamente el comportamiento de una instancia del proceso de negocio, proporcionando una respuesta automática en circunstancias predefinidas. Estas respuestas requieren que el sistema modifique la ejecución de una instancia de proceso. La modificación del flujo del proceso en tiempo de ejecución proporciona una mayor agilidad en el proceso de negocio ya que permite a los usuarios de negocio cambiar dinámicamente el comportamiento del proceso sin tener que volver a desplegarlo. En ciertos casos, se pueden utilizar reglas de negocio para desacoplar las decisiones del flujo del proceso [26].

Un concepto importante en relación a la modificación del flujo del proceso en tiempo de ejecución es el DCM que hace uso de un componente colaborativo, impulsados por eventos externos, no estructurados, que necesita cierta funcionalidad ad-hoc para solventar eventos impredecibles y que no pueden llevarse a cabo con definiciones rigurosas de los procesos. En estos casos, los procesos de negocio pueden ser compuestos en tiempo de ejecución basándose en el contexto, en los eventos, en las situaciones, en el conocimiento y razonamiento, y en la experiencia [19].

Esta fase tambien debe cubrir tareas relacionadas a la gestión, organización y asignación de recursos pertenecientes a la organización relacionados a los procesos de negocio. Los recursos de la organización incluyen personas, sistemas de software y hardware, documentos, artefactos, etc., y se caracterizan por ser capaz de llevar a cabo tareas específicas [27]. Esta gestión permite cambiar dinámicamente y en tiempo de ejecución los recursos asignados a los procesos de negocio [1].

1.8.7 Monitorización

1.8.7.1 Perspectivas de monitoreo

La fase de Monitorización se realiza desde una perspectiva técnica o de negocio.

Desde el punto de vista técnico es necesario identificar y resolver las causas técnicas de operaciones anómalas de los procesos de negocio. Esto hace referencia a fallos producidos en los sistemas que soportan la arquitectura empresarial y procesos de negocio: problemas con la red, de conexiones, fallos en servidores, bases de datos, datos inaccesibles, tiempos de respuesta o carga del sistema.

En cambio, desde la visión de negocio se hace referencia a los procesos de negocio en sí mismos y los problemas que ocurren cuando se ejecutan. Esta visión es muy importante ya que con la información que se obtiene se impulsa el análisis necesario como parte de la siguiente iteración del ciclo de mejora de los procesos de negocio [6]. En esta fase, los procesos en ejecución se miden para proporcionar información precisa sobre el estado de las instancias de los procesos de negocio y ayudar en la toma de decisiones basada en los KPIs definidos en ellos, permitiendo enfocarse en los objetivos definidos en la estrategia de la organización y ayudar a medir la agilidad, efectividad y eficacia para la mejora continua de los procesos [1][28]. También se puede implementar metodologías de mejora de rendimiento tales como Six Sigma, Lean, o combinaciones de ambos enfoques para proporcionar información detallada sobre la ejecución de los procesos que ayudan en la toma de decisiones [29].

1.8.7.2 Tipos de Monitoreo

Existen dos tipos de monitoreos:

I. La monitorización activa se lleva a cabo mediante técnicas BAM, que proporcionan monitoreo en tiempo real de los indicadores críticos de rendimiento empresarial a partir de las instancias de los procesos de negocio [30][31][32], aplicando técnicas de Procesamiento de Eventos Complejos (*Complex Event Processing* - CEP). CEP es un conjunto de teorías y tecnologías que permiten procesar los flujos de eventos de diferentes fuentes en tiempo real para correlacionarlos con los demás logrando eventos de alto nivel y una derivación de ellos [26].

II. La monitorización pasiva se realiza mediante la minería de procesos, que es una disciplina formada por el aprendizaje automático, la minería de datos y por el modelado y el análisis de procesos. Las técnicas de minería de procesos consisten en extraer conocimiento a partir de los registros disponibles en los sistemas de una organización. La minería de procesos establece enlaces entre los procesos y sus datos, y los modelos de procesos [33]. Puede considerarse como otra colección de técnicas de BI (*Business Intelligence*) que se centra en la consulta y presentación de informes combinados con técnicas de visualización que muestran cuadros de mando y de resultados, utilizando técnicas de minería de datos o mediante OLAP (*Online Analytical Processing)* [8]. Las técnicas de BI ayudan a realizar el análisis necesario para detectar fallos, desviaciones o ineficiencias en los procesos ejecutados, para luego mejorarlos en las fases de Definición y Diseño, iterando sobre el ciclo de mejora continua de los procesos de negocio.

Capítulo II – Análisis comparativo de herramientas BPMS y su elección

Una de las decisiones más importantes que debemos tomar en el marco de este trabajo, es determinar que tecnología BPM vamos a utilizar para implementar la integración con la herramienta de Gestión de Incidencias.
Por lo tanto, en este capítulo tomaremos dos tecnologías BPM y estaremos realizando un análisis sobre las mismas de acuerdo a ciertos criterios de comparación o parámetros que consideramos necesarios para poder llevar adelante dicha integración.
A continuación, se detallan las secciones a desarrollar:

- En la Sección 2.1 definiremos los criterios a utilizar en la comparación tecnológica BPM.
- En la Sección 2.2 describiremos el análisis comparativo de las tecnologías BPM.
- En la Sección 2.3 estaremos justificando el porqué de la tecnología seleccionada.
- Por último, en la sección 2.4 daremos una breve introducción acerca la herramienta BPM seleccionada.

2.1 Definición de parámetros de comparación

En base a la integración a desarrollar en este trabajo, a continuación, definimos los criterios principales de comparación a tener en cuenta a la hora de evaluar las tecnologías BPM:

I. El tipo de licenciamiento del producto: esto es si las herramientas son gratuitas o si requiere de algún tipo de inversión para su obtención.

II. Integración con Portales de Autenticación SSO (*Single Sign On*): Los sistemas SSO permiten a los usuarios utilizar los servicios de varios sitios web en diversas aplicaciones o servicios web sin identificación y contraseña si los usuarios se autentican en un sitio web. Por lo que SSO ofrece la reducción de costos y comodidad a los administradores de sitios web y a los usuarios [69].

III. Capacidad de distribución de procesos en sistemas externos.

IV. Administración de usuarios y roles de cada herramienta.

V. Componentes nativos para interoperabilidad con sistemas externos, servicios y bases de datos.

VI. Documentación y ayuda: La herramienta debe proveer documentación completa y mensajes de ayuda para la corrección de errores.

2.2 Análisis comparativo de las tecnologías BPM

El análisis comparativo de las tecnologías BPM se realizará entre dos herramientas muy conocidas, que tienen un gran impacto en el mercado ya que cuentan con

implementaciones de los procesos de negocio en grandes empresas y universidades del mundo.
Las aplicaciones a ser analizadas son Bonita Open Solution y Bizagi. Ambos entornos son potentes y útiles, son totalmente distintos a nivel de desarrollo y se basan en el estándar BPMN 2.0 mantenida por el OMG (*Object Management Group*) desde el 2005 [67].

A continuación, se presenta el análisis comparativo de las aplicaciones BPM de acuerdo a los parámetros de comparación definidos en la Sección 2.1.

2.2.1 Licencias

Bonita Open Solution: La aplicación está desarrollada en Java y dispone de una licencia GNU GPL v2 para la versión Community, por lo que podemos hacer uso de la herramienta para realizar nuestros propios desarrollos sin necesidad de realizar ningún pago. Adicionalmente, la empresa BonitaSoft cuenta con otras tres ediciones diferentes, bajo la modalidad de suscripción: Teamwork, Efficiency y Performance la cual posee utilidades adicionales para los desarrolladores que agilizan la producción y permite un control más exhaustivo de los procesos [51].

Bizagi: esta herramienta consta de tres módulos:

- Bizagi Modeler: permite modelar y documentar procesos de negocio.
- Bizagi Studio: permite convertir diagramas de Proceso en aplicaciones ejecutables (workflow).
- Bizagi Engine: Permite ejecutar y controlar los procesos de negocio ya desplegados en los módulos Modeler y Studio.

Bizagi Modeler y Bizagi Studio pueden utilizarse de manera gratuita. Bizagi Studio soporta un máximo de veinte Usuarios en ambientes de desarrollo y pruebas [69].

2.2.2 Integración con portales de autenticación SSO (*Single Sign On*)

Bonita Open Solution: en la versión Community solo está disponible el conector LDAP. Ediciones de subscripción Teamwork, Efficiency y Performance: está disponible la posibilidad de la sincronización de los datos de usuarios. [69]

Bizagi: en el módulo Studio existe un servidor de LDAP configurado en la entidad donde se ha instalado Bizagi, este se sincroniza con este servicio. [69]

2.2.3 Capacidad de distribución de procesos en sistemas externos

Bonita Open Solution: En todas las ediciones, a través de las extensiones API REST esta herramienta permite que sus recursos puedan ser exportados, importados, modificados y eliminados en el Bonita BPM Portal [69].

Bizagi: el módulo Modeler expone servicios webs para poder integrar esta herramienta hacia otros sistemas, sin embargo, es posible que otros sistemas desearan integrarse con el BPM. Para esto, Bizagi cuenta con su Capa de Integración Orientada a Servicios. En forma automática, cada proceso está disponible para ser invocado mediante servicios webs cuyo entrada y salida son documentos XML. Las aplicaciones que invocan la capa de servicios Bizagi pueden llevar a cabo tareas como la creación de nuevas instancias de

procesos, ejecutar actividades del proceso, ejecutar eventos, consultar o adicionar información sobre el proceso. Desde el punto de vista de arquitectura de sistemas, Bizagi es compatible con arquitecturas SOA, en la que las aplicaciones exponen sus principales funcionalidades como servicio para lograr integraciones entre sistemas que corren sobre plataformas heterogéneas. [69]

2.2.4 Administración de usuarios y roles de cada herramienta

Bonita Open Solution: En esta herramienta la creación de usuarios, grupos y roles es muy sencilla y se lo realiza de una manera ordenada ya que se crea una organización y de ahí los roles, grupos y luego los usuarios que participan dentro de cada proceso. Por lo tanto, de esta manera se puede crear varias organizaciones en la misma herramienta.

Bizagi: Esta aplicación es un poco más restringida la creación de usuarios. En el desarrollo se permite crear todos los usuarios que se necesitan, pero se lo crea por cada proceso, es decir que en cada proceso se tiene que crear el mismo usuario. A su vez, como la licencia limita el número de usuarios en el desarrollo se tiene que verificar el número de usuarios que van a utilizar el proceso.

2.2.5 Componentes nativos para interoperabilidad con sistemas externos, servicios y bases de datos

Bonita Open Solution: los conectores es una de las virtudes más fuertes de esta herramienta. Ademas, la herramienta al estar desarrollada en Java posee una gran comunidad que se dedica a crear conectores para la comunicación de Bonita con otras aplicaciones mediante Web Services, .jar, crear nuestros propios conectores, crear documentos, informes Jasper, conexiones a cualquier Base de Datos, etc.

Bizagi: esta herramienta es un poco más limitada en sus conectores, pero también permite incorporar componentes (librerías .dll) y también la comunicación mediante web services.

2.2.6 Documentación y Ayuda

Bonita Open Solution: es una herramienta muy popular, las actualizaciones de las mismas son constantes. Tiene una comunidad muy activa la misma se puede ver en sus foros, documentación muy completa y videos de tutoriales disponibles, tiene también un curso Online gratis [69].

Bizagi: tiene una comunidad muy activa donde hay un foro por cada módulo de la misma, proporciona una documentación muy completa (proporciona una guía de usuario muy completa), tutoriales en videos, la actualización de versiones es muy activa agregando nuevas funcionalidades y realizando mejoras al producto [69].

2.3 Justificación de la herramienta seleccionada

Como conclusión, una vez finalizado el análisis comparativo entre los entornos Bonita Open Solution y Bizagi vemos que son herramientas muy completas para la gestión de los procesos de negocio, pero a continuación, daremos una explicación de las razones por

las cuales optamos en elegir utilizar el software BonitaSoft en su versión 7.3.3 Community para implementar la integración con el sistema de Gestión de Incidencias:

- Para obtener una licencia del producto no es necesario realizar ningún pago y no existe restricciones en el momento de realizar desarrollos.
- La plataforma para su ejecución, al estar desarrollado en Java puede ejecutarse en cualquier plataforma.
- Permite la sincronización con un sistema SSO.
- El mecanismo de actualización de modelos de procesos es resuelto utilizando el repositorio SVN para el manejo de versiones.
- En la actualización de instancias en los procesos tiene en cuenta la posibilidad de cambios en la instanciación de los procesos.
- Brinda conectores nativos a bases de datos como HSQL, PostgreSQL, MySql, Oracle, SQL Server.
- La conexión con sistemas externos es uno de los puntos más fuertes de BonitaSoft y la facilidad de crear nuestros propios conectores y agregarlos a nuestros procesos.
- La ayuda para la creación de los procesos es muy fácil de encontrar ya que existe una comunidad de usuarios de la herramienta.
- Cuenta con una API REST completa para poder realizar integraciones con sistemas externos.

2.4 Breve descripción de la herramienta BonitaSoft

Esta suite es un proyecto de software fundado en 2009 por la empresa Bonitasoft. Bonitasoft es de código abierto [34], el producto tiene un gran crecimiento en descargas (más de 3 millones) con la atención de más de 1000 clientes en más de 60 países y cuenta con más de 150 empleados, localizados a nivel internacional. Una de sus principales características es que cubre los objetivos de cualquier BPMS de la actualidad, es decir, es social, colaborativo y no tiene la necesidad de realizar demasiada codificación. Se destaca por la facilidad de su utilización, presenta un diseño intuitivo de sus componentes, se basa en la notación estándar BPMN 2.0, es simple para integrarse con aplicaciones externas, permite manejar una gran cantidad de usuarios y existe una gran cantidad de ejemplos en la web que pueden ayudar a construir rápidamente aplicaciones BPM.

Bonita BPM tiene dos componentes principales:

1. Bonita BPM Studio es la herramienta de desarrollo para definición y diseño de los procesos de negocio.

2. Bonita BPM Platform es la plataforma de ejecución de los procesos de negocio, o sea, es el entorno operativo de ejecución administración y monitorización.

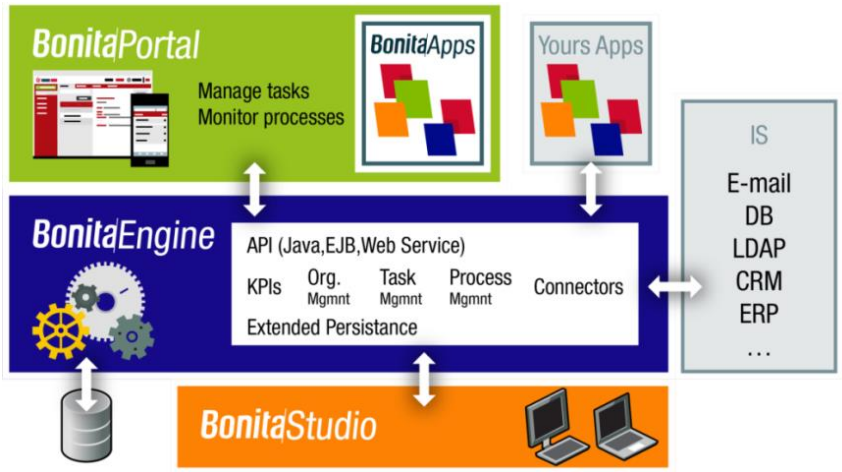

Figura 6. Componentes de Bonitasoft [48]

2.4.1 Bonita BPM Studio

Bonita BPM Studio es la herramienta gráfica para la definición y diseño de procesos de negocio utilizada por los analistas de negocio y los desarrolladores de aplicaciones.
Esta herramienta contiene varios componentes que permiten el modelado y diseño de las diferentes características que se deben incorporar a los procesos de negocio utilizando una notación BPMN 2.0.
Bonita BPM Studio tiene dos herramientas de diseño principales:

- La pizarra: se utiliza para dibujar un diagrama de flujo del proceso y definir el detalle de los pasos, transiciones, los puntos de decisión y otros elementos del proceso.

- El constructor de formularios: se utiliza para crear formularios utilizados en aplicaciones web de procesos.

Bonita BPM Studio también cuenta con características para la gestión, publicación, importación y exportación del modelo organizacional, en el que se pueden definir grupos, roles, usuarios y asociaciones de grupos y usuario a través de roles. Esto permite flexibilidad en la estructura de la organización.
A nivel de diseño, Bonita BPM Studio permite definir, gestionar, implementar, importar y exportar conectores personalizados, scripts desarrollados en Java, Groovy, filtros de actores, el modelo de datos de negocio y gestionar librerías.jar.

2.4.2 Bonita BPM Platform

Este componente está formado por dos componentes fundamentales, que interactúan entre sí, basándose en componentes externos para ejecutar los procesos de negocio definidos con Bonita BPM Studio:

- Bonita BPM Engine. Es el motor de ejecución de los procesos de negocio y es transparente para los usuarios.

- Bonita BPM Portal. Es la interfaz de usuario web. Este componente es usado por los usuarios de procesos de negocio que la utilizan para ver sus tareas e

interactuar con la plataforma. Esta herramienta también es utilizada por los administradores de procesos para instalar, desplegar y gestionar procesos.

Bonita BPM Platform es un portal web para facilita a los usuarios la interacción con el motor de procesos. En este componente dos perfiles: Usuario y Administrador. Ademas, proporciona una sección para la administración y configuración de la plataforma a través del portal web, que presenta la misma estructura visual que para los usuarios.

Adicionalmente, Bonita BPM Plataform cuenta con:

- Un Servidor de Aplicaciones. Por defecto, tiene configurado un servidor de aplicaciones Tomcat. Es recomendable, en ambientes de producción, utilizar servidores JBoss o Tomcat.

- Una Base de Datos. Por defecto, hace uso de una base de datos H2. Esta base de datos es utilizada en entornos de pruebas, en cambio, en ambientes de producción es recomendable utilizar sistemas de bases de datos más potentes que es soportada por la herramienta.

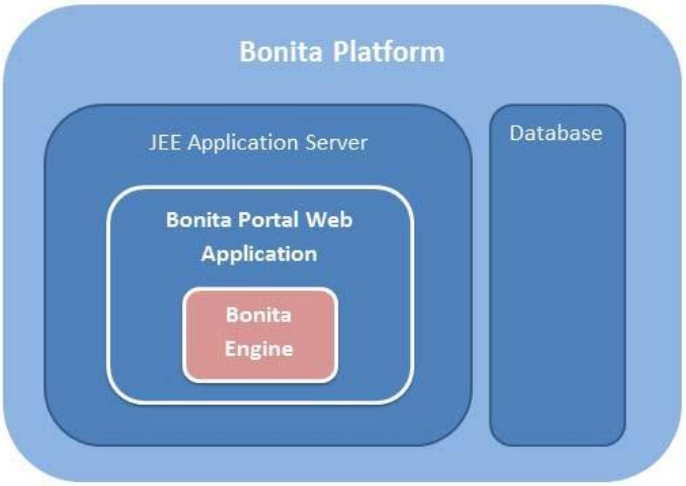

Figura 7. Arquitectura de Bonita BPM Platform

Capítulo III – Gestión de Incidencias

El segundo pilar de este trabajo tiene como finalidad poder identificar y seleccionar una herramienta para poder gestionar incidencias. Este producto nos tiene que brindar la posibilidad de minimizar los períodos de fuera de servicios de TI, registrar la información relevante de todas las incidencias que puedan ocurrir, incorporar las mejores prácticas de gestión de forma satisfactoria; y desde el punto de vista tecnológico, la herramientea deberá proveer una API de Servicios y contar con componentes o conectores nativos de integración compatibles con la herramienta BPM seleccionada Bonitasoft en la Sección 2.3.

En la actualidad, casi ningún entorno BPM cuenta con la capacidad de administrar incidencias en la Fase de Monitorización (Sección 1.8.7).
Tomando como premisa esta debilidad en la Fase de Monitorización del Ciclo de Vida BPM, al momento de lograr una integración con un Gestor de Incidencias contaremos con el potencial de detectar, clasificar, escalar, solucionar y cerrar incidentes identificados por personas técnicas y de negocios.
De acuerdo a lo definido en la Sección 1.8.7.1 Perspectivas de monitoreo, el personal técnico registrará incidencias relacionadas a fallos producidos en los sistemas que soportan la arquitectura empresarial y procesos de negocio; mientras que los analistas de negocios registrarán sugerencias de los procesos de negocio y todos los posibles problemas que ocurren durante su ejecución.

A continuación, se indica la temática a seguir para el capítulo:

- En la Sección 3.1 definiremos conceptos generales relacionados a la gestión de incidencias.
- En la Sección 3.2 analizaremos los marcos metodológicos ITIL v3, COBIT y CMMI utilizados para la gestión de incidentes.
- Luego, en la en la Sección 3.3 describiremos una serie de recomendaciones para la gestión de incidencias basándonos según lo definido en la Sección 3.2.
- En la Sección 3.4, Elección de la herramienta para la gestión de incidencias
- En la última sección de este capítulo, Sección 3.5, desarrollaremos una breve descripción de la herramienta seleccionada.

3.1 Conceptos básicos

3.1.1 Definición de Incidencia

Una incidencia es una interrupción no planificada o una reducción de calidad de un servicio de TI. El fallo de un elemento de configuración que no haya afectado todavía al servicio también se considera una incidencia. [35]

3.1.2 Definición de Problema

Un problema esla causa desconocida de uno o más Incidentes. Por lo regular, desconoce la causa al momento de crear un registro de problema y el proceso de la gestión de problemas es responsable de continuar con la investigación. [71]

3.1.3 Definición de Servicio

Un servicio es un medio para entregar valor a los clientes al facilitar los resultados que deseen obtener, sin la propiedad de costos y riesgos específicos. [71]

3.1.4 Definición de Valor

Valor es el aspecto esencial del concepto de servicio. Desde el punto de vista del cliente, el valor consta de dos componentes básicos: utilidad y garantía. La utilidad es lo que el cliente recibe, mientras que la garantía reside en cómo se proporciona. [71]

3.1.5 Definición de Gestión de Servicio

La gestión de servicios es un conjunto de capacidades organizativas especializadas cuyo fin es generar valor para los clientes en forma de servicios. [71]

3.1.6 Definición de Solicitud de servicio

Es una petición formal por parte de un usuario para que algo sea provisto. Las solicitudes de servicios son gestionadas por el proceso de cumplimiento de solicitud en conjunto con la mesa de servicios. Las solicitudes de servicio pueden estar vinculadas con una solicitud de cambio como parte del cumplimiento de la solicitud [36].

3.1.7 Definición de Acuerdo de Nivel de Operación

OLA (*Operation Level Agreement*) es un documento interno de la organización donde se especifican las responsabilidades y compromisos de los diferentes departamentos de la organización TI en la prestación de un determinado servicio. [72]

3.1.8 Definición de Acuerdo de Nivel de Servicio

SLA (*Service Level Agreement*) es un acuerdo suscrito entre cliente y proveedor donde se especifican los detalles de los servicios brindados. Debe tener definidos los siguientes aspectos: descripción, disponibilidad, niveles de calidad, tiempo, etc. [72]

3.1.9 Definición de Error

Es un incidente o problema para el cual la causa raíz es conocida y para el cual se ha identificado una solución permanente o temporal. [72]

3.1.10 Definición de Impacto

Es el grado de desviación sobre la operativa normal, en términos de número de usuarios o de procesos del negocio afectados por un incidente. [72]

3.1.11 Definición de Urgencia

Es la demora aceptable para el usuario o el proceso del negocio, para resolver un incidente. [72]

3.1.12 Mesa de Ayuda

La Mesa de Ayuda es un área en cualquier organización que ayuda a sus usuarios a encontrar respuesta a sus preguntas o soluciones a sus problemas. Estos usuarios son empleados de la misma organización que necesitan ayuda con respecto al software de sus computadoras, acceso a la red, problemas de impresión o cualquier otro problema técnico. [71]
La Mesa de Ayuda esta conformado por recursos humanos (personal especializado), tecnología y procesos, siendo una "Unidad Funcional", y no un proceso.

3.1.13 Mesa de Servicio

La Mesa de Servicio es el único punto de contacto entre los usuarios de la organización y la oficina de servicios de TI (también denominada gestión de servicios de TI). En su concepción más moderna, debe funcionar como centro de todos los procesos de soporte al servicio y adicionalmente, debe jugar un papel importante dando soporte al negocio identificando nuevas oportunidades en sus contactos con usuarios y clientes. [72]

3.1.14 Elemento de configuración

Es cualquier componente de servicio que debe ser gestionado con el fin de entregar un servicio de TI.
La información de cada elemento de configuración se almacena en un registro de configuración dentro del sistema de gestión de configuración y este se mantiene a lo largo de su ciclo de vida por la gestión de activos de servicio y configuración. Los elementos de configuración están bajo el control de la gestión del cambio.
Un elemento de configuración incluye servicios de TI, hardware, software, edificios, personas y documentación formal como documentación del proceso y los SLA [36].

3.1.15 Base de datos para la gestión de la configuración

Es una base de datos utilizada para almacenar los registros de configuración a lo largo de su ciclo de vida. El sistema de gestión de la configuración mantiene una o más bases de datos y cada base de datos almacena los atributos de los elementos de configuración junto con las relaciones a otros elementos de configuración [35].

3.1.16 Base de errores conocidos

Es una base de datos que tiene como finalidad almacenar el conocimiento sobre las incidencias o problemas registrados junto con sus resoluciones, de manera que sea posible diagnosticarlos y resolverlos en menos tiempos en caso de que se vuelvan a producir [35].

3.2 Marco teórico de las metodologías ITIL V3 2011, COBIT 5 y CMMI

En el marco teórico, se tomarán únicamente como referencia las buenas prácticas de gestión de incidentes contenidas en ITIL V3 2011, COBIT 5 y CMMI para servicios [35] [40][41].

3.2.1 ITIL V3

ITIL en la definición del proceso Operación del Servicio, en el subproceso 4.2 Gestión de Incidentes [37] explica las buenas prácticas para que el manejo de las incidencias sea exitoso.
ITIL maneja la gestión de incidentes dentro del ciclo de vida de la operación de servicio, en el mismo que brinda recomendaciones para el manejo de problemas [70].
Dentro de la terminología de ITIL, un incidente se define como una interrupción no planificada o la reducción de la calidad de un servicio de TI.

A continuación, se definen el conjunto de pasos para la gestión de incidencias especificadas por la metodología:

I. El proceso recomendado para una correcta gestión de incidentes inicia por la identificación del incidente. En este paso se debe definir cómo detectarlo, priorizarlo y categorizarlo.

II. Como segundo paso, se debe registrar el incidente. Todos los incidentes deben ser registrados con su información completa para mantener datos históricos de los mismos. La información a registrar de cada incidente debería estar compuesta por: número único de referencia, categoría, urgencia, impacto, prioridad, hora y fecha, nombre y/o ID de la persona que registró el incidente, método de la notificación, modo de contacto para la respuesta, descripción de los síntomas, estado, problemas relacionados, actividades para la resolución del incidente, fecha de resolución, categoría de cierre y fecha de cierre. En caso de que el proceso se pasa a otro grupo el proceso de registro debe ser igual de riguroso.

III. En el tercer proceso se deben categorizar los incidentes. Todas las empresas son únicas y la definición de las categorías pueden presentar sus particularidades. Sin embargo, se brinda una técnica que puede ser usada para ayudar a las organizaciones a definir un conjunto completo de categorías, siempre y cuando se arranque desde cero.

IV. El cuarto proceso tiene como objetivo asignarle una prioridad a las incidencias, la cual va a ser importante para determinar cuándo y cómo el incidente va a ser manejado por el departamento recomendado. Por lo general, las prioridades son determinadas de acuerdo a la urgencia de la incidencia y en nivel de impacto en el negocio. Otras cuestiones que se tienen en cuenta son el número de servicios impactados, el nivel de perdida en finanzas, el efecto de reputación del negocio y regulaciones del estado o locales.

V. En el quinto proceso se va a realizar un diagnóstico inicial. Si el incidente es reportado por el departamento de servicios comúnmente mientras se tiene a la persona en el teléfono se trata de brindar un primer diagnóstico y ver si es posible solucionarlo en la llamada; en caso contrario, el receptor del incidente deberá informarle al interesado sobre cómo se procederá y deberá darle un número de incidencia para que este le pueda dar seguimiento. Luego, el receptor del incidente podrá buscar una solución para el incidente registrado. En este paso también va a ser importante que la persona encargada del soporte busque dentro de la base de datos de incidentes si el incidente reportado se encuentra registrado y visualizar posibles soluciones o como este fue resuelto anteriormente. En caso de contar con una base de conocimientos se tiene como ventaja evitar procesos de investigación redundantes ya que se tiene disponible la resolución de incidencias que fueron previamente resueltas.

VI. En el sexto proceso se describe sobre la posibilidad de escalamiento del incidente. Esta puede ser de manera funcional que requiere el apoyo de un especialista de más alto nivel para resolver el problema o jerárquico que debe involucrar a un responsable de mayor autoridad para tomar decisiones que se escapen de las atribuciones asignadas a ese nivel.

