
UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN
CENTRO DE INVESTIGACION EN SISTEMAS DE INFORMACIÓN CISI

# DESARROLLO DE MODELO DE ARQUITECTURA ORIENTADA A SERVICIOS PARA LA GESTIÓN DE PROCESOS DE NEGOCIO. CASO DE ESTUDIO: GESTIÓN DE PROYECTOS

**Trabajo de Especial de Grado presentado ante la ilustre**
**Universidad Central de Venezuela por el**
**Br. Clayton Erikson Carbajo Veramendi**

**Para optar al título de Licenciado de Computación**
**Tutor: Prof. Franky Uzcátegui**

**Caracas, Julio 2019**

UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

ESCUELA DE COMPUTACIÓN

**ACTA**

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado titulado "Desarrollo de modelo de arquitectura orientada a servicios para la gestión de procesos de negocio. Caso de estudio: gestión de proyectos." y presentado por el bachiller: Br. Clayton Erikson Carbajo Veramendi, número de cédula: 20678921, a los fines de optar al título de **Licenciado en Computación**, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día 31 de Julio de 2019, a las 8:30 horas, para que el autor lo defendiera en forma pública, lo que esto hizo en la Sala PA3 de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondió a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de 19 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 31 de Julio de 2019

**Prof. Franky Uzcátegui**

**(Tutor)**

**Prof(a). Mercy Ospina**

**(Jurado)**

**Prof(a). Concettina Di Vasta**

**(Jurado)**

Agradecimientos

Agradezco a mi familia que son uno de los pilares más importantes del crecimiento personal que me permitió llegar a ser lo que soy hoy, su fuerza, apoyo e inspiración fueron factores fundamentales en la culminación de mis estudios, gracias Erika y Jhonnathan, pero sobre todo gracias a ti Beatriz Veramendi y Pablo Carbajo, este logro es de ustedes también.

Agradezco a todas aquellas personas que me acompañaron, guiaron, apoyaron y enseñaron durante toda mi carrera hasta el día de hoy, su compañerismo es muy valioso para mí.

Agradezco a todos los profesores de la facultad de ciencias por su excelente calidad de educación dejando en alto el nombre de esta alma mater incluso en tiempos difíciles, he aprendido mucho gracias a ustedes y estoy orgulloso de haber sido un alumno para los que me dieron clases en las distintas materias de la carrera.

Agradezco a la Universidad Central de Venezuela por recibirme en tan maravillosa casa de estudio, ha sido un honor pertenecer a la casa que vence a las sombras.

Agradezco a todas mis amistades y personas que compartieron conmigo, me regalaron momentos increíbles durante mi carrera, cada uno de ustedes es único y aprecio sus amistades.

Este trabajo es dedicado a mi familia, pero también a todos aquellos estudiantes de pregrado de las Universidades, ¡no se rindan! puede que a veces las cosas no sean fáciles y menos con la situación actual del país, pero siempre hay que levantarse con más fuerza y continuar, todo esfuerzo tiene su recompensa.

**Clayton Carbajo.**

# RESUMEN

## DESARROLLO DE MODELO DE ARQUITECTURA ORIENTADA A SERVICIOS PARA LA GESTIÓN DE PROCESOS DE NEGOCIO. CASO DE ESTUDIO: GESTIÓN DE PROYECTOS.

**Autor:** Clayton Erikson Carbajo Veramendi.
**Tutor**: Prof. Franky Uzcátegui.
**Fecha**: 31 de julio de 2019.

El continuo crecimiento de la Gestión de Procesos de Negocio (BPM) como principio de gestión para el control de los procesos y de la tecnología asociada, ha traído como consecuencia la creación de sistemas con múltiples componentes de software que deben ser coordinados. El presente Trabajo Especial de Grado estableció como objetivo, desarrollar un modelo de arquitectura orientada a servicios para la BPM el caso de estudio la Gerencia de Proyectos además de modelar el proceso básico de consulta de proyecto utilizado por los procesos pertenecientes a este caso de estudio, todo esto en base a la metodología del ciclo de vida BPM a través de la plataforma BPMS BonitaSoft como el orquestador de las funcionalidades, cuyos servicios serán gestionados por medio de una Arquitectura Orientada a Servicios (SOA) que haga uso de los beneficios que esta ofrece. La comunicación entre BonitaSoft y las distintas plataformas a través de SOA se realizará por medio de la herramienta de integración WSO2 Integrador Empresarial. Para controlar el desarrollo se utilizó la metodología SCRUM.

**Palabras Claves:** Gestión de Procesos de Negocio, Arquitectura Orientada a Servicios, WSO2, Gerencia de Proyectos.

# Índice

# Índice de Figuras

# Índice de Tablas

# INTRODUCCIÓN

La Gestión de Procesos de Negocio (BPM) se considera un conjunto de disciplinas, métodos y tecnologías que tienen como propósito mejorar el desempeño y la optimización de los procesos de negocio de una determinada organización. En la actualidad la Gestión de Procesos de Negocio (BPM) ha sido persistente en su crecimiento como una metodología corporativa que se encarga de controlar los procesos y la evolución que van teniendo los Sistemas de Gestión de Procesos de Negocio (BPMS), pero con su constante crecimiento también surgen una serie de desventajas en los sistemas de gran magnitud, perjudicando así la agilidad del negocio.

El problema para los sistemas de gran magnitud inicia cuando se mantienen o incorporan nuevas tecnologías y se realice algún cambio en ellas. Por la naturaleza de los procesos de negocio en los BPMS estos están atados a constantes modificaciones en su arquitectura, ya sea por el cambio de tecnología obsoleta a una novedosa, por cambios en los requerimientos sobre los que están modelados los procesos y que implican modificaciones en distintos componentes existentes, por el manejo de varios formatos para el envío de datos, entre otros, provocando así largos tiempos de desarrollo o adaptación, altos costos y tiempos de respuesta de negocio tardíos entre otras consecuencias.

En la Gerencia de Proyectos se pueden desarrollar BPMS's de gran magnitud ya que la Gerencia de Proyectos posee una gran cantidad de procesos complejos para asegurar la ejecución de un proyecto de manera eficiente y efectiva, todos estos procesos se comunican con distintas tecnologías externas para satisfacer las necesidades del negocio.

Los requerimientos sobre los que están modelado los procesos de negocio pueden ser construidos a partir de servicios web, los cuales son una tecnología que permite la comunicación e interoperabilidad entre los distintos componentes. Los servicios web se manejan bajo un estándar y con un formato de entrada y salida específico, estos pueden ser alojados o definidos en un sistema encargado de gestionarlos, son fáciles de reutilizar y mantener al igual que aplicar cambios sobre ellos. El uso de los servicios web abre paso a una Arquitectura Orientada a Servicios (SOA) la cual permite separar la lógica de negocio de un proceso como servicio del BPMS, permitiendo así obtener los beneficios que ofrece como lo son: la reutilización de los servicios, interoperabilidad, bajos costos de mantenimiento, tiempos de respuesta de negocio más rápidos y una mayor escalabilidad y crecimiento del sistema.

Es por lo antes mencionado que el presente Trabajo Especial de Grado propone el Desarrollo de un Modelo de Arquitectura de Servicios para la Gestión de Procesos de Negocio en base a la metodología del ciclo de vida BPM que permita que la plataforma BPMS BonitaSoft sea el orquestador de las funcionalidades de cada proceso como servicios, estos servicios serán gestionados bajo un modelo de Arquitectura Orientada a Servicios (SOA) el cual permitirá aprovechar al máximo los beneficios mencionados anteriormente, la comunicación

entre Bonitasoft y los distintos componentes del sistema se realiza a través del consumo de servicios donde su implementación está apoyada por un sistema de integración llamado WSO2 Integrador Empresarial. mientras que para la Gestión del Proyecto se utilizará la metodología SCRUM.

En el capítulo I, se plantea formalmente el problema a abarcar, dando un contexto completo y justificado, además se definen el alcance y los objetivos de este Trabajo Especial de Grado (TEG).

En el Capítulo II, se definen todos los conceptos necesarios para poder desarrollar la solución al problema presentado en el capítulo anterior y las técnicas necesarias que ayudarán a alcanzar los objetivos.

En el Capítulo III, se explica la metodología a utilizar para el desarrollo del trabajo.

En el Capítulo IV, se explica formalmente el desarrollo de la aplicación, siguiendo las actividades planteadas en la metodología, y se expone el producto final. Para finalizar, se ofrecen las Conclusiones alcanzadas y algunas propuestas para posibles trabajos futuros.

# CAPÍTULO 1
# PROBLEMA DE INVESTIGACIÓN

En este capítulo se esboza el problema a resolver, se explica cuál es el planteamiento del problema en el contexto actual, se especifican los objetivos, la justificación y por último el alcance definido para este TEG.

## 1.1. Planteamiento del Problema

La Gestión de Procesos de Negocio (BPM) ha obtenido un papel fundamental dentro de las organizaciones, el avance de la tecnología permite el uso de un Sistema de Procesos de Negocios (BPMS) para desarrollar y automatizar los procesos que existan en dichas organizaciones.

Estos procesos poseen funciones y características diferentes, cada uno con un objetivo en específico para cumplir con los requerimientos de la organización. Estos requerimientos obligan a que nuestro BPMS interactúe con otros componentes como gestores documentales, Sistemas Manejadores de Base de Datos (SMBD), servidores de correos electrónicos, entre otros para lograr un objetivo en común.

A medida que surgen más requerimientos o se identifican más procesos en una organización son más las conexiones que se deben hacer a estos componentes, ya sea porque se vuelve a necesitar un componente existente o se integra un nuevo componente.

En la, se **Figura 1.1** puede observar un escenario donde interactúan distintos componentes para satisfacer los requerimientos de la organización.


**Figura 1.1 Comunicación entre diversos componentes.**

En este caso existen BPMS con muchos procesos como lo son los BPMS asociados a la Gerencia de Proyectos, un ejemplo es el BPMS que está en actual desarrollo por la empresa Incognitio que se encarga de integrar todos los procesos del PMBOK, los cuales se identifican en distintos grupos de procesos utilizados para la iniciación, planificación, ejecución, control y cierre de un proyecto que son integrados la mayor parte del tiempo para proyectos de distintas organizaciones. Estos ocupan un nivel de actividad que para efectos del BPMS se traduce a una mayor o menor cantidad de requerimientos. En la Figura 1.2, se observa una gráfica con los grupos de procesos y su relación en cuanto al nivel de actividad.

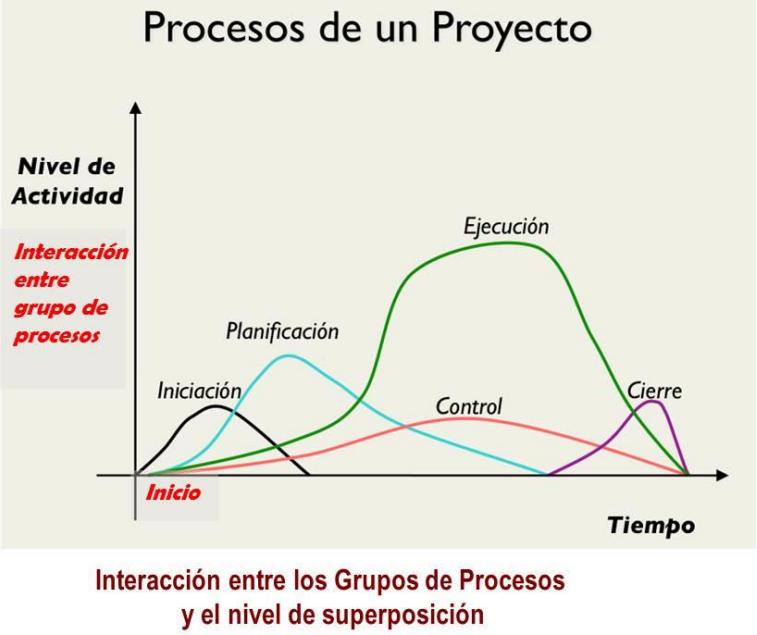


**Figura 1.2 Grafica de interacción entre Grupos de Procesos.**

Fuente: https://formulaproyectosurbanospmipe.wordpress.com/2012/04/25/3-la-interaccion-entre-los-procesos-de-la-direccion-de-proyectos-segun-la-guia-del-pmbok-26-03-2012-1ra-parte-la-guia-del-pmbok-capitulo-3/.

Sin embargo, estos procesos en su ciclo para ser modelados pueden presentar problemas al recibir constantes cambios por la naturaleza del ciclo de vida de los Procesos de Negocio, estos hacen uso frecuentemente de componentes funcionales para sus distintas entradas y salidas a través de conectores los cuales son elementos de conexión para un componente específico, al recibir algún cambio en la forma en cómo se accede a estos componentes puede ocasionar errores al momento de utilizar este conector. Al aumentar la cantidad de procesos y con ellos la cantidad de conectores para el acceso de datos, un cambio podría afectar a la agilidad de la aplicación, ocasionando una tasa mayor de errores, tiempos mayores para la adaptación de estos cambios, altos costos, entre otros.

Un ejemplo en este tipo de BPMS es el proceso simple de Consultar un Proyecto, para consultar un proyecto necesitamos un identificador como dato de entrada y como datos de salida podemos obtener información, documentos y correos relacionados a un proyecto, este proceso se utiliza N veces en la Gestión de Proyectos y puede existir un conector por cada vez que se necesite utilizar, por consecuencia cada vez que este proceso sufra una modificación y esto implique a los componentes entonces debe aplicarse una modificación para todos los conectores existentes pertenecientes a ese proceso.

Hoy en día existe una tecnología llamada servicios web que nos permiten intercambiar información en base a unos protocolos y estándares, los servicios web abstraen la lógica y el desarrollo de los accesos de datos entre los componentes para que así los BPMS solo deban consumirlos y no tengan que definir el cómo se utilizan y qué deben hacer asegurando que se comporten de la misma manera para todos los procesos que los utilicen.

El uso completo de los servicios web en una organización abre paso a una Arquitectura Orientada a Servicios (SOA) qué es un modelo de arquitectura tecnológica que nos ofrece un buen manejo de los servicios web para así obtener numerosos beneficios como lo son la reutilización, la interoperabilidad, una mayor escalabilidad y crecimiento del sistema, bajos costos de mantenimiento, un tiempo de respuesta más rápido a los cambios, mayor eficiencia, un manejo simple de los componentes, facilidad en las pruebas, reducción de errores, facilidad de integración, entre otros.

En la actualidad existen soluciones de Sistemas de Integración que nos dan una mayor facilidad para implementar una Arquitectura Orientada a Servicios, permitiendo crear, desarrollar, modificar, gestionar y monitorear servicios, así como otras características únicas de estos sistemas.

Bajo este contexto, es necesario contar con un Sistema de Integración que gestione la comunicación entre los distintos servicios, facilitando la integración y la interacción entre componentes heterogéneos con el BPMS cumpliendo así con el enfoque que plantea SOA para así eliminar el uso de conectores hacia componentes específicos.

## 1.2. Objetivos del TEG

### 1.2.1. Objetivo General

Desarrollar de un modelo de arquitectura orientada a servicios para la gestión de procesos de negocio. Caso de estudio: gestión de proyectos.

### 1.2.2. Objetivos Específicos

- Identificar servicios requeridos por los procesos del caso de estudio Gestión de Proyectos incorporado a la plataforma BPM.

- Diseñar esquema técnico de arquitectura orientada a servicios que permita gestionar la plataforma de base de datos, la plataforma de gestión documental y la interacción vía correo.

- Desarrollar esquema de servicios para ser solicitados desde la plataforma BPMS de acuerdo con el flujo de trabajo del proceso.

- Realizar pruebas funcionales, no funcionales, de calidad y de integración de la arquitectura orientada a servicios, a partir de procesos del caso de estudio Gestión de Proyectos.

## 1.3. Alcance

Este trabajo busca implementar una Solución de Integración para desarrollar un modelo de Arquitectura Orientada a Servicios para la Gestión de Proceso de Negocios que permita la comunicación entre varios componentes a través de servicios web en el caso de Gestión de Proyectos.

La Solución de Integración que utilizaremos será WSO2 Enterprise Integrator el cual tiene todo lo necesario para un modelo SOA.

Se desarrolla un proceso llamado Consultar Proyecto en el BPMS BonitaSoft qué hace uso de los componentes: PostgreSQL como Sistema Manejador de Base de Datos, Alfresco como Gestor Documental y GMAIL como medio para envío de correos.

Se hace uso de los servicios para eliminar el uso de conectores de BonitaSoft a componentes específicos para así obtener los beneficios que este modelo SOA conlleva.

# CAPÍTULO 2
# MARCO CONCEPTUAL

En este capítulo se definen los conceptos relacionados a este TEG, empezando por los Sistemas de Información ya que se hace uso de estos sistemas. A continuación, la definición de Sistemas de Información.

## 2.1. Sistemas de Información

Para poder abarcar el concepto de Sistemas de Información es necesario definir como concepto un dato, este es un hecho aislado que por sí mismo no tiene relevancia alguna, un dato puede convertirse en información el cual se define como "Los datos que se han modelado en una forma significativa y útil para los seres humanos" (Kenneth C.Laudon & Jane P.Laudon, 2012).

Por otra parte, existen los sistemas, un sistema de define como un conjunto de partes coordinadas y en interacción para alcanzar un conjunto de objetivos (Johansen, 2004). Sin la existencia de una relación entre estas partes y un objetivo en común solo tendremos un conjunto de objetos.

Al tener claro lo que significa sistema e información podemos definir Sistema de Información como "Un conjunto de componentes interrelacionados que recolectan (o recuperan), procesan, almacenan y distribuyen información para apoyar los procesos de toma de decisiones y de control en una organización" (Kenneth C.Laudon & Jane P.Laudon, 2012). Además, estos sistemas también sirven de ayuda para los gerentes y trabajadores en cuanto el análisis de problemas, temas complejos y la creación de nuevos productos.

Los Sistemas de Información contienen información sobre personas, lugares y cualquier cosa que sea importante para la organización ya sea dentro de ella o en el entorno que lo rodea.

### 2.1.1. Tipos de Sistemas de Información

Según Kenneth C.Laudon & Jane P.Laudon (2012) existen diversos tipos de sistema de información, cada uno con distintos objetivos, pero enfocado a la empresa y a sus diferentes niveles, Los Sistemas de Información que dan servicio a la gerencia operacional son Sistemas de Procesamiento de Transacciones (TPS), como los de nómina o de procesamiento de pedidos, que rastrean el flujo de las transacciones diarias de rutina necesarias para realizar negocios. Los Sistemas de Información Gerencial (MIS) producen informes que dan servicio a la gerencia de nivel medio, puesto que condensan la información de los TPS y éstos no son muy analíticos. Los Sistemas de Soporte de Decisiones (DSS) dan soporte a las decisiones gerenciales que son únicas y cambian con rapidez, mediante el uso de modelos analíticos

avanzados. Todos estos tipos de sistemas proveen una inteligencia de negocios que ayuda a los gerentes y empleados de la empresa a tomar decisiones más informadas. Estos sistemas para la inteligencia de negocios dan servicio a varios niveles de la gerencia, e incluyen Sistemas de Apoyo a Ejecutivos (ESS) para la gerencia de nivel superior, que proveen datos en forma de gráficos, diagramas y tableros de control, los cuales se ofrecen a través de portales mediante el uso de muchas fuentes de información internas y externas. En la Figura 2.1 se observan los distintos tipos de Sistemas de Información y hacia quién están dirigidos.

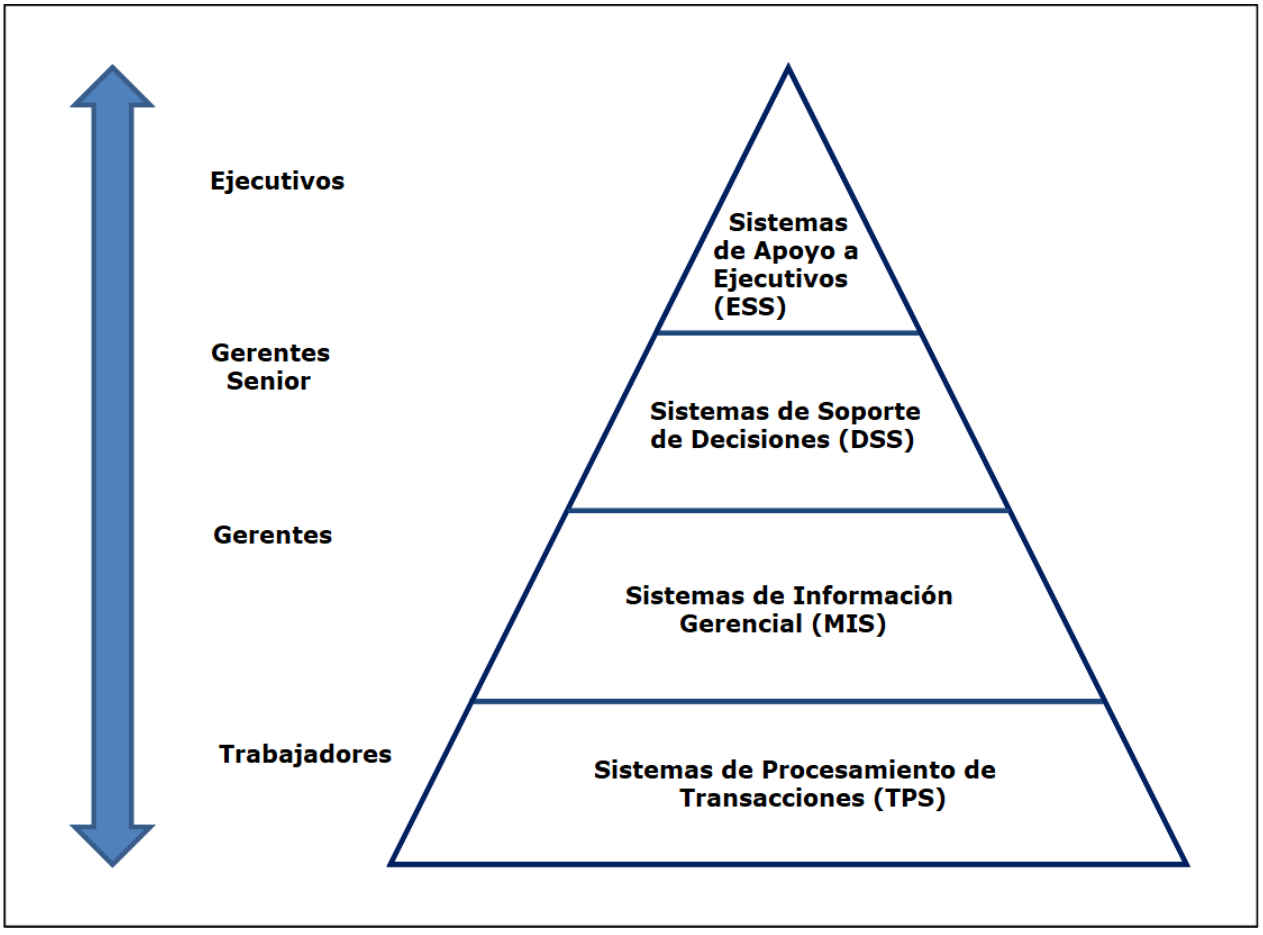


**Figura 2.1 Tipos de Sistema de Información.**

Fuente: https://sites.google.com/site/xatzilihc/sistemas-de-informacion-tecnologias-y-tipos.

A continuación, se explica cada tipo de Sistema de Información.

### 2.1.1.1. Sistema de Procesamiento de Transacciones (TPS)

Un Sistema de Procesamiento de Transacciones es un sistema computarizado que efectúa y registra las transacciones diarias de rutina necesarias para realizar negocios, como introducir pedidos de ventas, reservaciones de hoteles, nómina, registro de empleados y envíos. El principal propósito de los sistemas en este nivel es responder a las preguntas de rutina y rastrear el flujo de transacciones por toda la organización, por lo general la información debe estar fácilmente disponible, actualizada y precisa. En el nivel operacional, las tareas, recursos y metas están predefinidos y muy estructurados. Por ejemplo, la decisión de otorgar crédito a un cliente la realiza un supervisor de nivel inferior, de acuerdo con ciertos criterios predefinidos. Todo lo que se debe determinar es si el cliente cumple o no con los criterios.

#### 2.1.1.2. Sistemas de Información Gerencial (MIS)

Los Sistemas de Información Gerencial son el estudio de los Sistemas de Información en los negocios y la administración. El término Sistemas de Información Gerencial (MIS) también designa una categoría específica de sistemas de información que dan servicio a la gerencia de nivel medio. Los MIS proveen a los gerentes de este nivel reportes sobre el desempeño actual de la organización. Esta información se utiliza para supervisar y controlar la empresa, además de predecir su desempeño en el futuro.

Los MIS sintetizan e informan sobre las operaciones básicas de la compañía mediante el uso de datos suministrados por los TPS. Los datos básicos de las negociaciones que proporcionan los Sistemas de Protección de Alertas (TPWS) se comprimen y, por lo general, se presentan en informes que se producen en un itinerario regular. En la actualidad, muchos de estos reportes se entregan en línea.

Los MIS dan servicio a los gerentes que se interesan principalmente en los resultados semanales, mensuales y anuales. Por lo general estos sistemas responden a las preguntas de rutina que se especifican por adelantado y tienen un procedimiento predefinido para contestarlas. Por ejemplo, los informes del MIS podrían comparar las cifras de ventas anuales totales de productos específicos para objetivos planeados. En general, estos sistemas no son flexibles y tienen poca capacidad analítica. La mayoría de los MIS usan rutinas simples, como resúmenes y comparaciones, a diferencia de los sofisticados modelos matemáticos o las técnicas estadísticas.

#### 2.1.1.3. Sistemas de Soporte de Decisiones (DSS)

Los Sistemas de Soporte de Decisiones (DSS) brindan apoyo a la toma de decisiones que no es rutinaria. Se enfocan en problemas que son únicos y cambian con rapidez, para los cuales el proceso para llegar a una solución tal vez no esté por completo predefinido de antemano. Tratan de responder a preguntas como éstas: ¿Cuál sería el impacto en los

itinerarios de producción si se duplicarán las ventas en el mes de diciembre? ¿Qué ocurriría con nuestro rendimiento sobre la inversión si se retrasara el itinerario de una fábrica por seis meses? Aunque los DSS usan información interna de los TPS y MIS, a menudo obtienen datos de fuentes externas, como los precios actuales de las acciones o los de productos de los competidores. Estos sistemas usan una variedad de modelos para analizar los datos y están diseñados de modo que los usuarios puedan trabajar con ellos de manera directa.

#### 2.1.1.4. Sistemas de Apoyo a Ejecutivos (ESS)

Los Sistemas de Apoyo a Ejecutivos (ESS) ayudan a la gerencia de nivel superior a tomar resoluciones de problemas. Se encargan de las decisiones no rutinarias que requieren de juicio, evaluación y perspectiva, debido a que no hay un procedimiento acordado de antemano para llegar a una solución. Los ESS presentan gráficos y datos de muchas fuentes a través de una interfaz sencilla de manejar para los gerentes de nivel superior. A menudo la información se ofrece a los altos ejecutivos por medio de un portal, el cual utiliza una interfaz Web para presentar contenido de negocios personalizado e integrado.

El Sistema de Gestión de Procesos de Negocio (BPMS) es un TPS ya que efectúa y registra las transacciones diarias de rutina necesarias para realizar negocios a través de procesos automatizados.

Por otra parte, estos sistemas deben almacenar datos en algún repositorio para ser utilizados como información, a continuación, se explica la definición de Base de Datos.

## 2.2. Base de Datos

Las Bases de Datos representan un componente fundamental para los Sistemas de Información que requieren manejar y almacenar pequeños o grandes volúmenes de información.

Di Vasta & Díaz (2001) definen una Base de Datos (BD) como un repositorio centralizado de datos lógicamente relacionados, que permite almacenar y organizar hechos o eventos y restituirlos a demanda de él, o los usuarios para producir información.

Esto quiere decir que una Base de Datos se puede ver como una colección (finita) de datos que están organizados y estructurados siguiendo un modelo de información el cual refleja los datos y las relaciones entre ellos.

A continuación, se define Modelo de Datos.

### 2.2.1. Modelos de Datos

Un modelo de datos es una colección de herramientas conceptuales para la descripción de datos, relaciones entre datos, semántica de los datos y restricciones de consistencia, es decir, los modelos de datos no son algo físico, son una idea que permite la implementación de un Sistema de Base de Datos.

Existe un Modelo de Datos relacional que son utilizados por las BD relacionales, a continuación, se explica la definición este concepto.

### 2.2.2. Base de Datos Relacionales (RDB)

El modelo relacional se ha establecido actualmente como el principal Modelo de Datos para las aplicaciones de procesamiento de datos debido a su simplicidad.