VII. En el proceso siete se recomienda la investigación y análisis del incidente. Todos los grupos involucrados en el manejo de incidencias deben investigar y diagnosticar cual fue el motivo para que ocurra la incidencia y deberán documentar en detalle toda la información relevada y también cualquier acción tomada para resolver el incidente. En el proceso de investigación se deberá tener en cuenta que fue lo que anduvo mal o que hizo mal el usuario, entender el orden de las cosas que se hicieron, confirmar el impacto del incidente, identificar si hubo eventos que pudieron dispararse debido al incidente y especificación de búsquedas detalladas de conocimiento en busca de ocurrencias anteriores mediante la búsqueda de incidentes.

VIII. En el proceso ocho se encuentra la resolución y recuperación del incidente. Cuando la solución es encontrada deberá aplicarse y probarse. Las acciones que deben llevaste a cabo y las personas que van a participar en la toma de las acciones de recuperación pueden variar dependiendo de la naturaleza de la falla, pero se podrían involucrar las siguientes acciones: pedirle al usuario que realice algunas actividades, solucionarlo desde el equipo de soporte, solicitar la solución a un grupo de especialistas o solicitar el servicio a un proveedor.

IX. Por último, en el noveno proceso se requiere el cierre del incidente. En esta etapa se debe comprobar que las incidencias estén solucionadas y esto se lo debe verificar con los usuarios involucrados. También se deberán validar los siguientes procesos: categorización, encuesta de satisfacción al cliente, documentación del incidente y verificar si el incidente es recurrente.

3.2.2 Cobit 5

Cobit permite comprender el gobierno y la gestión de las Tecnologías de Información de una organización.
En el manual de buenas prácticas *IT Governance Institute*, en la sección de Gestión y el dominio Entrega, Servicio y Soporte define un proceso llamado Gestionar Peticiones e Incidentes de Servicio que cuales describen como proveer una respuesta oportuna y efectiva a las peticiones de los usuarios y a la resolución de todo tipo de contratiempos [38].

El proceso tiene los siguientes objetivos:

- Recuperar el servicio normal
- Registrar y completar las peticiones de usuario
- Registrar, investigar, diagnosticar, escalar y resolver incidentes

Además, define cual es el propósito del proceso en el que indica que se debe lograr mayor productividad y minimizar las interrupciones mediante la rápida resolución de consultas de usuarios e incidentes.
El proceso se basa en la consecución de un conjunto de objetivos principales de TI, que se subdividen en la definición de Riesgos de Negocio de TI y en la entrega de Servicios de TI, de acuerdo a los requisitos del negocio.
Para la resolución de incidentes y peticiones, el proceso toma como base la satisfacción y los niveles de servicios acordados por el usuario.

Como prácticas clave para la gestión se tienen el siguiente conjunto de subprocesos [40]:

Subproceso 1: se definen los esquemas de:

- Clasificación de incidentes y peticiones de servicio
- Registrar, clasificar y priorizar peticiones e incidentes
- Verificar, aprobar y resolver peticiones de servicio
- Investigar, diagnosticar y localizar incidentes
- Resolver y recuperarse de incidentes
- Cerrar peticiones de servicio e incidentes y el seguir el estado
- Emitir informes

Subproceso 2: Se deben definir los esquemas y modelos de clasificación de incidentes y peticiones de servicio.
Las principales actividades a realizar son las siguientes:

- Definir los esquemas de clasificación y priorización de incidentes, reclamos y criterios de servicio en cuanto al registro de problemas.
- Se deben asegurar enfoques consistentes para el tratamiento de incidencias, informar a los usuarios y realizar análisis de tendencias.
- Definir modelos de incidentes para errores conocidos teniendo como finalidad facilitar su resolución eficiente y efectiva.
- Definir modelos de peticiones de servicio según el tipo de petición para facilitar la “auto-ayuda” y las peticiones estándar.

- Definir reglas y procedimientos de escalado de incidentes importantes y de seguridad.
- Fijar fuentes de conocimiento de incidentes e instancia y su uso.

Subproceso 3: para el subproceso identificar, registrar y clasificar peticiones de servicio e incidentes se definen las siguientes actividades principales:

- Registrar todos los riesgos y peticiones de servicio.
- Registrar la información relevante de forma que pueda ser manejada de manera efectiva y mantener un registro histórico que posibilite el análisis de tendencias, clasificar incidentes y peticiones de servicio e identifique el tipo y categoría.
- Priorizar peticiones de servicio e incidentes, según la definición de impacto en el negocio del ANS y la urgencia.

En este subproceso se deben seleccionar los procedimientos adecuados para el manejo de las peticiones y verificar que se cumplan los criterios de solicitud definidos. En las actividades se tiene que verificar que las autorizaciones sean las adecuadas para realizar peticiones de servicio usando un flujo de proceso predefinido. Para esto es necesario seguir con el procedimiento del reclamo seleccionado o utilizar opciones automáticas o modelos predefinidos para los elementos solicitados con frecuencia.

Subproceso 4: Terminado el paso anterior, se deben:

- Identificar y registrar los síntomas de incidentes.
- Determinar las posibles causas y asignar recursos para su resolución. Para eso, se necesita tener definidas las actividades de identificación y descripción del síntoma en detalle para poder establecer las causas del mismo.

En esta instancia, se deberá tener en cuenta y hacer referencia a:

- Los recursos de conocimiento disponibles (incluyendo errores y problemas conocidos) para identificar posibles resoluciones de incidentes.
- Registrar un nuevo problema si está relacionado o si es un error conocido que aún no exista.
- En caso de que el incidente satisface los criterios acordados para el registro de problemas se debe asignar a funciones específicas.

Subproceso 5: Una vez finalizado el subproceso 4 se debe comenzar con las tareas de documentar, solicitar y probar las soluciones identificadas o temporales, como así también ejecutar las acciones de recuperación para restaurar el servicio TI relacionado.
En esta instancia se deben desarrollar las siguientes actividades:

- Seleccionar y aplicar las resoluciones de incidentes más apropiadas (soluciones provisionales y/o soluciones permanentes).
- Registrar si se usaron soluciones temporales para resolver los incidentes.
- Ejecutar acciones de recuperación.
- En caso de ser requerido, se debe documentar la resolución del incidente y evaluar si se pudiera usar como una fuente de conocimiento en el futuro.

Subproceso 6: Luego de esto, se realizan dos actividades:

- Verificar la correcta resolución de los incidentes.
- Realizar el cierre de incidentes.

Las dos actividades mencionadas requieren ser llevarlas adelante con los usuarios afectados para corroborar que la petición de servicio se completó correctamente o el incidente se resolvió de manera satisfactoria: Luego de obtener la aceptación por parte del usuario las peticiones de servicio e incidentes son cerrados.

Subproceso 7: Por último, se debe hacer un seguimiento para analizar e informar las incidencias y tendencias de cumplimiento de peticiones con el objetivo de proporcionar información para la mejora continua y actividades de supervisión. Para lograrlo esto se debe hacer un seguimiento escalado sobre los incidentes, las resoluciones y los procedimientos de gestión de resoluciones. Adicionalmente se debe utilizar la información como entrada para la planificación de la mejora continua, distribuir informes relacionados al tiempo de resolución o proporcionar acceso controlado a datos en línea.

3.2.3 CMMI

Las buenas prácticas del modelo se centran en actividades para proveer servicios de calidad a clientes y usuarios finales. CMMI integra un conjunto de conocimientos que son esenciales para proveedores de servicios. Estos modelos los desarrollan equipos del producto con personas procedentes de la industria, del gobierno y del Instituto de Ingeniería de Software [41].

La definición de resolución y prevención de incidencias busca asegurar que las incidencias del servicio se resuelvan a tiempo y de forma eficaz y que las incidencias dentro del servicio se puedan evitar según sea apropiado [42].

Para CMMI las incidencias son eventos que si no se tratan pueden causar el incumplimiento de los compromisos de servicio adoptados por la organización proveedora, por lo tanto, ésta última debería tratar a las incidencias de una manera oportuna y eficaz de acuerdo a los términos que se establecieron. Es importante identificar y tratar las causas subyacentes que son la condición o el evento el cual contribuye a que una o más incidencias ocurran. No todas las causas subyacentes tienen como resultado incidencias de forma inmediata.
Las prácticas que propone CMMI se categorizan en tres objetivos principales [41]:

I. Preparar la resolución y prevención de incidencias

Lo primero que debemos hacer es establecer un enfoque para la resolución y prevención de incidencias en donde se describen funciones organizativas en base a resolución y prevención de incidencias, procedimientos, herramientas de soporte y asignación de responsabilidades durante el ciclo de vida de una incidencia.
A continuación, se enumeran las subprácticas a desarrollar asociadas a este proceso:

- Definir criterios para determinar qué es una incidencia.
- Definir categorías para las incidencias, así como también criterios para determinar a qué categorías pertenece cada incidencia.

- Describir cómo se asigna y transfiere la responsabilidad del procesamiento de las incidencias.
- Identificar uno o más mecanismos para que los clientes y usuarios finales puedan reportar incidencias.
- Definir métodos y adquirir herramientas a utilizar para la gestión de incidencias.
- Describir el modo de notificar a todos los clientes y usuarios finales que puedan estar afectados por una incidencia reportada.
- Definir criterios para determinar los niveles de gravedad y prioridad.
- Identificar requisitos en el acuerdo de servicio acerca del tiempo disponible que está definido para resolver las incidencias.
- Documentar los criterios que definen cuando deberían cerrarse las incidencias.

Luego, se debe seleccionar un sistema de gestión de incidencias que permita procesar y seguir la información de las incidencias, administrar datos históricos de incidencias, causas subyacentes de incidencias, enfoques conocidos para tratar incidencias y soluciones temporales para dar soporte a incidencias [43].

II. El segundo objetivo es identificar, controlar y tratar a cada incidencia

Se centra en gestionar las incidencias individuales a medida que ocurren con el objeto de restaurar el servicio o de resolver las incidencias tan rápido como sea posible.
Tratar a las incidencias puede incluir recopilar y analizar datos para buscar incidencias potenciales o simplemente esperar a que los usuarios finales o clientes reporten incidencias. Dentro de este objetivo se espera identificar, registrar y categorizar incidencias junto con su información descriptiva.
Luego se deben analizar los datos de cada incidencia para determinar el curso de acción a tomar. Esta práctica se centra en resolver las incidencias a medida que ocurren por medio de un curso de acción que sea lo suficientemente oportuno y eficaz para cubrir las necesidades inmediatas de la petición de servicio.
Para esto, como subprácticas se sugiere:

- Analizar los datos sobre la incidencia.
- Determinar qué grupo es el más adecuado para tratar la incidencia.
- Planificar las acciones a realizar.

Como siguiente práctica se deberán resolver las incidencias siguiendo el curso de acción determinado por el análisis de cada incidencia. Las subprácticas recomendadas son:

- Tratar la incidencia utilizando el mejor curso de acción.
- Gestionar las acciones hasta que el impacto de la incidencia llegue a un nivel aceptable.
- Registrar las acciones y el resultado.
- Revisar las acciones realizadas que dieron como resultado cambios.

La siguiente recomendación es monitorizar el estado de las incidencias hasta su cierre, a lo largo de toda la vida de la incidencia. Su estado debería registrarse, seguirse, escalarse según se necesite y cerrarse. Las subprácticas recomendadas son:

- Documentar las acciones y monitorizar.
- Escalar las incidencias según se necesite.
- Revisar la resolución y confirmar los resultados con las partes interesadas relevantes.
- Cerrar las incidencias que cumplen los criterios de cierre.

Como última práctica dentro de este objetivo es comunicar el estado de las incidencias. La comunicación con la persona que reportó la incidencia y posiblemente a quienes se vean afectados por esta incidencia deberían tenerse en cuenta a lo largo de toda la vida del registro de la incidencia en el sistema de gestión de incidencias.

III. Analizar y tratar las causas e impactos de las incidencias seleccionadas y buscar la reducción de este o las ocurrencias de incidencias futuras

En este objetivo se manejan los términos causa subyacente y causa raíz [41].
La primera práctica que se sugiere es analizar las incidencias seleccionadas y a su vez las causas subyacentes. El propósito de realizar el análisis causal de las incidencias es determinar el mejor curso de acción para tratar las incidencias a futuro, de forma que sus impactos se minimicen con mayor eficacia. Como subprácticas podemos encontrar:

- Identificar las causas subyacentes de las incidencias.
- Registrar información sobre las causas subyacentes de una incidencia.
- Realizar los análisis causales con las personas responsables de realizar las tareas relacionadas.
- Determinar el mejor enfoque global para abordar las incidencias seleccionadas en el futuro.

La siguiente práctica es establecer soluciones para responder a futuras incidencias. Las subprácticas recomendadas en este caso son:

- Determinar cuál es el grupo más adecuado para establecer y mantener una solución reutilizable.
- Planificar y documentar la medida útil.
- Verificar y validar la solución disponible para asegurar que trata la incidencia de forma eficaz.
- Comunicar a las partes interesadas relevantes la salida aprovechable.

Por último, se tiene la práctica de establecer y aplicar resultados con el objetivo de reducir la ocurrencia de incidencias. Después de que el análisis determine las causas subyacentes de las incidencias, si existen acciones a realizar, éstas se planifican y ejecutan. En la planificación se tiene en cuenta quién actuará, cuándo, y cómo. Como sub-práctica se deberá determinar qué grupo es el más adecuado para tratar la causa subyacente.

3.3 Recomendaciones para la gestión de incidencias

En base al análisis realizado en los procesos para la gestión de incidencias según ITIL V3 2011, COBIT 5 y CMMI para servicios de TI en la Sección 3.2, definimos un proceso general

para la gestión de incidencias. Este proceso está formado por un conjunto de fases (pasos) obtenidos desde las tres buenas prácticas de gestión estableciendo un camino bien definido para asegurar el éxito cuando se gestionan incidencias en los servicios de cualquier organización.
El proceso general se clasificó en cuatro subprocesos con nombres explicativos y concisos en su forma de operar, tal como se especifica en la figura 8.
El proceso recomendado este compuesto por cuatro procesos:

I. Definición de Políticas
II. Registro
III. Resolución
IV. Cierre

Figura 8. Etapas del proceso recomendado para la gestión de incidencias

Proceso 1. Se inicia con el proceso de Definición de Políticas donde se deberán definir los pasos iniciales y parámetros que deberán estar establecidos antes de empezar con el proceso de gestión de incidentes.

Proceso 2. El segundo proceso es el Registro de Incidentes se deberán nombrar todas las actividades para registrar las incidencias.

Proceso 3. El tercer proceso es el Resolución donde se deberá incluir todos los pasos relacionados al aseguramiento de la resolución de las incidencias.

Proceso 4. Para finalizar, en el último proceso Cierre se deberán incluir todos los procedimientos relacionados con el fin de la incidencia para un cierre exitoso.

3.3.1 Descripción del flujo de trabajo

3.3.1.1 Proceso I. Definición de Políticas

El primer subproceso inicia con la definición de los criterios para definir incidencias especificadas en ITIL y CMMI. Lo importante en esta actividad poder es diferenciar entre un incidente y una solicitud de servicio.
Cada organización debe de tener bien definido cada concepto. Por su lado, ITIL [37] recomienda señalar como incidente una interrupción no planificada de un servicio de TI o reducción de la calidad de un servicio de TI. Si bien ambos tipos de solicitudes de servicio serán reportadas a la mesa de servicio la diferencia radica en que estas no van a representar una interrupción en un servicio determinado.

En la planificación, se tendrá como actividad el definir las categorías a utilizar, por lo tanto, en el momento que ingresa una incidencia pueda ser categorizada. ITIL y CMMI describen una serie de recomendaciones para manejar un catálogo de categorías. En el proceso actual se recomienda utilizar los pasos definidos por ITIL [37] debido a la facilidad con la que se logra armar el catálogo de categorías y la forma de acoplamiento según las necesidades de la empresa donde se quiere adoptar el proceso de gestión de incidencias. Para esto, se debe crear una lluvia de ideas entre las personas involucradas

pertenecientes al área de soporte, los supervisores del área de incidentes y los administradores de problemas internos de la organización. Como resultado de la reunión se deberá contar con una lista preliminar de categorías que se pondrán a prueba por un lapso de tiempo acotado. Dentro de esta técnica, se recomienda también hacer un análisis de las incidencias reportadas durante un período de tiempo que permitirá analizar a fondo si el catalogo definido inicialmente cuenta o no con las categorías necesarias en la organización. Una vez finalizado el período de prueba, se tendrá un catálogo con las categorías de alto nivel. Luego, a partir de las categorías de alto nivel se pueden crear las categorías de bajo nivel siempre y cuando sean necesarias. Se recomienda que el catálogo de categorías definido debería pasar a análisis cada 3 o 5 meses para poder tenerlo actualizado según las necesidades del negocio.

En el paso siguiente, se deberá describir como se asigna una incidencia al grupo de soporte cuya actividad se encuentra específica tanto en COBIT como en CMMI. Adicionalmente, en CMMI se especifica que las actividades deben ser asignadas para poder colaborar con la solución de la incidencia [41]. En la descripción de la asignación se debería tener la siguiente información:

- Quien es el responsable de monitorizar y darle seguimiento al estado de las incidencias
- Quién puede escalar la incidencia
- Como se asigna y se transfiere la responsabilidad sobre todos los elementos

Luego de haber definido el catálogo de categorías, se le deben asignar prioridades a los incidentes para que tomen cierta relevancia. Para esto se tienen que definir un conjunto de prioridades y la gravedad de los incidentes. Esta práctica se encuentra en las tres metodologías mencionadas. En este caso específico, la recomendación se basa en tomar lo mejor de las tres buenas prácticas y crear una actividad que ayude a clasificar los incidentes. CMMI recomienda usar escalas numéricas del 1-5, similar a lo que propone ITIL. Lo importante es utilizar números que nos indiquen la prioridad y gravedad debido a que esta va a ser una escala bastante fácil de entender por todas las personas implicadas. La priorización de un incidente se va a determinar en base a la urgencia del incidente, es decir que tan rápido el negocio necesita la solución y el nivel de impacto que el incidente está causando. Para emplear valores numéricos es necesario usar como patrón de impacto el número de usuarios afectados según la incidencia.

Es importante también tener bien definidos los mecanismos de reportes para las incidencias, práctica que se recomienda hacer en las tres metodologías descriptas.
La recomendación planteada toma en cuenta lo positivo de las tres buenas prácticas. CMMI recomienda tener un sistema encargado específicamente para este trabajo, sin embargo, se busca tener una guía que sirva para cualquier organización sin importar si se tiene o no el software o los recursos para adquirirlo [41].
La recomendación es tener un sistema que se encargue de esto sin importar si es en papel o automatizado siempre y cuando los usuarios puedan fácilmente hacer llegar su incidencia al departamento que corresponde, asegurando que el sistema de gestión de incidencias permita almacenar, actualizar, recuperar, y reportar la información de las incidencias para resolverlas y prevenirlas. Para darle seguimiento a las incidencias a través de su ciclo de vida se debe definir un conjunto de estados ITIL propone hacer uso de los siguientes estados: abierto, en progreso, resuelto, cerrado y re-abierta [37].

3.3.1.2 Proceso II. Registro

En el segundo proceso se debe realizar la identificación de las incidencias. Para poder desarrollar este proceso se tomaron las recomendaciones de ITIL y CMMI.

El objetivo es siempre tratar a la incidencia antes de que esta ocurra. Esto se puede lograr si se hace un monitoreo constante sobre los servicios críticos de las organizaciones. La identificación de incidencias se puede realizar por medio de monitoreos, análisis de anomalías en datos recogidos, detecciones por medio de un sistema automático, reportes por formulario web o llamadas telefónicas. En el caso de tener establecido la forma o medio por el cual las incidencias van a llegar se necesitará establecer un registro de incidencias que recolecte información suficiente para dar soporte a las actividades de análisis y solución sobre las mismas.

ITIL recomienda recolectar esta información y luego almacenarla para tener un registro completo sobre la incidencia y así, en caso de que la incidencia tenga que ser enviada a futuro a otro grupo o proceso se contará con la información necesaria para agilizar el proceso evitando la generación de re-procesos. La información relacionada a un incidente es la siguiente: número de referencia, categoría, urgencia, impacto, prioridad, día y hora, nombre de quién recibió la incidencia, método de notificación, forma de contacto, descripción detallada del problema, estado de la incidencia, incidencia relacionadas, actividades que se hicieron para resolver la incidencia, fecha de resolución y categoría de cierre. En esta actividad también se realizará una categorización de la incidencia basado en las categorías previamente definidas y en base a lo recolectado para el registro del mismo se le asignará una categoría que durante el ciclo de vida de la incidencia este valor podrá ser modificado. Teniendo el incidente debidamente registrado se procede a realizar un diagnóstico inicial del mismo. [37]

3.3.1.3 Proceso III. Resolución

En el tercer subproceso, Cobit recomienda iniciar con una verificación de los derechos para realizar peticiones de servicio utilizando un flujo de proceso predefinido [38].
En esta instancia se trata de definir por medio de perfiles de usuarios los roles de usuarios que garanticen poder contar con una mejor administración en la mesa de servicio. Además, se tiene en cuenta lo que define CMMI respecto al trato de una incidencia mediante el uso del mejor curso de acción lo que nos indica que para esta actividad el personal de la mesa de trabajo deberá contar con todas las herramientas necesarias para determinar el mejor camino para la resolución del incidente [41].
Para la resolución de incidencias ITIL especifica muy a detalle que dependiendo de la naturaleza de la falla pueden variar las acciones específicas a tomar como también el personal a cargo, por lo que se recomienda tener en cuenta el listado de acciones del cómo llevar a cabo ese soporte para obtener una correcta resolución. [38]
En el caso de que la incidencia no pueda ser resuelta se deberá realizar un proceso de escalamiento. Cobit y CMMI hacen mención a esta actividad, pero no está bien especificada por lo que se tomará como base la recomendación de ITIL que define dos niveles de escalamiento: el funcional y el jerárquico.
Para el escalamiento funcional y considerando que la organización cuenta con grupos de soporte funcionales con mayor experiencia; un participante encargado de solucionar incidencias puede escalarlas al nivel funcional adecuado. Si la incidencia requiere de un nivel de conocimiento más profundo o un grupo de soporte no pudo resolverla dentro de los tiempos establecidos entonces el incidente debe ser escalado inmediatamente al

siguiente grupo en nivel funcional. Es importante saber que las reglas de escalonamiento y el manejo de las incidencias deben ser acordados en los OLA con cada grupo de soporte. Algunos incidentes pueden requerir múltiples grupos de soporte para ser resueltos. Los grupos de apoyo pueden ser internos o de terceros (proveedores de software, fabricantes de hardware o personal de mantenimiento). Es importante tener en cuenta que el incidente permanece en la mesa de soporte sin importar donde haya sido referido para su resolución siendo la mesa de soporte sigue siendo la responsable del incidente en todo momento, dando seguimiento del progreso, mantener a los usuarios informados y en última instancia hacer el cierre de la incidencia.

En el escalonamiento jerárquico si hay incidentes que sean de mucha importancia (incidentes de prioridad alta) los administradores apropiados de TI deben ser notificados al menos para propósitos informativos. Este tipo de escalonamiento se da cuando la mesa de servicio no puede resolver el incidente, por lo tanto, el incidente se traslada hacia los niveles de mando superiores para que estén enterados y puedan tomar las acciones necesarias.

Continuando con el proceso, se recomienda incluir también lo que describe Cobit en cuanto a completar las peticiones siguiendo el procedimiento de petición seleccionado utilizando cuando sea posible menús automáticos de auto-ayuda y los modelos de petición predefinidos para los elementos solicitados con mucha frecuencia. En el momento que la base de datos de conocimiento de la organización tenga un nivel de madurez aceptable se podrán implementar los formularios y menúes de auto-ayuda obteniendo como beneficio la rápida resolución de incidencias conocidas. [38]

Por último, se recomienda la última actividad propuesta por ITIL en donde se menciona que en caso de que la resolución del incidente haya sido resuelta por otro grupo de soporte de la organización, éste deberá pasar de nuevo el incidente a la mesa de servicio para que se proceda con la acción de cierre de la incidencia [37].

3.3.1.4 Proceso IV. Cierre

Para el último proceso se recomienda utilizar las actividades definidas en ITIL, ya que Cobit las menciona, pero no con el nivel de detalle deseado.

ITIL indica en sus buenas prácticas que la mesa de servicio debe realizar la comprobación de que la incidencia está resuelta y que los usuarios están satisfechos y dispuestos a aceptar el cierre de la misma. En esta actividad se deben tener en cuenta a todos los involucrados en el incidente y además se tendrán que realizar las revisiones y comprobaciones acerca de las decisiones tomadas en la mesa de servicio respecto a la categorización de la incidencia e indicar los pasos a seguir en caso de que esa categorización haya sido incorrecta. [37]

A continuación, se deberán generar encuestas para comprobar la satisfacción del cliente por medio de una llamada telefónica o el envío de un correo electrónico, verificar que la documentación de la incidencia esté correctamente documentada. Si la indecencia informada ocurre con frecuencia se deberá estudiar con los grupos de soporte si el incidente se resolvió sin determinar la causa raíz, por lo tanto, se necesitará generar una cantidad mayor de medidas preventivas para evitarlo. Si no se creó un expediente del problema en cuestión, se deberá generar un registro del problema con el proceso de administración de problemas para reiniciar la acción preventiva.

Como último paso, se deberá cerrar el registro de la incidencia. Algunas organizaciones pueden optar es por el uso de un período de cierre automático de incidencias. Esta acción deberá ser discutida, acordada y publicada con los usuarios, para que todos estén conscientes de esto.

Como última actividad recomendada y presentada únicamente por ITIL es la definición de reglas para reabrir incidencias. En ciertas ocasiones las incidencias por más cuidado que se tenga se repiten a pesar de que han sido cerradas. Para este caso se debe recolectar toda la información relacionada al trabajo realizado para la resolución de la incidencia. Para proveer la opción de reapertura de incidencias se recomienda establecer reglas claras, predefinidas y documentadas sobre cuando un incidente puede ser reabierto. Una vez que todas reglas se encentran establecidas es importante comunicárselas a todo el personal de la mesa de servicio para que puedan aplicarlas de forma correcta.

3.4 Elección de la herramienta para la gestión de incidencias

El software Atlassian Jira será la herramienta que estaremos utilizando como Gestor de Incidencias para implementar la integración dentro del entorno de Bonitasoft en el marco del trabajo.

Para la elección de dicho software nos basamos en los siguientes fundamentos:

- En una primera instancia, tomamos los fundamentos teóricos relacionados al análisis de las buenas prácticas y recomendaciones basado en ITIL, Cobit y CMMI definidos en las Secciones 3.2 y 3.3.
- En segundo lugar, analizamos las características tecnológicas y funcionales de la herramienta teniendo en cuenta una visión tecnológica y de integración con sistemas externos.

A continuación, enumeramos los puntos de mayor importancia:

- Las características de la herramienta. Es una solución basada en la web para gestionar tareas y permite la gestión operativa de proyectos, es una herramienta orientado a incidencias, adaptable y configurable, dispone de una interfaz de usuario intuitiva, permite elaborar informes, cuenta con una base de conocimientos, permite gestionar flujos de trabajo y control de versiones, permite manejar la integración continua, es compatible con dispositivos móviles y admite acoplar otros productos de Atlassian como Jira Service Desk, Zephyr y Hipchat.
- Atlassian Jira puede adoptar el modelo de proceso para la gestión de incidencias planteado en la Sección 3.3.
- Un punto de gran importancia es esta herramienta ofrece una API donde expone todas las funcionalidades necesarias para poder integrarse con sistemas externos.

- Gestión de Usuarios. La herramienta brinda de sincronización con *Active Directory*. Permite a los usuarios finales generar cuentas de inicio de sesión de *ServiceDesk* o restringir la generación de cuentas de usuario final a los administradores u operarios de Mesa de Ayuda. Puede importar en *NetSupport* ServiceDesk departamentos de usuario y compañías de otros sistemas. Se pueden asignar operarios de Mesa de Ayuda a compañías y departamentos específicos en la jerarquía de usuario y definir previamente a los miembros para cada categoría.

- Crea y gestiona repositorios Git, configura permisos y colabora en el código fuente de forma segura, rápida y robusta, en nuestros propios servidores. Las claves de tareas de JIRA mapean automáticamente las tareas con el código fuente. Se pueden ver los cambios en el código, supervisar el progreso del trabajo y navegar de tareas al código con un solo clic. Incluso se puede conectar JIRA a GitHub Enterprise con el Conector DVCS gratuito.

- Complementos. Hay más de mil complementos que mejoran JIRA, Confluence y el resto de nuestras herramientas. Se pueden encontrar todos en el Atlassian Marketplace. NetSupport ServiceDesk cumple todos los requisitos como instrumento de gestión de problemas e incidentes completamente compatibles con la funcionalidad de ITIL, y a la vez se puede integrar con facilidad con NetSupport DNA un conjunto de programas de gestión de activos de TI. Ideal para la pequeña y mediana empresa, y completamente escalable para grandes empresas, NetSupport ServiceDesk ha sido desarrollado para minimizar el período de inactividad de sistemas a la vez que ofrece un soporte eficaz para todos los usuarios.

- Ampliaciones y notificaciones. Envía notificaciones por correo electrónico utilizando las plantillas de correo electrónico personalizables. Ampliación automática de incidentes basada en reglas específicas de cliente.

- El tiempo que la herramienta está disponible en el mercado y la gran cantidad de usuarios que la emplean dentro de las organizaciones.

- La experiencia con la que contamos en el uso de la herramienta.

- La variedad de manuales, videos y ejemplos de consulta que están disponibles en la web para usuarios y desarrolladores.

3.5 Breve descripción de la herramienta Atlassian Jira

Atlassian Jira es una herramienta utilizada para la administración de incidencias y proyectos.

Jira es un producto propietario desarrollada por Atlassian que proporciona soporte y código fuente de la plataforma a clientes que adquieran una licencia comercial. Esta herramienta maneja un grupo de usuarios que serán los encargados de crear

incidencias y otro grupo de usuarios, los operadores, que serán quienes lleven a cabo el trabajo sobre las incidencias.

Cada tipo de incidencia que se maneja estará asociada a distintos flujos de trabajo: los operadores que dispongan de ciertos permisos podrán mover la incidencia en la red de estados definida en el flujo de trabajo asociado.
Esta herramienta de gestión es una aplicación versátil, flexible y ofrece la posibilidad de configurar cualquier parámetro en cuanto a las incidencias, flujo de trabajo y las ventanas a través de las cuales los usuarios trabajan sobre las incidencias.

3.5.1 Componentes básicos

Jira proporciona un conjunto de tipos básicos de incidencias y un flujo de trabajo estándar, pero a diferencia de otras herramientas proporciona flexibilidad a la hora de definir los parámetros de trabajo permitiendo así configurar cualquier aspecto del entorno de trabajo.

Es importante que los usuarios comprendan cuales son los puntos fuertes y las funcionalidades de la herramienta. En base a esto, los usuarios podrán desarrollar e implementar los requerimientos que necesiten según su necesidad.
En cuanto a los usuarios administradores necesitarán tener en claro el conjunto de esquemas de configuración disponibles para poder modelar la funcionalidad de un proyecto de forma correcta.

Por esta razón, a continuación, se describirán los conceptos básicos manejados en Jira, tales como proyecto, incidencia y flujo de trabajo, y las relaciones que se establecen entre ellos con el objetivo de familiarizar al usuario con estas entidades" y la organización de conceptos dentro de la plataforma.

3.5.1.1 Proyectos

Un proyecto es el contenedor de las incidencias donde los usuarios pueden abrir incidencias o trabajar sobre ellas.
En el proyecto se definen las configuraciones que satisfacen los requerimientos y las funcionalidades solicitadas por los usuarios. En el momento que un usuario opera sobre una incidencia lo hace en el contexto de un proyecto.
Desde el punto de vista de un usuario, un proyecto puede verse como una instancia de privada de Jira ya que, aislada de los demás proyectos, por lo tanto, los usuarios podrán definir sus requerimientos libremente y solicitar a los administradores la personalización de su proyecto para ajustarlo según las necesidades del negocio que necesitan gestionar.

3.5.1.2 Componentes

Un proyecto puede dividirse opcionalmente en componentes. Un componente es una agrupación lógica de incidencias y una incidencia puede pertenecer a ninguno, uno o muchos componentes al mismo tiempo.

Un componente está formado por los atributos nombre, descripción y responsable del componente.
Un componente puede verse como un sub-proyecto, con un responsable y un conjunto de incidencias a gestionar. Jira proporciona un cuadro de mando de componentes,

idéntico al cuadro de mando de proyecto, donde se visualizan los indicadores de actividad del proyecto y otro tipo de estadísticas.
El responsable de un componente puede ser un operador que, por defecto, se le asignen si se crean incidencias en el sistema sin ser asignadas a otros usuarios.

La subdivisión de un proyecto en componentes será un requerimiento de los usuarios para reflejar la organización del equipo. La gestión de los componentes puede ser administrada por los usuarios que tengan los permisos "Administradores de Proyectos" (*Administer Projects*). Aunque es común efectuar esta operación al principio del proyecto, Jira permite modificar la lista de componentes en cualquier momento y añadir, en caso de que sea necesario nuevos componentes que se vayan necesitando a lo largo del tiempo.

3.5.1.3 Versiones

Un proyecto se puede subdividir en fases llamadas versiones.
Una versión está formada por los siguientes atributos:

I. Nombre
II. Descripción
III. Fecha de entrega
IV. Planificación

Al mismo tiempo, una versión puede tener asiganda uno de los siguientes estados:

I. *Unreleased*: la versión todavía no se ha entregado
II. *Released*: la versión se ha entregado
III. *Archived*: la versión se ha archivado

Las versiones no están organizadas en jerarquías, sino que son un conjunto de atributos de un proyecto que pueden relacionarse con las incidencias en el campo tipo versión. En Jira, las incidencias llevan por defecto dos campos relacionados con las versiones: *Affects version(s) y Fix version(s)*.
Sin embargo, un usuario puede solicitar la creación de campos personalizados de tipo versión y definir la relación que necesite.

3.5.2 Incidencias

Las incidencias son los objetos de negocio que se manejan en la herramienta Jira.
El término incidencia está asociado a proyectos relacionados con el desarrollo y el control de calidad del software. Sin embargo, Jira permite manejar proyectos e incidencias de cualquier naturaleza.
En resumen, una incidencia en la herramienta está asociado a proyectos e incidencias de cualquier naturaleza.

3.5.2.1 Tipos de incidencia

Todas las incidencias asociadass a un proyecto tienen asignado un tipo de incidencia.
El tipo de incidencia es uno de los parámetros fundamentales que maneja el comportamiento de las incidencias en un proyecto ya que es la entidad con la que se

establecen enlaces con las demás categorías de parámetros de configuración relacionados con las incidencias tales como:

- El comportamiento de las incidencias de cada tipo en un flujo de trabajo.
- Los datos almacenados en cada incidencia en una configuración de campos.
- Las pantallas utilizadas por los usuarios en una configuración de pantallas.

Por defecto, Jira proporciona los tipos de incidencia *Bug, Improvement, New feature y Task*.
Dependiendo de los requerimientos de los usuarios, se podrán crear nuevos tipos de incidencias en base la necesidad de cada proyecto.

3.5.2.2 Campos

Por defecto, las incidencias tienen un conjunto de campos predeterminados los cuales no pueden borrarse ni modificarse. En caso de que sea necesario, se puede esconder ciertos campos, siempre que sean campos opcionales de las incidencias.
Sin embargo, los usuarios no están limitados a utilizar exclusivamente los campos de incidencia estándar de la herramienta, ya que gracias a la flexibilidad que ofrece la plataforma, los usuarios pueden adicionar campos personalizados y asignarlos a los tipos de incidencias que crean convenientes según la necesidad del negocio.

3.5.2.3 Prioridad

La prioridad de una incidencia es un campo que permite establecer la importancia relativa de una incidencia con respeto a otras incidencias en el mismo proyecto.

Por defecto la herramienta proporciona las siguientes prioridades en orden decreciente de importancia:

- *Blocker*: prioridad asignada a incidencias que bloquean el progreso de otras actividades.
- *Critical*: prioridad asignada a incidencias de importancia crítica que, al momento, no están bloqueando otras actividades.
- *Major*: prioridad asignada a incidencias de importancia mayor.
- *Minor*: prioridad asignada a incidencias de importancia menor.
- *Trivial*: prioridad asignada a incidencias de importancia trivial.

El administrador de Jira puede definir niveles adicionales de prioridad, aunque estos niveles no serán limitados al contexto de un proyecto sino a toda la plataforma.
El campo prioridad se utiliza en operaciones de filtrado de incidencias, asignación y reporting.

3.5.2.4 Estado

El estado de una incidencia es un “bandera” que indica en que punto de su flujo de trabajo se encuentra una determinada incidencia.

Por defecto, los estados definidos en el flujo de trabajo son los siguientes:

- *Open*: es el estado inicial de una incidencia antes de que se empiece a operar sobre ella.

- *In Progress*: es cuando un operador está trabajando sobre una incidencia.
- *Resolved*: es cuando una incidencia que ha sido resuelta.
- *Reopened*: es el estado de una incidencia que se ha vuelto a abrir después de haber estado resuelta o cerrada.
- *Closed*: es cuando una incidencia está cerrada.

3.5.2.5 Resolución

El campo resolución define "como" y "si" una incidencia se solucionó. Por defecto, las resoluciones que Jira define son las siguientes:

- *Fixed*: la incidencia ha sido resuelta.
- *Won't fix*: la incidencia no se resolverá.
- *Duplicate*: la incidencia no se resolverá porqueestá duplicada.
- *Incomplete*: la incidencia no se resolverá porque fue descripta de forma incompleta.
- *Cannot reproduce*: la incidencia no se resolverá porque no se puede reproducir siguiendo las instrucciones proporcionadas por el usuario que reportó la incidencia.

En caso de que los usuarios necesiten definir nuevos tipos de resoluciones podrán solicitárselas a los administradores de la herramienta y las resoluciones creadas se añadirán al conjunto de resoluciones que existen en la plataforma.

Las resoluciones son un parámetro global de la herramienta, por lo tanto, serán visibles para todos los proyectos.

3.5.3 Flujos de trabajo

Los flujos de trabajo definen la máquina de estados asociada a uno o más tipos de incidencias.
Por defecto, Jira proporciona un flujo de trabajo que no puede modificarse.

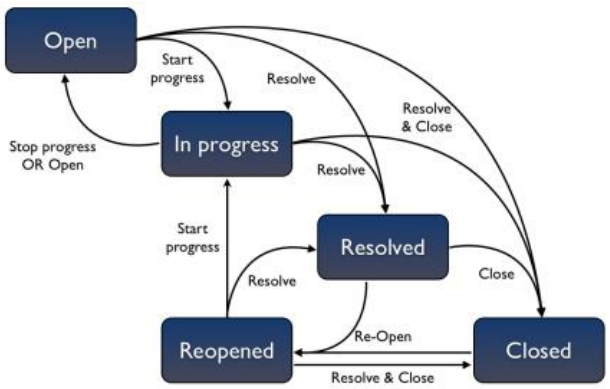

Figura 9. Flujo de Trabajo

En la Figura las cajas representan a los estados de la máquina y las flechas representan a las transiciones entre ellos.

Los estados *Resolved* y *Closed* son dos permisos distintos dentro de un proyecto para resolver y cerrar incidencias. Esta flexibilidad puede utilizarse para otorgar el permiso de cerrar incidencias solo a un subconjunto de usuarios que tienen como responsabilidad la supervisión del trabajo de los operadores.

La creación de flujos de trabajos personalizados permite implementar cualquier comportamiento que los usuarios requieran para todos los tipos de incidencias gestionados en su proyecto, siempre que dicho comportamiento se pueda modelar con una máquina de estados.

Un flujo de trabajo se modela de forma sencilla utilizando el interfaz web de Jira. Los usuarios deberán definir los siguientes requerimientos y enviarlos a los administradores de la plataforma:

- Los estados de la máquina de estados.
- Las transiciones entre estados.
- La relación entre estado de una incidencia y estado del flujo de trabajo.

Los campos estándar de las incidencias no pueden borrarse ni modificarse. En caso de que sea necesario, se puede esconder ciertos campos, siempre que sean campos opcionales de las incidencias.
Sin embargo, los usuarios no están limitados a utilizar exclusivamente los campos de incidencia estándar de la herramienta, ya que gracias a la flexibilidad que ofrece la plataforma, los usuarios pueden adicionar campos personalizados y asignarlos a los tipos de incidencias que crean convenientes según la necesidad del negocio.

3.5.3.1 Prioridad

La prioridad de una incidencia es un campo que permite establecer la importancia relativa de una incidencia con respeto a otras incidencias en el mismo proyecto.

Por defecto la herramienta proporciona las siguientes prioridades en orden decreciente de importancia:

- *Blocker*: prioridad asignada a incidencias que bloquean el progreso de otras actividades.
- *Critical*: prioridad asignada a incidencias de importancia crítica que, al momento, no están bloqueando otras actividades.
- *Major*: prioridad asignada a incidencias de importancia mayor.
- *Minor*: prioridad asignada a incidencias de importancia menor.
- *Trivial*: prioridad asignada a incidencias de importancia trivial.

El administrador de Jira puede definir niveles adicionales de prioridad, aunque estos niveles no serán limitados al contexto de un proyecto sino a toda la plataforma.
El campo prioridad se utiliza en operaciones de filtrado de incidencias, asignación y reporting.

3.5.3.2 Estado

El estado de una incidencia es un “bandera” que indica en que punto de su flujo de trabajo se encuentra una determinada incidencia.

Por defecto, los estados definidos en el flujo de trabajo son los siguientes:

- *Open*: es el estado inicial de una incidencia antes de que se empiece a operar sobre ella.
- *In Progress*: es cuando un operador está trabajando sobre una incidencia.
- *Resolved*: es cuando una incidencia que ha sido resuelta.
- *Reopened*: es el estado de una incidencia que se ha vuelto a abrir después de haber estado resuelta o cerrada.
- *Closed*: es cuando una incidencia está cerrada.

3.5.3.3 Resolución

El campo resolución define "como" y "si" una incidencia se solucionó. Por defecto, las resoluciones que Jira define son las siguientes:

- *Fixed*: la incidencia ha sido resuelta.
- *Won't fix*: la incidencia no se resolverá.
- *Duplicate*: la incidencia no se resolverá porqueestá duplicada.
- *Incomplete*: la incidencia no se resolverá porque fue descripta de forma incompleta.
- *Cannot Reproduce*: la incidencia no se resolverá porque no se puede reproducir siguiendo las instrucciones proporcionadas por el usuario que reportó la incidencia.

En caso de que los usuarios necesiten definir nuevos tipos de resoluciones podrán solicitárselas a los administradores de la herramienta y las resoluciones creadas se añadirán al conjunto de resoluciones que existen en la plataforma.

Las resoluciones son un parámetro global de la herramienta, por lo tanto, serán visibles para todos los proyectos.

3.5.4 Notificaciones

En respuesta a determinados eventos la herramienta envía notificaciones a los usuarios que se hayan asociados a cada uno de ellos. El esquema de notificaciones establece una relación entre los tipos de eventos y los usuarios que recibirán la notificación.
En cualquier momento del ciclo de vida del proyecto, los usuarios pueden asignar eventos no solo a cuentas de usuarios (o grupos) sino a las identidades pertenecientes al entorno. En caso de que un usuario necesite ser reportado de inmediato cuando se produce algún evento se podrá utilizar el mecanismo de notificaciones y "subscribirse" a las fuentes de eventos que le interesan.

3.5.5 Políticas de seguridad

El esquema de permisos es la herramienta que permite definir las políticas de seguridad que aplican en el contexto de un proyecto. Jira tiene definido un conjunto de permisos que permiten construir políticas de seguridad de grano muy fino y organizar los usuarios que acceden a un proyecto en todos los roles que se consideren necesarios.

Un punto de importancia a tener en cuenta, es que los responsables de cada proyecto pueden solicitar la definición de roles de proyectos y políticas de seguridad que les permitan proteger, de acuerdo a sus necesidades la información almacenada.

Los permisos están divididos en las siguientes categorías:

- Permisos para proteger proyecto.
- Permisos para proteger incidencia.
- Permisos para proteger votantes y observadores.
- Permisos para proteger comentarios.
- Permisos para proteger ficheros adjuntos.
- Permisos para proteger el registro de trabajo.

Los permisos permiten proteger las transiciones de flujo de trabajo asignándoles políticas de seguridad y las acciones que pueden efectuarse sobre las entidades (acciones de creación, actualización y eliminación)

3.5.6 Niveles de seguridad en las incidencias

Las herramientas para definir las políticas de seguridad permiten crear políticas de "grano muy fino" pero no permiten distinguir entre las incidencias. En ciertos momentos puede surgir la necesidad de restringir el acceso a incidencias concretas. Para lograr esto se deberían crear diferentes proyectos para poder proteger las distintas incidencias aplicando distintas políticas de seguridad. Para solventar este problema, Jira permite definir esquemas de nivel de seguridad.

Un esquema define un conjunto de niveles de seguridad y el acceso a cada uno de los niveles puede otorgarse a un conjunto de identidades como:

- Informar y reportar sobre la incidencia.
- Usuario.
- Grupo.
- Rol de proyecto.
- Responsable de proyecto.
- Asignatario de la incidencia.
- Usuario indicado en un campo personalizado de la incidencia.
- Grupo indicado en un campo personalizado de la incidencia.

Los niveles de seguridad permiten definir tantas políticas de seguridad como el proyecto requiera para proteger a sus incidencias. La incidencia a proteger podrá asociarse a un nivel de seguridad y esta asociación podrá ser cambiada cuando necesario. En ese momento, el acceso estará restringido a las identidades especificadas por la política de seguridad del nivel asignado. La asociación de una incidencia con un nivel de seguridad solamente puede ser realizada por un usuario con el permiso "*set issue security*".

Capítulo IV – Identificación de conceptos claves para la integración

Luego de haber elegido el entorno Bonitasoft como BPM en la Sección 2.3 y la herramienta Atlassian Jira como Gestor de Incidencias en la sección 3.4, en este capítulo estaremos realizando un estudio para poder identificar cuales deberían ser los puntos claves a tener en cuenta para lograr una arquitectura integradora entre ambas partes, permitiendo así que las tecnologías puedan adaptarse de manera adecuada y otorgar un nivel de contacto flexible y eficiente.

Al momento de contar con una solución integrada, la aplicación BPMS nos brindará la posibilidad de tener disponibles funcionalidades como la registración de tareas o incidencias surgidas para cada instancia, pudiendo así gestionarlas desde el mismo entorno y manteniendo un historial auditable de las mismas. Otro punto a tener en cuenta tiene ver con la fusión de dos entornos en solo uno solo obteniendo así mayor facilidad y practicidad a los usuarios del BPMS.

De acuerdo a la investigación realizada, identificamos una serie de aspectos que debemos considerar al momento de relacionar ambas tecnologías tal como se detalla a continuación:

- En la Sección 4.1 definiremos un modo comunicación entre el entorno BPM y el Gestor de Incidentes.
- En la Sección 4.2 daremos una definición de los posibles esquemas de datos que se pueden implementar.
- En la Sección 4.3 indicaremos como será la interacción entre procesos y proyectos.
- En la Sección 4.4 se dará una explicación de una posible alternativa para la gestión de usuarios.
- Por último, en la sección 4.5 describiremos las conclusiones finales y pasos a seguir de acuerdo a lo descripto en las secciones anteriores.

4.1 Comunicación entre componentes

Para lograr una interacción entre una herramienta BPMS y un Sistema de Gestión de Incidentes necesitamos definir algún tipo de mecanismo de comunicación que nos permita definir un conjunto de operaciones en forma eficiente y transparente relacionadas a:

I. Procesos con Proyectos
II. Participantes con Usuarios
III. Incidentes

Una manera de lograr esta interacción sería por medio de la utilización, definición e implementación de servicios webs que abstraigan el producto específico que se encuentra en cada lado del modelo integrador.

Otra alternativa de lograr una comunicación entre ambos entornos sería implementar algún otro tipo de conector que permita intercambiar información a través de las APIs y la utilización de las aplicaciones clientes provistos por ambas partes.

4.2 Modelo de datos

En el momento que ejecutamos operaciones relacionadas a la creación, modificación y visualización de tareas e incidentes desde el entorno BPMS estas acciones impactan directamente en el modelo de datos de ambos aplicativos (BPMS y Sistema de Gestión de Incidentes).
Al definir e implementar las operaciones sobre un entorno BPMS y teniendo una tarea o incidente asociado, nos vemos ante la necesidad de sincronizar ambas herramientas para administrar datos en común.
De acuerdo a los modelos de datos de cada tipo de ambiente, identificamos la posibilidad de plantear dos posibles metodologías para sincronizar la información:

I. Dos modelos de datos: gestionar ambos modelos en forma separada y sincronizar la información en común (ida y vuelta). Para este tipo de solución, se necesitará la definición de algún módulo intermedio encargado de unir ambos modelos.

II. Un único modelo de datos: el foco estará siempre sobre el modelo de datos del BPMS, por lo tanto, el sistema de Gestión de Incidentes deberá interactuar directamente con el mismo, sin la necesidad de que exista algún componente intermedio que se encargue de lidiar con cuestiones específicas de implementación para cada caso.

4.3 Asociación de Procesos – Proyectos

En el momento de ejecutar una operación relacionada a un incidente (creación, modificación o visualización) en el transcurso de las tareas de la ejecución de un proceso, deberemos contar con algún componente que permita asociar un proceso implementado en el BPMS con proyectos registrados en el Sistema de Gestión de Incidencias.
Por lo tanto, tendremos que contar con algún entorno al que un “Administrador” pueda gestionar el enlace entre los procesos pertenecientes al entorno BPMS y los proyectos almacenados en el repositorio del Sistema de Gestión de Incidencias.
Para solventar esta función, se necesitará entonces una base de datos particular que almacene ambos vínculos, manteniendo actualizada la información de los procesos y proyectos consultando tanto al BPMS como al Sistema de Gestión de Incidencias respectivamente.

4.4 Asociación de Participantes – Usuarios

Las herramientas BPMS proveen mecanismos de administración de usuarios que interactuarán en los distintos procesos, permitiendo otorgar diferentes roles y vincularlos así con permisos específicos para ejecutar una tarea.
Por otro lado, los sistemas de Gestión de Incidencias también cuentan con su propio módulo para el manejo de usuarios. Cada usuario tendrá un conjunto de permisos

asociados para poder operar sobre tareas o incidentes, los cuales están asociados a un proyecto en particular. Un usuario puede realizar las siguientes operaciones: crear, editar o visualizar tareas o incidentes.
Una posible solución es pensar en unir ambos repositorios en uno solo, ya que nos facilitará la administración y mantenimiento del conjunto de usuarios del dominio para los casos en que se agreguen nuevos usuarios o dejen de existir otros. Además, con el manejo de roles podemos separar los permisos que tendrá cada usuario.
El administrador del sistema integrador será el encargado de gestionar tal repositorio.

4.5 Conclusión

Una vez mencionados los puntos de conexión podemos pensar en unificarlos bajo una arquitectura bien definida que nos sirva de modelo cada vez que intentemos vincular tecnologías BPMS y con herramientas de Gestión de Incidencias.

La sincronización del modelo de datos nos proveerá consistencia en la información que manejan las dos herramientas y que pertenecen al mismo dominio, por tanto, solo se tendrá que elegir el mecanismo que mejor se adapte al objetivo de integración en cuestión.
Al tener un módulo encargado de la comunicación que sea configurable permite que podamos unir distintos productos implementando los mecanismos que sean necesarios en cada caso.

En cuanto a la posibilidad de relacionar a los usuarios en un único repositorio centralizado evitamos tener que mantener los dos entornos en cuestión, permitiendo además que los usuarios del BPMS también puedan interactuar bajo el entorno de Gestión de Incidencias.
Al identificar los conceptos claves que conlleva la interacción de las herramientas abarcadas en este trabajo, se logra alcanzar una propuesta arquitectónica que los engloba y cuyos detalles de implementación se detallarán en el próximo capítulo.

Capítulo V - Arquitectura de Integración entre el BPM Bonitasoft con el Gestor de Incidencias Atlassian Jira

Este capítulo tiene como misión poder definir y explicar el modelo de arquitectura propuesto para lograr la integración entre los entornos de Bonitasoft y Atlassian Jira.
De acuerdo a la investigación realizada en el Capítulo I - Los Procesos de Negocio y su Gestión, Capítulo III - Gestión de Incidencias y Capítulo IV - Identificación de conceptos claves para la integración dividiremos al capítulo de la siguiente manera:

- En la Sección 5.1 definiremos la arquitectura de integración.
- En la Sección 5.2 daremos una explicación de todos los componentes que forman parte de la arquitectura definida en la Sección 5.1.
- En la Sección 5.3 estaremos explicando cómo será asociación y/o gestión de "Procesos con Proyectos" y "Participantes con Usuarios" de la arquitectura propuesta tomando como base los conceptos descriptos en el Capítulo IV.
- Por último, en la Sección 5.4 desarrollaremos un caso de éxito que nos permita demostrar la correcta integración entre ambos sistemas de acuerdo al modelo de arquitectura implementado.

5.1 Definición de arquitectura de integración

De acuerdo al estudio realizado en el Capítulo IV “Identificación de conceptos claves para la integración”, definimos la siguiente propuesta arquitectónica con el objetivo de obtener una visión integral para lograr una interacción adecuada entre los sistemas Bonitasoft y Atlassian Jira.

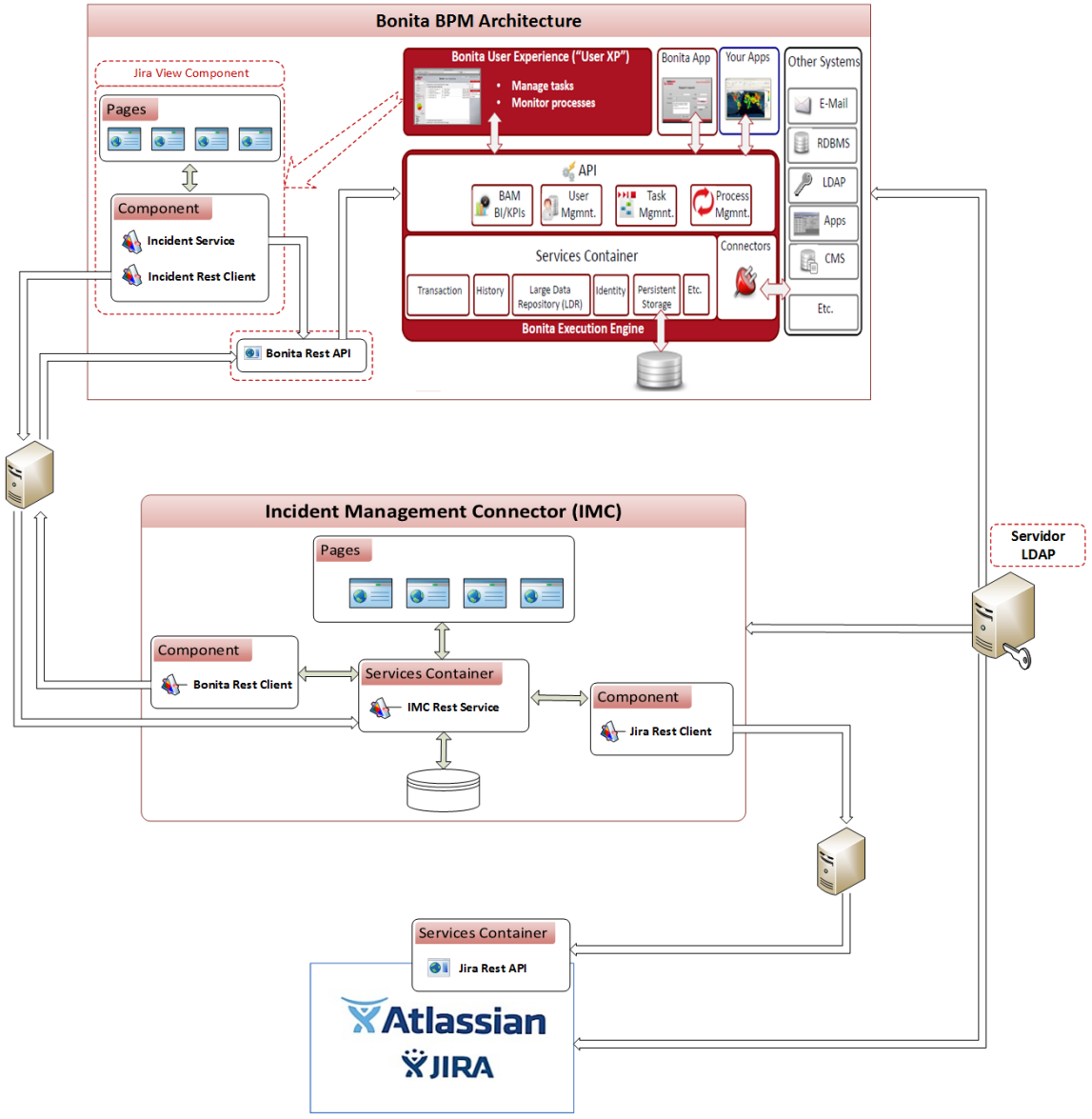

Figura 10. Arquitectura de Integración

A partir de este modelo se tendrá la capacidad de poder administrar operaciones (creación, modificación y actualización) relacionadas a incidencias desde el entorno de Bonita XP *(Bonita User Experience*) sin la necesidad de ingresar a la herramienta externa Atlassian Jira. Las operaciones mencionadas podrán ser ejecutadas a partir de una instancia de un proceso de negocio que se encuentra desplegado en el entorno Bonita XP. Luego, la información perteneciente a la operación realizada será registrada en el entorno de Atlassian Jira.

Tomando como base los conceptos especificados en las Secciones 4.1, 4.2, 4.3 y 4.4 para obtener una integración adecuada, definimos y relacionamos los siguientes puntos:

I. Asociación de Participantes-Usuarios

Por un lado, llamaremos “Participantes” a aquellas personas que interactúen con las diferentes instancias de procesos de negocio desde el entorno Bonita XP.

Por otro lado, denominaremos “Usuarios” a aquellas personas que estarán asociadas a proyectos pertenecientes al entorno de Jira, pudiendo llevar adelante operaciones de creación, modificación y eliminación de incidencias.