Una base de datos relacional consiste en un conjunto de tablas, donde a cada una ellas se le asigna un nombre exclusivo. Cada tabla tiene una estructura con varias columnas con nombre único entre ellas, cada fila de la tabla representa una relación entre un conjunto de valores y son representados como registros. En la Figura 2.2 se muestra una representación de la tabla relacional.

| *nombre de la sucursal* | *ciudad de la sucursal* | *activos* |
|---|---|---|
| Galapagar | Arganzuela | 7.500 |
| Centro | Arganzuela | 9.000.000 |
| Becerril | Aluche | 2.000 |
| Segovia | Cerceda | 3.700.000 |
| Navacerrada | Aluche | 1.700.000 |
| Navas de la Asunción | Alcalá de Henares | 1.500 |
| Moralzarzal | La Granja | 2.500 |
| Collado Mediano | Aluche | 8.000.000 |

**Figura 2.2 Representación de tabla relacional.**

Fuente: Fundamentos de bases de datos Silberschatz Korth Sudarshan (2002).

Las RDBs son implementadas a través de un sistema que permita la creación y gestión de una BD, a continuación, se explica la definición de Sistemas Manejadores de Base de Datos.

### 2.2.3. Sistemas Manejadores de Base de Datos (SMBD)

Para implementar el modelo de datos explicado anteriormente, es necesaria una herramienta de Software que ejerza el papel de interfaz entre el usuario, los modelos y el sistema físico. Esta es la función que ejercen los SMBD. Estos sistemas tienen la capacidad de crear y gestionar las Bases de Datos, manejar el control de accesos a la información y brindar herramientas para la manipulación de los datos de acuerdo a las necesidades de los usuarios.

De acuerdo con De Miguel, Piattini y Marcos (2002), los SMBD se pueden definir como el conjunto de programas, procedimientos, lenguajes, entre otros, que suministran, tanto a los usuarios no informáticos como a los analistas, programadores o administradores, los medios necesarios para describir, recuperar y manipular los datos almacenados en la Base de Datos, manteniendo su integridad, confidencialidad y seguridad.

Entonces, las Bases de Datos pueden ser creadas, mantenidas y gestionadas por un SMBD, siguiendo unos parámetros que permiten que pueda ser consultada por otros usuarios. Entre las principales funciones de los SMBD se tienen:

- Almacenar, recuperar y actualizar los datos en el disco.
- Manejar la integridad y consistencia de los datos cuando estos son actualizados.
- Coordinar la concurrencia de acceso de los diferentes usuarios a la Base de Datos, garantizando que los datos no se dañen.

Se puede decir que, en términos ideales, los SMBD que sean considerados completos, que manejen grandes cantidades de datos y que soporten múltiples usuarios accediendo a ellos, deben cumplir con dichas funciones.

En la actualidad existen múltiples SMBD por lo que en el presente trabajo se realizó un estudio de las características más relevantes de los principales SMDBR. En la Tabla 2.1 se presenta una comparación entre PostgreSQL y los SMBD más populares actualmente.

Tabla 2.1. Comparación entre SMBDs.

| Nombre | Microsoft SQL Server | MySQL | Oracle | PostgreSQL |
|---|---|---|---|---|
| Desarrollador | Microsoft | Oracle (2010) | Oracle | PostgreSQL Global Development Group |
| Año de inicio | 1989 | 1995 | 1980 | 1989 |
| Licencia | Comercial | Open Source | Comercial | Open Source |
| Lenguaje de implementación | C++ | C y C++ | C y C++ | C |

| | | | | |
|---|---|---|---|---|
| Soporta XML | Si | Si | Si | Si |
| Script lado servidor | Transact SQL y lenguajes .NET | Si | PL/SQL | Usuario define (elije el lenguaje) |
| Triggers | Si | Si | Si | Si |
| Sistemas Operativos de servidor | Linux<br>Windows | FreeBSD<br>Linux<br>OS X<br>Solaris<br>Windows | AIX<br>HP-UX<br>Linux<br>OS X<br>Solaris<br>Windows<br>z/OS | FreeBSD<br>HP-UX<br>Linux<br>NetBSD<br>OpenBSD<br>OS X<br>Solaris<br>Unix<br>Windows |

En este trabajo se usará PostgreSQL ya qué podemos hacer uso de la herramienta pgAdmin qué nos ofrece un monitoreo de las solicitudes a datos sobre una base de datos, además PostgreSQL tiene una licencia OpenSource y no esta bajo las restricciones de la licencia de Oracle como MySQL.

A continuación, se presenta en detalle el Sistema Manejador de Base de Datos que se utilizará en este trabajo.

#### 2.2.3.1. PostgreSQL

PostgreSQL es un SMBD objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Al ser Software libre, el desarrollo de este sistema es dirigido por una comunidad de desarrolladores apoyados por organizaciones comerciales. PostgreSQL utiliza un modelo cliente/servidor y usa multiproceso en vez de multihilos para garantizar la estabilidad del sistema. De esta forma un fallo en uno de los procesos no afectará al resto y el sistema continuará funcionando.

- **PgAdmin**

PgAdmin es la plataforma de administración y desarrollo más popular para PostgreSQL, este está diseñado para usuarios de PostgreSQL novatos o experimentados ofreciendo una interfaz gráfica que simplifica la creación, mantenimiento y uso de las bases de datos.

Los datos e información son utilizados por los procesos de negocio para cumplir con los requerimientos del negocio. A continuación, se explica la definición de Procesos de Negocio.

## 2.3. Procesos de negocio

Para entender la explicación de un Proceso de Negocio se debe explicar primero la definición de Proceso.

### 2.3.1. Proceso

Existen muchas definiciones para el término proceso donde cada una se diferencia, en el contexto de BPM un proceso es un conjunto de actividades sistemáticas que llevan un "evento de negocio" a un resultado exitoso, también es válido definir proceso como una colección de actividades de negocio que crean valor para un cliente.

Thompson, R.J y Redstone, L (2002) definen un proceso como una serie de actividades interconectadas en busca de un propósito. En un contexto de negocios, el propósito de los procesos será sus clientes internos con sus requerimientos de una manera oportuna. En la Figura 2.3 se puede observar una representación del proceso.

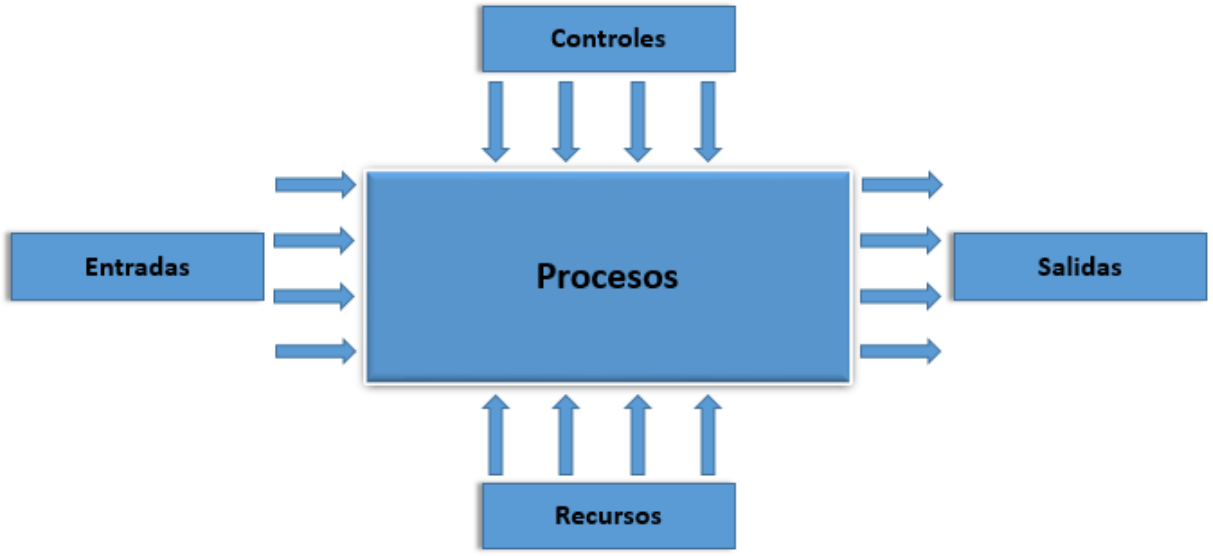

**Figura 2.3 Representación de proceso.**

Fuente: https://iso9001calidad.com/elementos-de-un-proceso-30.html.

Los procesos pueden tener valor para una organización, a continuación, se explica la definición de Procesos de Negocio.

## 2.3.2. Proceso de Negocio (BP)

Sánchez Schenone (2011) explica que en la actualidad las organizaciones tienen como un concepto fundamental la orientación al cliente por lo que poseen procesos que van añadiendo valor a un producto o servicio destinado al mismo, estos se denominan procesos de negocio.

Un proceso de negocio puede ser parte de otro proceso mayor que lo abarque 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 a varios niveles de granularidad.

Cada organización deberá identificar estos procesos dependiendo de las actividades que realice y teniendo en mente en todo momento la orientación al cliente final, estos procesos se denominan procesos operativos de negocio o centrales.

También es clave definir claramente las relaciones cliente-proveedor existentes entre los diferentes procesos ya que los objetivos de cada uno de ellos se establecerán en función sobre todo de este aspecto, estos procesos se denominan procesos de apoyo o soporte o incluso de gestión o estratégicos.

Los Procesos de Negocio son un recurso importante para el desempeño y la subsistencia de la competitividad en las empresas. Para representar Procesos de Negocio, en los últimos años se han mejorado lenguajes y han aparecido nuevas notaciones.

Sánchez Schenone (2011) explica que un proceso de negocio está conformado por varios componentes, a continuación, se explica la definición de ellos.

### 2.3.2.1. Componentes de un Proceso de Negocio

- Actividades: son las tareas que debe hacer una persona (Tareas humanas), o debe hacer un sistema dentro del proceso de negocio (tareas automáticas). Por ejemplo: "Revisión de documentos personales" (Tarea humana), o "Envío de correo automático" (Tarea automática).

- Roles y Usuarios: son los responsables de ejecutar las tareas. Por ejemplo: un "Cliente" de un Supermercado.

- Objeto de Negocio: es la información o documento que fluye a través del proceso de negocio. Por ejemplo: la "Solicitud de Afiliación".

- Decisiones: criterios para tomar distintas opciones en los procesos, distintas direcciones en el flujo.

- Subproceso: otro proceso interno, es parte de un proceso de mayor nivel que tiene su propia meta, propietario, entradas y salidas.

Los BP pueden ser gestionados y obtener un mejor resultado para las organizaciones. A continuación, se explica la definición de Gestión de Procesos de Negocio.

## 2.4. Gestión de Procesos de Negocio (BPM)

Para que un proceso sea gestionable se deben establecer los siguientes aspectos: misión o razón de ser del proceso y objetivos e indicadores, entradas o proveedores de las mismas, salidas, propietario del proceso y procedimiento o instrucciones que describen el inicio, desarrollo y final del proceso

Gestión de Procesos de Negocio es la traducción del término Businnes Process Management (BPM), según Sánchez Schenone (2011) BPM busca identificar, diseñar, ejecutar, documentar, monitorear, controlar y medir los procesos de negocios que una organización implementa. El enfoque contempla tanto procesos manuales como automatizados y no se orienta a una implementación de software.

Algo importante a tener presente es que BPM no es una tecnología de software, pero se apoya y hace uso de las mismas para su implementación efectiva.

Dependiendo del uso del enfoque y su aplicación, BPM puede verse como una metodología, como una herramienta estratégica o bien como conjunto de herramientas tecnológicas, no existe definición precisa, todo depende del prisma que utilicemos para ver la realidad.

BPM se basa en muchos principios o consideraciones que atacan a problemas típicos del día a día en empresas y el desarrollo de sistemas de información dentro de las mismas. Todas estas consideraciones llevadas a la práctica efectiva mitigan estos problemas diarios.

BPM considera fundamental el monitoreo del proceso para estar midiendo su performance y detectar posibles falencias. Mediante el monitoreo se puede determinar si el proceso genera los resultados esperados en función de los objetivos del negocio. La creación y uso de métricas y KPIs (Key Performance Indicators), es clave para realizar un control detallado de cada proceso.

Dado el dinamismo externo al cual están expuestas las empresas, es necesario tener una documentación detallada de cada actividad organizacional. BPM impulsa el entendimiento y modelado de todos los procesos por medio de la documentación. Apoyándose en herramientas de software de modelado de procesos este punto puede ser alcanzado eficientemente.

A continuación, se define la explicación de Sistemas de Gestión de Procesos de Negocio.

### 2.4.1. Sistemas de Gestión de Procesos de Negocio (BPMS)

BPMS es una solución tecnológica que apoya la estrategia de BPM en la organización, una compañía centrada en procesos que utilice unos BPMS logra que la ejecución de ellos está basada en tecnología para soportar la operación logrando capturar la información en tiempo real.

BPMS nos permite trabajar con BPM de forma automatizada gracias a un motor de procesos integrando a esto a personas, dispositivos, recursos informáticos y todo tipo de tecnología.

Para soportar esta estrategia es necesario contar con un conjunto de herramientas que den el soporte necesario para cumplir con el ciclo de vida de BPM, estas herramientas son las siguientes:

- Modelador Gráfico de Procesos: (Business Modeler) permite modelar los procesos de negocio, simular su ejecución, definir métricas para el monitoreo, y exportar a Lenguaje de Ejecución de Procesos de Negocio. Ambiente Integración y Desarrollo:(Integration Developer) es la herramienta que permite implementar los procesos, y servicios. Esta herramienta permite integrar las pantallas (para interacción de un participante), y los servicios (interacción con sistemas legados).

- Servidor de Procesos de Negocio: (Process Server) es el motor que permite ejecutar los procesos de negocio, aquí se ejecutan las Aplicaciones Compuestas (flujos BPM), los Workflows tradicionales, y la Orquestación de Servicios (procesos compuestos solo por servicios). Este servidor también es el encargado de generar los datos de las métricas, y de monitoreo. Permite intervenir los procesos en tiempo real: balancear carga, cambiar flujo de negocio, y realizar acciones correctivas (según reglas de negocio).

- Monitor de Actividades de Negocio: (BAM, Business Activity Monitor) esta es una aplicación de administración que permite gestionar los procesos y servicios, gráficamente se pueden ver indicadores de performance y Acuerdos de Nivel de Servicios (SLA - Service Level Agreements). Se puede además definir alertas y triggers de acuerdo a eventos de negocio que sucedan en el proceso. También puede proveer datos reales a los modelos (Business Modeler) para ajustar las simulaciones (y lograr mejoramiento continuo).

Para poder modelar los BP en un BPMS se utiliza un modelo y una notación que facilita el entendimiento del proceso. A continuación, se explica la definición de Modelo y Notacion de Procesos de Negocios.

## 2.4.2. Modelo y Notación de Procesos de Negocios (BPMN)

Business Process Model and Notation (BPMN), en español Modelo y Notación de Procesos de Negocio, es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo (workflow). BPMN fue inicialmente desarrollada por la organización Business Process Management Initiative (BPMI), y es actualmente mantenida por el Object Management Group (OMG), después de la fusión de las dos organizaciones en el año 2005.

A continuación, se nombran las características de BPMN.

### 2.4.2.1. Características

- BPMN es un estándar, un lenguaje que se ha establecido para representar diferentes situaciones de negocio, NO es una metodología.
- BPMN tiene poder de expresión, un proceso puede ser descrito de forma precisa utilizando este lenguaje.
- BPMN es importante porque permite a su organización tener un diagrama completo y preciso en fácil entendimiento con todos los interesados del negocio.
- BPMN es un estándar internacional de modelado de procesos aceptado por la comunidad.
- BPMN es independiente de cualquier metodología de modelado de procesos.
- BPMN crea un puente estandarizado para disminuir la brecha entre los procesos de negocio y la implementación de estos.
- BPMN permite modelar los procesos de una manera unificada y estandarizada permitiendo un entendimiento a todas las personas de una organización
- BPMN está planeada para dar soporte únicamente a aquellos procesos que sean aplicables a procesos de negocios. Esto significa que cualquier otro tipo de modelado realizado por una organización con fines distintos a los del negocio no estará en el ámbito de BPMN.

- BPMN 2.0 es la última especificación de OMG en el modelamiento de procesos de negocio.

En la **Figura 2.4** se puede observar un breve resumen de los elementos de BPMN 2.0.

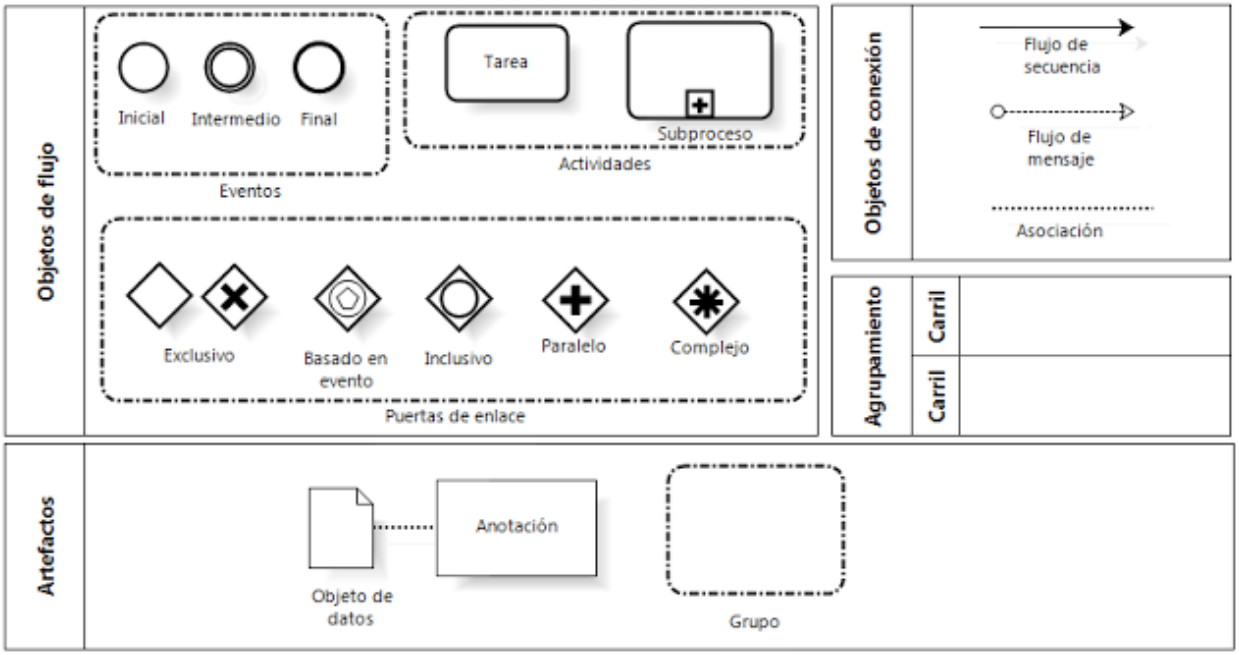


**Figura 2.4. Elementos BPMN 2.0.**

Fuente: https://informatica763.webnode.mx/news/actividad-6-objetos-de-flujo-bpm-business-process-model-and-notation-bpmn-/

### 2.4.2.2. Beneficios de una Herramienta BPM en una Organización

Automatizar los procesos supone una gran mejora en la actividad diaria de los empleados, facilitando la comunicación entre los distintos ejecutores que deben interactuar durante el ciclo de vida del proceso. Mejorar la comunicación entre los distintos elementos de la empresa conlleva una optimización de recursos, ya que los empleados se centrarán en la realización de la actividad y no en la recopilación de la información necesaria para llevarla a cabo. A mayores, se procede a la asignación de las actividades que debe realizar cada trabajador, acotando los límites de cada una y estableciendo pautas a seguir durante el flujo del proceso.

Entre otros beneficios esta:

- Mejorar los sistemas de calidad de la empresa.

- Mejorar el proceso de producción.
- Reducir los tiempos de ejecución de las actividades.
- Establecer puntos críticos como cuellos de botella.
- Mejorar la comunicación interna de la propia organización.
- Ayudar al cumplimiento de las distintas legislaciones vigentes.
- Restringir el acceso a la información: copias controladas, protección de datos, sistema de permisos.
- Monitorización y trazabilidad de procesos.
- Automatización de los procesos.
- Optimizar los recursos de la organización.
- Mayor alineación entre negocio y sistemas.

Los BPs existen en herramientas que crean soluciones basadas en BPM, a continuación, se explica algunas herramientas que utilizan BPM.

### 2.4.3. Herramientas BPM

En la actualidad existen diversas herramientas BPM que nos permiten generar una solución basada en la gestión de procesos de negocio, todas ellas tienen características similares y diferencias que al momento de seleccionar una de ellas debemos elegir según nuestras necesidades, características y limitaciones.

Algunas de estas herramientas son:

- BonitaSoft.
- Bizagi.
- Intalio.
- ProcessMaker.

- AquaLogic BPM Suite.
- IBM Business Process Manager.

En la **Tabla 2.2** se compara BonitaSoft con otras herramientas BPM utilizadas actualmente.

Tabla 2.2. Comparación entre Bonita, Bizagi e Intalio.

| Nombres | BonitaSoft | Bizagi | Intalio |
|---|---|---|---|
| Servidor | Contenedor de Servlets (JBoss, Tomcat, etc) | Versión .NET – IIS (Microsoft Internet Information Services)<br>Versión J2EE – Weblogic / Websphere / JBoss | Apache Gerónimo J2EE |
| Entorno de desarrollo | Propio basado en Eclipse (Multiplataforma, Java) | Propio (Multiplataforma, Java, .NET) | Propio Basado en Eclipse |
| Bases de datos | Hsql, PostGreeSql, MySql, Oracle, SQL Server | SQL Server, Oracle. | MySql |
| Formularios | Muy configurables | Configurables a nivel del desarrollo | Configurables a nivel del desarrollo |
| BPMN 2.0 | Si | Si | Si |
| Validaciones | Si | Si | Si |
| FrontEnd | Envió y recepción de notificaciones simple, Posibilidad de crear un FrontEnd independiente utilizando Bonita como motor de aplicaciones. | Envió y recepción de notificaciones completo y elaborado. | Envió y recepción de notificaciones simple. |

En este TEG se usará BonitaSoft como BPMS por su completa documentación, gran comunidad y fácil configuración.

#### 2.4.3.1. BonitaSoft

Plataforma para aplicaciones basada en BPM que proporciona un entorno único integrado para el desarrollo y despliegue de procesos de negocio a través de sus tecnologías vanguardistas:

- Herramienta de modelización con la notación BPM (BPMN).
- Motor de procesos (flujo de tareas).
- Ejecución en tiempo real de las decisiones de negocio.
- Diseñador de las interfaces de usuario.
- Acceso móvil a los procesos.
- Monitoreo y optimización.
- Funcionalidades de integración para conectar las aplicaciones con servicios de terceros (bidireccional).
- **Beneficios**
    - Los usuarios clave diseñan el proceso de negocio con una interfaz sencilla, durante el desarrollo del proceso el equipo trabaja conjuntamente para definir el modelo de datos y luego se formaliza el contrato para poder trabajar en paralelo la interfaz de usuario y la lógica de negocio.
    - Bonita BPM hace más fácil la integración de la aplicación con otros sistemas de información como el email la gestión de documentos e incluso la sincronización con la organización corporativa.
    - Bonita BPM permite a los desarrolladores construir paneles e informes para que los managers puedan monitorizar la actividad de negocio y seguir de cerca los indicadores de rendimiento claves.

- Con Bonita BPM se puede construir aplicaciones que se adapten fácilmente y en tiempo real a las nuevas necesidades de negocio, nuevos entornos software, nuevas necesidades de usuario y cambios en el entorno de negocio.

- Con Bonita BPM los desarrolladores emplearan más tiempo añadiendo valor al negocio y menos tiempo implementando y manteniendo aplicaciones.

- **Arquitectura de BonitaSoft**

A continuación, en la figura 2.6 se muestra la arquitectura de BonitaSoft.

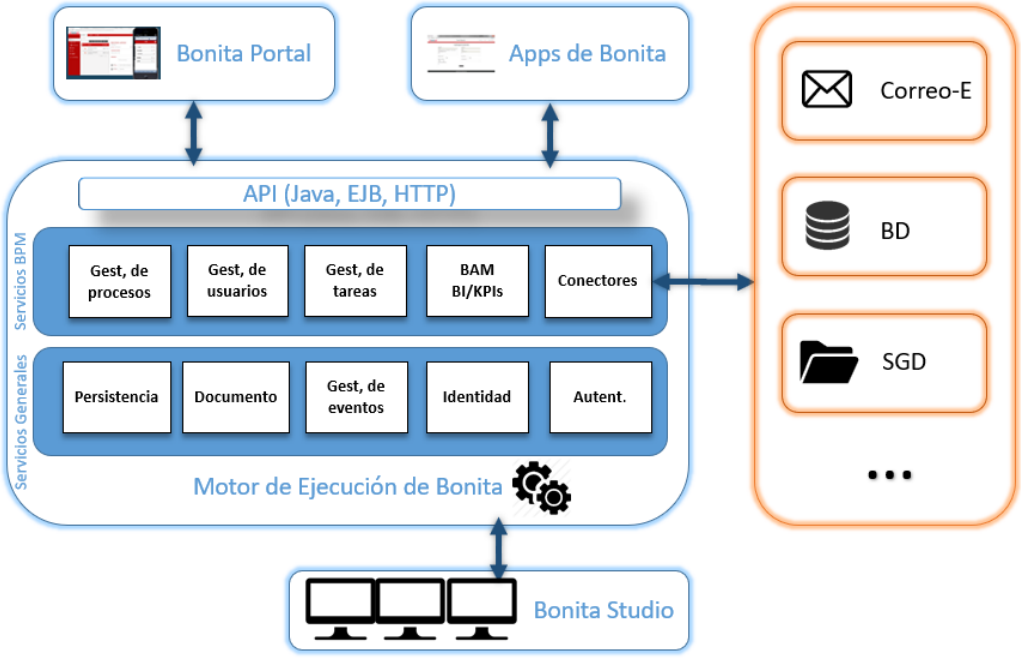


**Figura 2.5 Arquitectura de BonitaSoft.**

En este trabajo nos enfocaremos en el módulo de BonitaSoft que contiene a los conectores.

- Motor de Ejecución de Bonita:

El módulo de Motor de Ejecución de Bonita es el encargado de permitir que podamos desarrollar nuestros procesos, encargándose de la conexión de los procesos existentes, su despliegue y su ejecución en el sistema.

En este módulo están los Conectores, estos permiten que BonitaSoft interactúe con componentes o sistemas externos permitiendo la integración de ellos. Tiene un conector

especializado para cada tipo de componente los cuales son configurables por medio de una interfaz, entre estos conectores podemos conseguir conexiones con Base de Datos, Gestores Documentales y envío de correo electrónicos.

En este TEG se utiliza el BPMS BonitaSoft con un modelo de Arquitectura Orientada a Servicios, a continuación, se explica la definición.

## 2.5. Arquitectura Orientada a Servicios (SOA)

SOA (Arquitectura orientada a servicios) se define como un marco de trabajo conceptual para la integración de componentes (fperezsoa 2009), que permite a las organizaciones facilitar el cumplimiento de los objetivos de negocio, además su flexibilidad de integración permite que un sistema pueda convivir con sistemas legados o procesos de negocio.

Utilizando SOA obtenemos una reducción de costos de implementación, innovación al utilizar servicios web, una respuesta rápida ante cambios, además se puede reaccionar rápidamente cuando emergen nuevas tecnologías ya que son aplicaciones independientes, permitiendo que los componentes del proceso se integren y se coordinen de manera efectiva y rápida.

Hoy en día la competencia entre organizaciones es más fuerte obligándolos a responder de manera más rápida y efectiva ante las nuevas tecnologías. Saber actuar ante los cambios que afectan de manera natural a los negocios, optimizar los procesos, reducir los costos de TI, y lograr la flexibilidad son algunos de los factores claves para la competitividad y el crecimiento de las organizaciones.

Para lograr lo antes mencionado se necesita un marco de trabajo conceptual para facilitar la integración de componentes y sistemas heterogéneos, este debe tener un modelo bien establecido, para lograr un nivel óptimo de integración, dando como resultado un fácil manejo de cambios sobre la infraestructura que puedan surgir debido a una nueva tecnología para la organización.

SOA es un concepto que se empezó a conocer en los años 80, cuando llegaron al mercado la computación distribuida y las llamadas a procedimientos remotos.

Gartner describe la arquitectura orientada a servicios por primera vez en 1996, lo cual tomó importancia por la nueva tecnología que se manejaba llamada Servicios Web.

Existieron Arquitecturas para los Sistemas Distribuidos en los 90 que no tuvieron una buena aceptación, pero fue en el 2003, donde SOA entra al fin por completo en el mundo de las TI empresariales, a través de los servicios web.

### 2.5.1. Roles de SOA

Los roles que integran SOA se muestran a continuación en la Figura 2.6.