Para lograr la relación Participantes con Usuarios planteamos la posibilidad de contar con un único repositorio común entre ambos sistemas. Para esto, decidimos utilizar un protocolo de tipo cliente-servidor LDAP (*Lightweight Directory Access Protocol*) para acceder a un servicio de directorio.
En el marco de esta arquitectura haremos uso del servidor OpenLDAP ya que es una implementación libre y de código abierto de LDAP. Otra característica de importancia, en que OpenLDAP cuanta con una API que nos ofrece la capacidad de comunicación entre componentes de software, utilizadas como bibliotecas o librerías. [65]

II. Implementación del módulo Incident Management Connector (IMC)

En nuestro modelo definimos al módulo IMC como una aplicación intermedia desarrollada bajo la tecnología Java. Este aplicativo será el encargado de sincronizar los modelos de datos, realizará la comunicación entre los diferentes componentes y brindará un conjunto de servicios específicos para ambos entornos.

En el sistema IMC tanto los servicios como las funcionalidades serán administrados a través de un único usuario “Administrador”. Además, contará con un modelo de datos donde se podrá administrar toda la información relacionada a los servicios que éste provee.

La finalidad principal de este sistema es poder lograr la asociación de Procesos con Proyectos de manera correcta y eficiente. Para esto, IMC deberá tener la capacidad de poder comunicarse con el entorno Bonita BPM para obtener toda la información necesaria de los procesos de negocio, como así también obtener toda la información de los proyectos definidos en la herramienta Jira. Luego de contar con la información mencionada, el usuario Administrador tendrá la posibilidad de gestionar procesos con proyectos según corresponda.

5.2 Componentes de la arquitectura propuesta

A continuación, daremos una explciación de cada uno de los componentes y servicios implementados según la arquitectura definida en la Sección 5.1.

5.2.1 Bonitasoft

5.2.1.1 Jira View Component

En el ámbito de Bonita BPM Portal definido en la Sección 2.4.2, desarrollamos un nuevo componente llamado “Jira View Component” que será el encargado de adminitrar la interacción entre los participantes de procesos y las incidencias asociadas a los mismos.

Desde el punto de vista tecnológico, el componente se implementó utilizando el framework Angular JS; el cual podrá se visualizado en el momento que un participante selecciona una determinada tarea que éste puede tomar. Luego el participante deberá seleccionar la opción “Incidencias” según se muestra en la Figura 11.

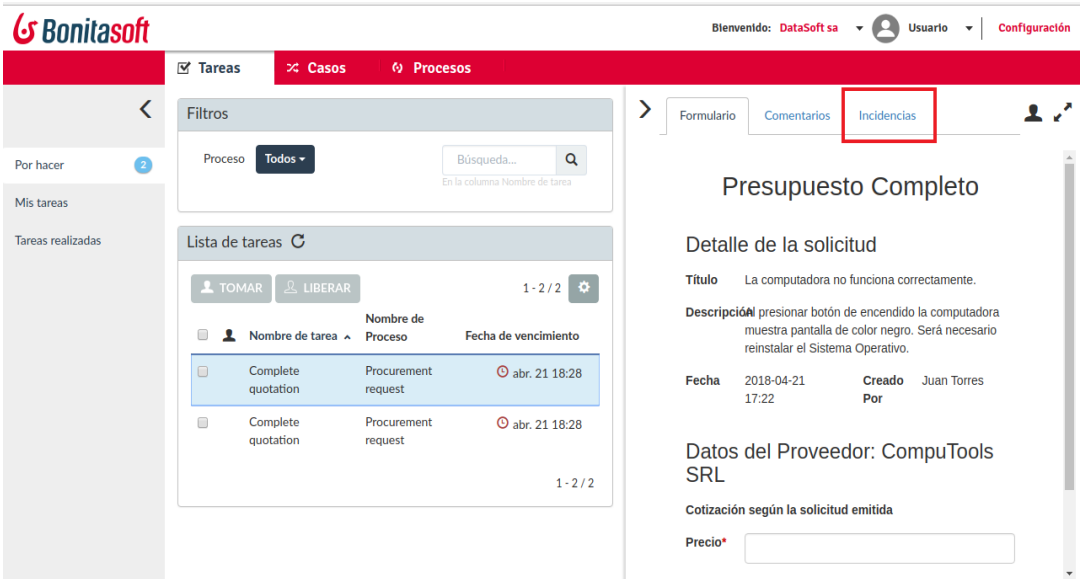

Figura 11. Jira View Component

5.2.1.2 Pages

Dentro del portal web Bonita XP, implementamos un conjunto de páginas webs con el objetivo de que un participante pueda interactuar con las incidencias. Estas páginas, tienen comunicación directa con el componente “Incident Service” que se utiliza para recuperar y enviar información asociadas a las incidencias desde el entorno de Jira.

A continuación, se detallan las páginas desarrolladas para poder cubrir con la operatoria relacionada a las incidencias:

- Visualización de Incidencias registradas

Esta página fue implementada en el archivo issues.home.html, ofreciendo las siguientes funcionalidades:

I. Cuenta con un filtro de búsqueda de incidencias.
II. Muestra una lista de incidencias de acuerdo a la tarea seleccionado.
III. Desde la lista permite modificar una determinada incidencia.
IV. Desde la opción “Crear”, permite acceder al formulario de creación de incidencias.

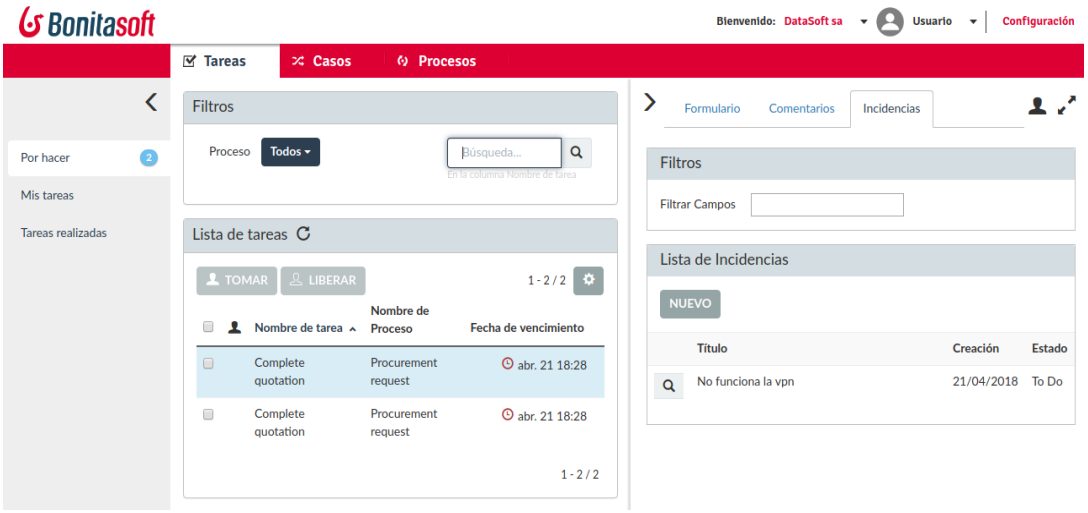

Figura 12. Página web - Visualización de Incidencias

- Creación de Incidencias

Esta página fué implementada en el archivo issues.createIssue.html.
Es un formulario para ingresar información relacionada a una incidencia. Para poder registrar una incidencia es necesario contar con la siguiente información: proyecto, tipo de tarea, prioridad, asignado a, resumen, descripción y evidencias.

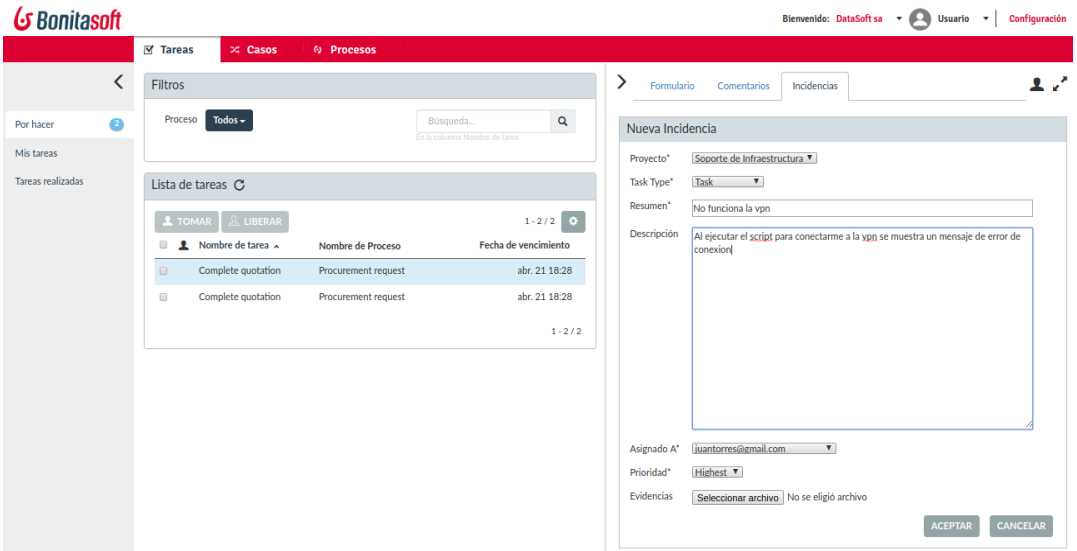

Figura 13. Página web - Creación de Incidencias

- Modificación de Incidencias

Esta página fué implementada en el archivo issues.viewIssue.html.

Es un formulario que permite la visualización y edición de la información asociada a una incidencia. Además, desde esta página se tendrá la posibilidad de adicionar, modificar o eliminar comentarios.
Los campos disponibles en este fomulario son los siguientes: proyecto, tipo de tarea, prioridad, asignado a, resumen, descripción, evidencias y comentarios.

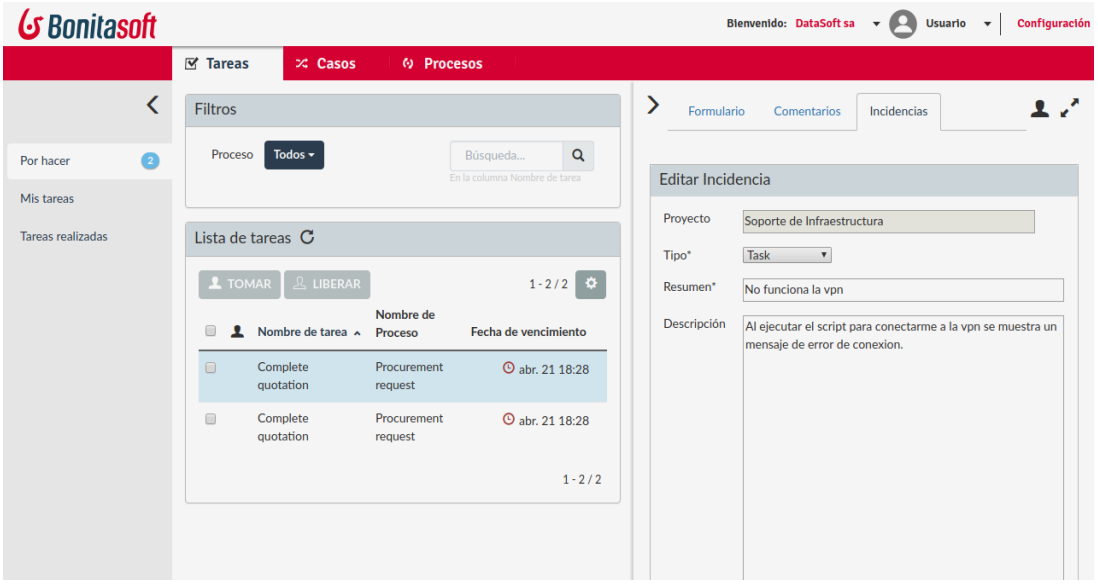

Figura 14. Página web - Actualización de datos generales de incidencias

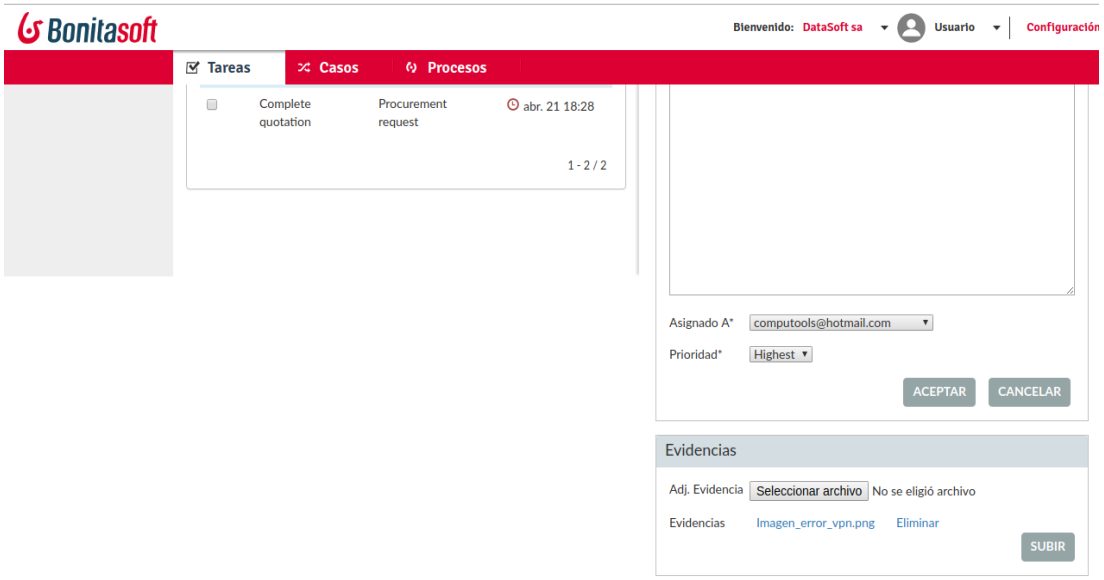

Figura 15. Página web – Adjuntar documentos de incidencias

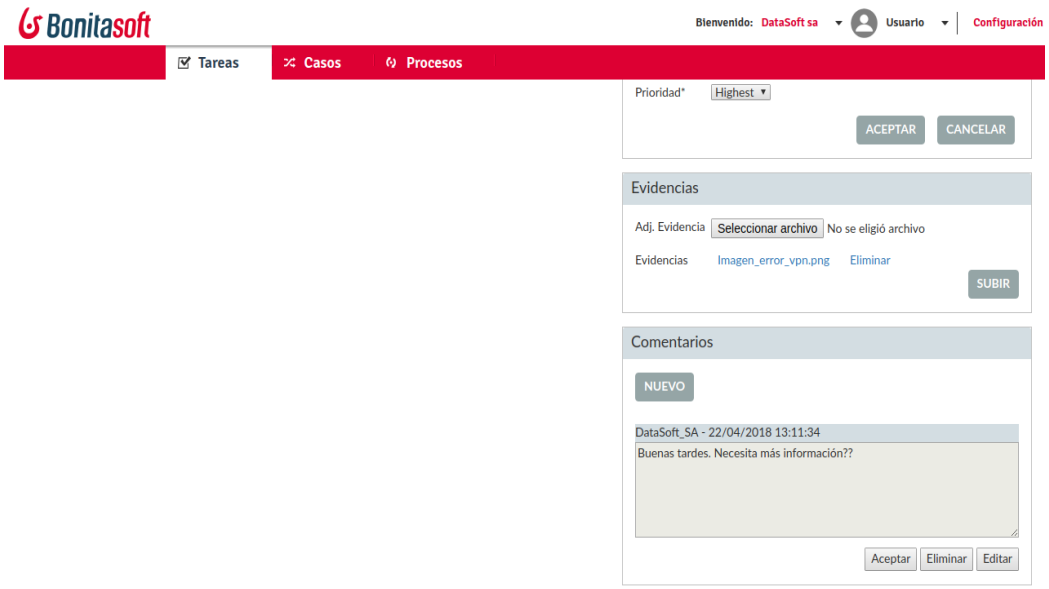

Figura 16. Página web – Adición o modificación de comentarios

5.2.1.3 Template Principal

Este template funciona como Master Page de las siguientes páginas webs:

- issues.home.html
- issues. createIssue.html
- issues. viewIssue.html

El template principal fué implementado en el archivo issues.html.

Para incluir las páginas webs en el entorno Bonita XP y enlazar el template principal con el controlador IssuesController del modelo MVC tuvimos que desarrollar una directiva Angular en el archivo issues.directive.js.
Esta directiva la incluimos en la sección de tab's pertenecientes al achivo task-details.html para poder visualizar la nueva solapa "Incidencias" dentro del entorno Bonita XP.

5.2.1.4 Incident Service

Este componente maneja la lógica de negocio relacionada a las incidencias del lado de Front-End en el entorno Bonita XP y es el encargado de evaluar qué página deberá ser presentada a un participante según las solicitudes que éste realice.
Además, este componente tiene comunicación con:

- El componente "Pages".
- El componente "Incident Rest Client".

Este servicio se encuentra implementado por:

I. IssuesController

Este controlador recibe eventos efectuados desde las páginas webs y delega la resolución de las peticiones al servicio IssuesService. Luego el servicio le retornará una *promise* al controlador para que tenga disponible la información a exhibir en las páginas webs.

Los métodos que utilizamos en el controlador tienen como objetivo poder manejar toda la información y las peticiones relacionada a las incidencias.
Una petición está asociada a una de las siguientes acciones:

- Crear una incidencia
- Modificar una incidencia
- Visualizar una incidencia
- Adicionar o eliminar comentarios de una incidencia
- Adjuntar o eliminar evidencias de una incidencia

Los métodos principales definidos para administrar incidencias son los siguientes:

Tabla 1. Métodos del controlador IssuesController

Métodos	Descripción
getTemplate ()	Determina cuál es la vista a presentar en base a las acciones ejecutadas por el participante.
createIssueAccept ()	Envía la información relacionada a una nueva incidencia.
editIssueAccept ()	Envía la información de una incidencia modificada por un participante.
uploadAttachment ()	Permite seleccionar una evidencia (documento) y asociarla a una incidencia existente.
deleteAttachment ()	Permite eliminar una evidencia asociada a una incidencia.
addComment ()	Permite crear un comentario y asociarlo a una incidencia existente.
editComment ()	Permite modificar un comentario existente.
deleteComment ()	Permite eliminar un comentario asociado a una incidencia.

Este controlador fué implementado en el archivo issues.controller.js.

II. IssuesService

El servicio IssuesService es el responsable de definir la lógica de negocio en la capa de presentación (Front-End), respondiendo a las llamadas realizadas desde el controlador IssuesController.

Además, este servicio tiene comunicación con la aplicación Incident Management Connector (IMC) a través del cliente “Incident Rest Client”.

Los métodos principales definidos en este servicio son los siguientes:

Tabla 2. Métodos del servicio IssuesService

Métodos	Descripción
createIssueAccept ()	Crea un FormData con la información relacionada a una nueva incidencia para ser enviado al IMC.
editIssueAccept ()	Crea un FormData con la información de una incidencia modificada para ser enviado al IMC.
uploadAttachment ()	Crea un FormData con el archivo seleccionado para ser enviado al IMC.
deleteAttachment ()	Permite eliminar una evidencia asociada a una incidencia.
addComment ()	Permite crear un comentario y asociarlo a una incidencia existente.

editComment ()	Permite modificar un comentario existente.
deleteComment ()	Permite eliminar un comentario asociado a una incidencia.

5.2.1.5 Incident Rest Client

Incident Rest Client es un cliente que tiene como único objetivo actuar como capa de comunicación con el servicio IMC Rest Service a través de llamadas HTTP.
Para lograr esta comunicación utilizamos el servicio $http del framework Angular JS que retorna una *promise* que es manejada por el controlador IssuesController del componente "Incident Service".

En el momento de implementar el cliente creamos una factory con el siguiente conjunto de operaciones relacionadas a las incidencias:

Tabla 3. Métodos del cliente Incident Rest Client

Métodos	Descripción
getProjects ()	Retorno la lista de proyectos a visualizar en la página de creación de incidencias.
getPriorities ()	Retorno la lista de prioridades a visualizar en las páginas de creación y edición de incidencias.
createIssue ()	Envía un FormData con la información relacionada a una nueva incidencia desde la página de creación de incidencias.
editIssue ()	Envía un FormData con la información de una incidencia modificada desde la página de edición de incidencias.
getIssues ()	Retorna la lista de incidencias a exhibir en la página de visualización de incidencias.
getIssue ()	Retorna una determinada incidencia para ser presentada en la página edición de incidencias.
getAssignableUsers ()	Retorna una lista de usuarios que pueden ser asignados a una incidencia. Esta lista será presentada en las páginas de creación y edición de incidencias.
getIssueTypesByProject ()	Retorna una lista de tipos de incidencias según el proyecto seleccionado. Esta lista será presentada en las páginas de creación y edición de incidencias.
uploadAttachment ()	Envía un FormData con el archivo seleccionado desde la página de edición de incidencias.
deleteAttachment ()	Permite eliminar una evidencia asociada a una incidencia en la página de edición de incidencias.
addComment ()	Permite crear un comentario y asociarlo a una incidencia existente desde la página de edición de incidencias.
editComment ()	Permite modificar un comentario existente desde la página de edición de incidencias.
deleteComment ()	Permite eliminar un comentario asociado a una incidencia desde la página de edición de incidencias.
getComments ()	Retorna una lista de comentarios de una determinada incidencia. Esta lista será presentada en la página de edición de incidencias.

5.2.1.6 Bonita Rest API

Para poder integrar una aplicación externa con una solución de Bonitasoft, esté o no implementada con tecnología Java, podemos hacerlo mediante el uso de una API REST Web provista por el producto.

Esta API proporciona acceso a todos los objetos del entorno Bonita BPM como procesos, tareas, usuarios, conectores, etc. para ejecutar operaciones en ellos (crear, recuperar, actualizar o eliminar). A través del uso de estas operaciones podemos crear un flujo de trabajo en el entorno Bonita BPM e integrarlo posteriormente con una aplicación externa.
Bonita BPM Engine es el responsable de ejecutar la lógica del flujo de trabajo (conectores, puertas de enlace con condiciones, mensajes, temporizadores, etc.) definido desde el entorno Bonita Studio, mientras que con la aplicación externa se podrá acceder al flujo de trabajo [44].

5.2.2 Incident Management Connector

Incident Management Connector (IMC) es una aplicación desarrollada en Java que actúa como eje central de integración entre el entorno Bonitasoft y la herramienta Atlassian Jira.

Por un lado, IMC tiene disponible una interfaz web implementada en Angular JS que nos permite realizar las asociaciones de Procesos con Proyectos.
Por otro lado, el backend expone un conjunto de servicios que pueden ser consumidos mediante request's HTTP utilizados por el componente "Jira View Component".

Además, la aplicación puede comunicarse con:

- El motor de Bonita (*Bonita Execution Engine*) mediante el uso de Bonita Rest API.
- La herramienta Atlassian Jira utilizando Jira Rest API.
- El servicio de directorio LDAP a través de la clase InitialDirContext del paquete javax.naming.directory perteneciente al ámbito nativo de Java.

Para poder lograr la comunicación con los entornos mencionados, en el contexto del IMC tuvimos que implementar los siguientes módulos que actuarán como Capa de Abstración:

- El cliente "Bonita Rest Client" va a estar interactuando de forma directa con Bonita Rest API.
- El cliente "Jira Rest Client" va a estar interactuando de forma directa con Jira Rest API.
- Desarrollamos la clase LdapAuthenticationManagerImpl que va a estar interactuando de forma directa con InitialDirContext.

5.2.2.1 IMC Rest Service

Es el servicio central de integración que nos permite realizar todas las operaciones relacionadas a incidencias, procesos, proyectos, participantes y usuarios, exponiendo su funcionalidad a través de servicios REST.

Al mismo tiempo, este servicio se comunica con los componentes “Bonita Rest Client” y “Jira Rest Client” teniendo como objetivo principal poder interactuar con los entornos Bonitasoft y Atlassian Jira. Adicionalmente, el servicio se encargará de manejar los posibles errores de comunicación que puedan surgir con las aplicaciones externas.

IMC Rest Service está implementado por cuatro clases princpales:

I. IncidentController

Es la clase encargada de recibir todas las peticiones HTTP desde el entorno Bonita XP y el componente Page pertenecientes al IMC.
En cada petición se recibe:

- Un token o un usuario y contraseña
- La operación relacionada a una incidencia

En segundo lugar, luego de recibir una petición delega su resolución al servicio SessionService para temas relacionados al manejo de sesiones y al servicio JiraServiceImpl para temas relacionado a incidencias.

Por último, luego de ejecutar la operación solicitada, como resultado se retorna la siguiente información:

- El código de estado HTTP correspondiente
- El estado de la operación
- La información de la operación

Los métodos definidos para el controlador son los siguientes:

Tabla 4. Métodos del controlador IncidentController

Nombre	Descripción
getProjects ()	Retorna los proyectos en los cuales un determinado participante tiene permisos.
createIssue ()	Crea una indicencia en Jira.
editIssue ()	Actualizauna indicencia en Jira.
getIssues ()	Retorna la lista de incidencias asociadas a una determinada tarea en Jira.
getIssue ()	Retorna una indicencia a partir de un identificador de tarea en Jira.
getIssueTypes ()	Retorna el conjunto de tipos de incidencias definidas en Jira.
getAssignableUsers ()	Retorna la lista de usuarios a quienes se pueden asignar una tarea determinada en Jira.
getPriorities ()	Retorna la lista de prioridades de acuerdo a la definición en Jira.
uploadAttachment ()	Adiciona una evidencia (documento) a una incidencia determinada en Jira.
deleteAttachment ()	Elimina una evidencia (documento) de una incidencia determinada en Jira.
addComment ()	Adiciona un comentario a una incidencia existente en Jira.
editComment ()	Edita un comentario de una incidencia en Jira.

deleteComment ()	Elimina un comentario de una incidencia en Jira.
getAllProcess ()	Retorna la lista de procesos de *Bonita Execution Engine*.
getAllProyects ()	Retorna la lista de proyectos de Jira.
addProcessProjectAssociation ()	Realiza la asociación entre un proceso y proyecto registrándolo en la base de datos IMC Data Base.
login ()	Retorna un token a partir de un usuario y contraseña generado en la clase SessionService.

II.JiraServiceImpl

JiraServiceImpl es la clase que se encarga de manejar la la lógica de negocio relacionada a las incidencias.
Esta clase recibe las peticiones del controlador IncidentController y se comunica con:

- La capa de persistencia donde esta definida la clase ProcessDAOImpl que permite registrar, actualizar, eliminar o recuperar información relacionada a incidencias, procesos, tareas y proyectos.
- El cliente "Bonita Rest Client".
- El cliente "Jira Rest Client".

En caso de ocurrir algún tipo de error en la comunicación, la clase permite capturar las excepciones y enviar un código de respuesta con una descripción del error ocurrido al controlador IncidentController.

Los métodos principales definidos para esta clase son los siguientes:

Tabla 5. Métodos de la clase JiraServiceImpl

Nombre	Descripción
getProjects ()	Método transaccional. Retorna los proyectos en los cuales un determinado participante tiene permisos basándose en la relación que existe entre los procesos y proyectos.
createIssue ()	Método transaccional. Crea una indicencia en Jira y guarda información en la base de datos IMC Database. Este método tiene implementada la excepción AssociationNotFoundExeption que se produce cuando no se encuentra la asociación de un proceso con un proyecto.
editIssue ()	Actualizauna indicencia en Jira.
getIssues ()	Método transaccional. Retorna la lista de incidencias asociadas a una determinada tarea en Jira.
getIssue ()	Método transaccional. Retorna una indicencia a partir de un identificador de tarea en Jira.
getIssueTypes ()	Retorna el conjunto de tipos de incidencias definidas en Jira.
getAssignableUsers ()	Retorna la lista de usuarios a quienes se pueden asignar una tarea determinada en Jira.
getPriorities ()	Retorna la lista de prioridades de acuerdo a la definición en Jira.

uploadAttachment ()	Adiciona una evidencia (documento) a una incidencia determinada en Jira.
deleteAttachment ()	Elimina una evidencia (documento) de una incidencia determinada en Jira.
addComment ()	Adiciona un comentario a una incidencia existente en Jira.
editComment ()	Edita un comentario de una incidencia en Jira.
deleteComment ()	Elimina un comentario de una incidencia en Jira.
getAllProcess ()	Retorna la lista de procesos de *Bonita Execution Engine*.
getAllProyects ()	Retorna la lista de proyectos de Jira.
addProcessProjectAssociation ()	Método transaccional. Realiza la asociación entre un proceso y proyecto registrándolo en la base de datos IMC Database.

III. SessionService

SessionService es una clase que se encarga de manejar las sesiones de usuarios utilizando un token.
Este servicio se comunica con la clase LdapAuthenticationManagerImpl para realizar la autenticación de usuarioscon el servicio de directorio LDAP.
Para guardar las sesiones se utiliza la clase CopyOnWriteArrayList que es una implementación List del paquete java.util para acceso concurrente.

Los métodos definidos en este servicio son los siguientes:

Tabla 6. Métodos del servicio SessionService

Nombre	Descripción
login ()	A partir de un nombre de usuario y una contraseña se valida si el participante existe en el servicio de directorio LDAP. Si existe se genera y se retorna un token.
getUserByToken ()	Dado un token de sesión, retorno al usuario correspondiente.

IV. LdapAuthenticationManagerImpl

LdapAuthenticationManagerImpl es una clase que se conecta con un servicio de directorio LDAP y autentica a un usuario.
Para poder conectarnos y autenticarnos con el servicio de directorio LDAP debemos crear una instancia de la clase InitialDirContext con la siguiente información:

- url_LDAP
- user_LDAP
- password_LDAP
- user
- password

En nuestro caso, la información de los campos url_LDAP, user_LDAP y password_LDAP estará en el archivo de configuración LDAPAuthenticator.config.
Para poder autenticar un usuario definimos el siguiente método:

Tabla 7. Métodos de Autenticación de Usuarios

Nombre	Descripción
authenticate ()	Valida un usuario y su contraseña (user, passowrd) en el servicio de directorio LDAP.

5.2.2.2 Bonita Rest Client

Bonita Rest Client es un cliente que utilizamos para poder realizar la comunicación desde la aplicación IMC con Bonita Rest API definido en el entorno Bonita BPM.
Para esto, hicimos uso del componente "Engine API", el cual es un módulo implementado en Java que proporciona una Capa de Abstracción con la API REST de Bonita. El objetivo de su uso es poder obtener información de los procesos de negocio y de los participantes para interactuar programáticamente entre ellos [45].

Los métodosprincipales que utilizamos en este cliente fueron los siguientes:

Tabla 8. Métodos del cliente Bonita Rest Client

Nombre	Descripción
getAllProcess ()	Devuelve el conjunto de procesos de negocio deployados en Bonita BPM.

5.2.2.3 Jira Rest Client

Jira Rest Client es un cliente que nos permite realizar la comunicación con los servicios de la herramienta Atlassian Jira.
Esta comunicación la realizamos de dos maneras diferentes:

I. Utilizando el cliente JRJC (*Jira Rest Java Client)* proporcionado por la empresa Atlassian que permite comunicarse con la herramienta utilizando su propia API REST. El método de autenticación que utiliza es *Basic Authentication*. Este cliente ofrece un modelo de objetos de dominio de entidades de Jira asociadas por el lado del cliente. Los objetos representados pueden ser incidentes, prioridades, estados, usuarios, etc. [46].

II. Utilizando la api Jira Rest API. El motivo por el cual tuvimos que hacer uso directo de la API REST de Jira es que el cliente JRJC no nos ofrecía todas las operaciones que necesitábamos para alcanzar el objetivo propuesto. Para poder invocar a los servicios que expone esta API utilizamos el framework de Servicios Web RESTfull definido por Jersey.

El desarrollo de este cliente tiene como finalidad poder realizar las siguientes operaciones relacionadas al sistema de Gestión de Incidencias:

- Obtener información de proyectos.
- Obtener información de usuarios.
- Crear, actualizar, listar y visualizar incidencias.

A continuación, se detallan los métodos implementados para poder cubrir las necesidades mencionadas:

Tabla 9. Métodos del cliente Jira Rest Client

Nombre	Descripción
getProjects ()	Devuelve la lista de proyectos registrados en Jira, utilizando JRJC para la comunicación.
createIssue ()	Crea una incidencia con la opción de adjuntar evidencias en Jira, utilizando JRJC para la comunicación.
editIssue ()	Actualizauna incidencia en Jira, utilizando JRJC para la comunicación.
getIssues ()	Retorna la lista de incidenciasde una determinada tarea, utilizando JRJC para la comunicación.
getPriorities ()	Devuelve la lista de prioridades registradas en Jira, utilizando JRJC para la comunicación.
getAssignableUsers ()	Devuelve la lista de usuarios a los cuales se les puede asignar una incidencia a partir de un identificador de proyecto, utilizando la API REST para la comunicación.
getIssueTypesByProject ()	Devuelve la lista de tipos de incidencias disponibles para un determinado proyecto, utilizando JRJC para la comunicación.
getJiraIssue ()	Retorna una incidencia a partir de un identificador de tarea, utilizando JRJC para la comunicación.
uploadAttachment ()	Adiciona una evidencia (documento) a una incidencia determinada en Jira, utilizando JRJC para la comunicación.
deleteAttachment ()	Elimina una evidencia (documento) de una incidencia determinada en Jira, utilizando la API REST para la comunicación.
addComment ()	Adiciona un comentario a una incidencia en Jira, utilizando JRJC para la comunicación.
editComment ()	Actualiza un comentario a partir de un identificador de comentario, utilizando la API REST para la comunicación.
deleteComment ()	Elimina un comentario de unaincidencia en Jira, utilizando la API REST para la comunicación.

En ciertos momentos, cuando realizamos una comunicación con la herramienta Atlassian Jira podemos encontrarnos con problemas en la red, servicios caídos, problemas en el servidor, problemas de autorización, etc., por lo tanto, se pueden producir excepciones. Con el objetivo de minimizar estos problemas, creamos en el paquete unlp.tesis.exception las siguientes excepciones:

- ConnectJiraException: se lanza esta excepción en el caso de que se produzca un error de conexión en la herramienta Atlassian Jira.
- LoginJiraException: se lanza esta excepción en caso de haber error de autenticación en la herramienta Atlassian Jira.

En caso de producirse algún error diferente a los mencionados se relanzará la excepción Exception del paquete java.lang pertenecientes al Core de Java.

5.2.2.4 IMC Database

IMC Database es una base de datos definida en el motor de base de datos MySQL 5.7.18 que nos permite administrar y relacionar información referente a:

- Procesos
- Tareas
- Proyectos
- Incidencias

En la figura 17 se muestra el modelo de datos que desarrollamos para cubrir la necesidad planteada:

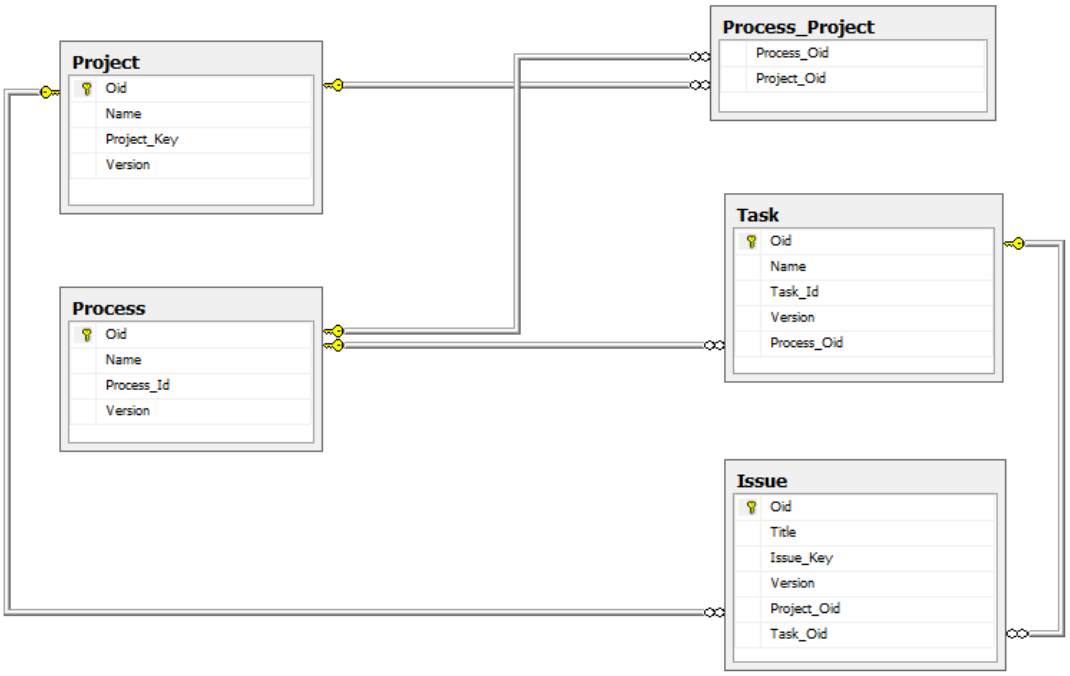

Figura 17. IMC Database - Modelo de base de datos

5.2.2.4.1 Tabla Process

Campos

- Oid: identificador único de la entidad en la aplicación IMC.
- Name: nombre del proceso de negocio registado en el entorno de Bonitasoft.
- Process_Id: identificador único de un proceso de negocioregistado en el entorno de Bonitasoft.
- Version: Utilizado por Hibernate para el manejo de concurrencia.

5.2.2.4.2 Tabla Project

Campos

- Oid: identificador único de la entidad en la aplicación IMC.
- Name: nombre del proyecto registado en la herramienta Atlassian Jira.

- Project_Key: identificador único de un proyecto registado en la herramienta Atlassian Jira.
- Version: Utilizado por Hibernate para el manejo de concurrencia.

5.2.2.4.3 Tabla Process_Project

Campos

- Process_Oid: identificador único de un proceso de negocio registrado en la aplicación IMC.
- Project_Oid: identificador único de un proyecto registado en la aplicación IMC.

5.2.2.4.4 Tabla Task

Campos

- Oid: identificador único de la entidad en la aplicación IMC.
- Name: nombre de una tarea definida en un proceso de negocio registado en el entorno de Bonitasoft.
- Task_Id: identificador único de una tarea definida en un proceso de negocio registado en el entorno de Bonitasoft.
- Version: Utilizado por Hibernate para el manejo de concurrencia.
- Process_Oid: identificador único de un proceso de negocio registrado en la aplicación IMC.

5.2.2.4.5 Tabla Issue

Campos

- Oid: identificador único de la entidad en la aplicación IMC.
- Title: nombre de una incidencia asociada a un proyecto registado en la herramienta Atlassian Jira.
- Issue_Key: identificador único de una incidencia asociada a un proyecto registado en la herramienta Atlassian Jira.
- Version: Utilizado por Hibernate para el manejo de concurrencia.
- Project_Oid: identificador único de un proyecto registrado en la aplicación IMC.
- Task_Oid: identificador único de una tarea registrada en la aplicación IMC.

5.2.3 Atlassian Jira

5.2.3.1 Jira Rest API

Jira Rest API tiene como finalidad interactuar con la herramienta Atlassian Jira programáticamente.
Esta API proporciona acceso a recursos (entidades de datos) a través de rutas URI. Para usar una API REST, su aplicación realizará una solicitud HTTP y analizará la respuesta. La Jira Rest API usa JSON como formato de comunicación y los métodos HTTP estándar como GET, PUT, POST y DELETE. Los URI para el recurso API REST de Jira tienen la siguiente estructura [47]:

Tabla 10. Estructura URI utilizado por la API Rest de Jira

http://host:port/context/rest/api-name/api-version/resource-name

En cuanto a los métodos de autenticación admitidos por esta API son los siguientes:

- Basic Authentication
- Cookie-based Authentication
- OAuth

A continuación, se listan los recursos utilizados de la API:

Tabla 11. Recursos de Jira Rest API

Operación	Descripción	Método	Recurso
Crear Incidencia	Crea una incidencia a partir de una representación JSON.	POST	/rest/api/2/issue
Editar Incidencia	Edita una incidencia a partir de una representación JSON dado id de incidencia.	PUT	/rest/api/2/issue/{issueIdOrKey}
Obtener Incidencia	Retorna una representación completa de una incidencia dado id de incidencia.	GET	/rest/api/2/issue/{issueIdOrKey}
Obtener conjunto de Incidencias	Obtiene un conjunto de incidencia dado un determinado JQL.	GET	/rest/api/2/search
Eliminar Comentario	Elimina un comentario dado un id de comentario.	DELETE	/rest/api/2/issue/comment/{commentId}
Obtener Proyectos	Retorna todos los proyectos por el cual son visibles por el usuario logueado. Si ningún usuario es logueado, retorna una lista de proyectos que son visibles cuando se usa acceso anónimo.	GET	/rest/api/2/project
Obtener Prioridades	Retorna una lista de todas las prioridades.	GET	/rest/api/2/priority
Buscar Usuarios	Retorna una lista de usuarios activos que machean con el dado string de búsqueda y permisos especificados.	GET	/rest/api/2/user/permission/search
Adjuntar archivo	Adjunta un archivo a una incidencia dado un id de incidencia.	POST	/rest/api/2/issue/{issue-key}/attachments
Eliminar adjunto	Elimina un adjunto de una incidencia dado un id de adjunto.	DELETE	/rest/api/2/attachment/{id}

5.3Definición de la arquitectura según los conceptos de integración

En esta sección estaremos explicando como se diseñó la arquitectura integración entre los entornos Bonitasoft y Atlassian Jira según lo definido en la Sección 5.1 “Arquitectura de integración Bonita BPM con el gestor de incidencias Jira” considerando los siguientes puntos de vista:

- Asociación entre Procesos y Proyectos.
- Asociación entre Participantes y Usuarios.

5.3.1 Asociación de Procesos y Proyectos

Según la definición provista en la Sección 5.2.2 Incident Management Connector, la asociación de Procesos y Proyectos se gestionará en la aplicación intermedia IMC.

Para poder realizar la carga inicial y asociar los procesos de negocio disponibles en el entorno Bonita XP con los proyectos disponibles en la herramienta Atlassian Jira en la base de datos “IMC Database” creamos la clase LoaderProcessProject.java que contiene la siguiente información:

I. Para el ambito de Bonita XP encontraremos desplegado el siguiente proceso de negocio:

   - Proceso:
     1. Procurement Request
   - Tareas:
     1. Specify need & identify suitable suppliers
     2. Complete quotation
     3. Review quotations & select supplier
     4. Update request after quotations received
     5. Update request after supplier selected
     6. Abort request

II. En la herramienta Atlassian Jira tendremos disponibles los siguientes proyectos:

   - Compras de Insumos informáticos.
   - Reparaciones de Pc’s.

5.3.2 Asociación de Participantes y Usuarios

La necesidad de contar con la asociación de los Participantes pertenecientes a Bonitasoft con los Usuarios pertenecientes al entorno Atlassian Jira se debe a que nos encontramos con la problemática de tener dos repositorios de usuarios diferentes y sin relación alguna. De forma nativa, tanto Bonitasoft como Atlassian Jira administran a sus usuarios con su propia base de datos de su propio ámbito, pero ambos entornos cuentan con la característica de poder conectarse con otras fuentes de datos para administrar usuarios como por ejemplo utilizando servidores de directorios LDAP.

Según lo definido en la Sección 4.4 Asosiación de Participantes – Usuarios, es necesario implementar un proceso de integración entre los participantes y usuarios para poder contar con un único gestor de usuarios centralizado entre ambos sistemas.

En base a lo mencionado, tomamos la decisión de utilizar un servidor de directorios LDAP para la gestión de usuarios centralizado con el objetivo de administrar y sincronizar participantes y usuarios. Luego de haber elegido el tipo de servicio debemos elegir que implementación vamos a utilizar para la integración. Siguiente la línea del software libre [63], en el marco del trabajo utilizaremos la implementación OpenLDAP.

El siguiente paso que realizamos fue ejecutar pruebas de concepto y para esto instalamos el servidor de directorio OpenLDAP en un Sistema Operativo Linux Ubuntu 16.04 tal como se explica en el Anexo 1.3. Como complemento tuvimos que instalar el software PhpAdmin como herramienta front-end para crear, modificar y visualizar la estructura de grupos y usuarios del sistema.

Una vez finalizada la instalación del software base, para poder administrar y sincronizar los entornos de Bonitasoft y Atlassian Jira debemos realizar las siguientes tareas:

I. Crear una organización en OpenLDAP.
II. Configurar el entorno Bonitasoft en base a la organización definida.
III. Configurar la herramienta Atlassian Jira en base a la organización definida.

5.3.2.1 Creación de una organización en OpenLDAP

A continuación, explicaremos los pasos a seguir para poder desarrollar el modelo de organización en OpenLDAP.

Paso 1. En una primera instancia, creamos una organización llamada Tesis_Organization.

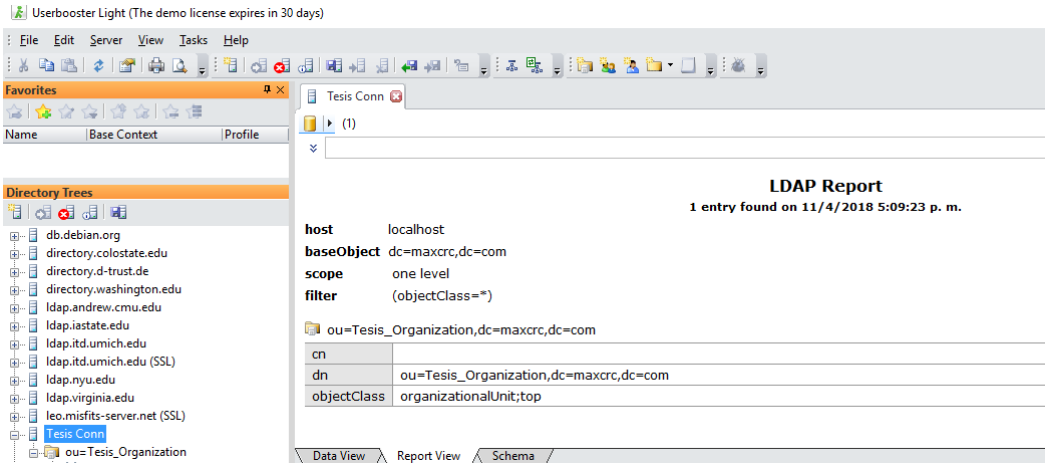

Figura 18. OpenLDAP - Organización Tesis_Organization

Paso 2. Luego, creamos dos grupos dentro de la organización creada en el Paso 1 con un conjunto de usuarios para cada uno de ellos.

- Grupo Solicitantes
  - Juan Torres
  - Adrían López
  - Marcos Pérez

- Grupo Proveedores
    - DataSoft SA
    - TecnoPoint SA
    - CompuTools SRL

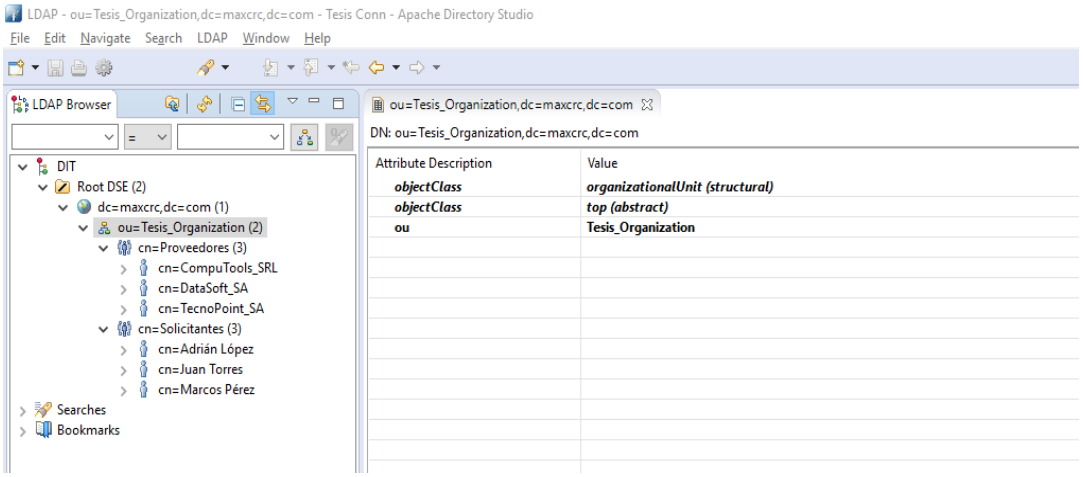

Figura 19. OpenLDAP – Grupos y Usuarios de la organización Tesis_Organization

Paso 3. Luego de haber creado la organización por completo estamos en condiciones de poder implementar las configuraciones en los sistemas de Bonitasoft y Atlassian Jira.

5.3.2.2 Configuración de Bonitasoft con el servidor OpenLDAP

En el transcurso de la investigación nos dimos cuenta que Bonitasoft en su versión Community no soporta integración directa con un servicio de directorios LDAP. Por esta razón, tuvimos que utilizar el componente "LDAP Synchronizer & LDAP Authenticator" provisto por bpms.help. Este componente consiste en un proceso de sincronización de usuarios con LDAP y cuenta con una libreria adicional que permite autenticar usuarios en LDAP cuando se loguean en Bonita BPM Portal.
LDAPSynchronizer y LDAPAuthenticator están diseñados para brindar integración LDAP con la version Community de Bonitasoft, cuenta con una sincronización periódica de usuarios y permiten el logueo dentro del Portal de Bonita haciendo uso de la contraseña almacenada en el servidor LDAP. Desde el punto de vista del negocio, estos módulos pueden ser usados de forma separada. [65]

Según lo mencionado anteriormente, el componente está formado por dos partes:

I. Un proceso que permite sincronizar usuarios.

II. Cuenta con el archivo bpms.help-bonita.authentication.ldap-1.0.0.jar que se encarga de realizar la autenticación en un servidor de usuarios LDAP.

Para poder realizar la integración de Bonitasoft con el servidor de directorio OpenLDAP debemos tener en cuenta dos conceptos:

I. Autenticación

El concepto de autenticación hace referencia a la validación del usuario y contraseña dentro del servicio de directorio OpenLDAP. La información relacionada a cada usuario estará registrada en el modelo de datos perteneciente al entorno de Bonitasoft donde se realizaremos una validación en cuanto a la existencia del usuario, pero no se verificará su contraseña dado que la misma se registró con anterioridad.

II. Sincronización

Con la sincronización de datos mantenemos actualizados la información almacenada en el entorno de Bonitasoft con los datos del servidor de directorios LDAP. Los datos sincronizados van en una sola dirección, solo se mueven desde el servicio de directorios LDAP al entorno Bonitasoft.

Para cumplir con los conceptos de sincronización y autenticación tuvimos que implementar y modificar diferentes archivos pertenecientes al entorno de Bonitasoft para poder operar con el servicio de directorio OpenLDAP.
Ahora detallaremos los pasos que desarrollamos para lograr la integración:

I. Autenticación Bonitasoft – OpenLDAP

Paso 1. Como el sistema Bonitasoft utiliza proyectos Maven creamos una dependencia con el archivo bpms.help-bonita.authentication.ldap-1.0.0.jar provisto por bpms.help con las siguientes características:

Tabla 12. Dependecias en bpms.help

ArtefactId: ldapAuthentication groupId: help.bpms.bonita.authentication version: 1.0.0

Paso 2. Agregamos la dependencia en el archivo pom.xml del proyecto console-server para poder tener acceso al artefacto.

Paso 3. Se modificó la clase AuthenticationManagerFactory del proyecto console-server para que podemos tener una referencia a la clase LdapAuthenticationManagerImpl y hacer uso de la implementación de autenticación.

Paso 4. Tuvimos que modificar la clase AuthenticationServiceImpl dentro del proyecto bonita-authentication-api-impl para que no realice la validación de usuario y contraseña contra la base de datos de Bonitasoft dado que esta validación se hizo previamente contra el servicio de directorioOpenLDAP.

Paso 5. Por último, tuvimos que desarrollar una configuración en el archivo LDAPAuthenticator.config provisto por bpms.help tal como se indica a continuación:

Tabla 13. Configuración del archivo LDAPAuthenticator.config

ldap_connection_host = localhost ldap_connection_port = 389 ldap_connection_user = cn=admin,dc=maxcrc,dc=com ldap_connection_pass = Tesis ldap_auth_search_base = dc=maxcrc,dc=com

```
ldap_auth_search_filter=(&(objectClass=inetorgperson)(cn=@username@))
bonita_fake_userpassword = 7H@NK$_T()_www.bpms.help
```

Donde:

- ldap_connection_host: Campo Requerido. Dirección del servicio de directorio OpenLDAP.
- ldap_connection_port: Campo Requerido. Puerto del servicio de directorio OpenLDAP.
- ldap_connection_user: Campo Requerido. Identificación del usuario administrador del servicio de directorio OpenLDAP.
- ldap_connection_pass: Campo Requerido. Contraseña del usuario del servicio de directorio LDAP.
- ldap_auth_search_base: Campo Requerido. Elemento raíz del árbol LDAP para buscar usuarios (DN).
- ldap_auth_search_filter: Campo Requerido. Secuencia de consulta LDAP para buscar el usuario especificado en el formulario de inicio de sesión.
- bonita_fake_userpassword: Campo Requerido. Contraseña que se almacena en la base de datos de Bonitasoft para cada usuario LDAP.

II. Sincronización Bonitasoft – OpenLDAP

Paso 1. La primera acción que llevamos a cabo fue importar el archivo LDAPSynchronizer-1.0.bos provisto por bpms.help dentro del entorno Bonita Studio.

Paso 2. Ahora explicaremos los parámetros de configuración que utilizamos para el proceso LDAPSynchronizer.config.

Tabla 14. Configuración del archivo LDAPSynchronizer

```
ldap_connection_host = localhost
ldap_connection_port = 389
ldap_connection_user = cn=admin,dc=maxcrc,dc=com
ldap_connection_pass = Tesis
ldap_groups_search_dn=Tesis_Organization,dc=maxcrc,dc=com
ldap_groups_search_filter= (&(objectClass=groupOfNames))
ldap_mapping_firstname=givenName
ldap_mapping_lastname=sn
ldap_mapping_perso_email=mail
ldap_mapping_username=cn
```

Donde:

- ldap_connection_host: Campo Requerido. Dirección del servicio de directorio LDAP.
- ldap_connection_port: Campo Requerido. Puerto del servicio de directorio LDAP.

- ldap_connection_user: Campo Requerido. UID de usuario LDAP (el usuario debe tener suficientes privilegios en el servicio de directorio LDAP).
- ldap_connection_pass: Campo Requerido. Contraseña del usuario LDAP.
- ldap_groups_search_dn: Campo Requerido. Elemento raíz del árbol LDAP para buscar grupos (DN).
- ldap_groups_search_filter: Campo Requerido. Secuencia de consulta LDAP para filtrar sólo las entradas de grupo válidas.
- ldap_mapping_firstname: Campo Opcional. Nombre del usuario en LDAP.
- ldap_mapping_lastname: Campo Opcional. Apellido del usuario en LDAP.
- ldap_mapping_perso_email: Campo Opcional. Email del usuario en LDAP.
- Configuramos el schedule del proceso para que realice la sincronización automáticamente con una periodicidad de 1 hora.
- Exportamos el proceso desde Bonita Studio generando un archivo LDAPSynchronizer--1.0.bar.

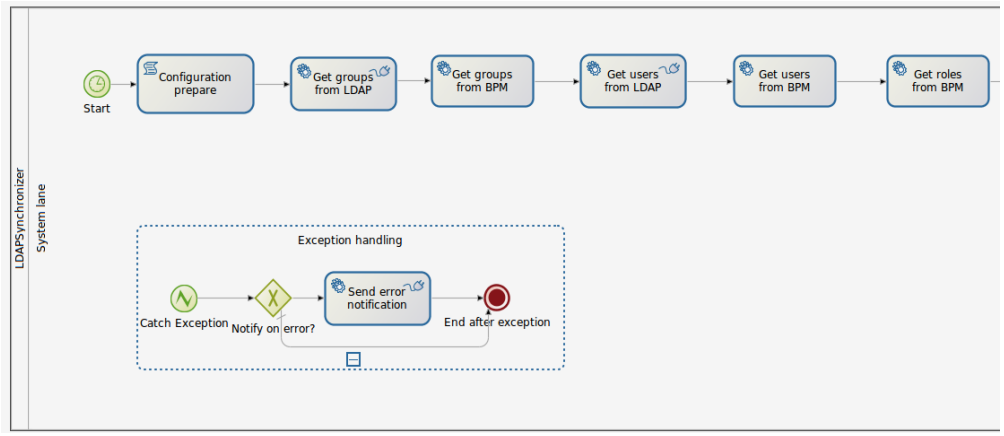

Figura 20 (I). Proceso LDAPSynchronizer

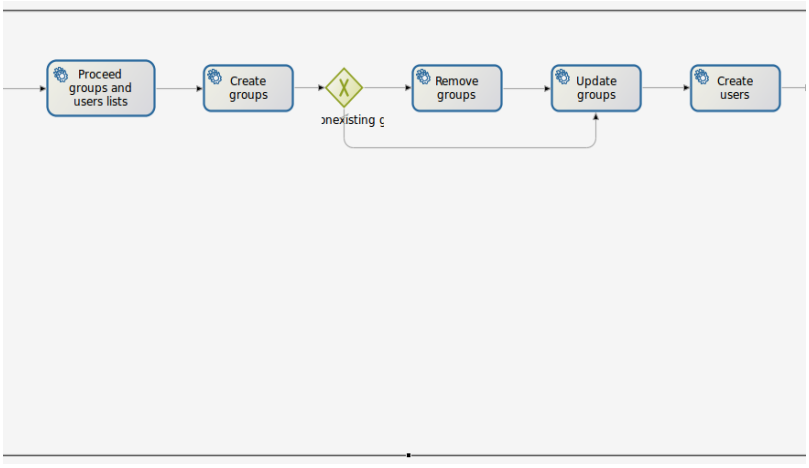

Figura 20 (II). Proceso LDAPSynchronizer

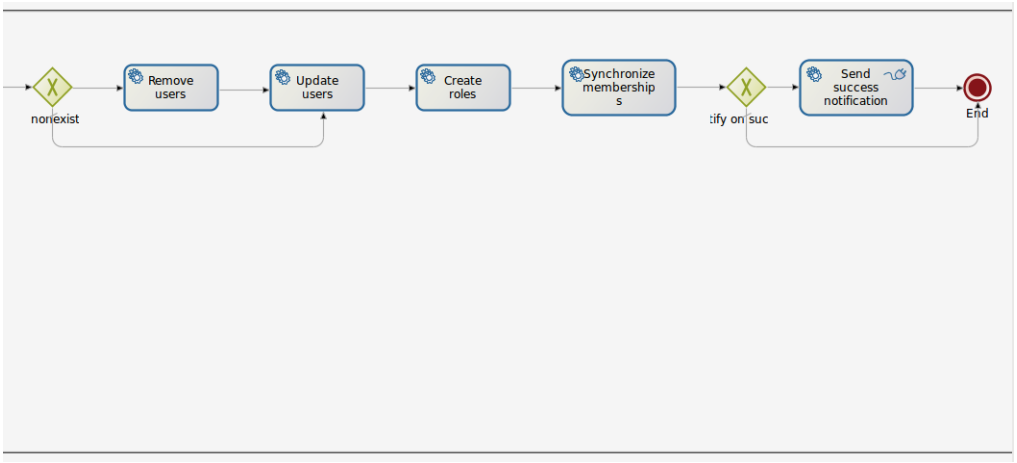

Figura 20 (III). Proceso LDAPSynchronizer

Configuración de Atlassian Jira con el servidor OpenLDAP

Paso 1. La configuración con el servicio de directorio OpenLDAP se inicia dentro de la opción de menú “Administración de Usuarios” perteneciente a la sección Administración de la herramienta Atlassian Jira.