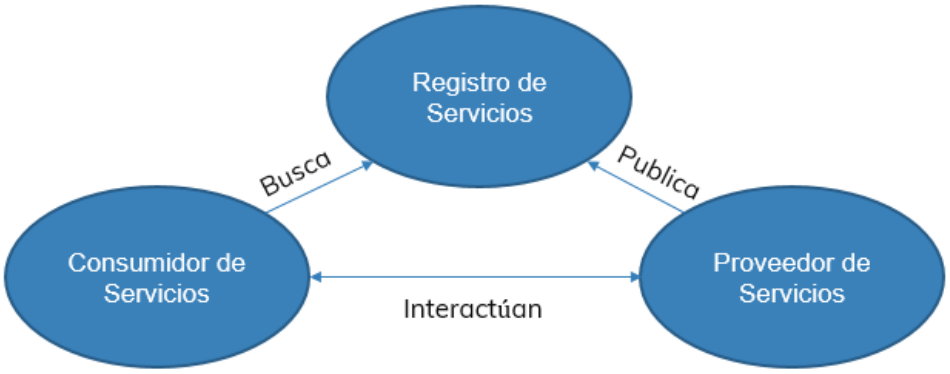


**Figura 2.6 Roles de SOA.**

Fuente: https://www.researchgate.net/figure/SOA-actors-and-roles_fig1_258210036.

Consumidor de servicios: Es una aplicación, un módulo de software u otro servicio que demanda la funcionalidad proporcionada por un servicio, y la ejecuta en una interfaz definida.

Proveedor de servicios: Es una entidad accesible a través de la red que acepta y ejecuta consultas de consumidores y publica sus servicios y su contrato de interfaces en el registro de servicios para que el consumidor pueda descubrir y acceder al servicio.

Registro de servicios: Es un repositorio de servicios disponibles y permitiendo visualizar las interfaces de los proveedores de servicios a los consumidores interesados.

SOA hace uso de servicios web, a continuación, se explica la definición de estos.

### 2.5.2. Servicios Web

Un servicio Web es un conjunto de protocolos y estándares abiertos utilizados para el intercambio de datos entre aplicaciones o sistemas. Los servicios Web son utilizados por aplicaciones realizadas en distintos lenguajes de programación y que se ejecutan en diferentes plataformas para interoperar a través de Internet con el fin de ofrecer servicios entre ellas. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. (W3C)

En los servicios web se hace uso de diversos formatos (XML, JSON, etc.) para representar los datos o información a intercambiar, así como también se hace uso de un conjunto de protocolos sobre los cuales se establece la comunicación.

#### 2.5.2.1. Ventajas de los Servicios Web

Como ventajas de los servicios web tenemos:

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.

- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.

Existen tipos de servicios web los cuales son explicados a continuación.

#### 2.5.2.2. Servicios SOAP

SOAP (Simple Object Access Protocol) es una especificación de protocolo de mensaje que nos permite intercambiar información de manera estructurada obteniendo extensibilidad e independencia.

Estos servicios hacen uso de un lenguaje XML para la estructura de la información. XML (Extensible Markup Language) es un lenguaje de marcado que define un grupo de reglas para la codificación de documentos.

#### 2.5.2.3. RESTful API

REST (Representational State Transfer) a diferencia de SOAP no es un protocolo sino un estilo de arquitectura para los sistemas hipermedias distribuidos. El uso más común es en el campo de las APIs (Application Programming interface) el cual es un software intermediario que permite que dos aplicaciones se comuniquen entre ellos. Los Servicios REST API se utilizan por medio de un URI (Uniform Resource Identifier) específico.

Estos servicios hacen uso mayormente de JSON para obtener o generar operaciones, JSON (JavaScript Object Notation) es un formato ligero de intercambio de data el cual permite la facilidad de lectura, parseo y generación de datos.

REST hace uso de 4 operaciones las cuales son: POST (crear), GET (leer y consultar), PUT (editar) y DELETE (eliminar).

Se debe seguir una estrategia sobre los servicios que nos proporcione eficacia. A continuación, se explica la definición de Gobernabilidad SOA.

### 2.5.3. Gobernabilidad SOA

La gobernabilidad de la Arquitectura Orientada a Servicios, se refiere a la estrategia sobre los servicios de negocio, en la gestión del ciclo de vida de estos servicios, políticas, prácticas y en el uso eficaz de los recursos para implementar SOA en la organización.

Para una gobernabilidad SOA eficaz se debe tomar las decisiones para asegurar el uso eficaz de las Tecnologías de Información (TI), así como tener un encargado para la toma de decisiones y una manera de supervisar y hacer estas decisiones.

La estrategia de orientación a servicios proporciona las condiciones para llevar a la empresa a niveles altos de competitividad y crecimiento, ya que a aporta la flexibilidad necesaria para mantener el éxito en el futuro, terminando además con la desintegración e inflexibilidad, la cual genera costos, reduce la capacidad de respuesta ante los clientes y afecta la productividad de la empresa.

### 2.5.4. Características

Entre las características de SOA tenemos:

- Su capacidad para reutilizar los componentes existentes para nuevos servicios es que se puedan crear bajo la infraestructura de TI existente.

- Su interoperabilidad, que hace posible que los servicios puedan ser consumidos por los clientes en componentes o procesos de negocio distintos.

- Aumenta la eficiencia en los procesos de la organización.

- Reduce costes de mantenimiento.

- Fomenta la innovación orientada al desarrollo de servicios.

- Simplifica el diseño, optimizando la capacidad de organización.

En la Figura 2.7 se observa el ciclo de vida de la gobernabilidad SOA.

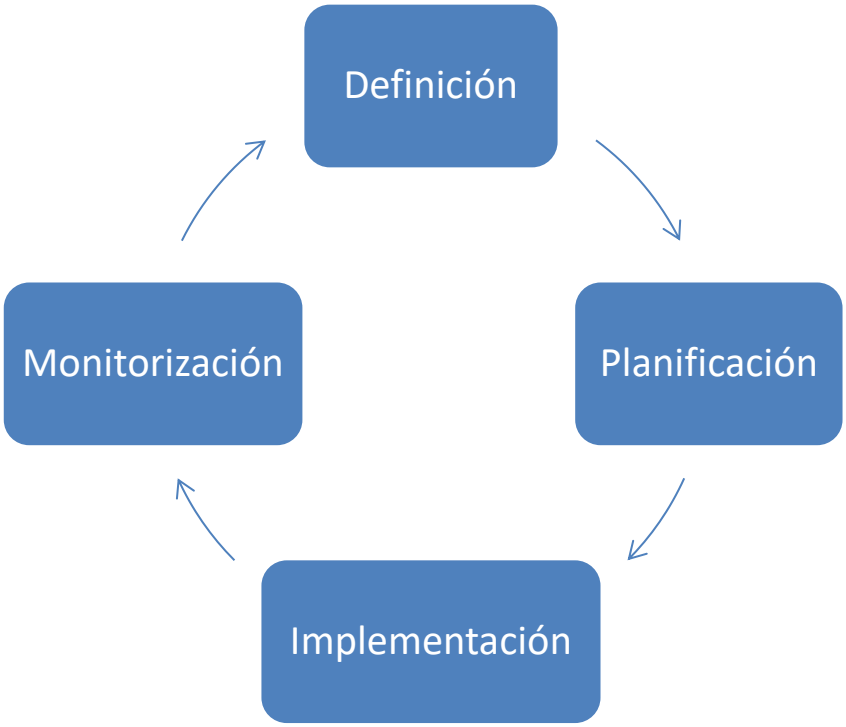


**Figura 2.7 Ciclo de vida de la Gobernabilidad SOA.**

A continuación, se explica cada fase.

- Fase de planificación: Esta se encarga de identificar y analizar las áreas principales de gobernabilidad que se deben mejorar, también se establece los objetivos y medidas necesarias para lograr el avance propuesto, así como también se evalúa avances ya implementados en busca de mejoras.

- Fase de definición: Esta se encarga de definir los planes del Modelo de Gobernabilidad SOA requeridos para cumplir con los objetivos definidos durante la fase de planificación.

- Fase de implementación: Esta se encarga de implementar los planes definidos, incluyendo la implementación de procesos, la organización y los aspectos tecnológicos del Modelo de Gobernabilidad SOA.

- Fase de monitoreo: Esta se encarga de monitorear la efectividad del resultado actualmente implementado y verifica si está cumpliendo con el propósito específico para el que se lo creó. De ser necesario se vuelve a la fase de planificación para una mejora.

### 2.5.5. Beneficios de SOA

SOA ofrece una gran cantidad de beneficios tanto para la empresa como para la organización TI.

Entre los beneficios para la empresa tenemos:

- Una mejor toma de decisiones ya que al manejar la información a través de servicios se puede obtener un monitoreo que nos permita tener una información más exacta y actualizada permitiendo a la organización reaccionar más rápido ante un problema que surja en el sistema.

- Una mayor productividad por parte de los empleados de la organización ya que al tener fácil acceso a los servicios una modificación será más sencilla y rápida, además tienen nuevas formas de acceder a la información de un componente.

- Un aumento en las relaciones con clientes y proveedores por la fácil integración de nuevos componentes a la organización a bajos costos, convirtiéndose en una oferta atractiva para los clientes al momento de querer comercializar u optimizar su negocio.

Entre los beneficios de organización de TI tenemos:

- Un Sistema más productivo y flexible debido a una buena gobernabilidad SOA en donde componentes y sistemas ya existentes sean mejorados para una mayor productividad.

- Un desarrollo de componentes más rápido y económico por la eliminación de redundancia reduciendo así los tiempos de desarrollo y de ciclos de prueba.

- Un sistema más seguro y manejable debido al modelo y la documentación de SOA que nos permite un desarrollo de servicios seguros y gestionables, el acceso ya no será directo a los componentes sino a los servicios.

- Mejora de la capacidad para innovar y diferenciarse desarrollando funcionalidades nuevas de manera más rápida.

- La interoperabilidad entre las componentes y tecnologías heterogéneas.

A continuación, se explica como BPM y SOA pueden trabajar en conjunto.

### 2.5.6. BPM con SOA

El uso de BPM y SOA permite que los servicios sean consumidos por los procesos de negocio orquestados a través de un BPMS mientras que un Solución de Integración gestione los servicios existentes. SOA se encarga de encapsular los componentes funcionales de los sistemas existentes y exponerlos como servicios eliminando los medios directos entre el BPMS y otro sistema. Los servicios además pueden ser reutilizados por el BPMS en todos sus procesos de negocio.

BPM y SOA son independientes. BPM se enfoca en la mejora del rendimiento del negocio a través de procesos mientras que SOA apunta a la agilidad del negocio involucrando las integraciones de TI, SOA mejora el diseño de desarrollo en una organización.

En la actualidad una organización que apunte a ser de gran tamaño y que quiera tener éxito deberá tener como pensamiento empresarial a BPM Y SOA en conjunto. Una gobernabilidad SOA ayudará tener una visión más clara de la arquitectura qué se desarrollará junto a una buena gestión que apoye a la visión de la organización y al cumplimiento de metas.

Existen herramientas que nos facilitan desarrollar un modelo SOA en nuestra organización, a continuación, se explica la definición WSO2 Enterprise Integrator.

## 2.6. WSO2 Enterprise Integrator

Hoy en día existen soluciones que nos facilitan implementar un modelo SOA en nuestra organización una de ellas es WSO2 Enterprise Integrator (EI), una solución de integración comprensiva que permite la comunicación entre aplicaciones dispares, en vez de tener que

comunicarnos directamente con cada uno de las aplicaciones mediante diferentes formatos, cada aplicación puede comunicarse con WSO2 EI.

WSO2 EI está distribuido bajo la Licencia de Apache Software Versión 2.0 además este producto de WSO2 puede ser desplegado de diversas formas: local, en cualquier infraestructura de nube, en nubes privadas e inclusive usando sistemas de contenedores.

WSO2 EI actúa principalmente como un Bus de Servicio Empresarial (ESB) para manejar la transformación de los mensajes y el enrutamiento de su destino apropiado, esto se denomina Perfil ESB en la solución.

### 2.6.1. Bus de Servicio Empresarial (ESB)

El ESB es un modelo de arquitectura de software estándar que permite implementar un modelo SOA y tiene como objetivo impulsar el crecimiento de una organización. Un ESB también se considera como una herramienta de middleware para distribuir el trabajo entre los distintos componentes conectados de una aplicación.

Existen diferentes definiciones dependiendo de la procedencia o el fabricante. Entre otras cosas, un ESB se define como:

- Un estilo de arquitectura de integración que permite la comunicación a través de un bus de comunicación común que consiste en una variedad de conexiones punto a punto entre proveedores y usuarios de los servicios.

- Una infraestructura que una compañía utiliza para la integración de servicios en el paisaje de la aplicación.

- Una arquitectura patrón que permite la interoperabilidad entre entornos heterogéneos, con orientación al servicio.

En la Figura 2.8 se puede observar una representación del Bus de Servicio Empresarial con sus entradas y salidas.


**Figura 2.8 Representación de Bus de Servicio Empresarial.**

En definitiva, un ESB debe ser capaz de reemplazar todo el contacto directo entre aplicaciones, consiguiendo que todas ellas se comuniquen a través del bus.

Los ESB transmiten y reciben mensajes basados en estándares, pero deben ser capaces de transformar mensajes a formatos que sean reconocidos por las distintas aplicaciones en el caso de que sea necesario, lo que se realiza a través de adaptadores, además el intercambio de mensajes debe ser independiente de la plataforma, esto permite al ESB integrar aplicaciones que se ejecutan en diversos sistemas operativos o mainframes.

### 2.6.2. Data Services

Otra funcionalidad que nos ofrece WSO EI es el uso de Data Services que son mecanismos de integración para los repositorios de datos con la finalidad de exponer sus operaciones como servicios web, estos mecanismos son configurables a través de una interfaz gráfica intuitiva que nos facilita la integración de los repositorios.

### 2.6.3. Razones para elegir WSO2 EI

WSO2 EI posee una cantidad de beneficios y facilidades para poder tener un BPMS bajo un modelo SOA, esta solución es ligera y escalable, tiene una documentación completa, una comunidad aceptable para la aclaración de dudas, es de código libre, posee herramientas completas de desarrollo y depuración para agilizar la comunicación entre componentes de un sistema, posee herramientas de monitoreo y análisis para hacer seguimiento a los servicios, entre otros.

En este TEG los servicios pertenecen a un proceso que forma parte de la Gestión de Proyectos, a continuación, se explica la definición de Gestión de Proyectos.

## 2.7. Gestión de Proyectos

Primero debemos saber que un proyecto según PMI (2013) es un esfuerzo temporal emprendido para crear un producto, servicio o resultado. De lo anterior podemos definir la Gestión de Proyectos como la aplicación de conocimiento, habilidades, herramientas y técnicas sobre un proyecto para asegurar sus objetivos.

Según PMI (Project Management Institute) en la Gestión de Proyectos se han identificado 5 grupos de procesos en el transcurso de un proyecto: Iniciación, planificación, control, ejecución y cierre. Estos grupos de proceso poseen una gran cantidad de subprocesos y tareas lo cual convierte a la Gestión de Proyectos en un concepto complejo de manejar y un candidato para que sus procesos sean gestionados por un BPMS.

La Gestión de Proyectos crea, edita, elimina y actualiza documentos relacionados a cada proyecto, como herramienta tecnológica existen los gestores documentales que facilitan estas tareas relacionadas a los documentos. A continuación, se explica la definición de Gestor Documental.

## 2.8. Gestor documental

Los gestores documentales son tecnologías que nos facilitan el uso de documentos de todo tipo que pertenezcan a una organización, ofreciendo una mayor facilidad de ubicación, creación, modificación, eliminación, ordenamiento y control de versiones de los documentos basado en normas y técnicas que se crearon a partir de la gestión de documentos manual.

A continuación, se presenta la definición del Gestor Documental Alfresco el cual se utiliza en este TEG por su documentación, licencia libre y gran comunidad.

### 2.8.1. Alfresco

Según Alfresco Software, Inc., (2019). Alfresco es una tecnología que administra todo tipo de contenido y es de código abierto, permite acelerar el flujo de negocio a partir de una gestión documental eficiente y óptima. Alfresco permite la búsqueda avanzada, la categorización de contenido, el manejo de versiones de documentos y posee una librería de servicios que permite el manejo y almacenamiento de contenido.

#### 2.8.1.1. Servicios de Alfresco

Los servicios son parte fundamental para el servidor de Alfresco, se utilizan en todos los componentes que interactúen con este servidor, como sus componentes para su interfaz en un navegador. Estos servicios pueden ser usados fuera del ambiente de Alfresco permitiendo que otros sistemas puedan interactuar y obtener los beneficios de esta solución.

Para conectar las tecnologías como el Gestor Documental Alfresco y una Base de Datos con BonitaSoft también se hace uso de lenguajes de programación. A continuación, se explica la definición de Lenguajes de Programación.

## 2.9. Lenguajes de Programación

Un lenguaje de programación es aquel elemento dentro de la informática que nos permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes.

Los lenguajes de programación se clasifican por su nivel de abstracción del procesador, en este caso la abstracción es un principio por el cual se aísla toda aquella información que no es resulta relevante a un determinado nivel de conocimiento.

En el caso de BonitaSoft se utiliza los lenguajes de programación JAVA y GROOVY. A continuación, se define cada uno.

### 2.9.1. JAVA

Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web que no funcionarán a menos que tenga Java instalado y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes.

### 2.9.2. GROOVY

Apache Groovy es un poderoso lenguaje de programación, utilizado para la plataforma Java para mejorar la productividad del desarrollador gracias a su sintaxis familiar y fácil de aprender. Se integra sin problemas a cualquier programa Java e inmediatamente proporciona poderosas características, incluyendo la capacidad de hacer scripts, el tiempo de compilación y corrida, así como la programación funcional.

# CAPÍTULO 3
# MARCO METODOLÓGICO

## 3.1. Introducción

Una metodología nos permite definir límites, un orden y un contenido. Construir un software complejo requiere de un gran esfuerzo, tecnología, dinero y recursos. Personas que interactúan entre sí, con diferentes grados de conocimiento, con diferentes roles, con diferentes intereses. Una metodología propone un esquema de trabajo que nos permite entender cuál es nuestro rol dentro del proyecto, nos acerca una cierta sensación de control y de seguridad. Sin un proceso no sabemos cómo comenzar y cuándo terminar.

El término metodología se define como el grupo de mecanismos o procedimientos racionales, empleados para el logro de un objetivo, o serie de objetivos que dirige una investigación científica. Este término se encuentra vinculado directamente con la ciencia, sin embargo, la metodología puede presentarse en otras áreas.

Existen una gran variedad de metodologías para el desarrollo de un software por lo que debemos saber seleccionar la metodología a usar, el objetivo de aplicar una metodología es mejorar la eficiencia por medio de la gestión sistemática de los procesos de negocio, los cuales se deben modelar, integrar, monitorizar y optimizar de manera continua.

## 3.2. Metodologías de desarrollo

Se refiere a un conjunto de técnicas y procedimientos apoyados por una documentación para ser empleados en el desarrollo de un sistema de información. Tiene como objetivo principal exponer una serie de técnicas que permitan el desarrollo de un software de calidad, entre ellas tenemos las metodologías tradicionales y las metodologías ágiles. En el presente capítulo se presenta el estudio de la metodología ágil SCRUM.

## 3.3. Metodologías Ágiles

El desarrollo ágil de software refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, estas metodologías son imprescindibles en un mundo en el que nos exponemos a cambios frecuentemente. Siempre hay que tener en cuenta como programadores que lo que es la última tendencia hoy puede que no exista mañana y por esto

existe la metodología ágil donde los requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinarios.

### 3.3.1. Scrum

La metodología ágil Scrum gestiona el desarrollo de software, teniendo como objetivo principal el mayor retorno de inversión posible para la organización.

Scrum permite obtener un mayor compromiso del cliente debido a que está incluido en cada iteración del ciclo. Así mismo le permite en cualquier momento realinear el software con los objetivos de negocio de su empresa, ya que puede introducir cambios funcionales o de prioridad en el inicio de cada nueva iteración sin ningún problema.

Scrum promueve el compromiso del equipo que forma parte del proyecto, la innovación y la motivación, convirtiéndolo en una metodología favorable para las organizaciones.

#### 3.3.1.1. Proceso y Roles de Scrum

El desarrollo es iterativo, eso quiere decir que se irá desarrollando mediante iteraciones, donde cada iteración tiene una duración entre 3 y 4 semanas denominadas Sprint. Cada Sprint se va modificando la funcionalidad construida y se van añadiendo nuevas características, estas características son seleccionadas bajo un orden de prioridades donde la primera es la que aporta mayor valor a la organización.

A continuación, en la Figura 3.1 se muestra la representación de un Sprint y se explica cada fase y los roles que se utilizan.

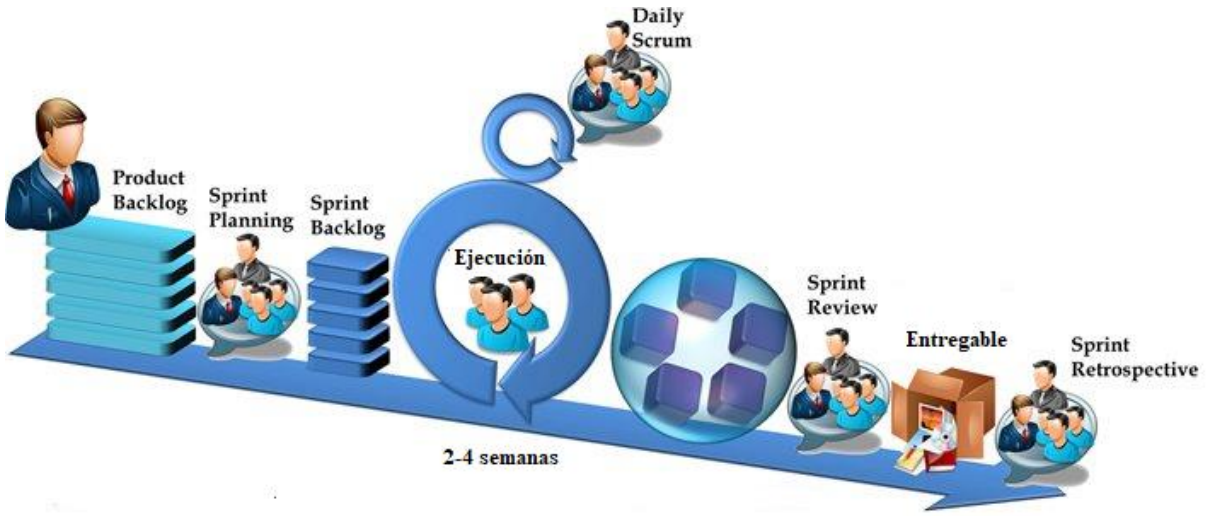


**Figura 3.1 Representación de Sprint en SCRUM.**

Fuente: https://www.stary.co/pm-scrum-intro.

A continuación, se explica en detalle la figura anterior.

- Product Backlog: Lista de funcionalidades ordenadas por prioridad de valor de negocio, visible para todos los miembros de SCRUM donde cualquier integrante puede añadir alguna funcionalidad a esta lista, esta lista suele cambiar por el cambio de prioridad.

- Sprint Planning: Al principio del Sprint se lleva a cabo una reunión para negociar sobre los ítems del Product Backlog que serán desarrollados, el Product Owner es el encargado de declarar cuáles son los ítems de mayor prioridad para la organización. El Team es el responsable de seleccionar la cantidad de trabajo qué sienten qué pueden implementar durante el Sprint.

- Sprint: Iteración de duración prefijada durante la cual el equipo trabaja para convertir los ítems del Product Backlog a las que se ha comprometido, en una nueva versión del software totalmente operativo.

- Sprint Backlog: Lista de los ítems seleccionados del Product Backlog qué van a ser desarrollados durante el Sprint, sirve de referencia para los Dailys Scrum Meeting.

- Daily Scrum meeting: Todos los días todo el equipo Scrum se reúne para inspeccionar el progreso de los objetivos del Sprint y crean un plan para ese día. los miembros del

Team comparten con los demás lo que han hecho el día anterior para alcanzar los objetivos, lo que harán ese día y si tienen algún impedimento.

- Spring Review: Al final del Sprint el Equipo Scrum se reúne para mostrar el incremento del desarrollo realizado en el producto a todo el que esté interesado, esta demostración no debe ser un reporte.

- Demo y retrospectiva: Todo Sprint termina con una retrospectiva, en esta reunión el equipo refleja su propio progreso, inspeccionan su resultado y toman acciones para adaptaciones en Sprint futuros.

En Scrum, existe un equipo encargado de construir el software de calidad. Este equipo se encarga de definir las características del producto y seleccionar en qué orden se van a desarrollar.

El equipo Scrum está formado por los siguientes roles:

- Scrum Master: Es la persona que actúa como facilitador para el Product Owner y el Team. El Scrum Master trabaja para remover los impedimentos que obstruyen al Team para lograr sus objetivos, además guía al equipo para que cumplan con las reglas y procesos de la metodología. Trabaja con el Product Owner para maximizar el retorno de la inversión.

- Product Owner: Es la persona con visión, autoridad y disponibilidad. Él es el responsable de comunicar la visión y las prioridades al equipo de desarrollo.

- Team: Es el grupo de personas responsables de organizarse y completar el trabajo. Poseen el conocimiento necesario para desarrollar el proyecto. El Team tiene la autonomía y autoridad para cumplir los objetivos del Sprint.

#### 3.3.1.2. Beneficios

Entre los beneficios de usar esta metodología tenemos:

- Mejor Calidad: Scrum tiene un feedback continuo del producto y muestras en vivo del producto lo que permite que la calidad sea la más alta posible.

- Menor tiempo para salir al mercado: Scrum suele entregar al cliente un producto más rápido que los métodos tradicionales debido al desarrollo temprano, el orden de prioridad de los requerimientos y la finalización de cada Sprint.

- Incremento del retorno de inversión: un beneficio producido por el beneficio de salir antes al mercado ya que se empieza a tener ingresos en tiempos menores.

- Alta satisfacción del cliente: esto debido a la colaboración y papel fundamental que tienen durante los Sprint, interviniendo y participando en las reuniones, observando los avances en vivo, entregando un producto más rápido, entre otros.

- Ánimos del Equipo altos: el equipo se siente motivado al participar en reuniones donde pueda opinar, innovar y aportar valor al producto, además de tener una auto organización y de tener apoyo del equipo al momento de presentarse un impedimento y se comunica en una reunión.

- Reducción de riesgos: Scrum ayuda a mitigar el riesgo de un proyecto fallido y se pierda grandes cantidades de dinero y tiempo sin retorno de inversión. Al desarrollarse primero las funcionalidades de mayor valor se manejan los errores que puedan surgir desde un inicio, además se tiene una visión de avance del equipo durante el desarrollo del producto.

- Cumplimento de expectativas: El cliente tiene sus expectativas al fijar qué ítems del Backlog se van a desarrollar primero y en cuanto tiempo en el inicio del Sprint, al igual que al final del Sprint comprueba que efectivamente se han cumplido con los desarrollos planificados en el producto.

- Flexibilidad a cambios: Alta capacidad de reacción ante los cambios de requerimientos generados por necesidades del cliente o evoluciones del mercado. La metodología está diseñada para adaptarse a los cambios de requerimientos que conllevan los proyectos complejos.

- Mayor productividad: Se consigue entre otras razones, gracias a la eliminación de la burocracia y a la motivación del equipo que proporciona el hecho de que sean autónomos para organizarse.

- Predicciones de tiempos: Conforme se van haciendo Sprints se puede conocer la velocidad del equipo desarrollando una funcionalidad, en base a estos tiempos se

pueden estimar para futuros Sprints los tiempos de funcionalidades que aún siguen en el Backlog.

En este trabajo se hace uso de la Metodología Scrum ya que se presentarán varios cambios por la naturaleza de los procesos de negocio.

## 3.4. Ciclo de vida BPM

Tradicionalmente se automatizan los procesos de negocio usando flujos de trabajo para implementar el proceso automatizado, pero en el ámbito de Gestión de Procesos de Negocios existe una metodología llamada ciclo de vida BPM cuyo objetivo es mejorar la eficiencia a través de la gestión sistemática de los procesos de negocio de una organización.

Esta metodología es propuesta por Smith y Fingar (2003) y nos indica que las actividades o tareas que se desarrollan en BPM pasan por distintas etapas, cada una con un enfoque específico. Como lo muestra la Figura 3.2, el ciclo comienza con un Descubrimiento (análisis), Diseño (modelado), Ejecución, Monitoreo, y Optimización.