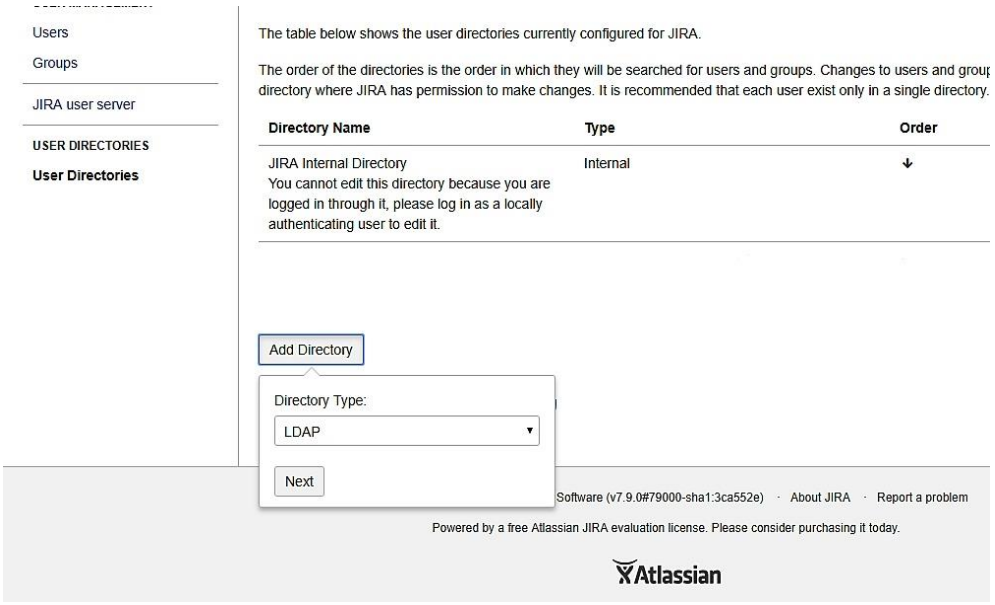

Figura 21. Pantalla Principal - Administración de Usuarios

Paso 2. Dado que la herramienta nos ofrece una pantalla amigable para poder realizar la configuración con el servicio de directorio OpenLDAP no necesitamos adicionar y/o modificar archivos de configuración en del ambiente. En primer lugar, especificamos la información relacionada a la configuración del servidor OpenLDAP.

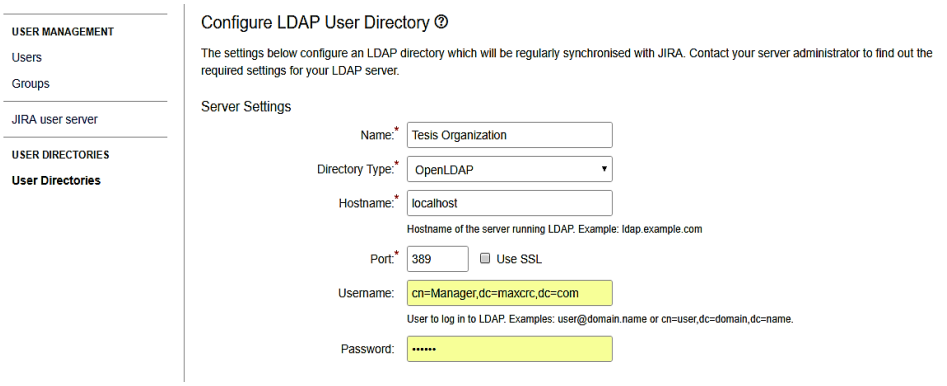

Figura 22. Configuración del ServidorOpenLDAP

Paso 3. Ahora, debemos describir la información relacionada al esquema LDAP utilizado.

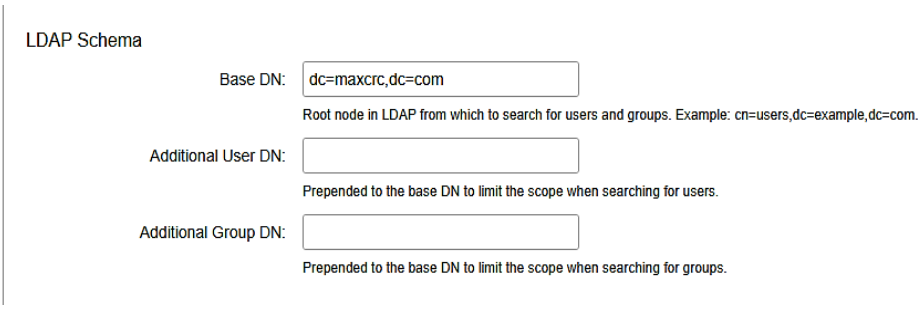

Figura 23. Esquema del Servidor OpenLDAP

Paso 4. En esta instancia, asignamos los permisos referentes al servicio del servidor LDAP.

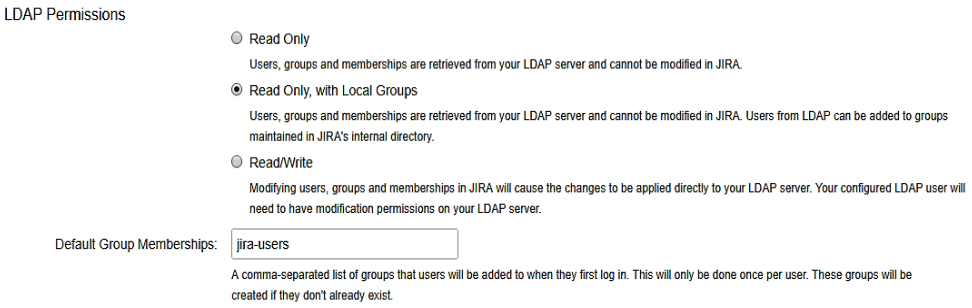

Figura 24. Permisos otorgados al Servidor OpenLDAP

Paso 5. En esta instancia, definimos mediante que objetos de Java obtendremos la información referente al Esquema de Usuario definido en el servidor LDAP. Este es reconocido automáticamente dentro el entorno de Atlassian Jira. En nuestro caso, se utilizará la clase “inetOrgPerson”.

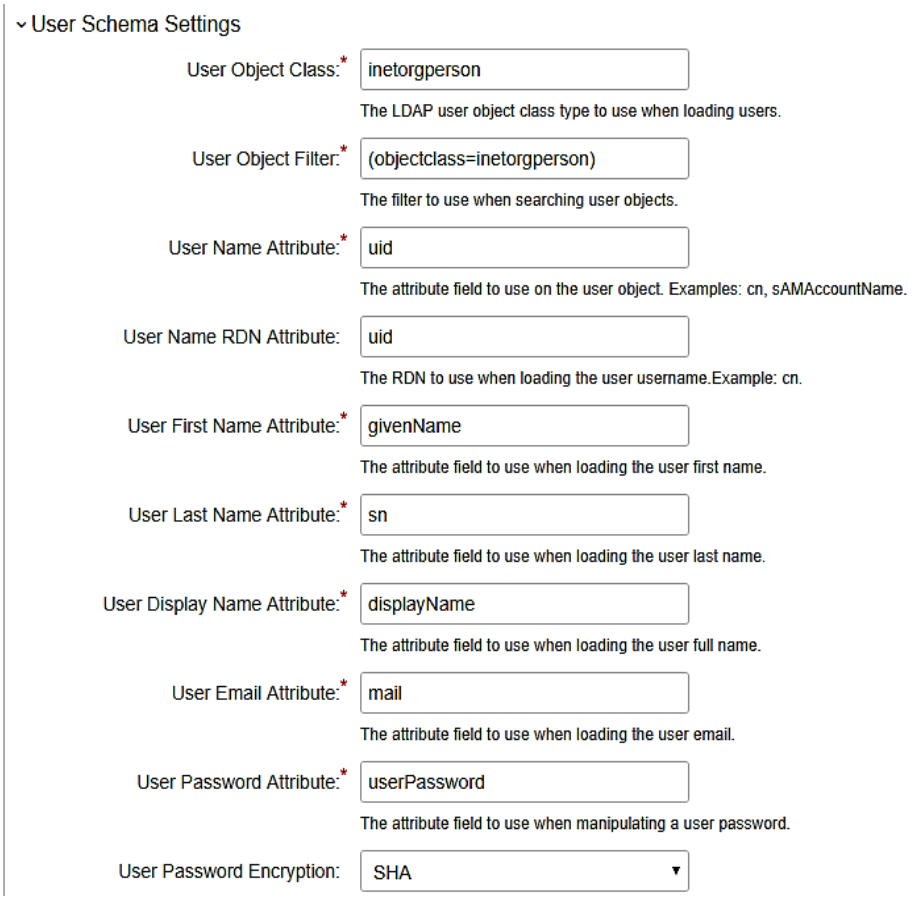

Figura 25. Seteo del Esquema de Usuarios

Paso 6. Luego ingresamos los datos para setear el esquema de grupos.

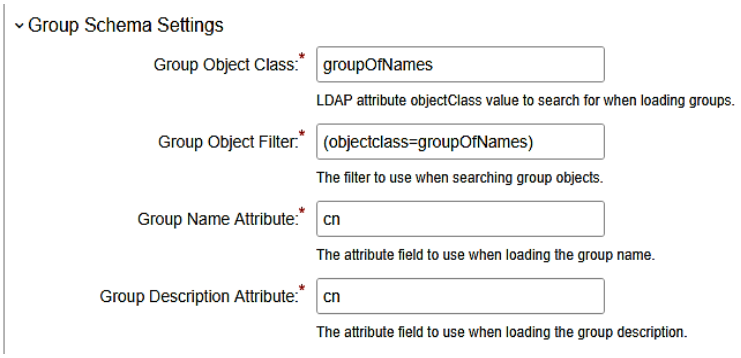

Figura 26. Seteo del Esquema de Grupos

Paso 7. Como último paso de configuración, seteamos el esquema de Membership y luego presionamos la opción “Save And Test” para guardar la configuración completa.

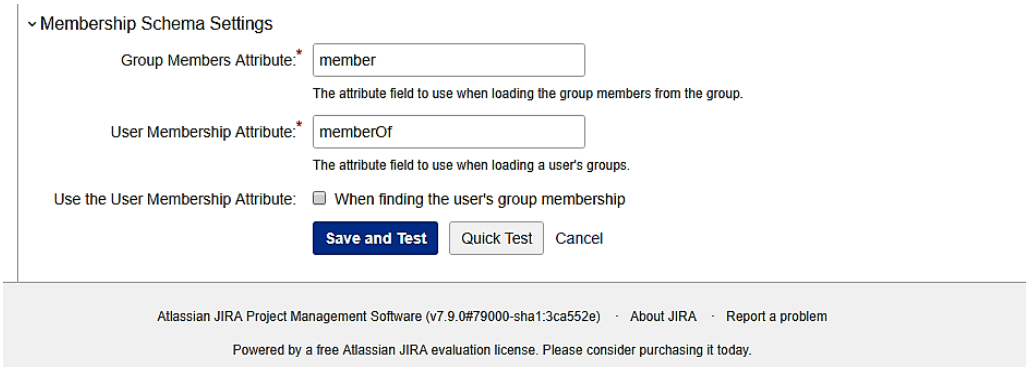

Figura 27. Seteo del Esquema Membership

Paso 8. Luego de haber configurado el esquema por completo, para comprobar que el circuito de autenticación es correcto debemos ingresar un nombre usuario y contraseña de un usuario que se encuentre registrado en el servidor OpenLDAP.

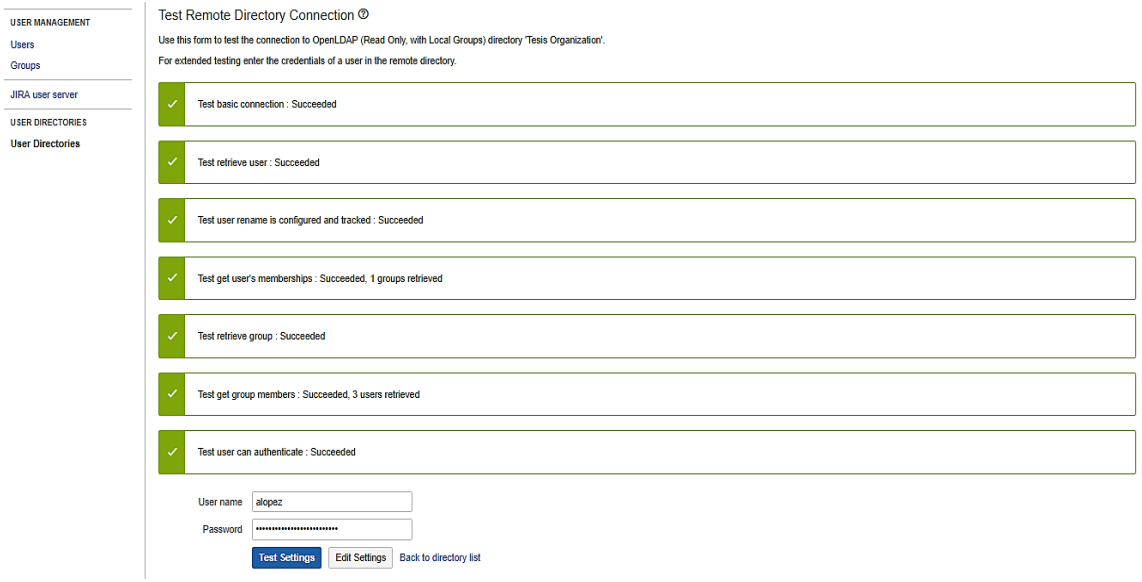

Figura 28. Test de Autenticación desde la herramienta Jira conOpenLDAP

Paso 9. Una vez que la prueba de Autenticación es satisfactoria, seleccionamos la opción “User Directories” y se podrá observar que hasta el momento no se encuentran sincronizados los usuarios del servicio de directorio OpenLDAP dentro del entorno de Atlassian Jira.

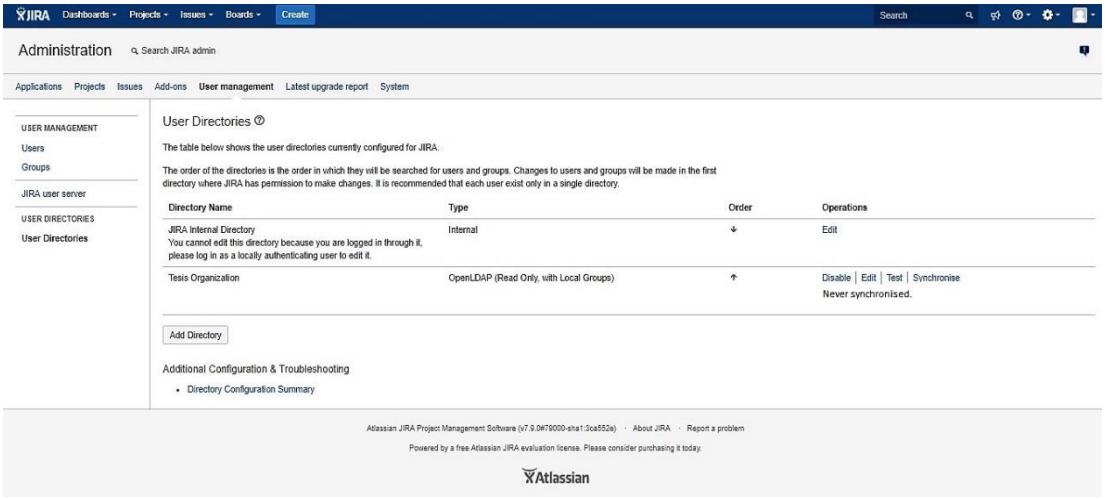

Figura 29. Directorio OpenLDAP No Sincronizado

Paso 10. Ahora seleccionamos la opción “Synchronise” del directorio Tesis_Organization para sincronizar todos usuarios almacenados en el servicio de directorio OpenLDAP con la herramienta Atlassian Jira.

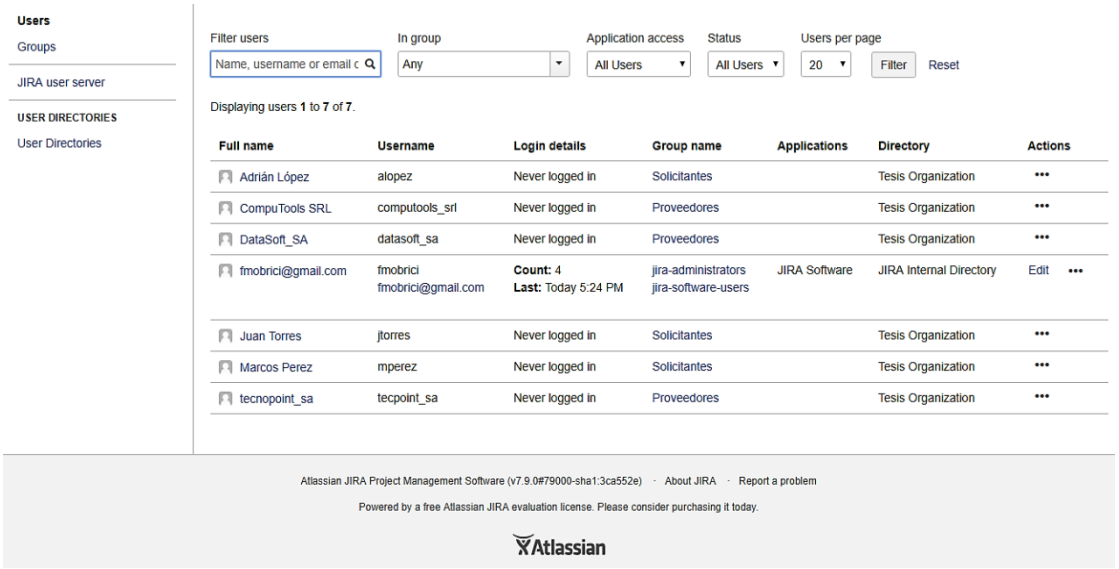

Figura 30. Lista de usuarios sincronizados con OpenLDAP

5.4 Caso de Estudio

Luego de haber presentado y explicado una arquitectura de integración entre los entornos de Bonitasoft y Atlassian Jira en la Sección 5.2 y haber realizado las instalaciones, la configuración, la carga inicial y sincronización de datos relacionados a procesos, proyectos y usuarios en los respectivos ambientes según lo definido en la Sección 5.3, estamos en condiciones de desarrollar un caso de estudio en miras a validar el esquema propuesto para esta tesis.

La demostración del caso de estudio estará dividida en varias etapas tal como se detalla a continuación:

- Etapa I. Se describe el proceso, las configuraciones realizadas en LDAP, actores en Bonitasoft, proyectos creados en Atlassian Jira y los registros creados en la base de datos IMC DataBase.
- Etapa II. Se visualiza como fueron sincronizados los usuarios y grupos tanto en el entorno de Bonitasoft como en Atlassian Jira.
- Etapa III. Se inicia el proceso y se muestra con distintos participantes las funcionalidades dentro de la sección de Incidencias.
- Etapa VI. Se exponen conclusiones.

5.4.1 Etapa I

El proceso de negocio que utilizaremos para el caso de estudio consiste en solicitar un pedido de cotización en el que intervienen deversos proveedores para un servicio determinado, el cual puede ser rechazado o aceptado eligiendo una cotización para un proveedor en particular. El proceso en cuestión se denomina "*Procurement Request*".

El proceso se inicia con un pedido de cotización el cual el solicitante completa los proveedores involucrados y el servicio que necesita. Luego, un participante puede, en base a la solicitud anteriormente creada, completar la cotización para cada uno de los proveedores. Una vez completada la cotización, un solicitante puede elegir una cotización para un proveedor en particular o rechazar todas las cotizaciones. De acuerdo a opción elegida por el solicitante, una solicitud quedará terminada de forma exitosa o cancelada.

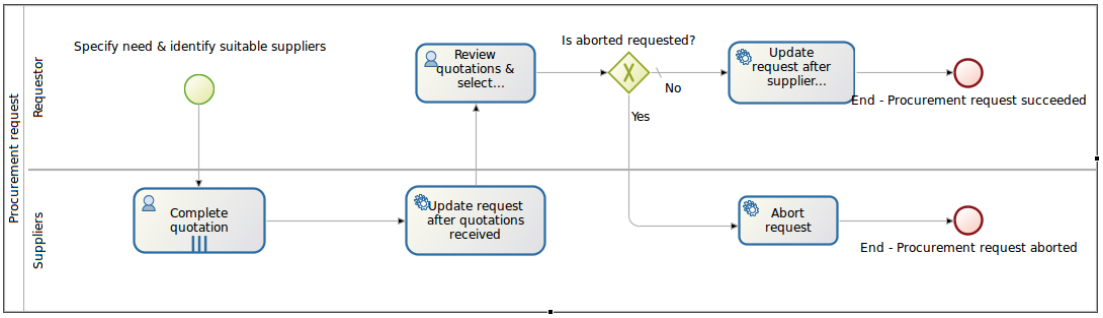

Figura 31 - Proceso Procurement Request

5.4.1.1 Configuración en OpenLDAP

Dentro del servidor de servicios OpenLDAP es necesario realizar la configuración de toda la organización para que pueda ser sincronizada por los ambientes Atlassian Jira y Bonitasoft.

Las entradas configuradas son:

I. Organización: Representa la organización donde van a estar configurados y agrupados todos los usuarios.
La organización configurada es: **Tesis_Organization**.

II. Grupos: Utilizados para estructurar la organización. Determina qué conjunto de usuarios pueden tomar determinada tarea y cuales trabajan únicamente atendiendo incidencias.
Los grupos configurados son:
   - Proveedores
   - Solicitantes
   - Soporte

III. Usuarios: Son todos los usuarios de la organización los cuales son clasificados a través de los grupos.
Los usuarios configurados por grupos son:

   - Proveedores: Daniel Acosta, Juan Romero.
   - Solicitantes: Adrián López, Juan Torres, Walter Bates.
   - Soporte: Carlos Martinez, David Acevedo.

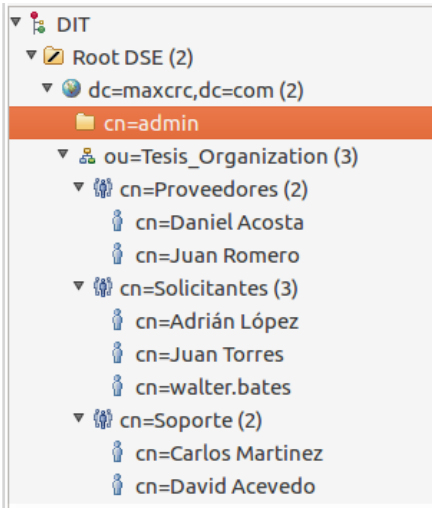

Figura 32 – Configuración OpenLDAP

5.4.1.2 Cofiguración de Actores en Bonitasoft

Dado que es necesario determinar qué conjunto de usuarios está habilitado para operar en cada paso del proceso se configuraron los siguientes actores:

I. Solicitante: Encargado de iniciar una solicitud de cotización y de elegir o rechazar una cotización. Está asociado con el grupo Solicitantes.

II. Proveedor: Encargado de completar una cotización para un proveedor determinado. Está asociado con el grupo Proveedores.

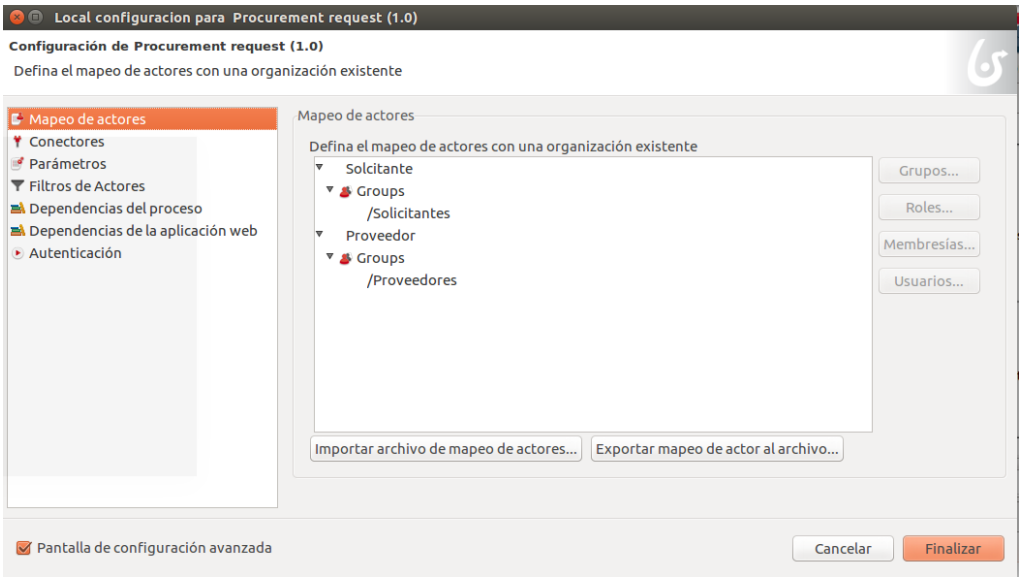

Figura 33 - Definición de mapeos de actores en Bonitasoft

5.4.1.3 Proyectos creados en Atlassian Jira

Dentro del entorno Atlassian Jira se crearon distintos proyectos los cuales podrán ser utilizados para vincular incidencias dentro de las tareas manuales del proceso creado. Los proyectos configurados son:

I. Soporte de Infraestructura: Utilizados para incidencias relacionadas a infraestructura como, problemas a conexión de vpn, problemas de red etc. La key configurada es SDI.

II. Soporte de Aplicaciones: Utilizado para incidencias relacionadas a problemas con sistemas informáticos como problemas de acceso, problemas de permisos, errores en determinadas secciones etc. La key configurada es SDA.

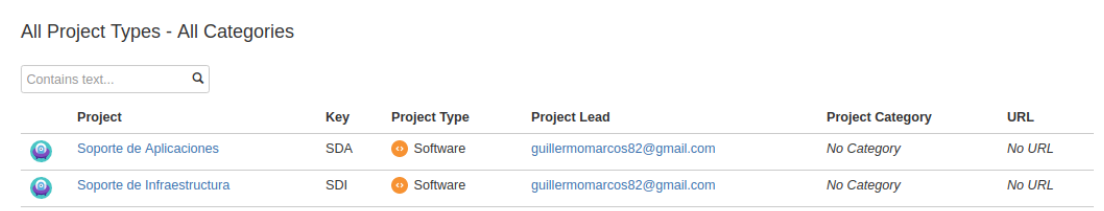

Figura 34 - Visualización de proyectos en Atlassian Jira

5.4.1.4 Creación de registros en la Base de Datos IMC DataBase

Debido a que el módulo “Asociación Procesos con Proyectos” se encuentra fuera del alcance de la tesis, realizaremos la registración manual a distintas tablas en la base de datos según el modelo definido en la Sección 5.2.2.4 IMC DataBase para cubrir esta funcionalidad.
Por lo tanto, a nivel conceptual, deberemos realizar las siguientes acciones:

- Daremos de alta los proyectos “Soporte de Aplicaciones” y “Soporte de Infraestructura” creados en el ambiente de Atlassian Jira.
- Daremos de alta al proceso “Procurement Request” perteneciente al entorno de Bonitasoft.
- Asociaremos los proyectos “Soporte de Aplicaciones” y “Soporte de Infraestructura” con el proceso “Procurement Request”.

A continuación, se detallan las tuplas registradas en la base de datos “IMC DataBase”:

Tabla PROCESS

Fila 1
OID: 1
VERSION: 0
NAME: Procurement request
PROCESS_ID: Procurement request

Tabla PROJECT

Fila 1
OID 1
VERSION: 0
NAME: Soporte de Aplicaciones
PROJECT_KEY: SDA

Fila 2
OID: 2
VERSION: 0
NAME: Soporte de Infraestructura
PROJECT_KEY: SDI

Tabla PROCESS_PROJECT

Fila 1
PROCESS_OID: 1
PROJECT_OID: 1

Fila 2
PROCESS_OID: 1
PROJECT_OID: 2

5.4.2 Etapa II

En esta sección mostramos cómo en base a la configuración de la organización descripta en la Sección 5.4.1.1 Configuración en OpenLDAP, implementaremos la sincronización de grupos y usuarios dentro de los entornos Bonitasoft y Atlassian Jira.