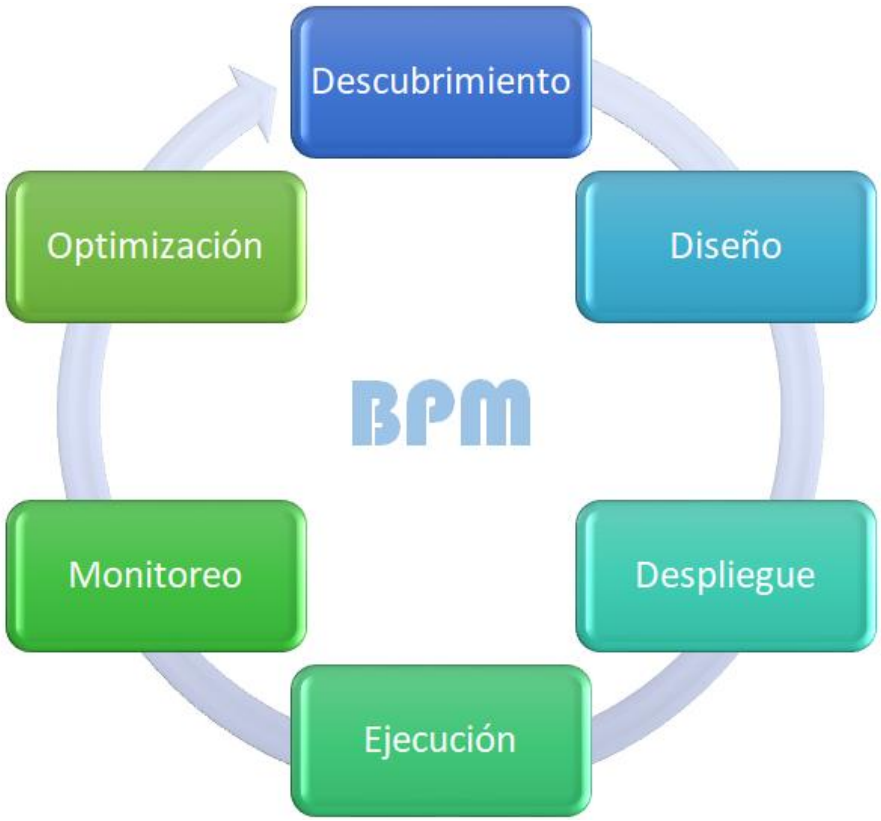

**Figura 3.2 Ciclo de vida BPM.**

A continuación, se explica en qué consiste cada etapa de la metodología del ciclo de vida BPM.

#### 3.4.1.1. Descubrimiento

En esta etapa se busca lograr una descripción de las diferentes actividades que componen el proceso. Estas actividades generalmente están inmersas en los participantes del proceso y en los sistemas que lo soportan y lo que se pretende es obtener el conocimiento contenido en estas fuentes para poder volver explícito el proceso. El descubrimiento intenta describir cómo se hacen las cosas en la actualidad, de una manera objetiva, en vez de conciliar las apreciaciones subjetivas de los diferentes participantes del proceso.

#### 3.4.1.2. Diseño

Se encarga de reunir suficientes detalles acerca del proceso para poder entenderlo conceptualmente, también se debe garantizar que el alto nivel de detalle es el correcto, así como tener claro sus funcionalidades, reglas de negocio y beneficios que se obtiene. Si el proceso ya existe se debe buscar mejoras o un rediseño del mismo. Este nuevo proceso debe tener un objetivo y no ser creado sin razón.

#### 3.4.1.3. Despliegue

Una vez se tenga el diseño del proceso entonces se debe hacer la implementación, esto se logra con el uso de las tecnologías y softwares que sean necesarios para permitir el despliegue.

#### 3.4.1.4. Ejecución

En esta etapa se busca administrar las actividades llevadas a cabo por cada participante del proceso para asegurar la correcta ejecución del mismo. El sistema administrador de procesos es el que se encarga de esta etapa y es responsable de que las diferentes plataformas o sistemas que soportan el proceso sean transparentes para el usuario. Además, deben asegurar la correcta comunicación entre los diferentes entes que intervienen en el proceso y el correcto funcionamiento de los mismos. Además, aquí es cuando se recolecta la información para el monitoreo de los procesos.

Es la etapa en donde se explota el proceso desarrollado previamente mediante:

- Espacio de trabajo.
- Lista de tareas pendientes.
- Gestión de excepciones.
- Inicio de procesos desde formularios.
- Configuración y personalización del espacio de trabajo.

#### 3.4.1.5. Monitoreo

Cada empresa tiene objetivos estratégicos. Y es en esta etapa del ciclo BPM que será posible descubrir si los procesos están alineados con estos objetivos, indicadores de seguimiento apropiados para la evaluación de los resultados obtenidos. Los indicadores más empleados implican generalmente 4 dimensiones: tiempo de duración del proceso, costo monetario del proceso, capacidad del proceso (cuanto produce el proceso efectivamente) y calidad, donde se analiza si hay muchos errores y variaciones que afectan una entrega satisfactoria a los clientes en el proceso.

#### 3.4.1.6. Optimización

Se busca el mejoramiento continuo del proceso, eliminando inconsistencias para asegurar que su ejecución corresponda con el diseño del mismo. En pocas palabras esta etapa pretende que la realidad del proceso sea lo más similar al diseño del mismo.

La manera de optimizar es tomando la información de la etapa de modelado y datos de desempeño de la etapa de monitoreo y se comparan, logrando identificar los problemas que puedan presentar los procesos y se proponen acciones correctivas. Dichas acciones se aplican en la etapa de descubrimiento.

En este TEG se hizo una adaptación del ciclo de vida BPM donde agrupamos las etapas y se convirtieron en Sprints. Por la naturaleza de los procesos de negocio y sus constantes cambios estas etapas volvían a cumplir ciclos por lo tanto se generaban se repetían Sprints. A continuación, en la **Figura 3.3** se muestra una representación de esta adaptación.

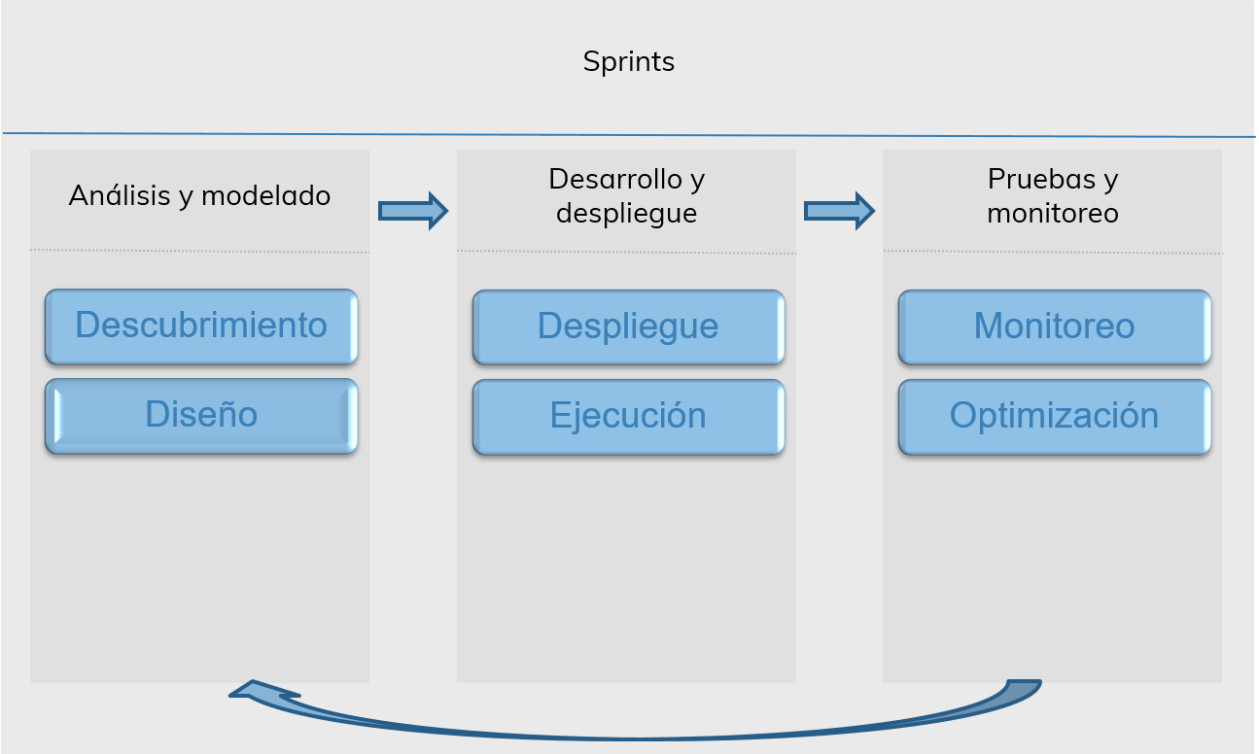


**Figura 3.3 Adaptación de metodologías.**

# CAPÍTULO 4
# MARCO APLICATIVO

En este capítulo se presenta la implementación de la propuesta de solución del TEG. Dicha propuesta se basa en la definición del modelo SOA sobre un BPMS utilizando como metodología de desarrollo una adaptación de Scrum.

## 4.1. Definición de Requerimientos

En el presente TEG, se desarrolla un modelo de Arquitectura de Servicios en conjunto de un sistema BPMS, esto haciendo uso los servicios web y el sistema de integración para satisfacer los requerimientos del proceso modelado "Consultar proyecto" que hace peticiones de datos a los componentes de correo, base de datos y gestor documental para así mostrar la información obtenida a través de un formulario.

El requerimiento principal es obtener los beneficios que nos ofrece SOA en un BPMS, eliminando el uso de conectores a componentes específicos sobre un proceso que normalmente los utilizaría para poder satisfacer las necesidades de una organización. En este trabajo se seleccionó el caso de Gestión de Proyectos por su complejidad en los BPMS y el uso de distintos componentes en sus procesos.

Se usa una adaptación de la metodología Scrum en conjunto de la metodología ciclo de vida BPM para el desarrollo de este TEG.

La tecnología BPM que se utiliza para desarrollar el BPMS fue la versión Comunnity de BonitaSoft, la cual permite el desarrollo y gestión del proceso de consultar proyecto permitiendo crear tareas para la solicitud, envió y recepción de datos.

El sistema que se utiliza para permitir la integración de servicios y así desarrollar un modelo de Arquitectura de Servicios fue la versión WSO EI la cual nos permitió el desarrollo e integración de servicios web para satisfaces los requerimientos del proceso consultar proyecto.

El Sistema Manejador de Base de Datos que se utiliza para alojar la información de un proyecto es PosgreSQL y su plataforma pgAdminn.

El sistema que se utiliza como Gestor Documental para el alojamiento de documentos pertenecientes a un proyecto es la versión Comunnity de Alfresco.

Para el envío a través de correos electrónicos de notificaciones acerca de la consulta de un proyecto se hizo uso de la API de Gmail.

## 4.2. Diseño Técnico

Existe varios BPMS que para satisfacer los requerimientos de sus procesos hacen solicitudes de datos a otros componentes para obtener la información que necesitan, sin embargo, esta comunicación de datos a través de conectores a componentes específicos se vuelve más compleja en BPMS de gran magnitud, ocasionando altos costos y mayores tiempos en alguna integración o mantenimiento de un componente.

Se decide modelar un proceso que sea común en la Gestión de Proyectos, que se utilice en los distintos procesos de la Gestión de Proyectos y que además sea complejo sus requerimientos con el uso de componentes externos. La consulta a un proyecto es un proceso fundamental en este caso ya que es utilizado cada vez que se necesite obtener información acerca de un proyecto en específico N cantidad de veces sobre N proyectos.

Tomando en consideración este proceso se decidió crear una Base de Datos (BD) que almacenara la información acerca de un Proyecto esta información abarca: id de proyecto, nombre, fecha de inicio del proyecto, cliente, descripción y fecha fin del proyecto. Se utiliza el Sistema Gestor de Base de Datos, PostgreSQL con su plataforma pgAdmin.

También se decide utilizar el Gestor Documental Alfresco ya que se vio la necesidad de almacenar documentos relacionados al proyecto que también puedan ser consultados como el diagrama de Gantt de un proyecto.

Para el envío de notificaciones al consultar un proyecto se utiliza el API de Gmail el cual permite el envío de correos electrónicos a través del consumo de un servicio web.

Se decide utilizar el sistema de integración WSO2 EI ya que es código libre, gratuito y además tiene una documentación completa acerca de los componentes que deben comunicarse con el BPMS, además tiene un componente llamado Bus de Servicio Empresarial que es el encargado de facilitar un modelo de Arquitectura de Servicios.

Realizando investigaciones se encuentra la manera de integrar las bases de datos relacionales y sus operaciones de lectura, escritura, inserción y actualización a través de Data Services, un componente de WSO2 EI que permite la integración de la BD de manera sencilla. También se encontró que Alfresco expone la gestión documental a través de servicios web los cuales fueron usados para la consulta de los documentos.

WSO2 EI permite exponer los servicios web como SOAP o como operaciones REST, por lo que se decidió utilizar servicios con operaciones REST ya que en Bonita BPM son más sencillas de consumir con la interfaz que nos proporciona el conector de REST.

A continuación, en la Figura 4.1 se muestra la arquitectura que se desarrolla en este TEG.

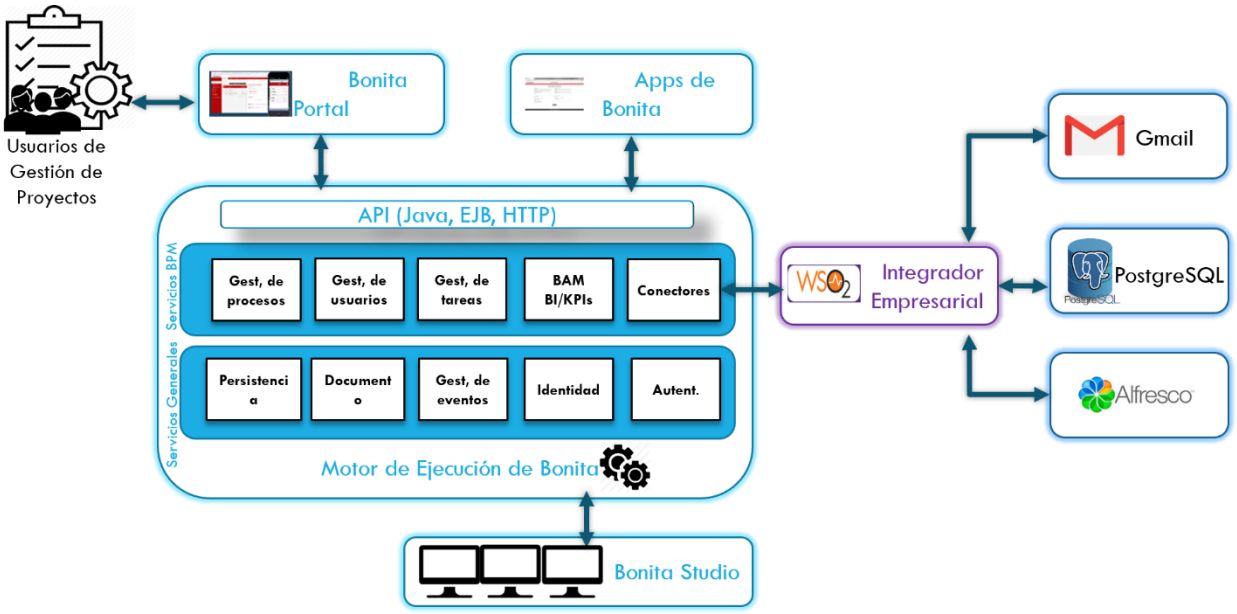


**Figura 4.1 Arquitectura desarrollada.**

## 4.3. Herramientas Tecnológicas

A continuación, se describen las herramientas seleccionadas para el desarrollo de esta esta arquitectura propuesta.

### 4.3.1. Bonita BPM

Bonita BPM es una herramienta que nos permite generar una solución basada en la gestión de procesos de negocio. (Ver más sobre Bonita en la Sección 2.4.3.1. ).

A continuación, se detallan los elementos de Bonita BPM que fueron utilizados en el modelado del proceso que se utiliza en este TEG:

- Contenedores y Compartimientos: representan a las entidades responsables de las actividades en un proceso. Los compartimientos son anidados a los contenedores para agregar divisiones. En la Figura 4.2 se muestra un ejemplo de este elemento.

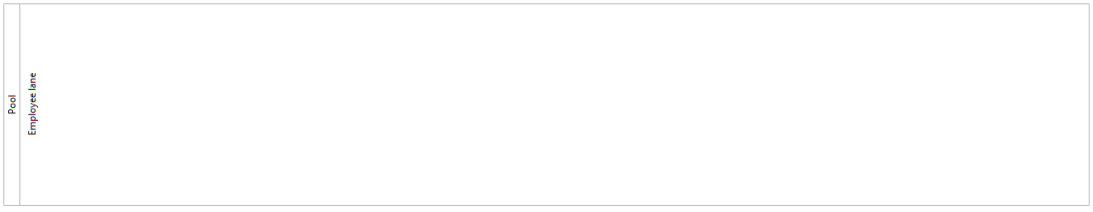

**Figura 4.2 Contenedores y Compartimientos, Bonita BPM.**

- Eventos simples de Inicio y Fin: Indican los puntos de inicio y fin de un proceso. En la Figura 4.3 se muestra un ejemplo de este elemento.


**Figura 4.3 Eventos Inicio y Fin, Bonita BPM.**

- Compuerta Exclusiva o XOR: representa una decisión donde se selecciona por cual flujo va a continuar entre las alternativas existentes a partir de la condición que se cumpla. En la Figura 4.4 se muestra un ejemplo de este elemento.


**Figura 4.4 Compuerta Exclusiva, Bonita BPM.**

- Tarea: es la unidad de trabajo o el trabajo a realizar. Esta tarea puede ser humana, donde se realiza un trabajo manual o de servicio, donde se realiza un trabajo automatizado. En la Figura 4.5 se muestra un ejemplo de este elemento.


**Figura 4.5 Tarea, Bonita BPM.**

- Actividad de llamada: es una referencia o llamada a un subproceso o tarea que se utiliza en el proceso actual. En la Figura 4.6 se muestra un ejemplo de este elemento.

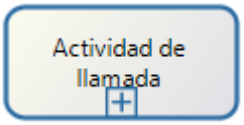

**Figura 4.6 Actividad de llamada, Bonita BPM.**

- Flujo de secuencia: define el orden de ejecución entre todas las actividades del proceso. En la Figura 4.7 se muestra un ejemplo de este elemento.

**Figura 4.7 Flujo de secuencia, Bonita BPM.**

### 4.3.2. PostgreSQL

Según (PostgreSQL Global Development Group, 2019), PostgreSQL es un SMBD relacional potente de código abierto que utiliza y amplia el lenguaje SQL combinado con muchas características que almacenan de forma segura y escalable los datos más complicada. (Ver más sobre Bonita en la Sección 2.2.3.1. ).

### 4.3.3. Alfresco

Según (Alfresco Software, Inc., 2019), Alfresco es un software de código abierto que ofrece a sus usuarios un modo de trabajo sencillo y eficaz, con la capacidad de acceder a la información que necesitan en el momento en que la necesitan. (Ver más sobre Bonita en la Sección 2.8.1. )

### 4.3.4. WSO2 EI

Según (WSO Inc., 2018) WSO2 Enterprise Integrator es una solución de integración comprensiva que permite la comunicación entre varias aplicaciones dispares. WSO2 EI nos permitirá exponer como servicios todo acceso de datos de un componente externo, permitiendo así desarrollar un modelo SOA.

## 4.4. Etapas del proyecto

En este TEG se hace una adaptación de la metodología SCRUM donde cada etapa del proyecto o fase de desarrollo es un ciclo o Sprint, además los tiempos de cada Sprint fueron

prolongados debido al ciclo de vida BPM y el manejo de procesos. Se fueron agregando ítems o requerimientos que surgieron en el Backlog, cada 2 semanas se hace una reunión para discutir los avances, las complicaciones y mostrar el desarrollo hasta ese punto de forma remota. En estas reuniones si el ítem o requerimiento ya estaba listo se discutía el siguiente requerimiento y se iniciaba un nuevo Sprint siendo una adaptación de un Sprint Planning, Review y retrospectiva. En caso de haber dudas o complicaciones se consultaba de forma remota convirtiéndose en una especie de meeting o Daily. Con respecto a los roles de Scrum se Team era conformado por 2 personas, mi persona y el tutor Franky Uzcátegui, el tutor era el dueño del proceso o Product Owner y mi persona el Scrum Master.

A continuación, se explica cada fase de desarrollo o Sprint realizados en este TEG:

### 4.4.1. Análisis y modelado

En esta fase se analiza la arquitectura de la herramienta WSO2 EI para entender cómo funciona, En la Figura 4.8 podemos observar esta arquitectura.

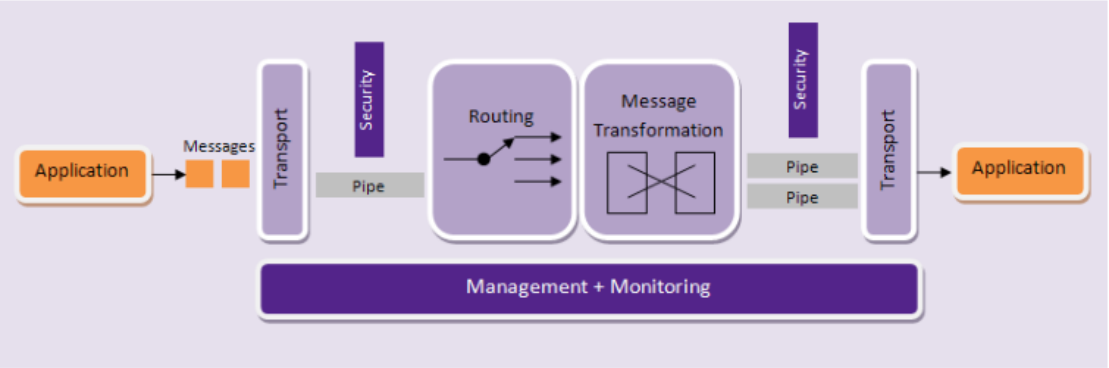


**Figura 4.8 Arquitectura de flujo de mensaje WSO2 EI.**

Fuente: https://docs.wso2.com/display/EI640/Overview.

Esta arquitectura representa el flujo de mensaje que ocurre cuando un componente o consumidor hace una solicitud de datos.

- Este mensaje es recibido por el perfil ESB de WSO2 EI, inicialmente por el transporte adecuado, es decir, aquel que fue configurado para ese tipo de mensajes y aplica los protocolos necesarios.

- Luego el transporte envía el mensaje a través de un pipe o canal el cual maneja calidad de servicio.

- Después el mensaje llega al framework mediador de WSO EI el cual es el encargado de realizar el enrutamiento y transformación de forma asíncrona.

- El mensaje es asignado a uno o más pipes de ser necesario, se utilizan los protocolos de transporte adecuados y se envía el mensaje transformado al proveedor del servicio.

- Este flujo de mensaje trabaja de forma inversa al recibir respuesta de los proveedores del servicio.

A partir de este flujo se procede a modelar un proceso receptor en Bonita BPM encargado del envió o la solicitud de datos para los procesos de la gestión de proyectos y la respuesta o recepción de datos generada por wso2 EI esto con el fin de que se maneje un solo proceso que haga el consumo de servicios. A continuación, en la Figura 4.9 se muestra el modelado del proceso.

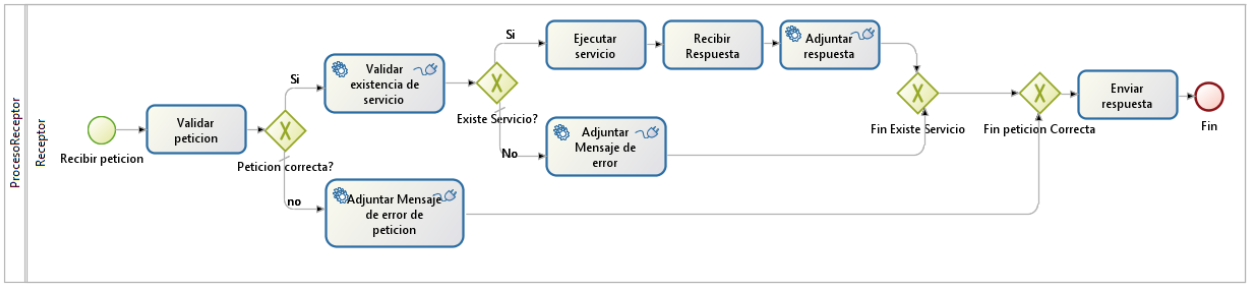


**Figura 4.9 Proceso Receptor en Bonita BPM.**

El proceso se inicia cuando es llamado por algún proceso de la gestión de proyectos, luego se valida la petición recibida verificando que los parámetros que se necesiten para el consumo del servicio sean los correctos, si son incorrectos enviara un mensaje de error. Si son correctos entonces se valida la existencia del servicio, si el servicio no existe se enviará un mensaje de error en caso contrario se ejecutará el servicio, luego se recibirá una respuesta y esta respuesta será adjuntada al mensaje que se enviará al proceso que lo llamo o ejecuto.

En este trabajo se selecciona uno proceso común que es utilizado por los procesos de la gestión de proyectos el cual es llamado Consultar proyecto, este fue modelado en bonita BPM. A continuación, en la Figura 4.10 se muestra el proceso.

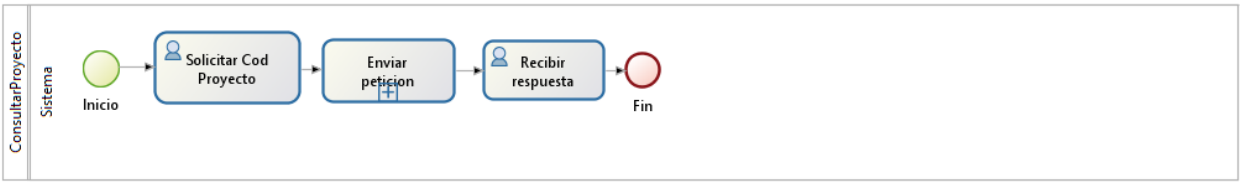


**Figura 4.10 proceso Consultar Proyecto en Bonita BPM.**

Este proceso solicita un código de proyecto el cual se decidió con el tutor que es el identificador único de cada proyecto, en base a este dato de entrada se envia la petición al proceso receptor y luego se recibirá una respuesta de este proceso.

Luego se procedió a utilizar PostgreSQL en conjunto de pgAdmin para crear una base de datos que almacenara datos acerca de un proyecto. En la Figura 4.11 se muestra la base de datos creada en pgAdmin llamada BD_GESTOR_DE_PROYECTOS.

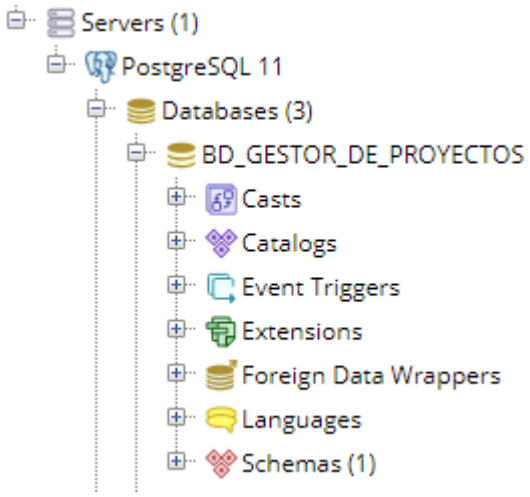


**Figura 4.11 Visualización de base de datos en PostgreSQL.**

En esta base de datos se crea una tabla denominada Proyecto el cual se decidió crear 6 atributos. En la Figura 4.12 se muestra la tabla y sus atributos.

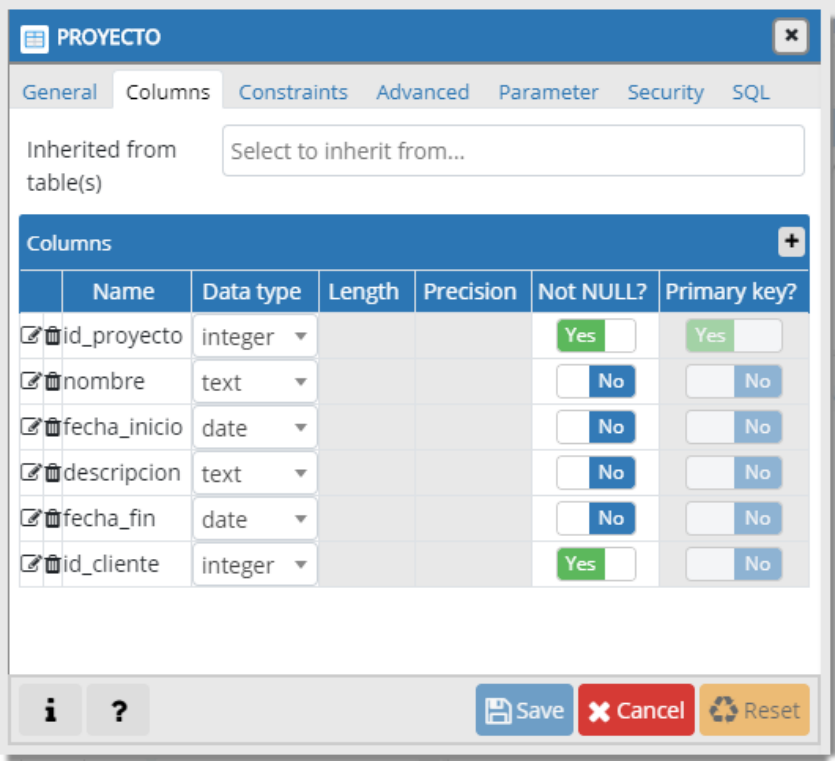


**Figura 4.12 Visualización de la tabla Proyecto en PostgreSQL.**

El id de proyecto es nuestro identificador de cada proyecto y es el dato utilizado para poder traer los datos asociados. Un proyecto tiene asociado un nombre, una fecha de inicio y una fecha fin, un cliente asociado a ese proyecto y una breve descripción.

Esta tabla se llena con datos de prueba para poder hacer pruebas al utilizar el componente de PostgreSQL.

También se creó la tabla Cliente con datos de prueba para relacionar un cliente al campo id_cliente. A continuación, en la Figura 4.13 se observa en detalle los campos de la tabla cliente.

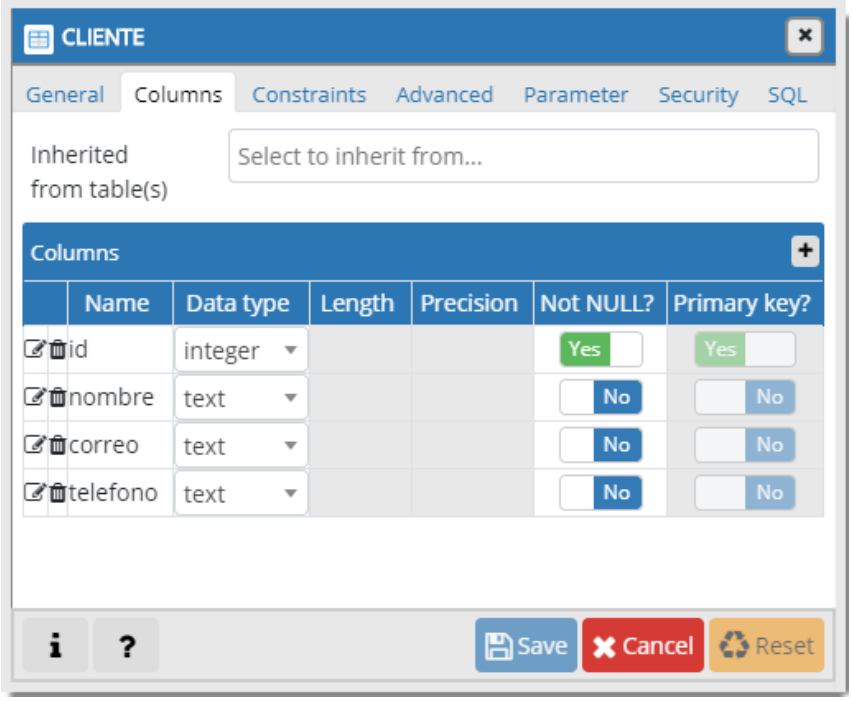


**Figura 4.13 Visualización de la tabla Cliente en PostgreSQL.**

Finalmente se define cuáles son los componentes que se usan en el proceso de consultar proyecto y que se comunican con bonita BPM para tener un modelo SOA, traer los datos de PostgreSQL hace uso de un componente de base de datos, las notificaciones por parte del proceso receptor se enviaran a través de un correo electrónico haciendo uso de un componente SMTP y se plantea la posibilidad de traer documentos relacionados a un proyecto a través de Alfresco. Estos documentos son los diagramas de Gantt de cada proyecto y en formato pdf.

### 4.4.2. Desarrollo y despliegue de procesos ejecutables

En esta fase se logró el desarrollo de los procesos modelados para que fueran ejecutables y poder enviar datos y recibir una respuesta haciendo uso de servicios web.

#### 4.4.2.1. Configuración de la herramienta WSO2 EI

Antes de utilizar este sistema para la integración y creación de los servicios web que deben ser creados para satisfacer los requerimientos de los procesos se hizo una instalación básica. En la Figura 4.14 se muestra la interfaz de instalación de WSO2 EI.

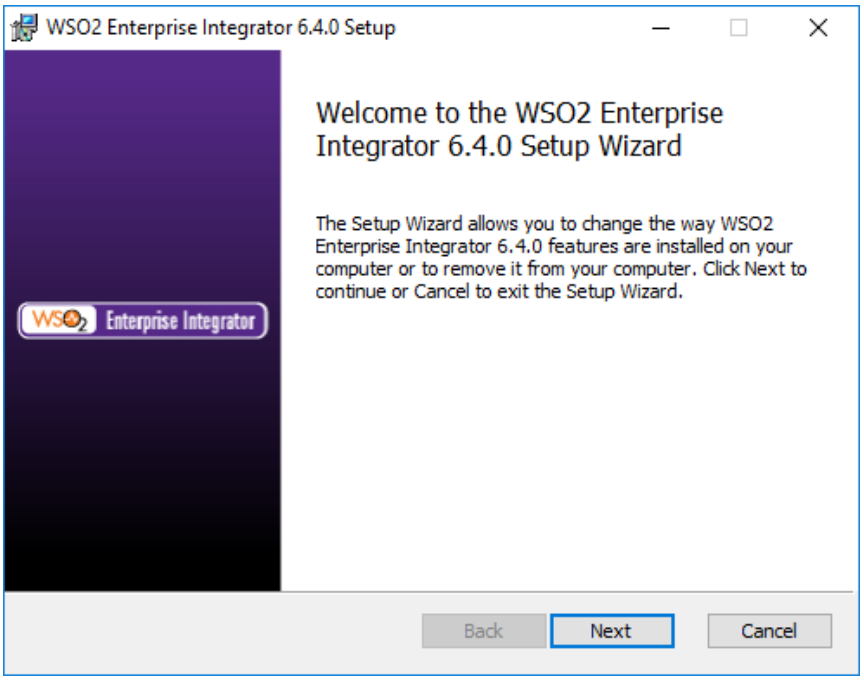


**Figura 4.14 Interfaz de instalación de WSO2 EI.**

Esta instalación es intuitiva y no necesita mayor detalle, una duda que surge durante la investigación de la herramienta fue el almacenamiento de información acerca de los servicios que sean integrado y creados en la herramienta. WSO2 EI almacena la mayoría de su información en archivos de configuración en su carpeta.

Una vez instalada la herramienta la ejecutamos y levanta el servidor, mostrándonos la dirección y puerto al cual debemos acceder en nuestro navegador, una vez nos identificamos en el login entonces llegaremos a la interfaz de la Figura 4.15.

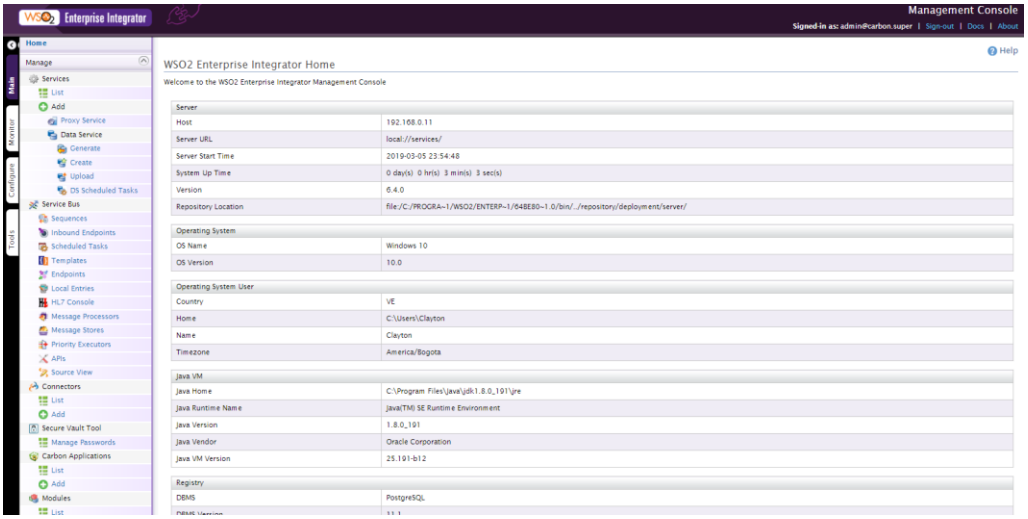

**Figura 4.15 Interfaz principal de WSO2 EI.**

Existen muchas funcionalidades mostradas en la barra de opciones, en este TEG mostraremos detalle de aquellas que fueron utilizadas, el resto de ellas se encuentra en la documentación de WSO EI en su página principal.

### 4.4.2.2. Creación de servicios para PostgreSQL

Para la creación de servicios de consulta en PostgreSQL se selecciona la opción de configuración que se muestra en la Figura 4.16.

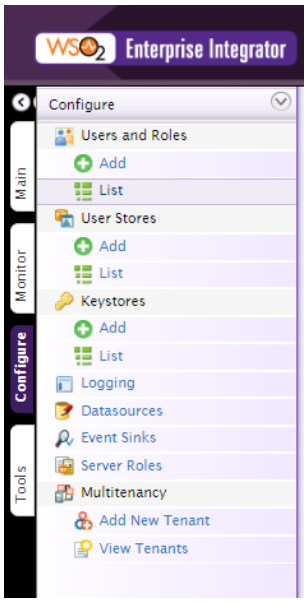


**Figura 4.16 Selección de pestaña de configuración en WSO2 EI.**

Como se muestra en la Figura 4.17, se seleccionó la opción de Datasources y nos muestra una vista de todas las conexiones a las bases de datos, le damos la opción de Add Datasource para poder agregar nuestra conexión a la base de datos.

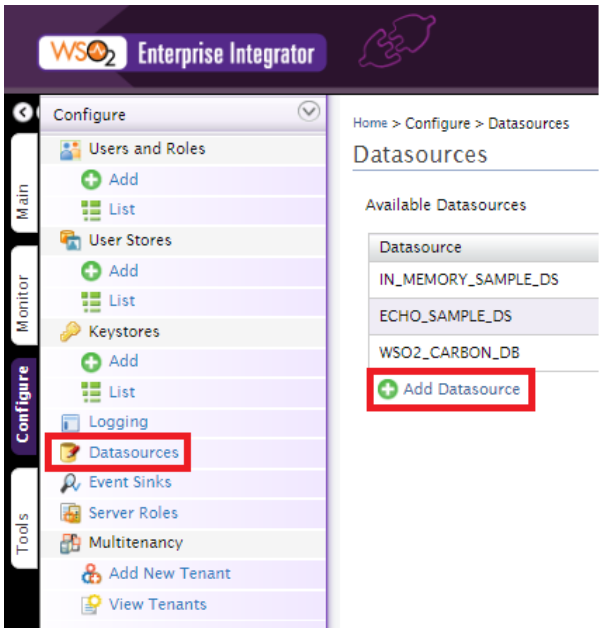


**Figura 4.17 Selección de la opción Datasources en WSO2 EI.**

En la Figura 4.18 se muestra una vista donde tendremos que completar con los datos de configuración de nuestra base de datos.

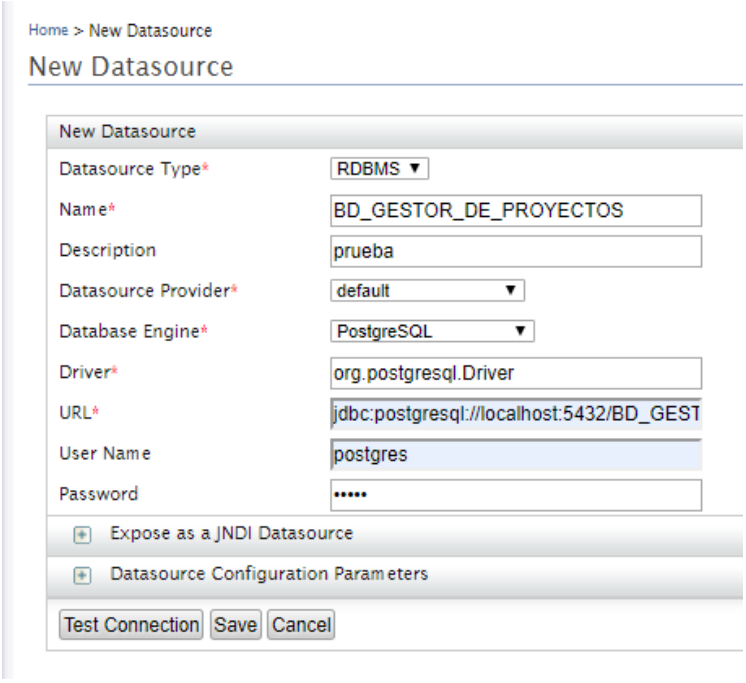


**Figura 4.18 Vista de configuración para conexión a PostgreSQL en WSO2 EI.**

En nuestro caso nuestro tipo de Datasource es un RDBMS o Sistema Manejador de Base de Datos Relacional, colocamos el nombre de la BD creada para almacenar nuestros proyectos, el proveedor será por defecto y motor de BD seleccionamos PostgreSQL. El driver se coloca el indicado en la Figura 4.18 si el url dependerá del puerto en que esta levantado el SMBD y el nombre de la BD, finalmente colocamos el nombre y contraseña del usuario que tenga los permisos para acceder y modificar los datos. Podemos hacer uso de la opción Test Connection para verificar que introducimos los datos correctamente.

Luego como se indica en la Figura 4.19 nos dirigimos a la pestaña principal o Main para seleccionar la opción Generate ubicada en la sección de Data Service.

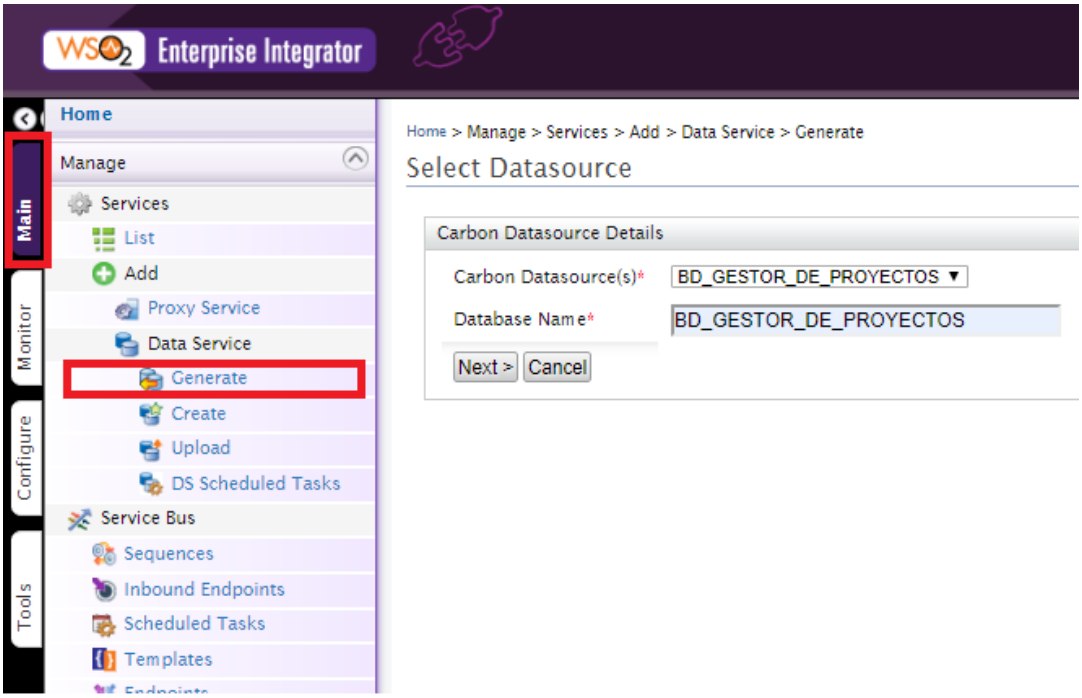


**Figura 4.19 Vista de generación de Data Service en WSO EI.**

Seleccionamos el Datasource creado anteriormente y colocamos el nombre la BD de los proyectos, hacemos clic en Next y colocamos el nombre del esquema como se indica en la Figura 4.20.

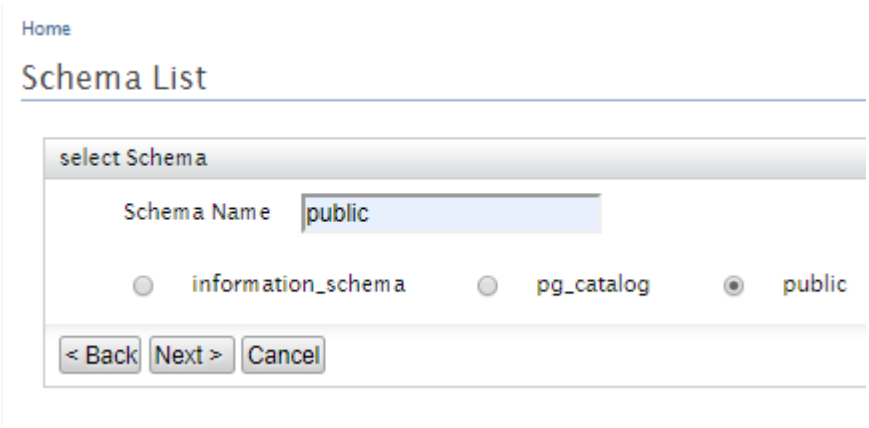


**Figura 4.20 Selección de Esquema en creación de Data Service.**

Continuamos y seleccionamos la tabla que queremos consultar para generar los servicios como se indica en la Figura 4.21.

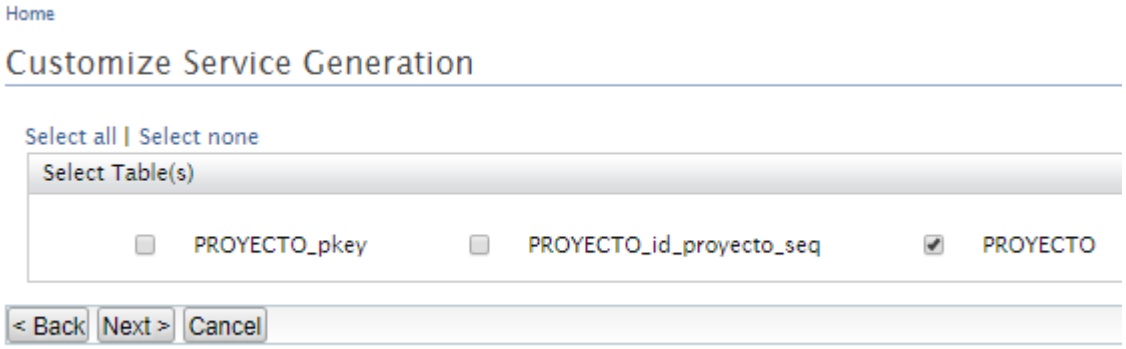


**Figura 4.21 Selección de tablas en creación de Data Service.**

Continuamos y seleccionamos la opción de múltiples servicios y colocamos el Namespace de nuestra BD como se indica en la Figura 4.22.

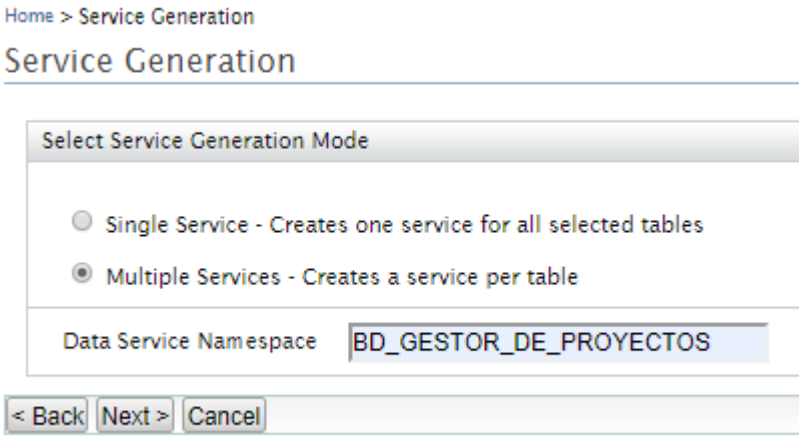


**Figura 4.22 Selección de modo de generación de Data Service.**

Finalmente continuamos y se crea nuestro Data Service como se muestra en la Figura 4.23.


**Figura 4.23 Creación de Data Service finalizada.**

Ahora nos dirigimos en la opción de lista en la categoría de servicios como se indica en la Figura 4.24 y podremos observar nuestro Data Service creado.

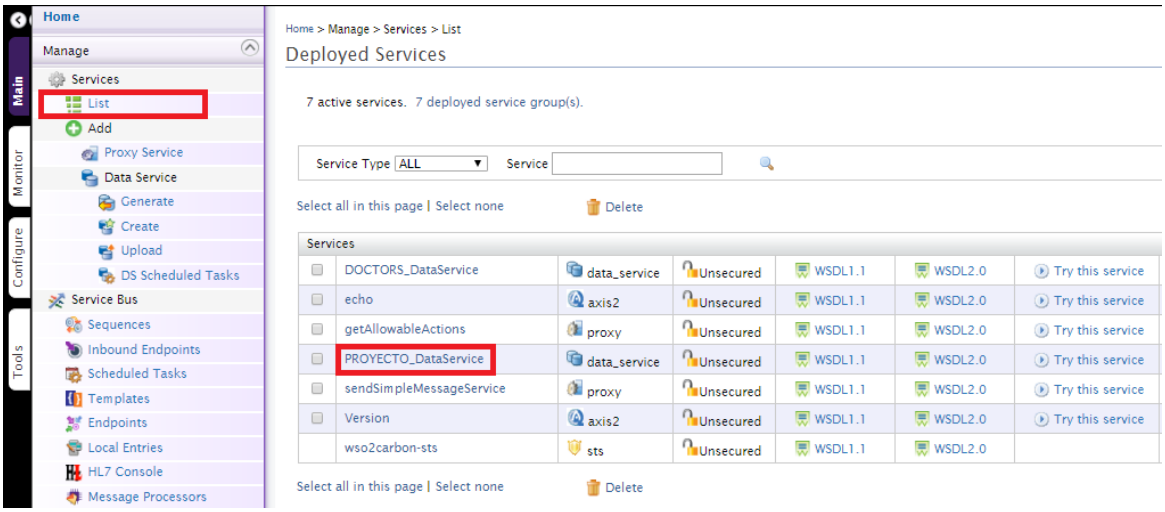


**Figura 4.24 Lista de servicios en WSO2 EI.**

Si seleccionamos el servicio nos llevara a la vista que se muestra en la Figura 4.25, donde podemos observar características de nuestro servicio creado, así como las estadísticas nos servirán en la fase de monitoreo para observar más detalles.

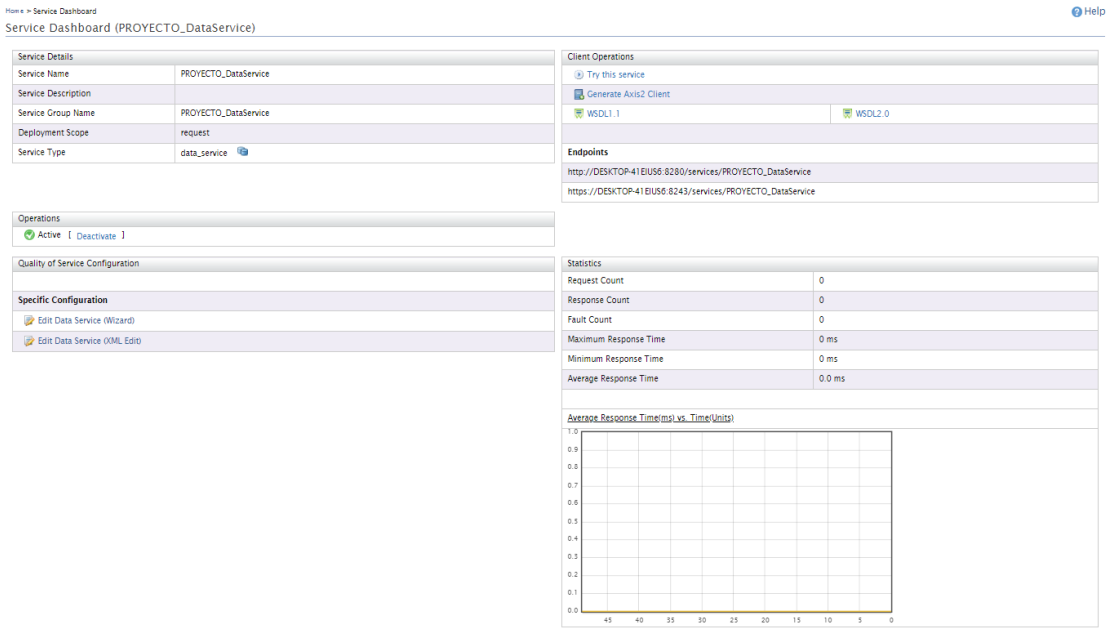


**Figura 4.25 Dashboard de un servicio en WSO2 EI.**

Si seleccionamos la opción de editar servicio en formato XML podremos observar el XML del servicio creado con todas sus operaciones como se muestra en la Figura 4.26. Esto permite editar el servicio vía XML.

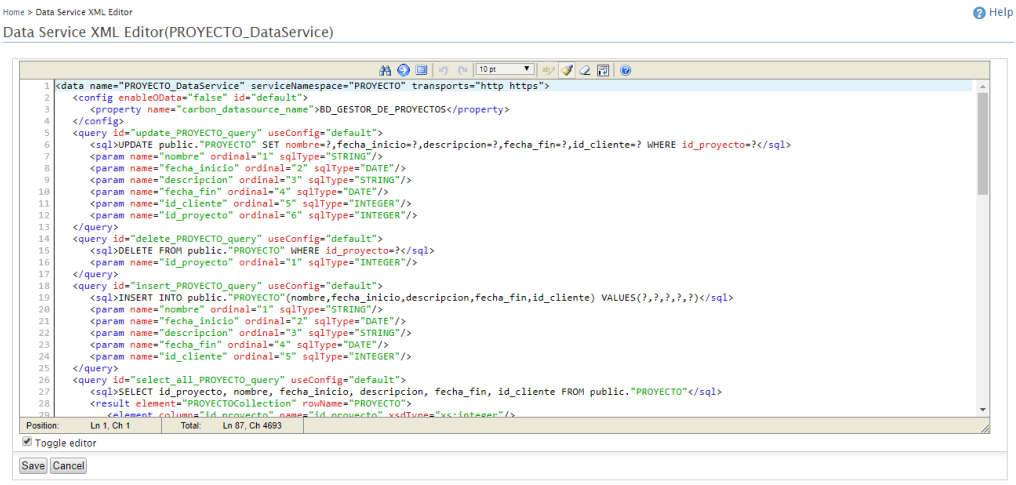


**Figura 4.26 vista en XML de un servicio en WSO2 EI.**

Si seleccionamos la opción de editar servicio por configuración tendremos una interfaz para cada parte del desarrollo de los servicios como se muestra en la Figura 4.27.

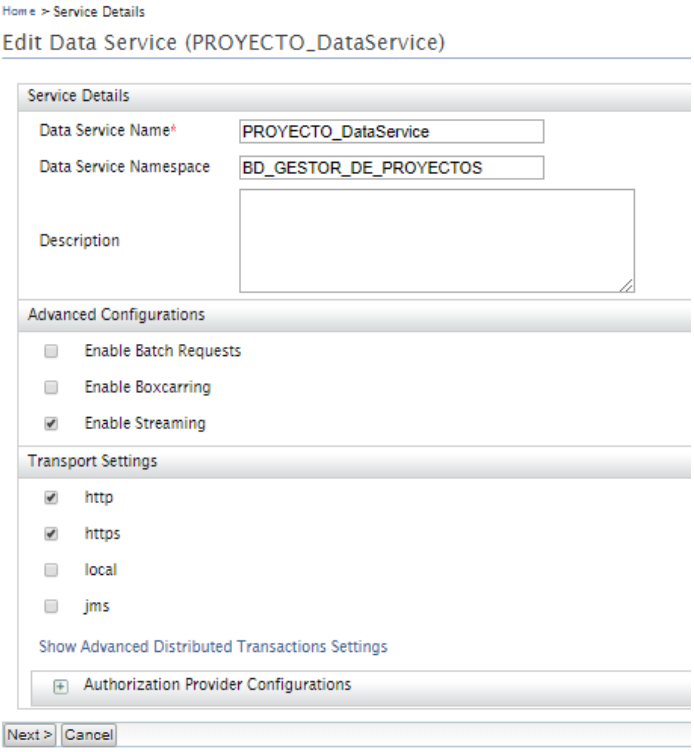


**Figura 4.27 Vista de edición del servicio por interfaz en WSO EI.**

Si le damos continuar hasta la sección de queries podremos observar todos los queries que se crearon al crear el Data Service, seleccionamos editar query en el servicio de consulta a un solo proyecto llamado select_with_key_Proyecto como se indica en la Figura 4.28.