5.4.2.1 Sincronización en Atlassian Jira

En la Figura 35, se muestra el listado de usuarios creados luego de ejecutar la funcionalidad de sincronización dentro de la sección “User Management” de Atlassian Jira.

David Acevedo	David Acevedo david.acevedo@gmail.com	**Count:** 2 **Last:** Today 9:07 PM	jira-administrators jira-software-users jira-users	JIRA Software	LDAP server	•••
Juan Romero	Juan Romero juan.romero@hotmail.com	**Count:** 11 **Last:** lunes 10:06 PM	jira-administrators jira-software-users jira-users Proveedores	JIRA Software	LDAP server	•••
Juan Torres	Juan Torres juantorres@gmail.com	Never logged in	Solicitantes		LDAP server	•••
~~Marcos Perez [X]~~	~~Marcos Pérez~~ (Inactive) marcos.perez@hotmail.com	Never logged in			LDAP server	•••
~~tecnopoint_sa [X]~~	~~TecnoPoint_SA~~ (Inactive) tecnopoint@gmail.com	Never logged in			LDAP server	•••
Walter Bates	walter.bates walter.bates@gmail.com	Never logged in	Solicitantes		LDAP server	•••

Figura 35 – Listado de Usuarios en Atlassian Jira

5.4.2.2 Sincronización en Bonitasoft

En la Figura 36, se muestra el listado de usuarios y grupos sincronizados luego de ejecutar el proceso “LDAPSynchronizer”.

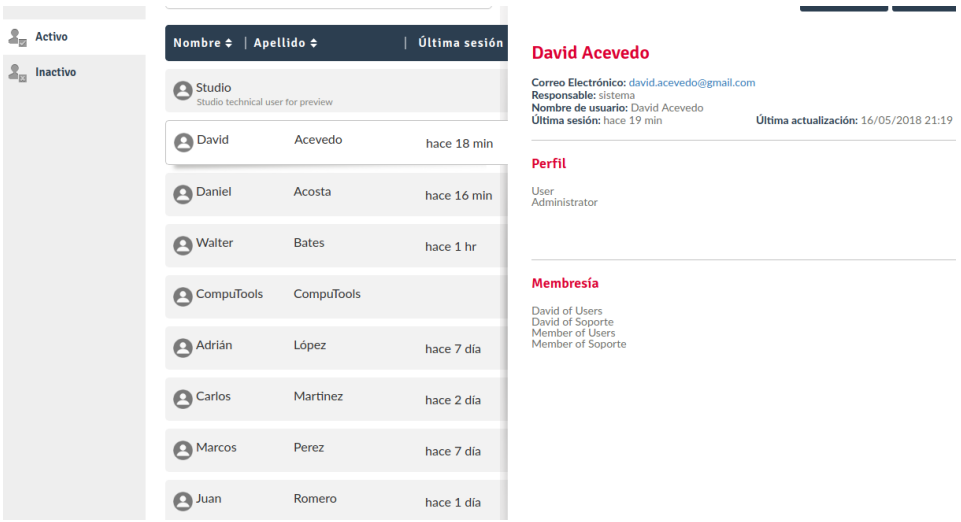

Figura 36 – Usuarios sincronizados en Bonitasoft

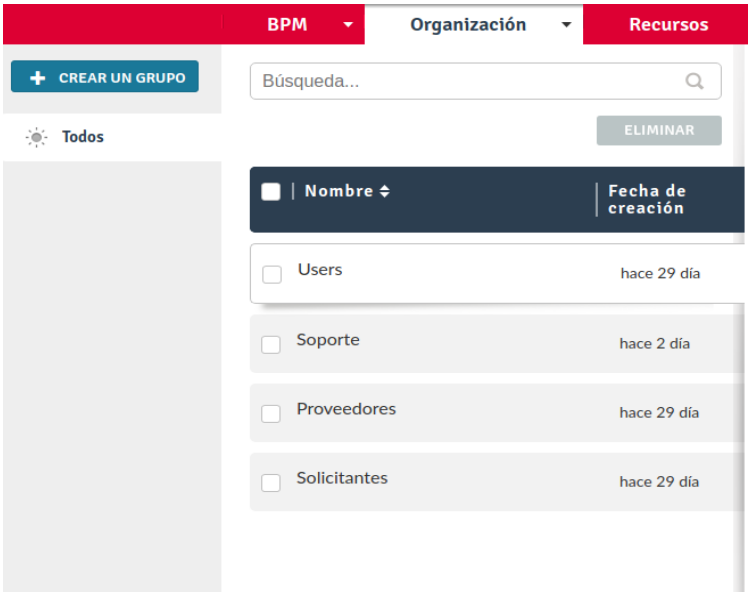

Figura 37 – Grupos sincronizados en Bonitasoft

En el momento de implementar la sincronización y autorización según lo definido en la Sección 5.4.1.1 Configuración de Bonitasoft con el servidor de directorio OpenLDAP podemos observar que centralizamos toda la gestión de la organización en un lugar único repositorio centralizado, por lo tanto, esto nos generará a futuro menor costo en la administración de usuarios ya que no tendremos que gestionar los entornos de Bonitasoft y Atlassian Jira por separado. Adicionalmente, con la implementación propuesta los participantes tendrán como ventaja que solo deberán loguearse en un único entorno, el de Bonitasoft, generando de esta manera una mejora de importancia en cuanto a la usabilidad y disminución de esfuerzo.

5.4.3 Etapa III

En esta sección instanciamos el flujo “Procurement Request” con el fin de realizar un caso de prueba donde un participante asociado al actor “*Proveedor”* al querer realizar la tarea “Completar Cotización” se encuentre con el problema de no poder acceder a la sección de catálogos de una aplicación externa.
De esta forma, mostraremos las interacciones de los participantes de Bonitasoft y los usuarios del área de soporte pertenecientes al entorno de Atlassian Jira haciendo uso de la nueva sección “Incidencias” dentro del portal de Bonita XP.

5.4.3.1 Creación de una Solicitud

En este caso de prueba, el proceso se inicia a través del participante “Juan Torres”, quién se encuentra configurado dentro el grupo de actores “*Solicitante”,* que desea completar una solicitud de cotización. Para esto, el participante deberá completar un título, la descripción del problema y los posibles proveedores involucrados tal como se especifica en las Figuras 38 y 39.

Figura 38 – Registro de una Solicitud de Servicio

Figura 39 – Selección de Proveedores

5.4.3.2 Creación de Incidencias

Una vez de completado el formulario por un actor de tipo solicitante *"Solicitante"*, la solicitud pasa a la tarea manual "Completar Cotización"; la cual deberá ser tomada por un actor de tipo "*Proveedor".* En este caso, seleccionamos al proveedor *"Daniel Acosta"* que toma la tarea, y en ese momento se ve imposibilitado de continuar con la gestión de la solicitud debido a problemas técnicos. Al identificar este inconveniente, el actor "Proveedor" deberá crear una incidencia realizando los siguientes pasos:

1. Selecciona la solapa "Incidencias".
2. Para crear una nueva incidencia, presiona el botón "Nuevo".
3. Selecciona un técnico (en nuestro caso asignamos al técnico Carlos Martínez perteneciente al área de Soporte).
4. Describe el problema.
5. Luego, registra la incidencia.

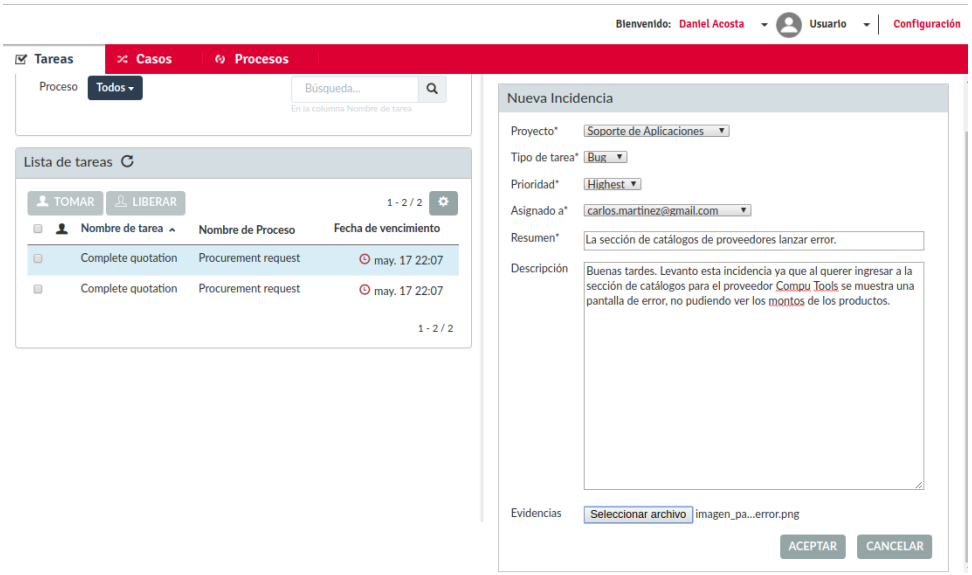

Figura 40 – Creación de una Incidencia

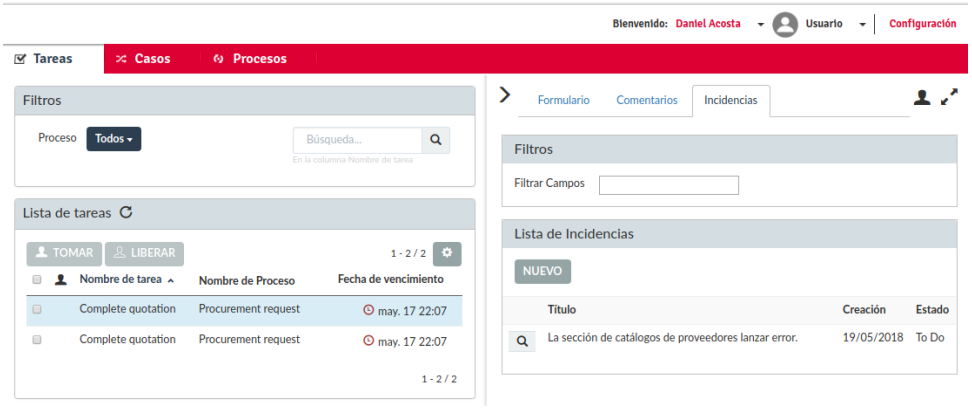

Figura 41 – Visualización de Incidencias

5.4.3.3 Registración de comentarios

Los participantes pertenecientes a un mismo grupo de actores podrán visualizar, seleccionar y editar todas las incidencias registradas que se encuentren asociadas a cada tarea en particular. En caso de ser necesario, un participante tendrá la posibilidad de adicionar comentarios o adjuntar evidencias a una incidencia ya registrada.

A modo de ejemplo, el proveedor *"Juan Romero"* registra una observación sobre la incidencia creada previamente por "*Daniel Acosta*".

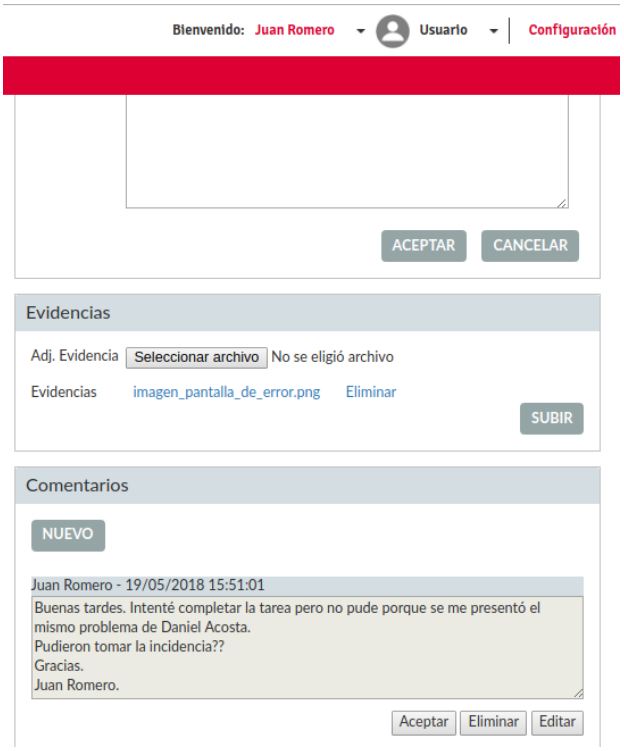

Figura 42 – Registración de Comentarios (I)

Luego de que el proveedor *"Juan Romero"* haya registrado un comentario sobre la incidencia creada por el proveedor "*Daniel Acosta*" en el portal Bonita UX, el técnico Carlos Martínez, perteneciente al área de Soporte en el sistema Atlassian Jira, podrá visualizar la trazabilidad completa de la incidencia registrada por todos los actores de Bonitasoft.

Luego, desde el entorno de Atlassian Jira, el técnico Carlos Martínez tendrá la posibilidad de gestionar las incidencias reportadas y responder comentarios a cada grupo de actores. Cabe mencionar que todas las acciones realizadas por los técnicos desde la herramienta Atlassian Jira se verán reflejados dinámicamente en el entorno de Bonitasoft.

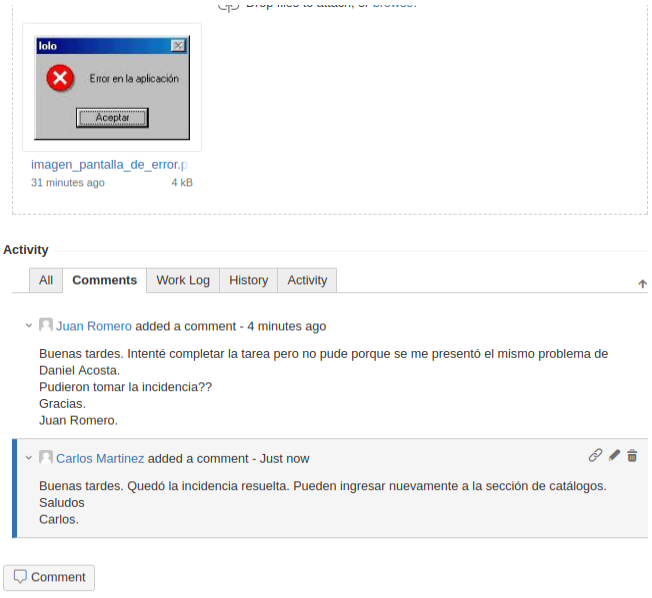

Figura 43 – Registración de Comentarios (II)

Capítulo VI – Resultados y Conclusiones

El capítulo final estará dividido de la siguiente manera:

- En la Sección 6.1 estaremos exponiendo un análisis de los resultados obtenidos luego haber realizado la integración.
- En la Sección 6.2 describiremos las conclusiones finales del trabajo.
- Por último, en la Sección 6.3 estaremos especificando algunas de las posibles extensiones o trabajos futuros que podrían llegar a investigarse y desarrollarse tomando como base la tesis en cuestión.

6.1 Resultados de la investigación

En esta sección, estaremos exponiendo un análisis de los resultados obtenidos luego de haber logrado la integración entre los entornos de Bonitasoft y Atlassian Jira, tomando como punto de partida los marcos teóricos de los sistemas Bonitasoft definido en la Sección 2.3, Atlassian Jira definido en la Sección 3.4 e Identificación de los Conceptos Claves para llegar a la integración definido en el Capítulo IV, tal como se especifica a continuación:

- Implementación de la aplicación Incident Management Connector (IMC). A través de este aplicativo logramos las siguientes funcionalidades de integración:
    - Unificación y sincronización de usuarios de los entornos Bonitasoft y Atlassian Jira en un único repositorio de usuarios.
    - Asociación de participantes pertenecientes al entorno de Bonitasoft con usuarios pertenecientes al entorno de Atlassian.
    - Asociación de procesos del entorno de Bonitasoft con proyectos definidos en el entorno de Atlassian Jira.
    - Interoperabilidad con la API de Bonitasoft.
    - Interoperabilidad con la API de Atlassian Jira.
    - Implementación de un conjunto de servicios específicos que nos brinda la posibilidad de gestionar y ejecutar funcionalidades en los entornos de Bonitasoft y Atlassian Jira.
- Implementamos la Autenticación SSO (Single Sign On) para los entornos de Bonitasoft y Atlassian Jira.
- En sistema Bonitasoft, dentro del ámbito de Bonita BPM Portal, implementamos una sección exclusiva para la creación, edición, visualización y consultas de todas las incidencias registradas en el entorno de Atlassian Jira las cuales están asociadas a los procesos de negocio desplegados.

6.2 Conclusiones

Antes de presentar la propuesta de nuestra tesis, el BPMS Bonitasoft no contaba con la posibilidad de comunicarse desde su entorno con herramientas relacionadas a la administración de tareas e incidencias a través de conectores nativos o utilizando API's externas, por lo tanto, en caso de que una organización tenga implementado ambos tipos de soluciones (un BPMS y un Gestor de Indecencias) los usuarios debían acceder a sistemas complementarios utilizando autenticación mixta, esfuerzo adicional en el aprendizaje de la herramienta de Gestión de Incidencias, demoras en la carga de tareas o incidencias y mayor complejidad en el seguimiento y resolución de las incidencias registradas.

Sin embargo, luego de haber implementado la solución integral entre las herramientas Bonitasoft y Atlassian Jira, podemos observar que el sistema Bonitasoft se logró potenciar en los siguientes aspectos:

Teniendo un enfoque teórico y conceptual

- Este proyecto nos permitió conocer más a fondo todas las características teóricas de la Gestión de Procesos de Negocio y de la Gestión de Incidencias, ya que, en asignaturas durante el desarrollo de la carrera, no se ven conceptos y aplicaciones básicas de lo que esto significa.

- La realización de este estudio nos permitió conocer en mayor profundidad las ventajas de la herramienta Bonitasoft y Atlassian Jira, demostrando que se pueden alinear las actividades de todos los usuarios participantes de un proceso, ordenando y controlando con otros tipos de herramientas que no estén relacionados a un BPMS.

Teniendo un enfoque sistémico

- Facilita la creación, modificación, consulta y visualización de incidencias desde el entorno BPM, evitando así el acceso a herramientas secundarias para poder gestionar incidencias.

- Al definir un único repositorio de usuarios y haber implementado el procedimiento SSO, se facilita la autenticación de usuarios para el acceso a múltiples sistemas. A través del modelo definido, evitamos redundancia de usuarios y tener que configurar a cada usuario en cada sistema de forma independiente.

- Al contar con un módulo genérico para la gestión de incidencias dentro del entorno Bonitasoft, los usuarios no tendrán la necesidad de aprender la herramienta Atlassian Jira de manera exhaustiva y optimizarán los tiempos en la carga de incidencias.

En conclusión, tras finalizar la integración descripta partiendo de dos herramientas existentes, creemos que la herramienta Bonitasoft ofrecerá una funcionalidad adicional

de gran importancia logrando un trabajo más eficaz y dinámico, la compartición de todas las incidencias registradas para cada proceso entre todos los usuarios, visibilidad de cada acción realizada en cada proceso y el estado actual en el que se encuentra cada incidencia asociada a cada proceso.

6.3 Trabajos Futuros

En miras a la mejora continua de este tipo de integraciones dentro del entorno Bonitasoft con herramientas relacionadas a la Gestión de Incidencias, este trabajo de final de carrera se podría ampliar de la siguiente manera:

- Ampliar las acciones relacionadas a las incidencias disponibles en la herramienta Atlassian Jira dentro de componente “Incidencias” en el entorno de Bonitasoft.
- Desarrollar el Front-End y Back-End del módulo Incident Management Connector (IMC).
- Desarrollar un conector estándar “Jira Connector” y adicionarlo en la categoría “Conectores” dentro del entorno Bonitasoft.
- Definir y desarrollar una serie de reportes estándares para obtener información relacionada a procesos e incidencias desde el entorno Bonitasoft.
- Tomando como base el marco teórico y el modelo de integración con la herramienta Atlassian Jira de este trabajo, desarrollar un conector “Generic Incident Manager Connector” que permita adaptar y configurar cualquier Gestor de Incidencias del mercado dentro del entorno Bonitasoft.

Anexos

1.1 Instalación de Bonitasoft

Distribuciones del producto

Bonita BPM se distribuye en varios paquetes según las funcionalidades de la herramienta que se requieran.

Por un lado, está la herramienta de diseño de procesos de negocio, Bonita BPM Studio. Este paquete se distribuye como instalador en las versiones de sistemas operativos soportados o como un archivo comprimido con todos los componentes necesarios para Bonita Studio, pero sin un instalador.

Por otro lado, está el paquete (bundle) para la puesta en marcha en el entorno de producción, Bonita BPM Platform que incluye el motor y el portal. En este caso se ofrecen dos distribuciones: la primera cuenta con los componentes previamente instalados en Tomcat o JBoss, y el segundo cuenta con todos los componentes necesarios para su despliegue sobre un servidor de aplicaciones existente.

Instalación

Paso 1: Ingresamos al sitio web GitHub a través de la siguiente url:

https://github.com/Bonitasoft-Community/Build-Bonita-BPM

Paso 2: Luego, seleccionamos el tag 7.3.3 y descargamos el archivo build-script-7.3.3.shdel proyecto Buid-Bonita-BPM.

Paso 3. Ahora debemos brindar permisos de ejecución para el archivo descargado con la siguiente línea de comando:

chmod +x build-script-7.3.3.sh

Paso 4. En esta instancia, antes de ejecutar el archivo build-script-7.3.3.sh debemos tener instalado en el entorno el componente Java 7+ y la herramienta Maven 3.3.9.

Paso 5. Luego debemos ejecutar el archivo build-script-7.3.3.sh por medio del siguiente comando:

./build build-script-7.3.3.sh

Paso 6: Para finalizar la instalación, debemos descomprimir el archivo BonitaBPMCommunity-7.3.3.zip ubicado en el directorio bonita-studio/all-in-one/target para poder disponible el entorno Bonita BPM para ser utilizado.
Mediante esta distribución contamos con el entorno Bonita Studio, Bonita User Experience, una base de datos (H2 Database Engine), y un contenedor de aplicaciones (Tomcat 7) embebidos para facilitar el desarrollo y las pruebas de los procesos modelados.

1.2 Instalación de Atlassian Jira

Instalación

Para poder comenzar con la instalación debemos bajar el instalador del la siguiente url seleccionado el sistema operativo deseado:

https://www.atlassian.com/software/jira/download

En nuestro caso, la instalación la vamos a realizar sobre el sistema operativo Ubuntu 16.04.

Paso 1: Primero debemos brindar permiso de ejecución al archivo atlassian-jira-software-7.2.4-x64.bin mediante la siguiente línea de comando:

chmod +x atlassian-jira-software-7.2.4-x64.bin

Paso 2: Luego, con permiso de "super usuario(sudo)" ejecutamos el instalador de la siguiente manera:

sudo ./atlassian-jira-software-7.2.4-x64.bin

Paso 3: Una vez ejecutados el instalador, debemos ingresar la letra "o" para dar incio a la instalación de la herramienta tal como se muestra en la Figura:

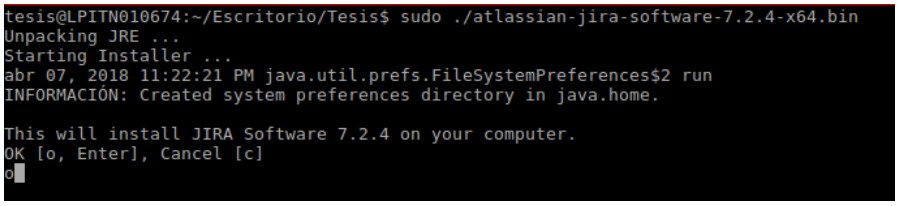

Figura 44. Inicio de la instalación Atlassian Jira

Paso 4: Ahora debemos seleccionar el tipo de instalación que deseamos ralializar. En nuestro elegimos la opción 2 "Custon Install".

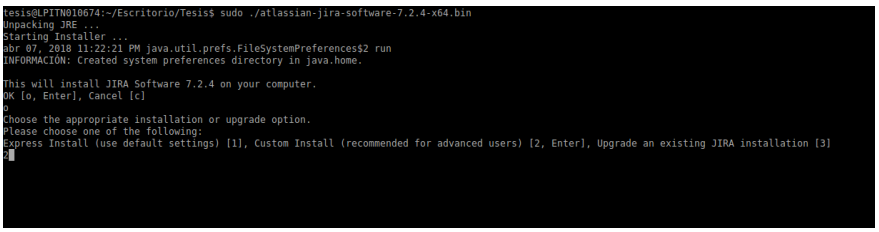

Figura 45. Tipo de Instalación – Custom Install

Paso 5: Como siguiente paso, debemos indicar cual será el directorio donde debemos instalar la herramienta. En nuestro caso el directorio a utilizar es /opt/atlassian/jira.

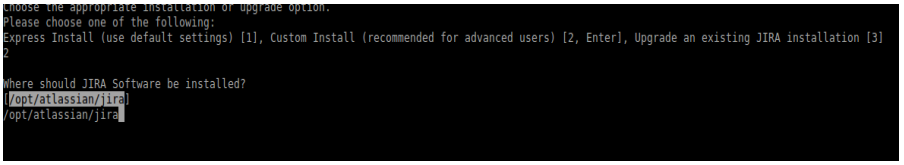

Figura 46. Directorio de instalación

Paso 6: Ahoradebemos especificar el directorio donde Jira va a almacenar todos los datos (proyectos, incidencias, usuarios, etc). En nuestro caso utilizaremos el directorio (por defecto) el directorio será /var/atlassian/application-data/jira.

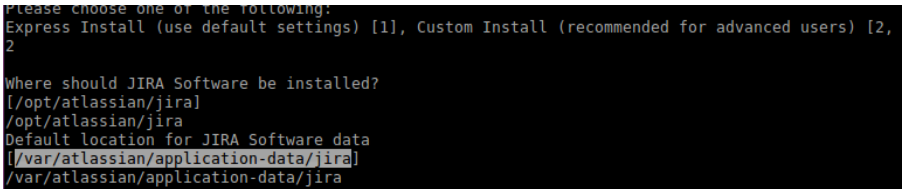

Figura 47. Directorio de almacenamiento de datos

Paso 7: Luegodebemos especificar si se deja la opción recomendada en cuanto a los números de puertos o bien customizarlos. Los puertos a configurar son los siguientes:

- Puerto HTTP: es el puerto utilizado para acceder a la herramienta Jira.
- Puerto de Control: es el puerto utilizado para levantar y bajar el servidor de la herramienta Jira.

En nuestro caso seleccionamos la opción 2 “Set custom value for HTTP and Control ports”.

Figura 48. Opción 2 - Configuración de puertos customizados

Luego seteamos los puertos con los siguientes valores:

- Puerto HTTP:9290
- Puerto de Control:9291

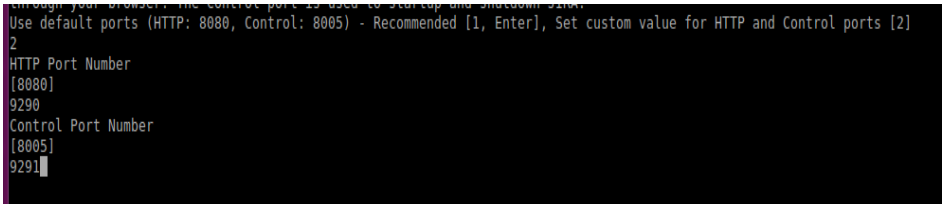

Figura 49. Configuración de puertos HTTP y Control

Paso 9: A continuación, debemos indicar si la herramienta Jira deberá ser instalada como un servicio. En nuestro caso seleccionamos la opción No.

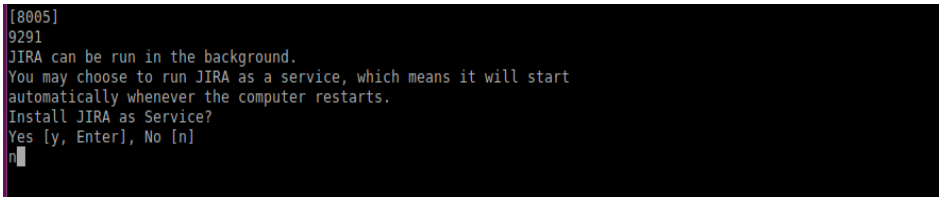

Figura 50. Opción de configuración de Jira como servicio

Paso 10: Por último, podemos visualizar la configuración completa de la instalación realizada. En caso de todos los pasos seleccionados sean los correctos debemos confirmarlo con la opción i. Luego de la confirmación la herramienta Jira esta lista para ser usada.