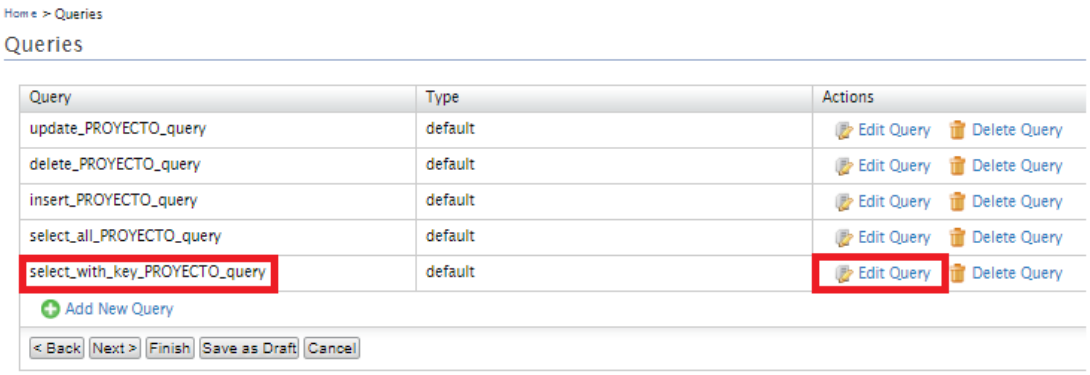


**Figura 4.28 Selección en editar query en WSO2 EI.**

En la vista que se muestra en la Figura 4.29 podemos observar el query generado automáticamente, así como asignación de los campos.

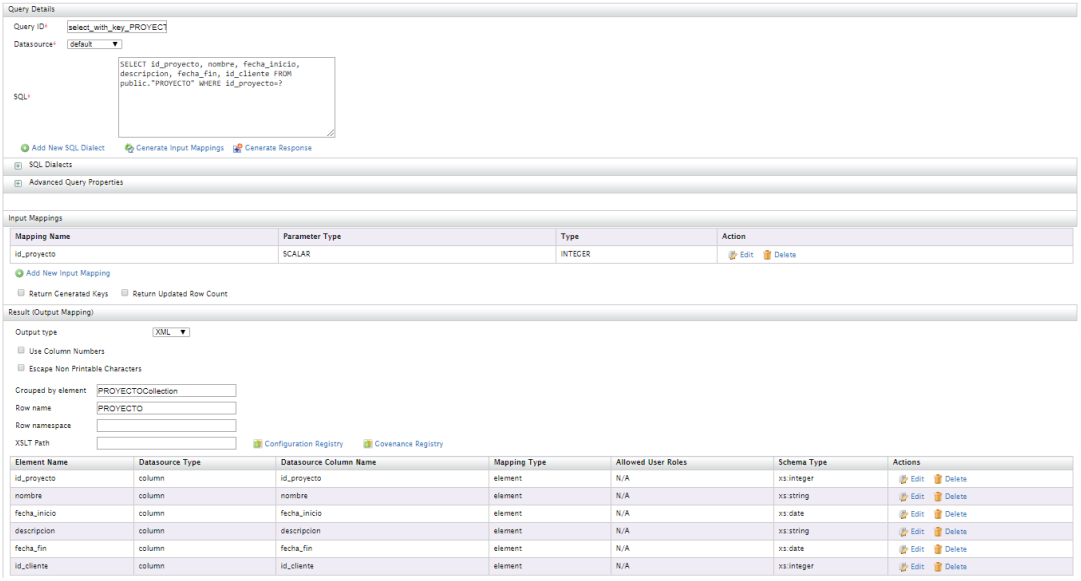


**Figura 4.29 Vista de edición de query en WSO EI.**

La siguiente sección es la de operaciones donde se definen las mismas, indicando los parámetros de entrada como se indica en la Figura 4.30. Estas también fueron generadas automáticamente en la creación del Data Service.

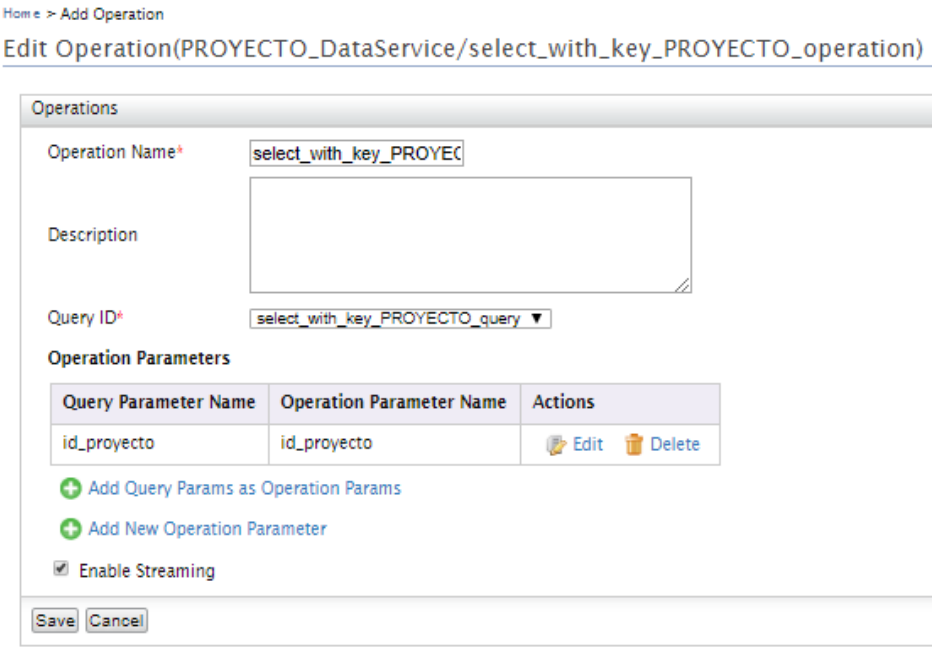


**Figura 4.30 Vista de edición de operación en WSO EI.**

Por último, tenemos la sección de Recursos como se muestra en la Figura 4.31, esta no es generada automáticamente por el Data Service. Con WSO2 EI la creación de un Data Service y cualquier servicio se crea para ser consumido como un servicio SOAP sin embargo en esta sección podemos crear un recurso que nos permite acceder a este servicio como un servicio RESTful API permitiendo que se pueda consumir a través de un URI o pasándole un formato JSON de entrada. Seleccionamos agregar nuevo recurso como se indica.

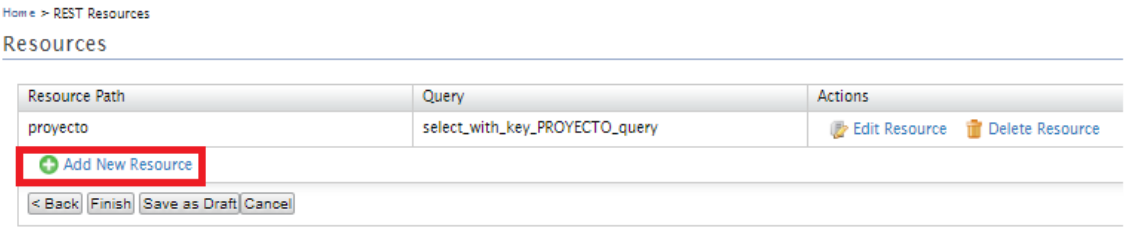


**Figura 4.31 Vista de sección de recursos en WSO2 EI.**

Como se muestra en la Figura 4.32 debemos colocar el nombre del path del recurso, seleccionamos el método del recurso que para la consulta del proyecto sería una operación GET, seleccionamos el query que consulta un solo proyecto y la vista mostrara los parámetros que se deben recibir al usar este recurso, guardamos y finalizamos la edición del servicio.

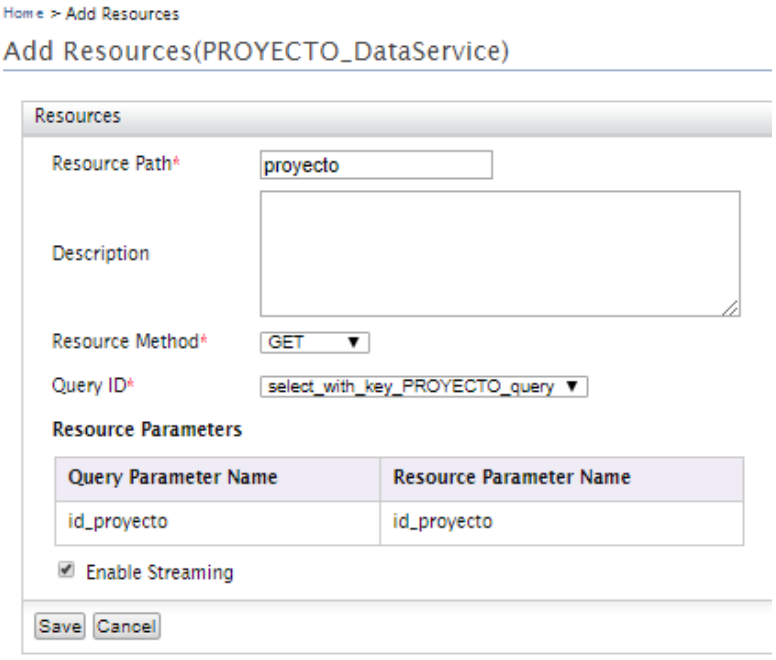


**Figura 4.32 Vista de agregar recursos en WSO EI.**

Para probar este servicio se hizo uso de la dirección o endpoint creado por WSO2 EI para este servicio, esta es la url final con el cual podemos acceder al servicio y consumirlo. Podemos encontrar el endpoint en el dashboard del servicio específicamente en la sección de operación de clientes como se indica en la Figura 4.33.

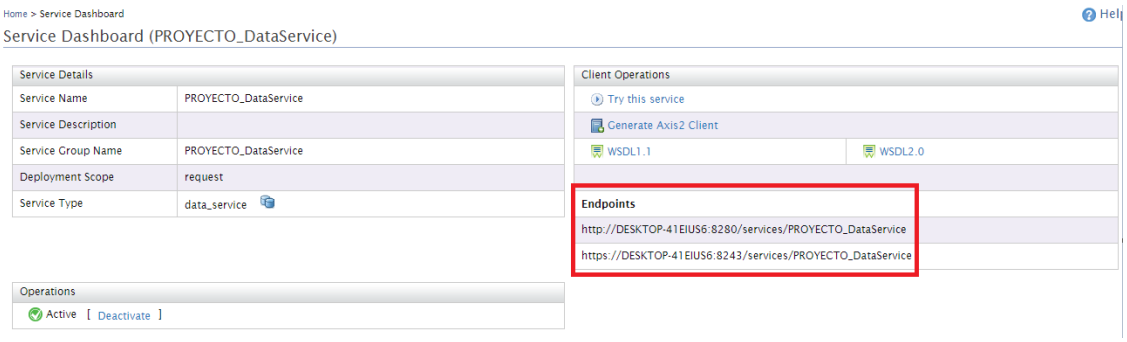


**Figura 4.33 Ubicación de endpoints de un servicio en WSO2 EI.**

Colocamos en el navegador esta url concatenando el llamado al recurso y el parámetro necesario para obtener la respuesta que se visualiza en la Figura 4.34 en formato XML.


**Figura 4.34 Resultado de consumo de servicio consultar proyecto vía browser.**

Para los servicios relacionados con el gestor documental Alfresco se consiguió en la documentación la librería de servicios que pueden ser utilizados, para poder acceder a través de otra aplicación a los servicios de gestión documental primero debemos consumir un servicio de autenticación que nos retornara una llave de acceso, luego con esta llave de acceso podemos consumir el servicio de descargar un documento en específico haciendo uso del api para visualización de archivos de Alfresco. A continuación, se muestran un ejemplo al llamado de estos los servicios.

```
1. http://localhost:8180/alfresco/s/api/login?u=admin&pw=admin
```

Donde el usuario y contraseña es el utilizado para autenticarse en Alfresco.

```
1. http://localhost:8180/alfresco/api/-default-
   /public/cmis/versions/1.1/browser/root/Shared/proyectos/gantt_"+codigo_p
   royecto+".pdf?alf_ticket="+keyAlfresco
```

Donde codigo_proyecto es el proyecto a consultar y keyAlfresco es el resultado anterior.

#### 4.4.2.3. Creación de servicio para envío de correos electrónicos.

Para el desarrollo de este servicio se utilizó el API de Gmail, WSO2 EI posee un repositorio de conectores en su página llamado WSO2 Store que podemos utilizar para facilitar la integración de un componente nuevo en nuestro sistema. En la Figura 4.35 se puede observar varios conectores entre ellos el conector de Gmail utilizado en este TEG.

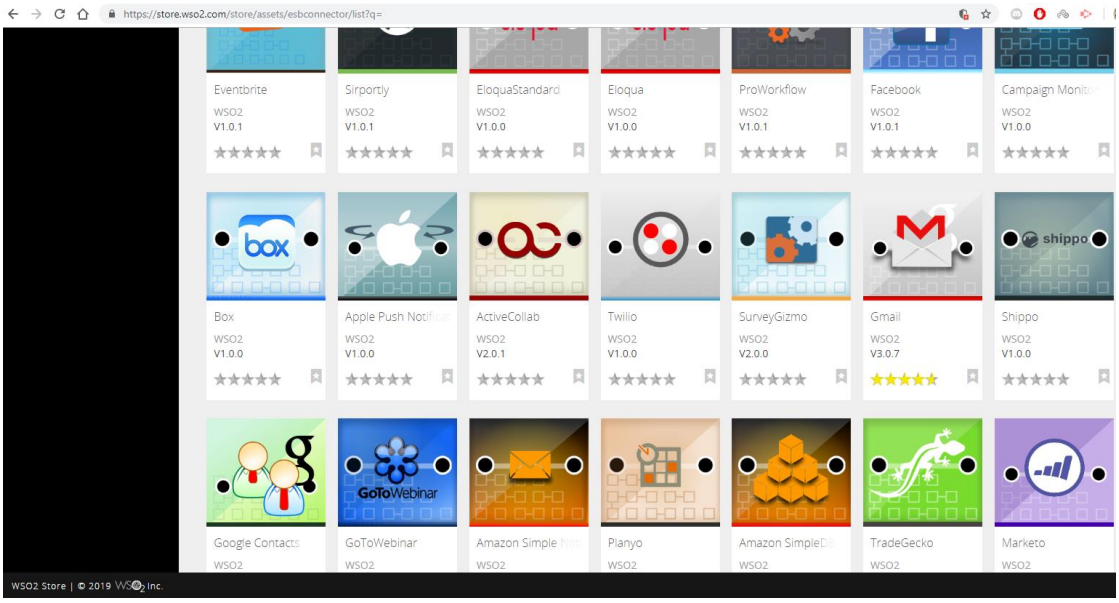


**Figura 4.35 Vista de WSO2 Store y sus conectores.**

Se descarga el conector y se agregó a WSO2 EI seleccionando Add en la sección de Conectores en la pestaña principal como se muestra en la Figura 4.36. Luego seleccionamos el archivo que contiene el conector el conector seleccionamos upload y se agregara a nuestra lista de conectores.

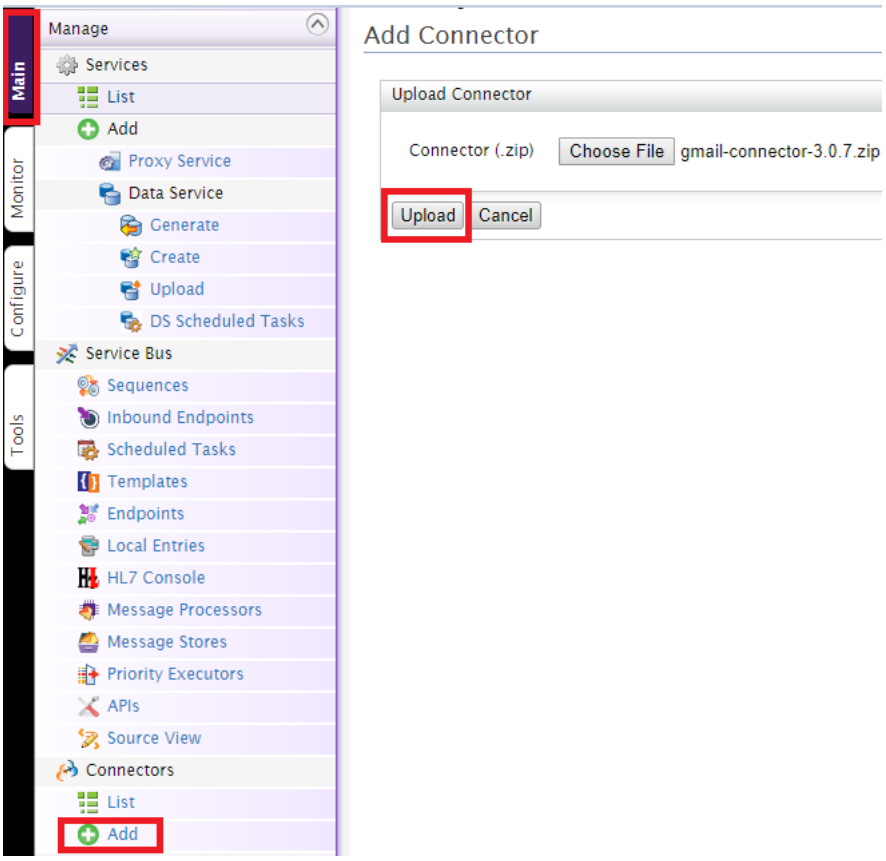


**Figura 4.36 Vista para agregar un conector en WSO2 EI.**

Luego cambiamos el estatus de nuestro conector seleccionando *enable* en la lista de conectores como se muestra en Figura 4.37.

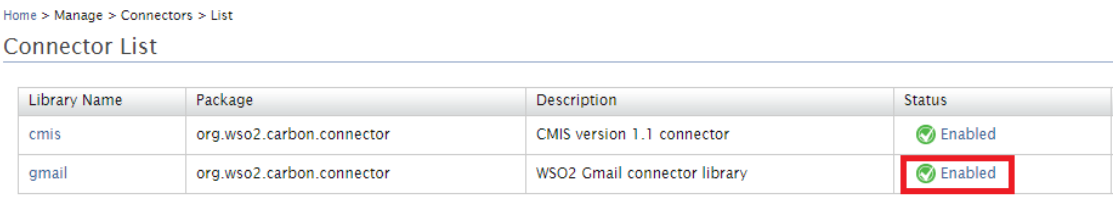


**Figura 4.37 Lista de conectores en WSO2 EI.**

Si seleccionamos el conector podemos observar las operaciones que pueden ser utilizadas y configuradas como se muestra en la Figura 4.38. Para este TEG se utilizó la operación de envío de correos simple.

| gmail | |
|---|---|
| Operations | Description |
| deleteMessages | delete the message |
| listLabels | List all labels |
| deleteLabel | Delete labels |
| sendMail | Send a plain mail |
| trashThreads | Send threads to trash |
| getUserProfile | Get the user profile |
| readDraft | read a particular draft |
| passwordLogin | Login to the Gmail account using username and password |
| deleteDraft | delete an existing draft |
| createDraft | create a plain draft |
| getAccessTokenFromRefreshToken | get the access token from refresh token |

**Figura 4.38 Algunas operaciones del conector de Gmail en WSO2 EI.**

Para configurar las operaciones de un conector se debe crear un Servicio Proxy que nos permitirá configurar los parámetros de una operación, así como los parámetros iniciales para poder hacer uso del conector. Para crear un Servicio Proxy se selecciona Proxy Service en la pestaña principal y se selecciona en este caso Custom Proxy como se muestra en la Figura 4.39.

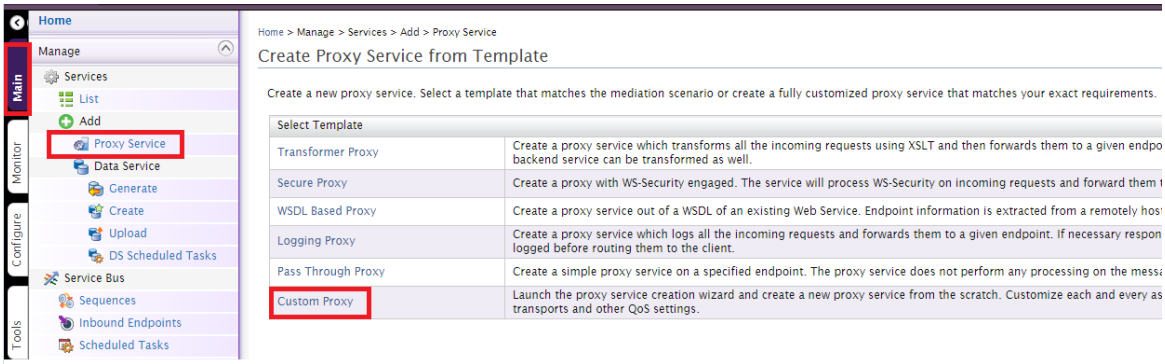


**Figura 4.39 Creación de servicio proxy en WSO2 EI.**

Luego seleccionamos la opción para cambiar la vista en XML como se muestra en la Figura 4.40.

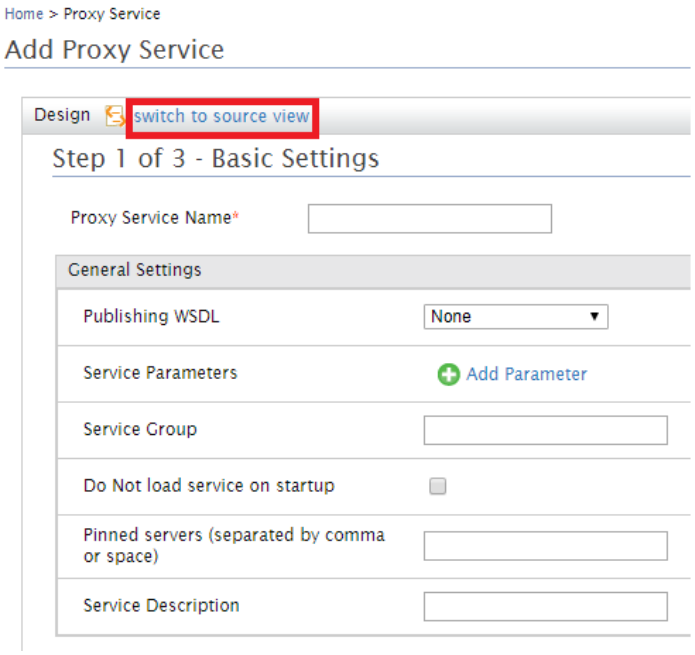


**Figura 4.40 Selección de cambio de vista a XML en Servicios Proxy.**

Luego se hizo una adaptación del ejemplo que se encuentra en la guía de uso de este conector, quedando como resultado final la Figura 4.41.

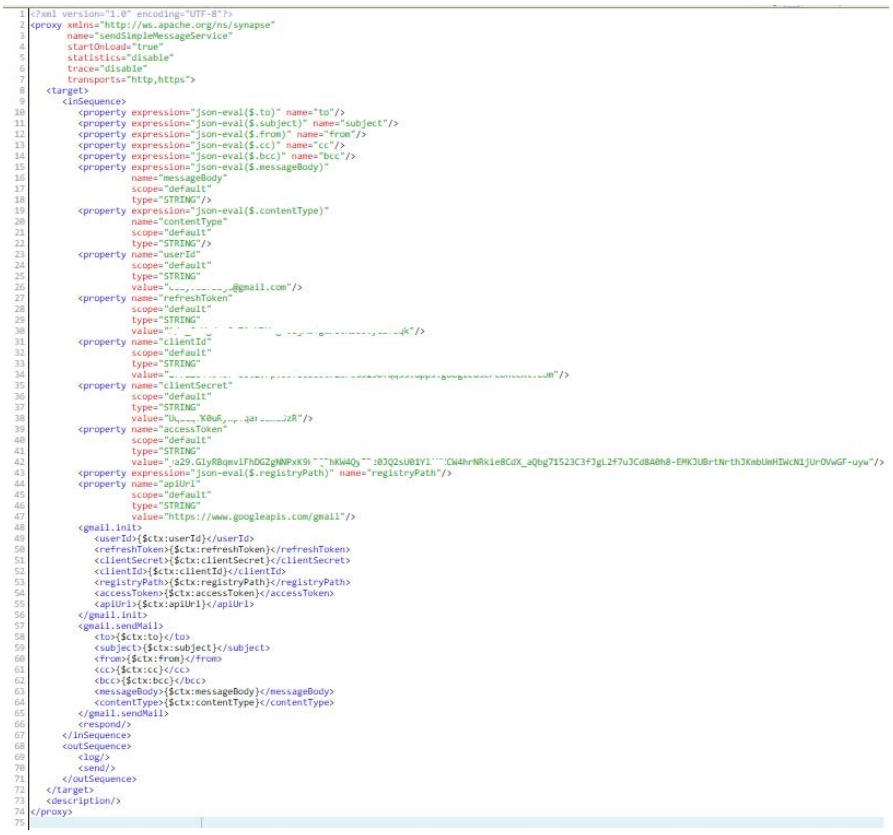

**Figura 4.41 Vista general del Servicio Proxy para Gmail en WSO2 EI.**

A continuación, se muestran Los parámetros que necesita el api de Gmail en la Figura 4.42.

```
<gmail.init>
   <userId>{$ctx:userId}</userId>
   <refreshToken>{$ctx:refreshToken}</refreshToken>
   <clientSecret>{$ctx:clientSecret}</clientSecret>
   <clientId>{$ctx:clientId}</clientId>
   <registryPath>{$ctx:registryPath}</registryPath>
   <accessToken>{$ctx:accessToken}</accessToken>
   <apiUrl>{$ctx:apiUrl}</apiUrl>
</gmail.init>
<gmail.sendMail>
   <to>{$ctx:to}</to>
   <subject>{$ctx:subject}</subject>
   <from>{$ctx:from}</from>
   <cc>{$ctx:cc}</cc>
   <bcc>{$ctx:bcc}</bcc>
   <messageBody>{$ctx:messageBody}</messageBody>
   <contentType>{$ctx:contentType}</contentType>
</gmail.sendMail>
```

**Figura 4.42 Parámetros necesarios para el uso del API de Gmail en WSO2 EI.**

Cada parámetro debe ser llenado para que se pueda utilizar este conector.

- userID: el email a ser usado para el envío de correos.
- refreshToken: el valor del Refresh Token que se utiliza para generar un nuevo Access Token cuando expira.
- clientSecret: el valor del Client Secret que se obtiene cuando registras tu aplicación con el API de Gmail.
- clientID: el valor del Client ID que obtienes al registrar tu aplicación con el API de Gmail.
- registryPath: el path donde el Access Token es guardado, se dejará vacío por default y el conector lo guardará en un path por default.
- accessToken: valor del Access Token para acceder al API de Gmail.
- apiUrl: el url del API de Gmail (https://www.googleapis.com/gmail).

Todos estos datos se consiguen creando las credenciales y asociando el correo que se vaya a usar en el API de Gmail, para mayor detalle consultar la guía de configuración del conector. (WSO2 Inc., 2019).

Los otros parámetros son pertenecientes a la operación y son aquellos datos básicos que usaremos para los envíos de correo, WSO2 EI permite el uso de constantes en la definición de parámetros para que no tengamos que enviar una y otra vez nuestro JSON o payload (archivo de solicitud de datos que es enviado al servicio) los mismos datos en parámetros que no suelen cambiar. Para ello seleccionamos el modo de vista normal y nos vamos a la sección 2 de In Sequence como indica la Figura 4.43, luego seleccionamos la opción editar.

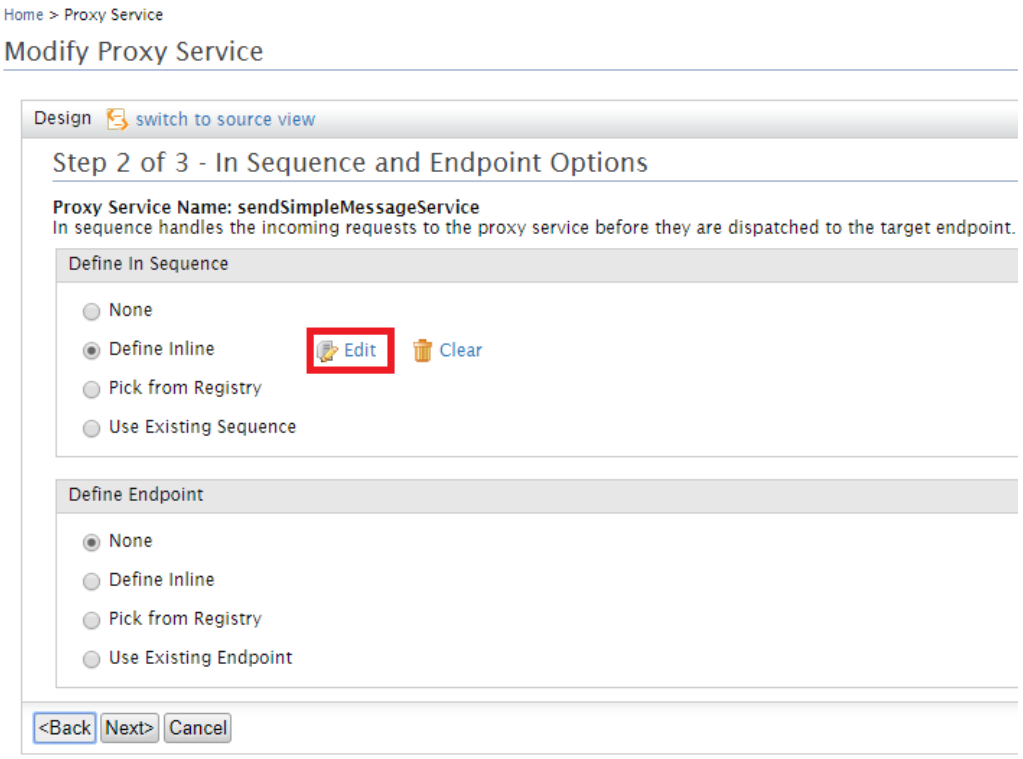


**Figura 4.43 Seleccionar editar In Sequence en WSO2 EI.**

Luego seleccionamos la propiedad que queremos como constante, elegimos el check de valor y en el campo de valor colocamos la constante, finalmente actualizamos como se indica en la Figura 4.44.

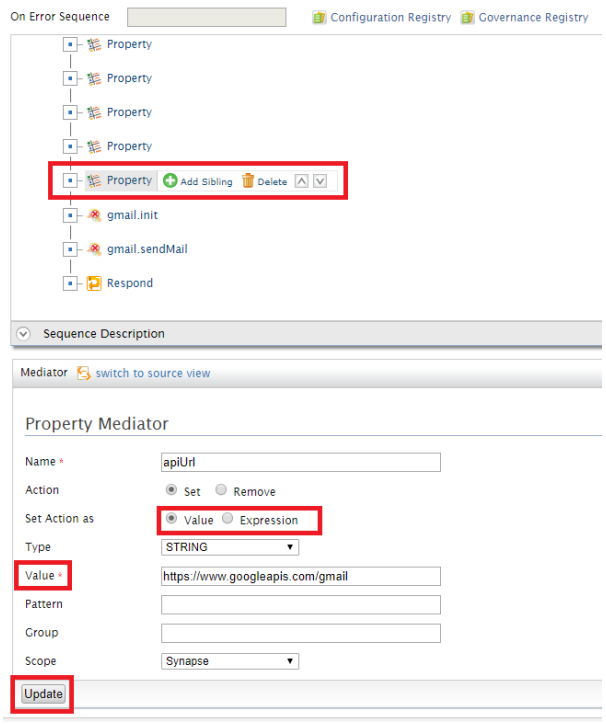


**Figura 4.44 Colocar constantes en los parámetros de Servicios Proxy.**

Las pruebas para este servicio se hicieron al momento de integrar los servicios al proceso receptor.

### 4.4.2.4. Integración de servicios en Bonita BPM

Una vez creado los servicios estos deben ser integrados al proceso en bonita BPM, primero se empezó con el desarrollo de tareas funcionales. A continuación, mostramos la primera tarea del proceso consultar proyecto en la Figura 4.45.

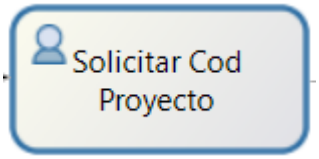


**Figura 4.45 Tarea Solicitar Cod Proyecto.**

En esta tarea como se muestra en la Figura 4.46 se creó un formulario para recibir el parámetro que necesitamos sobre un proyecto que es el id de proyecto. El parámetro será

asignado a una variable de contrato para luego ser manejada con una variable de proceso. El formulario es creado con UI Designer teniendo una caja de texto para escribir el parámetro y el botón aceptar como se muestra en la Figura 4.47.

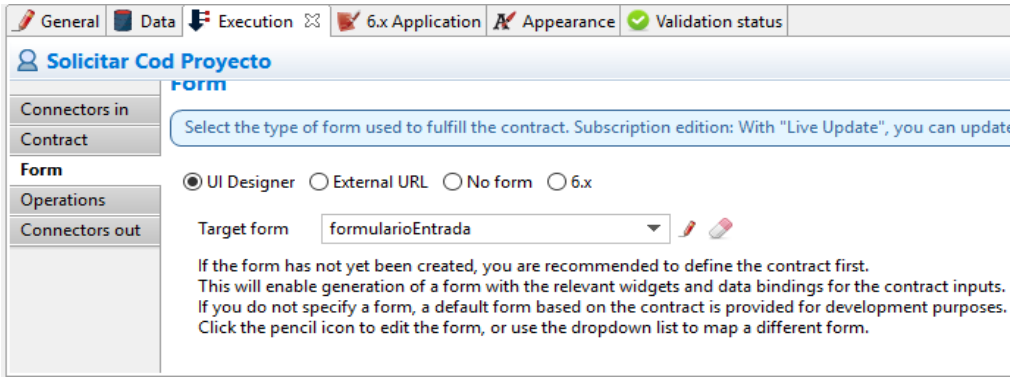


**Figura 4.46 Creación de formulario de entrada en Bonita BPM.**

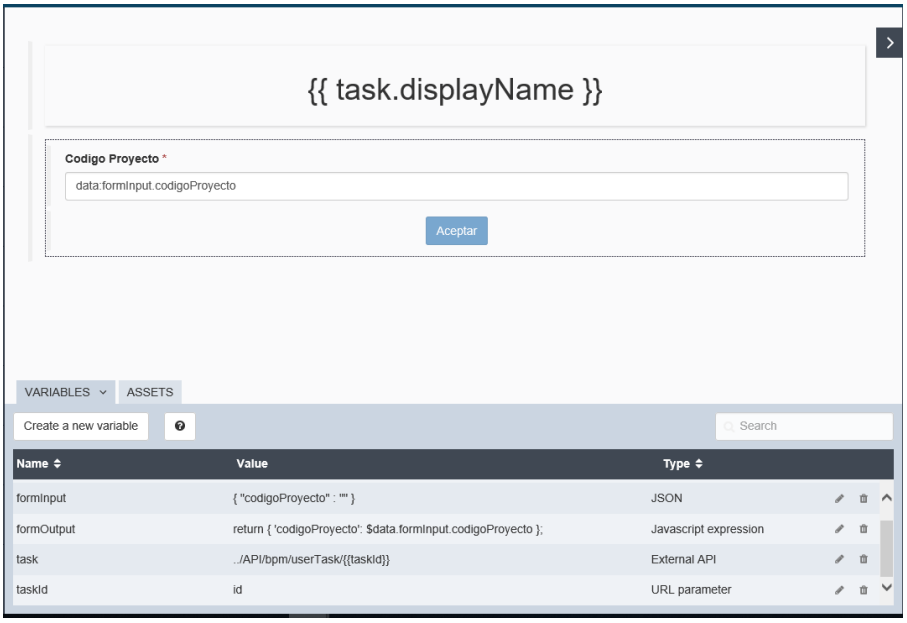


**Figura 4.47 Creación de formulario de entrada UI Designer.**

Como se muestra en la Figura 4.48 la variable de contrato se llama codigoProyecto, el cual almacena el resultado que se introduzca en el formulario. Luego este valor se asigna a la variable de proceso en la pestaña de operaciones como se muestra en la Figura 4.49.

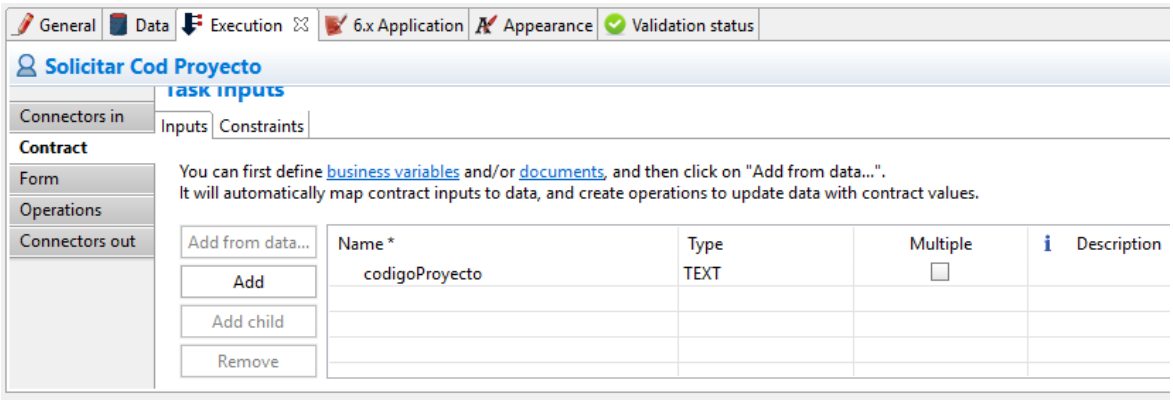


**Figura 4.48 Variable de contrato codigoProyecto.**

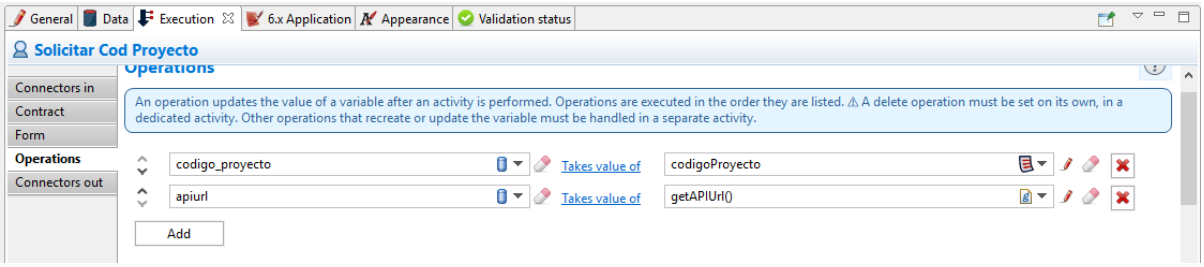


**Figura 4.49 Asignación de datos en Solicitar Cod Proyecto.**

Luego de asignar el resultado en la variable de proceso codigo_proyecto se asigna el url en la variable de proceso apiurl en base al código de proyecto y el Endpoint del servicio que creamos para consultar un proyecto en PostgreSQL con el siguiente método en Groovy.

```
1. return "http://localhost:8280/services/PROYECTO_DataService.SOAP12Endpoi
   nt/_getproyecto?id_proyecto="+codigo_proyecto
```

Después de tener la variable apiurl armada, la tarea Enviar petición hace una llamada de actividad como se muestra en la Figura 4.50.


**Figura 4.50 Tarea Enviar petición.**

Esta actividad recibe parámetros para enviar como se muestra en la Figura 4.51 y parámetros para recibir como se muestra la Figura 4.52 y se llenan cuando el proceso llamado

finaliza. La variable payload se mantiene vacía ya que en este proceso no es de utilidad, pero en la creación de futuros procesos puede hacer falta.

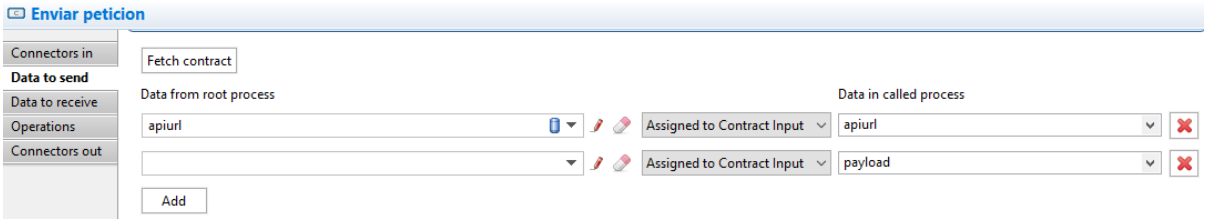


**Figura 4.51 Datos a enviar en Enviar petición.**

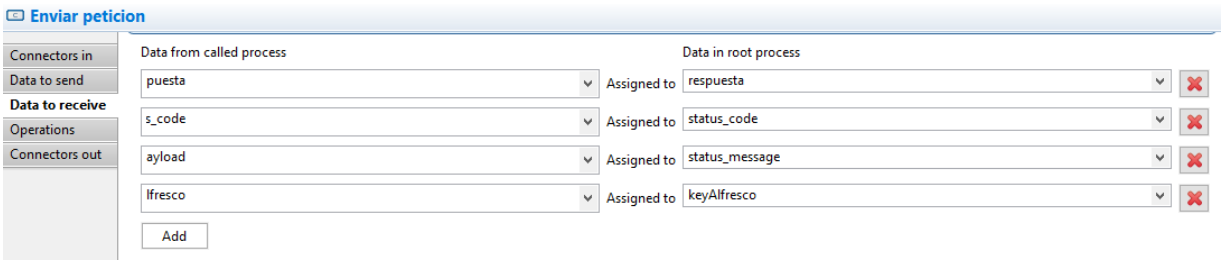


**Figura 4.52 Datos a recibir en Enviar petición.**

El sub proceso retorna una respuesta que esta será en formato XML y contiene la información de un proyecto almacenado en PostgreSQL, también retorna un código y mensaje de status que indicaran si el servicio fue consumido exitosamente o no. Luego de recibir la respuesta en estas variables, en la sección de operaciones como se muestra en la Figura 4.53 se crea un script de Groovy para cada columna que existe en la tabla de Proyecto extrayendo así la información de la variable respuesta.

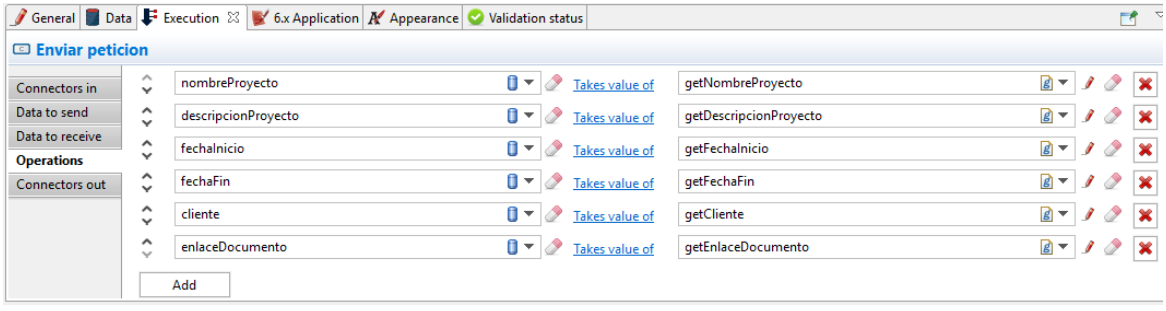


**Figura 4.53 Operaciones en tarea Enviar Petición.**

A continuación, se muestran los muestran los scripts de cada variable donde se hace uso de la clase XmlSlurper la cual nos facilita la extracción de datos de un archivo XML.

- Se crea un script para obtener el nombre del proyecto.

```
1. def PROYECTOCollection = new XmlSlurper().parseText(respuesta)
2. return PROYECTOCollection.PROYECTO.nombre.text()
```

- Se crea un script para obtener la descripción del proyecto.

```
3. def PROYECTOCollection = new XmlSlurper().parseText(respuesta)
4. return PROYECTOCollection.PROYECTO.descripcion.text()
```

- Se crea un script para obtener la fecha inicio del proyecto.

```
5. def PROYECTOCollection = new XmlSlurper().parseText(respuesta)
6. return PROYECTOCollection.PROYECTO.fecha_inicio.text()
```

- Se crea un script para obtener la fecha fin del proyecto.

```
7. def PROYECTOCollection = new XmlSlurper().parseText(respuesta)
8. return PROYECTOCollection.PROYECTO.fecha_fin.text()
```

- Se crea un script para obtener el id de cliente del proyecto.

```
1. def PROYECTOCollection = new XmlSlurper().parseText(respuesta)
2. return PROYECTOCollection.PROYECTO.id_cliente.text()
```

- Se crea un script para construir el url para el servicio de Alfresco y así obtener su diagrama Gantt al consumir el servicio.

```
1. return "http://localhost:8180/alfresco/api/-default-
   /public/cmis/versions/1.1/browser/root/Shared/proyectos/gantt_"+codigo_p
   royecto+".pdf?alf_ticket="+keyAlfresco
```

La siguiente tarea como se muestra en la Figura 4.54 es Recibir Respuesta donde se muestra a través de un formulario los datos que nos retornó el proceso receptor y que fueron cargados en sus variables respectivas en la tarea Enviar petición.

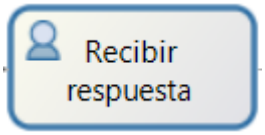


**Figura 4.54 Tarea Recibir respuesta.**

En esta tarea se crea un formulario de entrada como se indica en la Figura 4.55.

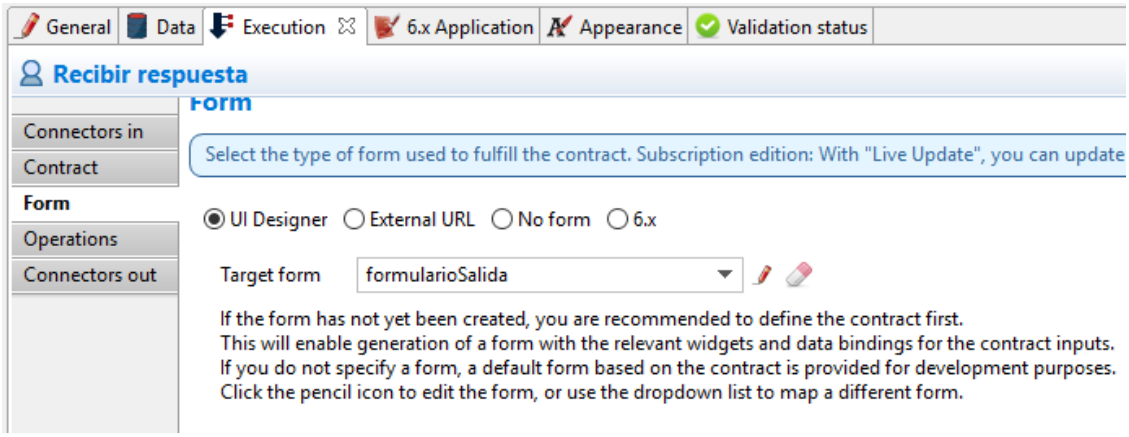


**Figura 4.55 Creación de formulario de salida en Bonita BPM.**

En este formulario se muestra la información que contiene cada variable de proceso perteneciente a un proyecto como se indica en la Figura 4.56.

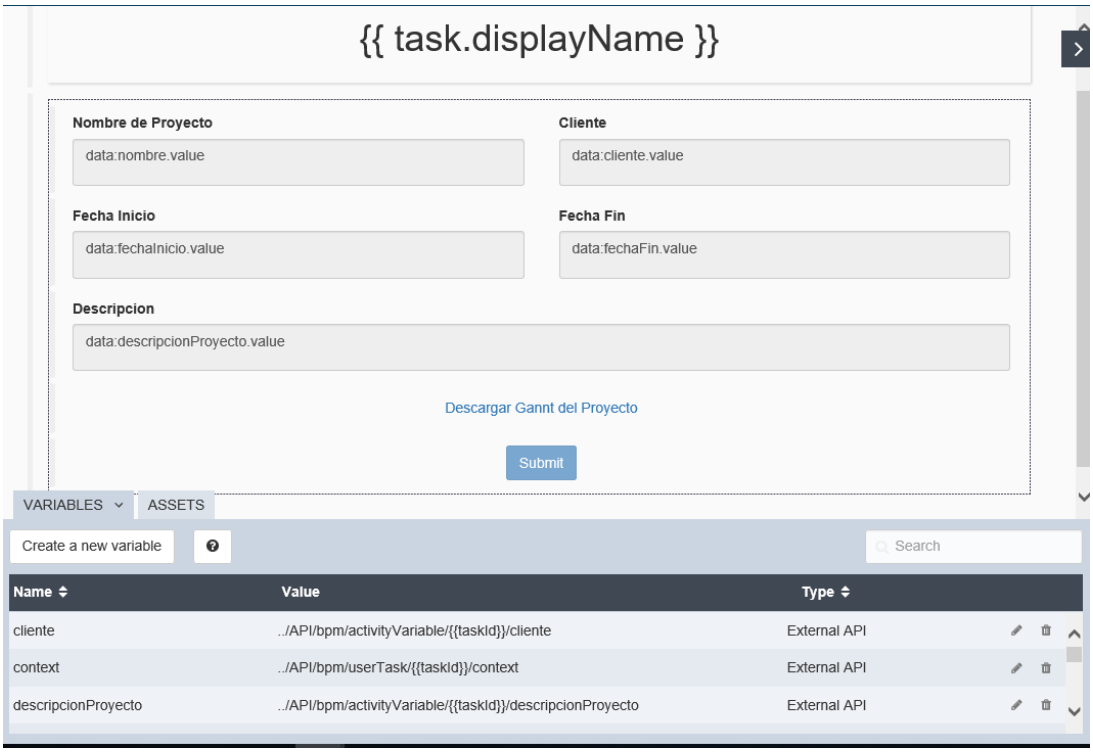


**Figura 4.56 Creación de formulario de salida UI Designer.**

Con respecto al proceso receptor se utiliza conectores de servicios en las tareas que se muestran a continuación en la Figura 4.57. Las tareas que no tienen función alguna no se eliminaron para mantener un BPMN fácil de entender.

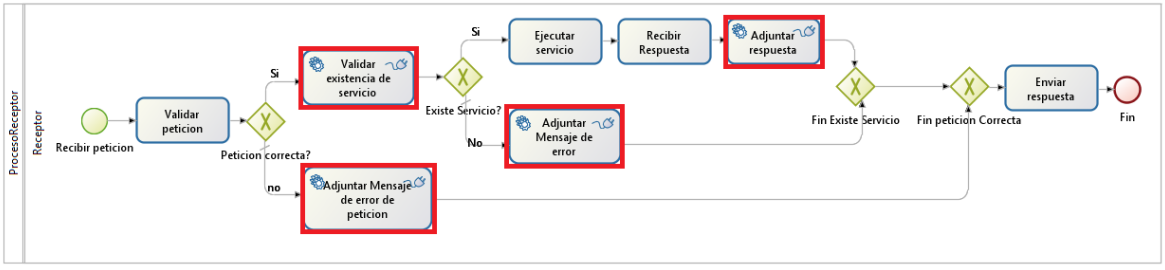


**Figura 4.57 Integración de servicios en el proceso receptor.**

El primer paso es validar si la petición es correcta para esto hacemos validaciones en la compuerta exclusiva de petición correcta que se muestra en la Figura 4.58.

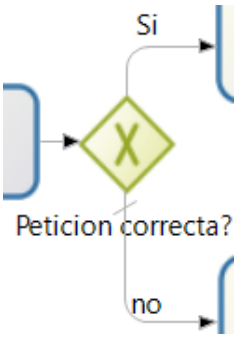


**Figura 4.58 Compuerta exclusiva de validación de petición.**

Se crea un script de para validar si la variable apiurl es vacía. En caso afirmativo no será una petición correcta y por lo tanto se ira por el camino por defecto que es el No, en caso de que la variable no está vacía entonces se puede continua en el flujo del proceso correcto.

```
1. return !apiurl.isEmpty()
```

En la Figura 4.59 se observa la tarea Adjuntar mensaje de error de petición la cual es la encargada de notificar por correo el error.

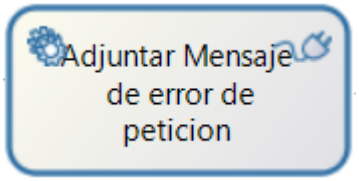


**Figura 4.59 Tarea Adjuntar mensaje de error de petición.**

Se crea operaciones para asignar un mensaje de error y un código de error a las variables status_message y status_code como se observa en la Figura 4.60, las cuales son utilizadas por el servicio y además devueltas al proceso de consultar proyecto.

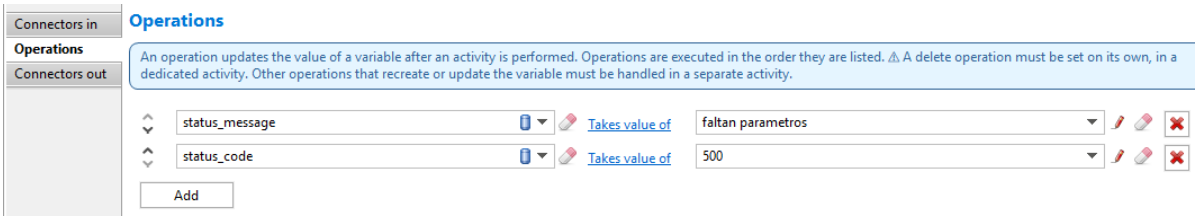


**Figura 4.60 Operaciones en Adjuntar mensaje de error de petición.**

En la sección de conectores de salida se agregó un conector REST POST llamado errorParametros, este conector en la parte de configuración de solicitud se introdujo el endpoint de nuestro servicio de correo, especificando una entrada o payload tipo json y además se especificó el payload con los parámetros y sus valores como se indica en la Figura 4.61.

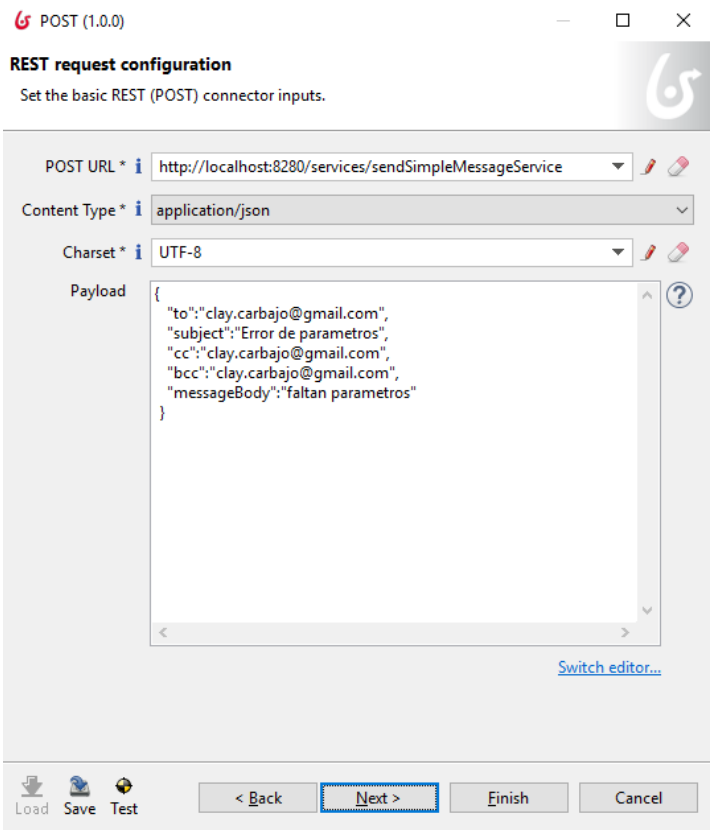


**Figura 4.61 Configuración de Solicitud para servicio de error de petición.**

Por otro lado, en la Figura 4.62 se observa la tarea Validar existencia de servicio, en esta tarea consumiremos el servicio para verificar si recibimos una respuesta.

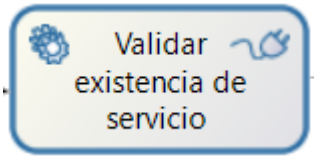


**Figura 4.62 Tarea Validar existencia de servicio.**

En la sección de conectores de entrada se agrega un conector REST GET donde en la parte de configuración de solicitud se llamó a la variable apiurl como se indica en la Figura 4.63

recordando que esta variable fue llenada por el proceso consultar proyecto y verificada en la compuerta exclusiva de validación de petición.

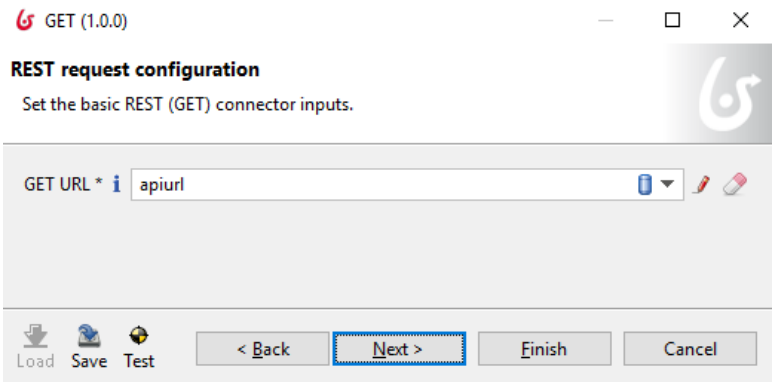


**Figura 4.63 Configuración de solicitud para servicio de consulta de proyecto.**

En la Figura 4.64 se observa la parte de operaciones de salida, donde se hizo la asignación de las variables de status_code, status_message y respuesta, donde respuesta será del tipo XML.