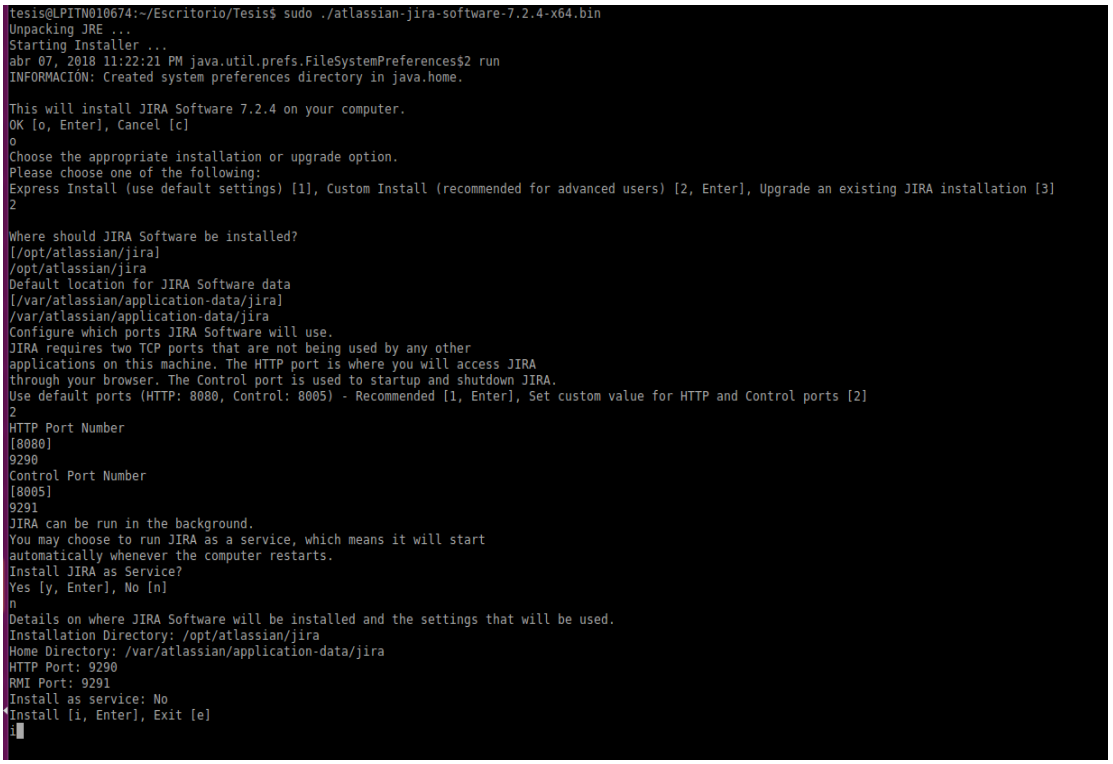

Figura 51. Visualización de la configuración de Jira

1.3 Instalación del Servicio de Directorio LDAP

Descarga de OpenLDAP

Para descargar el software, podemos hacerlo desde el siguiente enlace:

http://www.openldap.org/software/download/

Instalación del servidor OpenLDAP

La instalación del servidor OpenLDAP lo realizamos a través del uso de un asistente de instalación.

Para poder llevar adelante la instalación del servidor OpenLDAP necesitamos tener privilegios de superusuario y disponer de permisos de escritura en los directorios de instalación.

El servidor OpenLDAP se encuentra en los repositorios predeterminados de Ubuntu bajo el paquete "slapd", por lo que podemos instalarlo con el utilitario apt-get. Adicionamente necesitamos instalar algunas utilidades adicionales. Por tanto, ejecutamos las sentencias que se describen a continuación:

```
sudo apt-get update
sudo apt-get install slapd ldap-utils
```

Cuando se ejecutan los commandos anteriores, se solicita que ingresemos y confirmemos la contraseña de administrador para la cuenta LDAP.

En el siguiente paso, debemos reconfigurar el paquete slapd. Cuando completamos la instalación de OpenLDAP, necesitamos reconfigurar el paquete LDAP, por lo tanto, debemos escribir el siguiente commando para abrir la herramienta de configuración del paquete:

```
sudo dpkg-reconfigure slapd
```

Una vez ejecutados el comando anterior necesitaremos responder una serie de preguntas sobre cómo le deseamos configurar el software.

a. ¿Omitir la configuración del servidor OpenLDAP?
   Seleccionamos la opción “No” para poder configurar de forma customizada el servidor LDP.

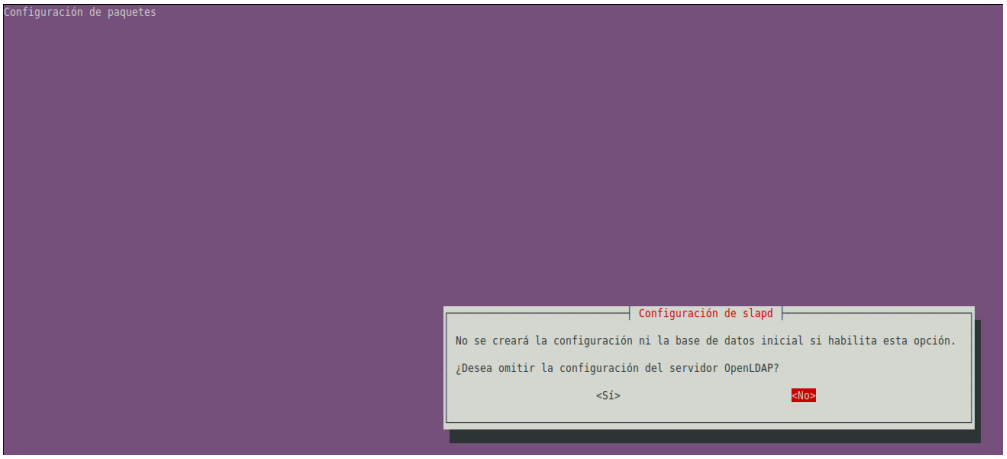

Figura 52. Configuración OpenLDAP

b. ¿Nombre de dominio DNS?
Esto creará la estructura base de su ruta de directorio.

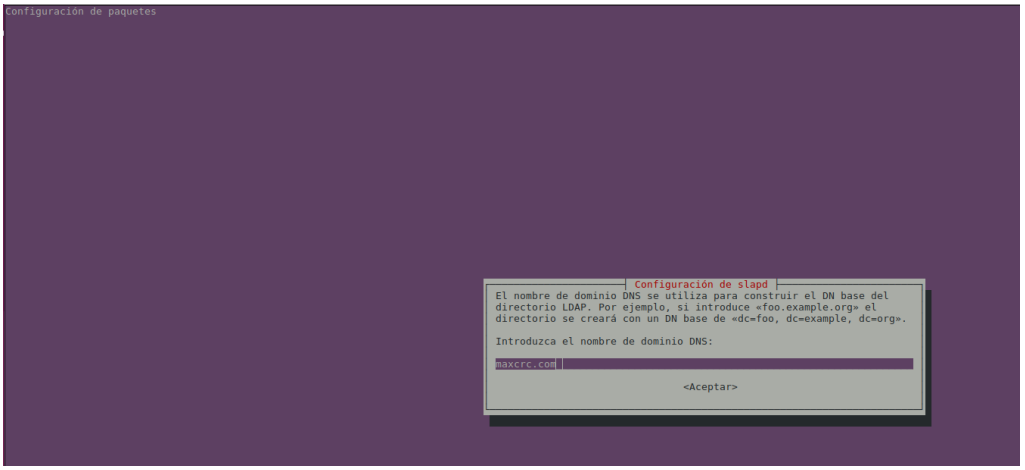

Figura 53. Configuración delDominio DNS

c. ¿Nombre de la organización?
En nuestro caso definimos la palabra BonitaOrg como nombre de la organización.

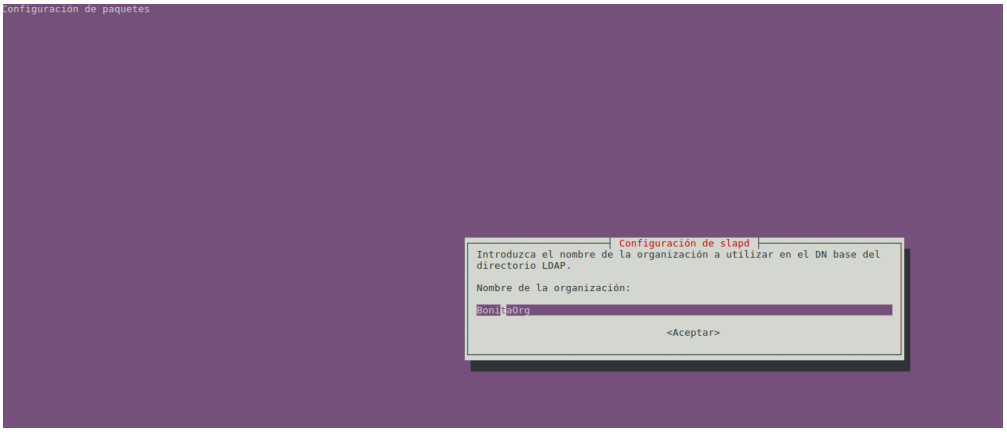

Figura 54. Configuración Nombre de la Organización

d. ¿Password de Administrador?
   En nuestro caso definimos la palabra Tesis como la contraseña de administrador.

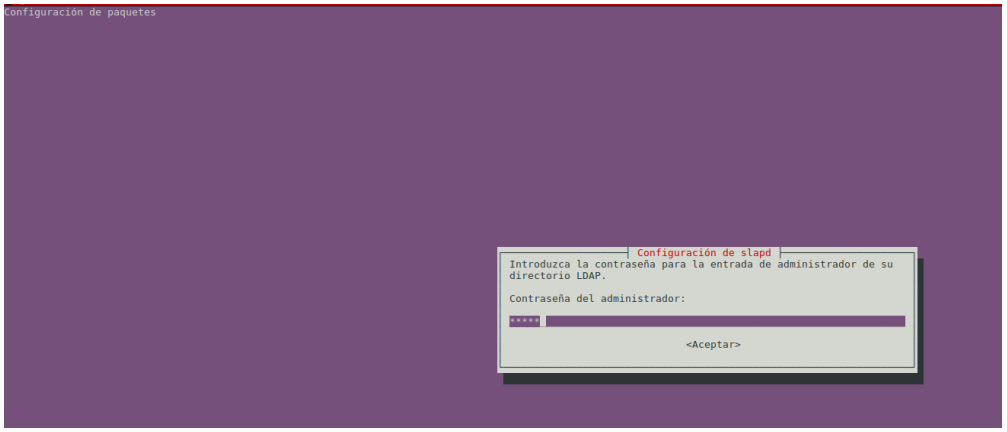

Figura 55. Configuración del password del Administrador

e. ¿Base de datos a utilizar?
   Debemos seleccionar la opción HDB.

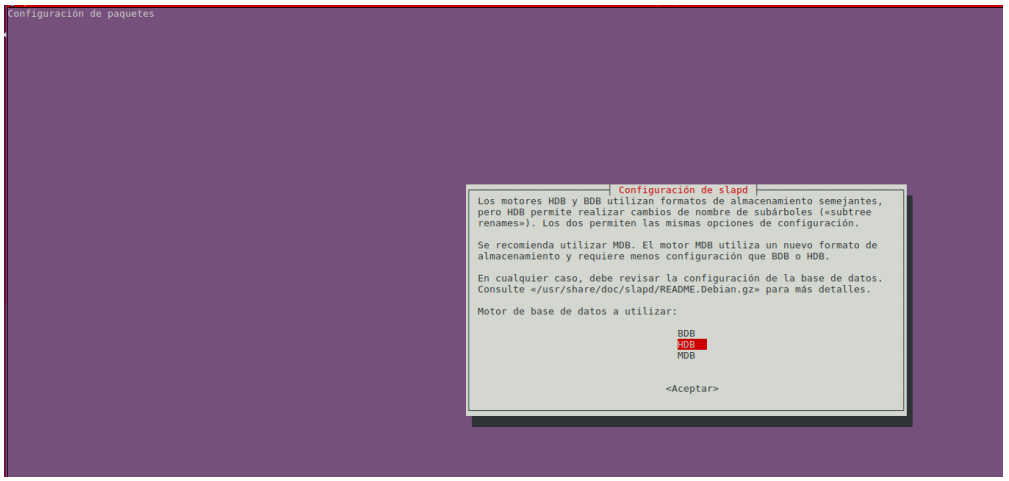

Figura 56. Configuración de la Base de Datos (I)

f. ¿Eliminar la base de datos cuando se actualiza slapd?
   Debemos seleccionar la opción No

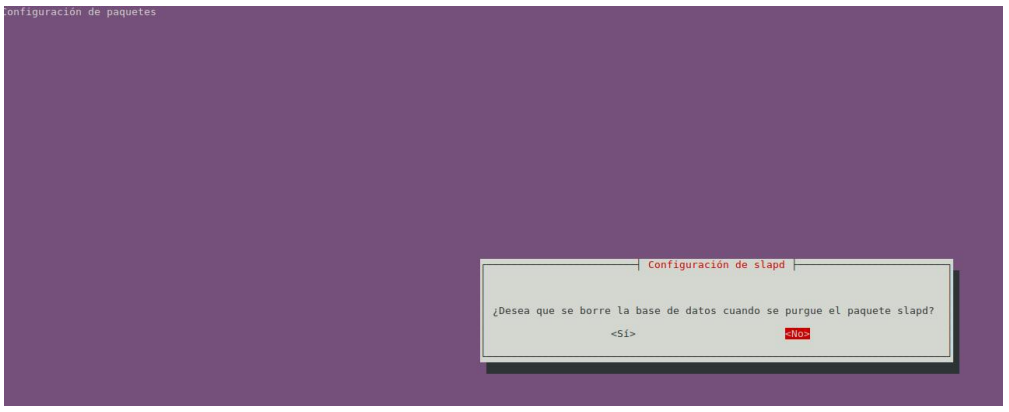

Figura 57. Configuración de la Base de Datos (II)

g. ¿Desea mover la base de datos anterior?
Debemos seleccionar la opción Sí

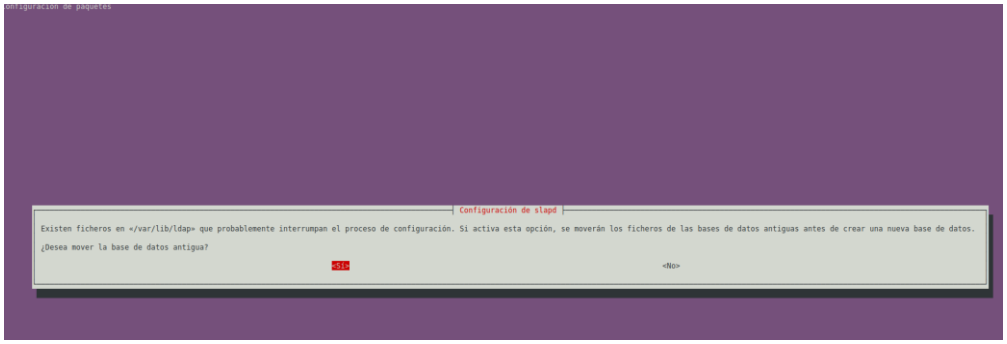

Figura 58. Configuración de la Base de Datos (III)

h. ¿Permite el protocolo LDAPv2?
Debemos seleccionar la opción No

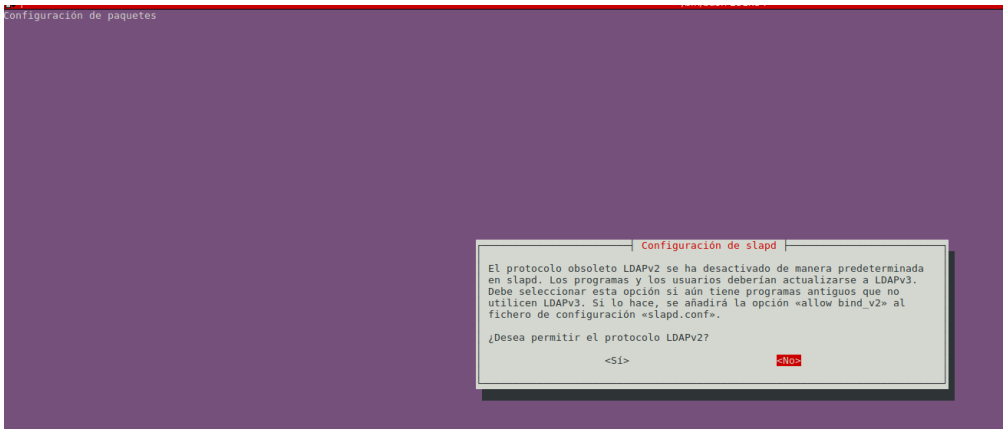

Figura 59. Configuración del protocolo LDAPv2

Una vez finalizado los pasos anteriores, tenemos disponible para hacer uso del servidor OpenLDAP.

Referencias

[1] Weske 2012. Mathias Weske. “Business Process Management Concepts, Languajes and Architectures”. 2ª edición, 2012, Springer-Verlag Berlin Heidelberg.

[2] James F. Chang. “Business Process Management Systems, Strategy and Implementation”, 2006. Auerbach Publications. Boca Raton, Florida, USA.

[3] Gartner, Inc. “Magic Quadrant for Business Process Management Suites 2007”, 2007. Disponible en https://www.appian.com/resources/.

[4] Paul Harmon. “Software Tools for BPM”, 2008. BPTrend SpotLight.
Disponible en http://www.bptrends.com/publicationfiles/spotlight_062008.pdf.

[5] Introducción a BPM para Dummies. Edición Especial, Garimella Kiran, Lees Michael, Williams Bruce, Publishing.

[6] Howard Smith, Peter Fingar. “Business Process Management: The Third Wave”, 2006. MK Press. Tampa, Florida, USA.

[7] P. Mathiesen, J.Watson et al. “Applying Social Tecnology to Business Process Lifecycle Management”. Business Process Management Workshops. Lecture Notes in Business Information Processing Volume 99, 2012. Springer-Verlag Berlin Heidelberg.

[8] C. Ballard, A. Abdel-Hamid, R. Frankus et al. “Improving Business Performance Insight with Business Intelligence and Business Process Management”, 2006. IBM RedBook. International Business Machines Corporation. USA.

[9] Stefan R. Koster. “An evaluation method for Business Process Management Products”, 2009. Master Thesis. University of Twente. Enschede, The Netherlands.
Disponible http://www.utwente.nl/ewi/trese/graduation_projects/2009/Koster.pdf.

[10] Zhen Chen. “Workflow Management Theories ans Techniques including Data Perspective”, 2012. Research Topics. University of Twente. Enschede, The Netherlands. Disponible http://www.utwente.nl/ewi/trese/graduation_projects/2012/RT-004.pdf.

[11] Marlon Dumas, Marcello La Rosa, Jan Mending, Hajo Reijers. “Fundamentals of Business Process Management”, 2013. Springer-Verlag Berlin Heidelberg.

[12] Michael Hammer, “What is Business Process Management?”. Handbook on Bussines Process Management Vol. 1, 2010, J. Von Brocke, M. Rosemann (editors). Springer-Verlag Berlin Heidelberg.

[13] John Hall. “Business Motivation Model. Reacting to Change”. OMG Business Rules Symposium, Minneapolis, June 2010.

[14] Object Management Group. “Business Motivation Model (BMM) v1.3”, 2015.
Disponible http://www.omg.org/spec/BMM/1.3.

[15] Object Management Group. “BPMN Specificacion v1.2”, 2009.
Disponible http://www.omg.org/spec/BPMN/1.2/PDF.

[16] Object Management Group. “BPMN Specification v2.0.2”, 2013.
Disponible http://www.omg.org/spec/BPMN/2.0.2/PDF.

[17] Gustav Aagesen and John Krogstie, “Analysis and Design of Business Processes Using BPMN”. Handbook on Bussines Process Management Vol. 1, 2010, J. Von Brocke, M. Rosemann (editors). pp. 213 – 235. Springer-Verlag Berlin Heidelberg.

[18] Fred A. Cummins. “BPM Meets SOA”. Handbook on Bussines Process Management Vol. 1, 2010, J. Von Brocke, M. Rosemann (editors). pp. 461 – 479. Springer- Verlag Berlin Heidelberg.

[19] Object Management Group, “Case Management Model and Notation v1.0”, 2014. Disponible http://www.omg.org/spec/CMMN/1.0/PDF.

[20] Mikael Lind and Ulf Seigerroth, “Collaborative Process Modeling: The Intersport Case Study”. Handbook on Bussines Process Management Vol. 1, 2010, J. Von Brocke, M. Rosemann (editors). pp. 279 – 298. Springer-Verlag Berlin Heidelberg.

[21] John Jeston, Johan Neils. “Business Process Management. Practial Guidelines to Successful Implementations”, 2006. Routledge. Abingdon, U. K.

[22] Theodore Panagacos, “The Ultimate Guide to Business Process Management”, 2012. Createspace.

[23] August-Wilhelm Scheer and Eric Brabänder. “The Process of Business Process Management”. Handbook on Bussines Process Management Vol. 2, 2010, J. Von Brocke, M. Rosemann (editors). pp. 239 – 265. Springer-Verlag Berlin Heidelberg.

[24] Oracle Coporation. “Oracle Practitioner Guide. Business Process Engineering, Release 3.0”, 2010.
Disponible http://www.oracle.com/technetwork/topics/entarch/oracle-pg-bpm-bus-proc-eng-r3-0-292099.pdf.

[25] Gartner, Inc. “Analysts to Discuss the Next Generation of BPM”. Gartner Business Process Management Summit 2012.
Disponible http://www.gartner.com/newsroom/id/1943514.

[26] Nathaniel Palmer. “Thriving Adaptavility: How Smart Companies Win in a Data-Driven World”. iBPMS: Intelligent BPM Systems. Impact and Opportunity, 2013. Layna Fischer (editor). Future Strategies Inc., Lighthouse Point, Florida, USA.

[27] Wil M. P. van der Aalst, Kees Max van Hee. “Workflow Management: Models, Methods and Systems”, 2002. The MIT Press Cambridge, Massachusetts London, England.

[28] Mathias Kirchmer, “Managements of Process Excellence”. Handbook on Bussines Process Management Vol. 2, 2010, J. Von Brocke, M. Rosemann (editors). Springer-Verlag Berlin Heidelberg.

[29] Chun Ouyang, Michael Adams, Moe Thandar Wynn and Arthur H. M. ter Hofstede, “Workflow Management”. Handbook on Bussines Process Management Vol. 1, 2010, J. Von Brocke, M. Rosemann (editors). Springer-Verlag Berlin Heidelberg.

[30] Joseph M. DeFee, Paul Harmon. “Business Activity Monitoring and Simulation”, 2004. BPTrends.
Disponible http://www.bptrends.com/bpt/wp-content/publicationfiles/02-04%20WP%20Simulation%20and%20BAM%20-%20DeFee-Harmon1.pdf.

[31] Fred A. Cummins. “Building the Agile Enterprise with SOA, BPM and MBM”, 2009. MK/OMG Press. Burlington, USA.

[32] Michael zur Müehlen, Robert Shapiro, “Business Process Analytics”. Handbook on Bussines Process Management Vol. 2, 2010, J. Von Brocke, M. Rosemann (editors). Springer-Verlag Berlin Heidelberg.

[33] Wil M. P. van der Aalst. “Process Mining. Discovery, Conformance and Enhancement of Business Process”, 2011. Springer-Verlag Berlin Heidelberg.

[34] Business Process Incubator. “BPMS Tools List”.
Disponible https://www.businessprocessincubator.com/tools/bpms.html.

[35] Van Bon, J., De Jong, A., Kolthof, A., Pieper, M., Tjassing, R., Van der Veen, A., & Verheijen, T. (2008a). Estrategia del servicio basada en ITIL® V3: Guía de Gestión. Zaltbommel: Van Haren Publishing.

[36] AXELOS (2011). Glosario y abreviaturas de ITIL Español (Latinoamericano). Disponible www.itilofficialsite.com/InternationalActivities/TranslatedGlossaries.aspx.

[37] ITILv3, F. (2011, Jul).
Disponible http://itilv3.osiatis.es/.

[38] IT Governance Institute, I. (2013). Cobit 5. Rolling Meadows.

[39] O’Toole, D. (2015). Incident management for i.t. departments. On Demand Publishing, LLC-Create Space.
Disponible https://books.google.co.cr/books?id=M2RargEACAAJ.

[40] Brand, K., & Boonen, H. (2004). It governance: A pocket guide based on cobit. Van Haren Publishing.
Disponible https://books.google.co.cr/books?id=LH77ZNT-MmEC.

[41] Forrester, E., Buteau, B., & Shrum, S. (2011). Cmmi for services: Guidelines for superior service. Pearson Education.
Disponible https://books.google.co.cr/books?id=ywvSVLmQmjoC.

[42] Kenett, R., & Baker, E. (2010). Process improvement and cmmi for systems and software. CRC Press.
Disponible https://books.google.co.cr/books?id=a7XS1GmuhWYC.

[43] Mutafelija, B., & Stromberg, H. (2008). Process improvement with cmmi v1.2 and iso standards. CRC Press.
Disponible https://books.google.co.cr/books?id=ErVuWU\U0SwC.

[44] Bonita BPM 7.3. REST API overview.
Disponible https://documentation.bonitasoft.com/bonita/7.3/rest-api-overview.

[45] Bonita 7.2+. Engine API.
Disponible https://documentation.bonitasoft.com/6.x-7.2/product-bos-sp/engine-api.

[46] Jira Rest Java Client Library.
Disponible https://ecosystem.atlassian.net/wiki/spaces/JRJC/overview.

[47] Jira Server Developer. REST APIs.
Disponible https://developer.atlassian.com/server/jira/platform/rest-apis/.

[48] Eclipse Marketplace. Bonita.
Disponible https://marketplace.eclipse.org/content/bonita-bpm.

[49] Bonita BPM.
Disponible https://en.wikipedia.org/wiki/Bonita_BPM.

[50] Bonitasoft.com. “Acerca de Bonitasoft”.
Disponible http://es.bonitasoft.com/ecosistema/empresa.

[51] Bonitasoft.com. “Bonita BPM Documentation”.
Disponible http://documentation.bonitasoft.com/.

[52] Bonitasoft.com. “Bonita BPM Community”.
Disponible http://community.bonitasoft.com/.

[53] Bazán P. “Implementación de Procesos de Negocio a través de servicios aplicando metamodelos, software distribuido y aspectos sociales”. Tesis de Doctorado en Ciencias Informáticas. Facultad de Informática. Universidad Nacional de La Plata. Febrero 2015.

[54] Atlassian, Atlassian JIRA Documentation.
Disponible http://confluence.atlassian.com/display/JIRA/JIRA+Documentation.

[55] Atlassian, Atlassian JIRA User’s Guide.
Disponible http://confluence.atlassian.com/display/JIRA/JIRA+User%27s+Guide.

[56] Atlassian, Atlassian Partner Directory.
Disponible http://confluence.atlassian.com/display/APW/Navigation.

[57] Atlassian, Atlassian Experts.
Disponible http://www.atlassian.com/resources/experts.

[58] Atlassian, End of support announcements for JIRA.
Disponible
http://confluence.atlassian.com/display/JIRA/End+of+Support+Announcements+for+JIRA.

[59] Atlassian, Pricing.
Disponible http://www.atlassian.com/software/jira/pricing.jsp.

[60] Atlassian Documentation, JIRA Extensions.
Disponible http://confluence.atlassian.com/display/JIRAEXT/JIRA+Extensions

[61] Atlassian Documentation, Atlassian Developer Network.
Disponible http://confluence.atlassian.com/display/DEVNET.

[62] Timothy A. Howes, Ph.D, Marck C.Smith, Gordon S.Good, Understanding and Deploying LDAP directory services, Estados Unidos, Addison-Wesley, Segunda Edición, 2003.

[63] OpenLDAP. Introduction to OpenLDAP Directory Services.
Disponible http://www.openldap.org/doc/admin24/intro.html.

[64] OpenLDAP.
Disponible https://www.openldap.org/.

[65] LDAPSynchronizer & LDAPAuthenticator.
Disponible http://bpms.help/instruction.php?module=bonita_ldapsynchronizer.

[66] Bizagi
Disponible http://www.bizagi.com/.

[67] OMG, «Object Management Group»
Disponible http://www.omg.org/.

[68] Soltel Group
Disponible http://www.soltel.es/es/blog/bonita_vs_bizagi.

[69] A. Rodríguez, P. Bazán y J. Díaz, Características funcionales avanzadas de los BPMS. Abril 2012.
Disponible www.linti.unlp.edu.ar/uploads/docs/caracteristicas_funcionales_avanzadas_de_los_bpms__analisis_comparativo_de_herramientas_.pdf.

[70] Persse, J. (2013). The it service management process manual. van Haren Publishing.
Disponible https://books.google.co.cr/books?id= 0FZeAgAAQBAJ.

[71] Jan van Bon, Arjen de Jong, Axel Kolthof, Mike Pieper, Ruby Tjassing, Annelies van der Veen, Tieneke Verheijen, 2008. Gestión de Servicios de TI basada en ITIL V3.
Disponible https://books.google.com.ar/books?hl=es&lr=&id=WdFEBAAAQBAJ&oi=fnd&pg=PR5&dq=related:iAgnRfcc31UJ:scholar.google.com/&ots=9kC7ZmxjFM&sig=25dK7iRW_83mIOxCWrFbjynJkH4#v=onepage&q&f=false.

[72] Gabriela Carolina Tueti Silva, 2010. ANÁLISIS Y PROPUESTA DE MEJORA DEL PROCESO DE GESTIÓN DE INCIDENTES DEL SERVICE DESK.