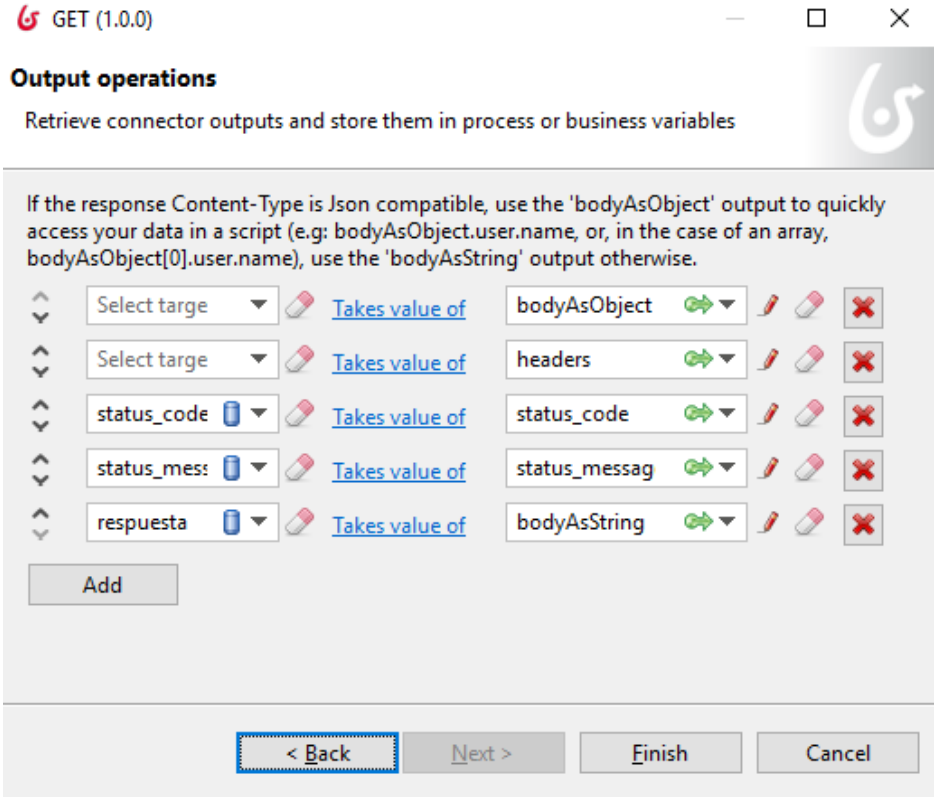


**Figura 4.64 Operaciones de salida para servicio consultar proyecto.**

Luego se hace una validación en la compuerta exclusiva que se muestra en la Figura 4.65.


**Figura 4.65 Compuerta exclusiva de validación sobre consultar proyecto.**

Esta validación es un script de Groovy que consulta la variable de código estatus que se guardó al consumir el servicio de consulta de proyecto. Quedando el código como:

```
1. return status_code == 200
```

Si el código de estatus es 200 significa que no hubo ningún error siguiendo el flujo del proceso, pero si es distinto entonces ocurrió un error interno y se ira por el flujo alterno continuando con la tarea Adjuntar mensaje de error que se muestra en la Figura 4.66.

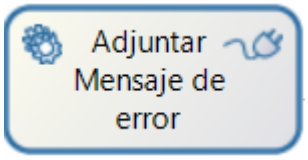


**Figura 4.66 Tarea Adjuntar mensaje de error.**

En esta tarea de desarrolla algo similar a la primera tarea de notificación de error, con la diferencia de que las variables del mensaje y código de estatus ya tienen un valor asignado. En la Figura 4.67 se observa cómo fue la configuración de solicitud para hacer él envió del correo a través de un servicio REST POST con un payload asociado.

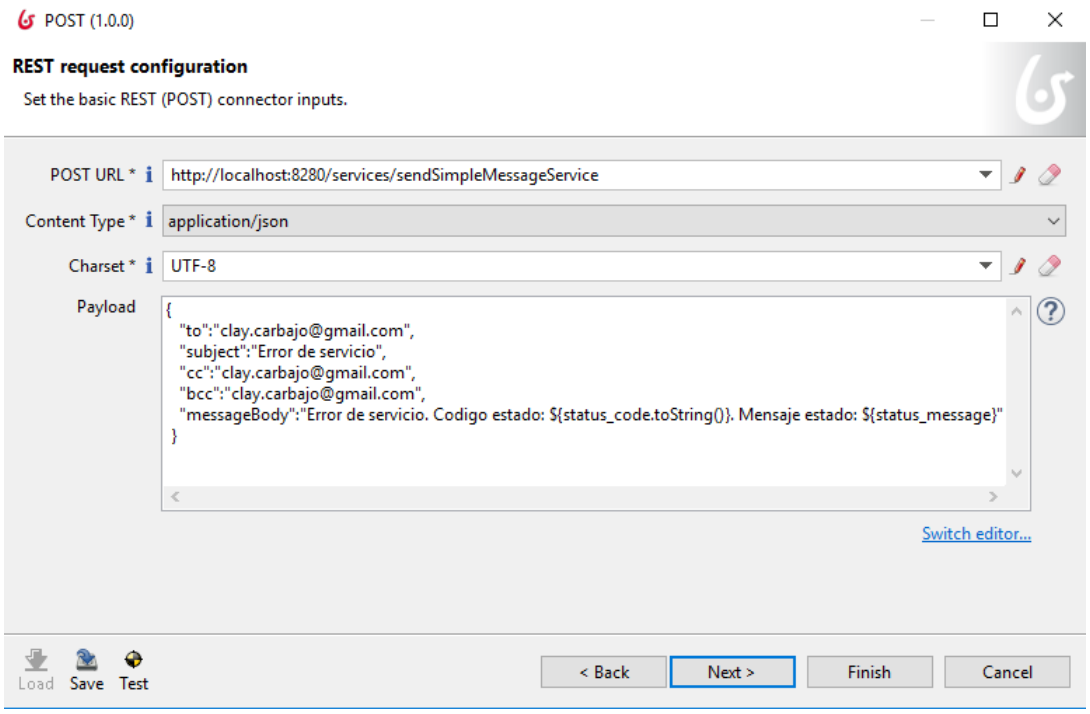


**Figura 4.67 Configuración de solicitud para servicio de consulta de proyecto.**

Por otro lado como se observa en la Figura 4.68, el flujo del proceso continua hasta llegar a la tarea Adjuntar respuesta, esta tarea se encarga de enviar una notificación por correo indicando que el consumo de servicio fue exitoso y además se consume un servicio de Alfresco para obtener su llave de acceso y poder consultar un documento en su repositorio.

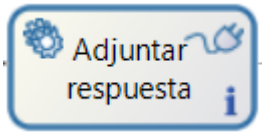


**Figura 4.68 Tarea Adjuntar respuesta.**

Para el servicio de Alfresco en la configuración de solicitud se colocó la url del api que retorna la llave de acceso como se muestra en la Figura 4.69.

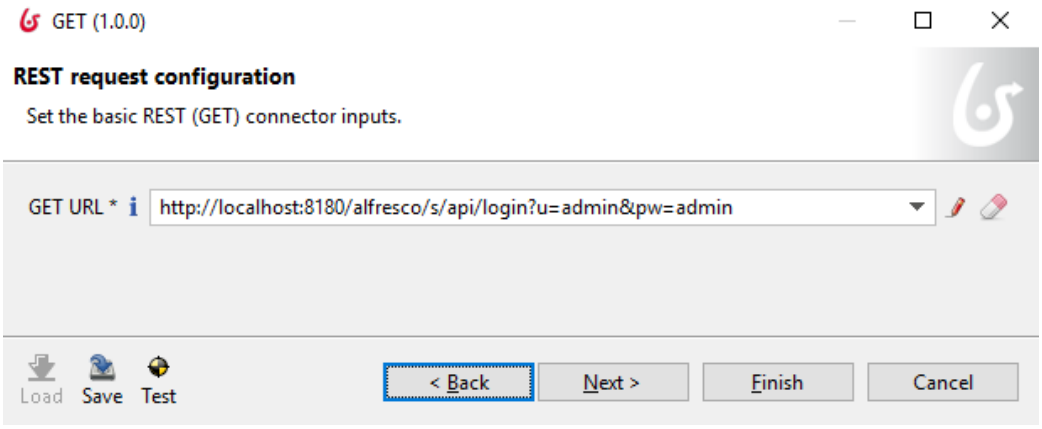


**Figura 4.69 Configuración de solicitud para servicio de llave Alfresco.**

Además, en la parte de operaciones de salida se creó un script de Groovy para extraer de la respuesta XML la llave de acceso y asignarla como se indica en la Figura 4.70.

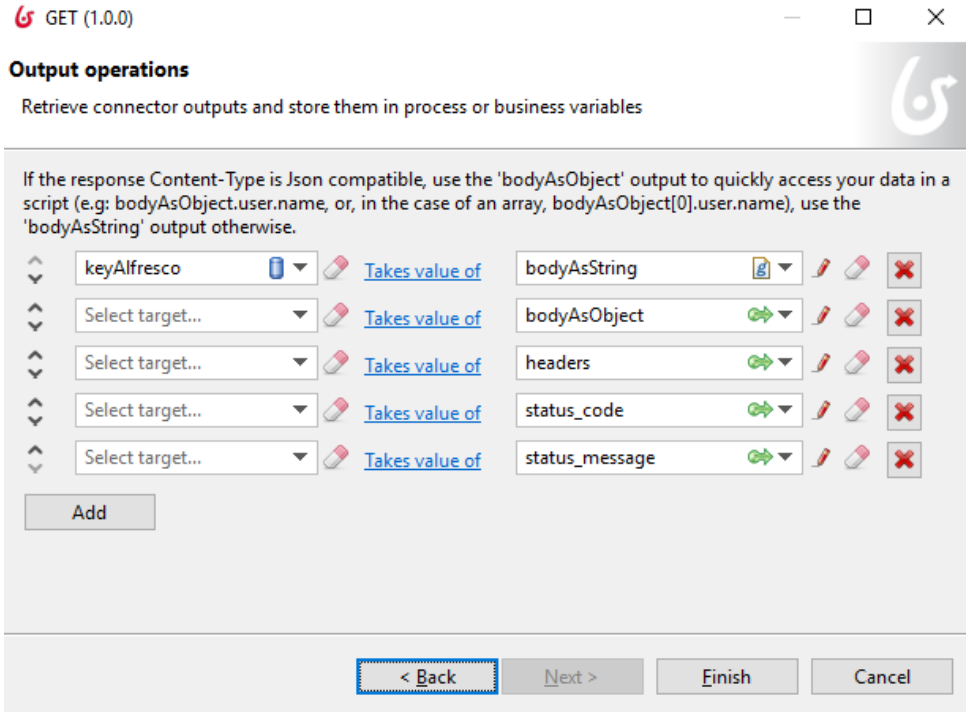


**Figura 4.70 Operaciones de salida para servicio de llave Alfresco.**

El script es el siguiente:

```
1. def PROYECTOCollection = new XmlSlurper().parseText(bodyAsString)
2. return PROYECTOCollection.text()
```

Luego de esta tarea sigue el flujo de proceso hasta terminar, las variables de proceso son asignadas a las variables de proceso de consultar proyecto como se especifica en las operaciones de la actividad de llamada.

### 4.4.3. Pruebas y monitoreo

Las pruebas que se realizaron sobre el proceso fueron a través del flujo principal del proceso y de los flujos alternos para validar su correcto funcionamiento. Para el proceso receptor existen 2 flujos alternos donde se verificó su funcionamiento. A continuación, en la Figura 4.71 se puede observar una representación de los flujos del proceso, siendo el color verde el flujo ideal y el rojo los flujos alternos.

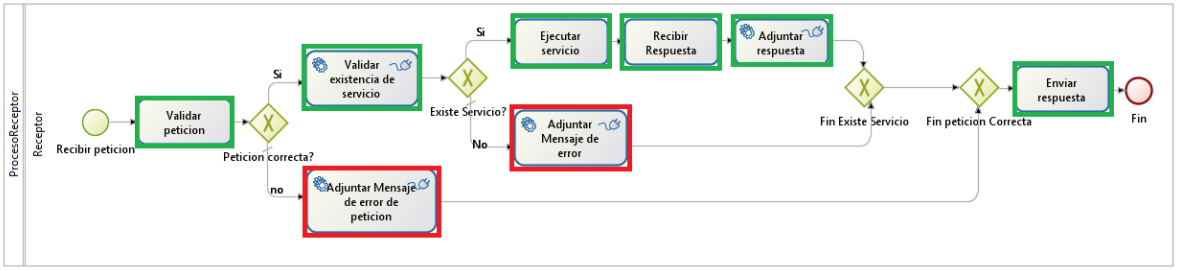


**Figura 4.71 Representación de flujos de proceso del proceso receptor.**

- Flujo principal:

Se empieza a ejecutar el proceso de consultar proyecto, donde nuestra primera interfaz es la solicitud de un código de proyecto como se muestra en la Figura 4.72.


**Figura 4.72 Interfaz de solicitud de código de proyecto.**

Se introduce un id de proyecto que sea válido y se selecciona aceptar. El proceso llama al proceso receptor y valida si la petición es correcta, como se introdujo un código de proyecto valido entonces continua a la validación de servicio donde se consume el servicio en base de datos, como se muestra en la **Figura 4.73**, este id de proyecto es válido y existe en la base de datos retornando los datos asociados al proyecto y un código de estatus 200 con mensaje OK, cumpliéndose la condición de la compuerta de la existencia de servicios. El flujo continuo

a través de las tareas de referencia hasta llegar a la tarea de adjuntar respuesta donde obtendremos la llave de acceso de Alfresco y enviaremos un correo indicando que la solicitud de datos fue exitosa. A continuación, se muestra la recepción del correo en la **Figura 4.74**.

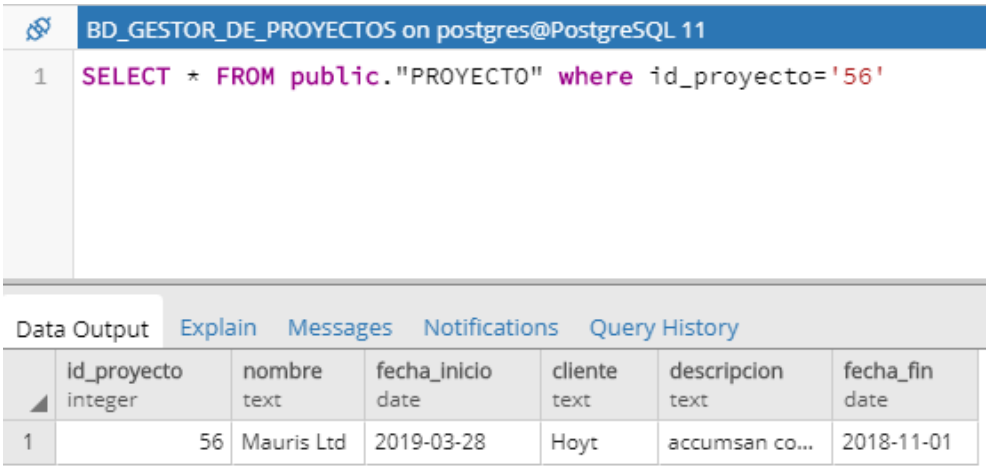


**Figura 4.73 Existencia de proyecto en la base de datos.**


**Figura 4.74 Recepción de correo de solicitud exitosa.**

Luego de terminar el proceso receptor entonces continua el proceso de consultar proyecto con la tarea de recibir respuesta, en esta tarea se mostrará el formulario con la información traída y el link disponible para la consulta del diagrama de Gantt del proyecto como se indica en la **Figura 4.75**. Al seleccionar el link nos mostrara una previsualización del documento y la opción de descargar como se muestra en la **Figura 4.76**.


**Figura 4.75 Interfaz de recepción de datos.**

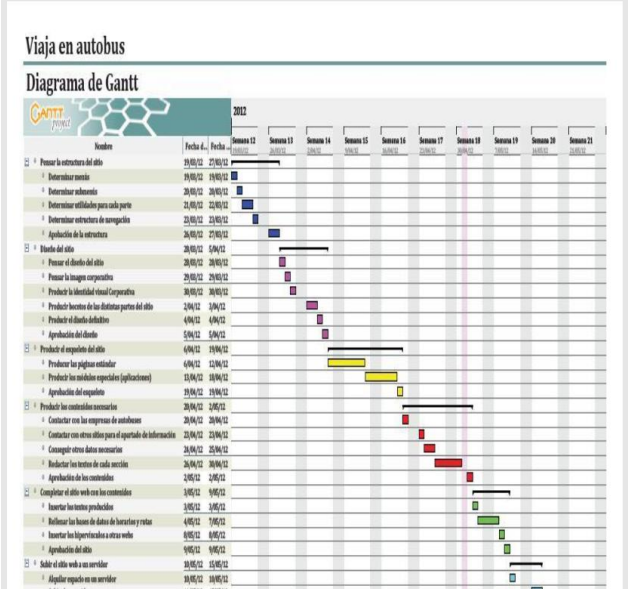


**Figura 4.76 Visualización del documento asociado al proyecto 57.**

- Flujo Alterno error de petición:

El proceso utiliza este flujo alterno cuando la url es vacía, esto fue de utilidad cuando se hizo la asignación de parámetros para enviar del proceso consulta de proyecto al proceso receptor y asegurar que efectivamente el parámetro url no estaba vacía. Para replicar este flujo se inició solo el proceso receptor, en la compuerta de validación de petición como el parámetro url nunca fue llenado por una actividad de llamada entonces continua por el flujo alterno, en este flujo alterno se envía un correo electrónico informando que hubo un error de validación como se muestra en la Figura 4.77.


**Figura 4.77 Recepción de correo de petición errónea.**

- Flujo Alterno error de existencia del servicio:

El proceso utiliza este flujo alterno cuando ocurre algún error con el servicio o cuando el servicio trae una respuesta invalida o vacía. Para replicar este flujo alterno se inició el proceso de consultar proyecto, se ingresó un código de proyecto que no existe en la base de datos y se seleccionó aceptar, el flujo de proceso continuara hasta llegar a la compuerta de existencia de servicio donde se valida si la variable de proceso llamada respuesta contiene información. A continuación, en la Figura 4.78 podemos observar el correo que se envía en este flujo alterno indicando que el resultado está vacío.


**Figura 4.78 Recepción de correo de falla en servicio.**

Cuando finaliza el proceso receptor y regresa al flujo del proceso principal, como no hay datos que mostrar entonces se muestra un mensaje en la interfaz indicando el error como se en la Figura 4.79.

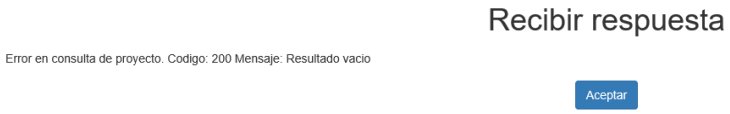


**Figura 4.79 Interfaz de recepción de datos errónea.**

Además de estas pruebas, se hicieron se manejaron las excepciones en las compuertas exclusivas de decisiones en caso de algún error de bonita BPM para que el proceso use utilice los flujos alternos sin detener la ejecución del proceso. A continuación, se muestran la validación para petición correcta con el manejo de excepciones.

```
1. try{
2.         return !apiurl.isEmpty()
3. }catch(Exception e){
4.         return false
5. }
```

Por otra parte, la validación de existencia de servicio es la siguiente.

```
1. try{
2.         def xml = new XmlSlurper().parseText(respuesta)
3.         return status_code == 200 && xml.PROYECTO.size() > 0
4. }catch(Exception e){
5.         return false
6. }
```

Por último, se agrega validaciones en el formulario de solicitud de datos para asegurar el ingreso de un código de proyecto válido.

Con respecto al monitoreo WSO2 EI nos muestra un contador de cuantas veces fue consumido o fallo un servicio, sus tiempos de respuesta y una gráfica con respecto al tiempo de respuesta. A continuación, en la Figura 4.80 se muestra estas características con respecto al servicio para consultas en la base de datos.

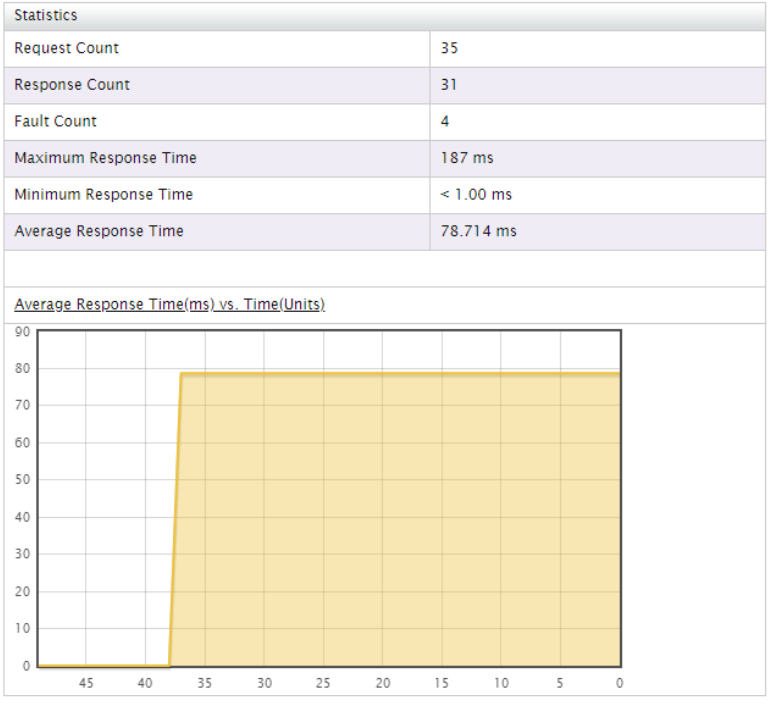


**Figura 4.80 Monitoreo de servicios en WSO EI.**

# CONCLUSIONES Y RECOMENDACIONES

Al finalizar la realización este Trabajo Especial de Grado, se desarrolló como solución el modelo de Arquitectura Orientada a Servicios para el Sistema de Gestión de Procesos de Negocio desarrollado bajo el enfoque de la gestión de proyectos, utilizando un sistema de integración que permitió sustituir el uso de conectores específicos en bonita BPM por el consumo de servicios para la comunicación entre los componentes de base de datos, gestión documental y envió de correos electrónicos, para así recibir los beneficios de SOA como lo son la interoperabilidad de los componentes y tecnologías heterogéneas, un mayor control sobre la comunicación entre los componentes, disminución de los tiempos de desarrollo, ciclos de prueba y facilidad de integración de nuevas funcionalidades.

El desarrollo de la solución se trabajó bajo una adaptación de la metodología Scrum donde los tiempos de sprints y meetings fueron modificados para culminar las distintas fases de desarrollo que van desde el análisis y modelado de requerimientos para luego, un desarrollo y despliegue de procesos que hacen uso de servicios, hasta la ejecución de pruebas y monitoreo de un sistema que posee un modelo de Arquitectura Orientada a Servicios, estas fases de desarrollo son una adaptación del ciclo de vida BPM donde las fases de este ciclo se agruparon y utilizaron para tener mayor orden. En la solución se utilizó el Sistema de Gestión de Procesos de Negocio Bonita BPM, el sistema de integración WSO2 Enterprise Integrator que permitió la creación de los servicios, el gesto documental Alfresco que permitió el almacenamiento y gestión de documentos de un proyecto y por último el Sistema Manejador de Base de Datos PostgreSQL que permitió el almacenamiento de datos relacionados a un proyecto.

Esta solución ofrece un aporte al desarrollo de los Sistemas de Gestión de Procesos de Negocio complejos que hacen uso de una gran cantidad de conectores específicos para cada componente, reutilizándolos en varios procesos y tareas provocando largos tiempos de desarrollo, difícil mantenimiento, mayores tiempos de respuesta y altos costos. Además, permite recibir otros beneficios que ofrece una Arquitectura Orientada a Servicios para las organizaciones y el desarrollo de sus soluciones.

Como recomendaciones se plantea el uso centralizado de procesos que disminuya a lo medida de lo posible las reutilizaciones de conectores, así como se desarrolló el proceso receptor de la solución que tiene el propósito de recibir las solicitudes de datos de distintos procesos y ejecutar los servicios correspondientes para retornar una respuesta al proceso que lo llamo.

# REFERENCIAS BIBLIOGRÁFICAS Y DIGITALES

Alfresco Software, Inc., (2019). About Alfresco, https://www.alfresco.com/es/company/about-alfresco

Bonitasoft (Sin fecha), Que es BonitaSoft. https://es.bonitasoft.com/business-process-management-bpm

Blog Powerdata (2014), Qué es la arquitectura orientada a servicios SOA. https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/394442/qu-es-la-arquitectura-orientada-a-servicios-soa.

Chakray (2017), Qué es WSO2 ESB. http://www.chakray.com/que-es-wso2-esb-la-solucion-de-integracion-empresarial/.

DB-Engines Ranking (2018). https://db-engines.com/en/ranking

Diego Sánchez Schenone (2011), Introducción a Business Process Management (BPM). https://www.ibm.com/developerworks/ssa/local/websphere/introduccion-bpm/index.html.

Derek M. (2006), Issues and Best Practices for the BPM and SOA Journey. http://ibiscom.net/index.php/es/ibpm/56-bpm-y-soa-dos-tecnologias-complementarias.

De Miguel A., Piattini M., Marcos E. (1999) Diseño de bases de datos relacionales. Ed. Ra-ma.

EcuRed (Sin fecha), Intalio. https://www.ecured.cu/Intalio

EcuRed (Sin fecha), BPM RAD. https://www.ecured.cu/BPM-RAD.

EcuRed (Sin fecha), Metodologías tradicionales. URL http://www.ecured.cu/Metodolog%C3%ADas_Tradicionales.

Formula proyectos urbanos (2012), Qué es pmi y qué es pmbok. https://formulaproyectosurbanospmipe.wordpress.com/2012/01/18/que-es-el-pmi-y-que-es-el-pmbok/.

Fperezsoa (2009), SOA Arquitectura Orientada a Servicios, https://fperezsoa.wordpress.com/2009/05/04/%C2%BFque-es-soa-la-arquitectura-orientada-a-servicios/

Groovy Main page (2019), ¿Qué es Groovy? htttp://groovy-lang.org/

Johansen, O. (2004). Introducción a la Teoría General de Sistemas (17ª. ed.). México DF: Ed. Limusa-Noriega.

Java Home page (2019), ¿Qué es Java? https://www.java.com/es/download/faq/whatis_java.xml

Kress J., Berthold M, Hajo N. Schmeidel D. Schmutz G, Trops B. (2013), Enterprise Service Bus. http://www.oracle.com/technetwork/articles/soa/ind-soa-esb-1967705.html.

Kenneth C.Laudon & Jane P.Laudon (2012) Sistemas de información Gerencial 12va

PMI (2013), Guía de los Fundamentos de la Dirección de Proyectos (Guía del PMBOK). 5ta edición.

PostgreSQL Global Development Group. (2019). About PostgreSQL, https://www.postgresql.org/about/

Roger C. (2011), Que BPM Suite uso. https://holisticsecurity.io/2011/07/21/jbpm-bonita-intalio-processmaker-activiti-que-bpm-suite-uso/

Silberschatz, A., K. H. and Sudarshan, S. (2002). Fundamentos de Base de Datos. Bombay. España: McGRAW-HILL., 4ta edición. Edition

Silva, R. (2009). Sistema de información. http://www.monografías.com.

Silva, R. (2010). Tipos de sistemas de información. http://si-2010-1-e4. blogspot.com/.

Singhal J. (2014), A Sprint Is More Than Just a Timebox. https://www.scrumalliance.org/community/articles/2014/may/sprint-is-more-than-just-a-timebox.

Smith, H.and Fingar, P. (2003). Business process management (bpm): The third wave. Technical report, Tampa, EUA: Meghan-Kiffer Press.

Universidad Unión Bolivariana (Sin fecha), programación extrema XP. http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html.

Vera M. (2014), Qué se entiende por SOA y cuáles son sus beneficios. http://www.i2btech.com/blog-i2b/tech-deployment/que-se-entiende-por-soa-y-cuales-son-sus-beneficios/.

Veyrat P. (2016), Conquiste resultados com as 6 fases do Ciclo de Vida BPM. http://www.venki.com.br/blog/ciclo-de-vida-bpm/.

WSO2 Inc., (2018). WSO2 EI Overview, https://docs.wso2.com/display/EI640/Overview

WSO2 Inc., (2019). Configuring Gmail Operations, https://docs.wso2.com/display/ESBCONNECTORS/Configuring+Gmail+Operations