Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Código 31105151 — Trabajo Fin de Máster

Generación automática de configuración de procesos en la herramienta JIRA

Autor: Enrique Font Pichardo
Director: Ismael Abad Cardiel
Curso: 2017 / 2018
Convocatoria: Septiembre 2018

Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Código 31105151 — Trabajo Fin de Máster

Generación automática de configuración de procesos en la herramienta JIRA

Tipo B: Trabajo específico propuesto por el alumno

Autor: Enrique Font Pichardo
Director: Ismael Abad Cardiel

DECLARACIÓN JURADA DE AUTORÍA DEL TRABAJO CIENTÍFICO, PARA LA DEFENSA DEL TRABAJO FIN DE MASTER

Fecha: 01/09/2018

Quién suscribe:

Autor(a): Enrique Font Pichardo

D.N.I/N.I.E/Pasaporte.: 77619917T

Hace constar que es la autor(a) del trabajo:

Generación automática de configuración de procesos en la herramienta JIRA

En tal sentido, manifiesto la originalidad de la conceptualización del trabajo, interpretación de datos y la elaboración de las conclusiones, dejando establecido que aquellos aportes intelectuales de otros autores, se han referenciado debidamente en el texto de dicho trabajo.

DECLARACIÓN:

- ✓ Garantizo que el trabajo que remito es un documento original y no ha sido publicado, total ni parcialmente por otros autores, en soporte papel ni en formato digital.
- ✓ Certifico que he contribuido directamente al contenido intelectual de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él.
- ✓ No he incurrido en fraude científico, plagio o vicios de autoría; en caso contrario, aceptaré las medidas disciplinarias sancionadoras que correspondan.

Fdo.

Impreso TFdM05_Autor. Autorización de publicación

y difusión del TFdM para fines académicos

Autorización

Autorizo/amos a la Universidad Nacional de Educación a Distancia a difundir y utilizar, con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firma del/los Autor/es

Juan del Rosal, 16
28040, *Madrid*

Tel: 91 398 89 10
Fax: 91 398 89 09

www.issi.uned.es

RESUMEN

El objetivo de este trabajo se centra en facilitar el proceso de configuración de procesos en JIRA, partiendo de la abstracción del problema para construir un modelo que permita la representación de una solución general.

En este trabajo se define un lenguaje específico de dominio que permita trabajar con los esquemas de configuración de procesos JIRA de una forma más sencilla. Se emplea el desarrollo dirigido por modelos para la generación automática de configuraciones JIRA que permitan la gestión de un proceso.

Se construye una herramienta encargada de proporcionar un entorno de trabajo a los administradores JIRA, donde la configuración de la aplicación está basada en un metamodelo, contribuyendo a la configuración de los procesos optimizando la productividad del equipo que administra JIRA, asegurando aspectos importantes como son la calidad, mantenibilidad y reutilización de elementos.

Lista de palabras clave

JIRA, Model-Driven Development (MDD), Domain-Specific Language (DSL), Domain-Specific Modeling (DSM), Abstracción, Generación automática de código, Ingeniería de software, metamodelo, modelo.

EXECUTIVE SUMMARY

The aim of this research is to make easier the procedure of configuration process in JIRA, starting from the abstraction of the problem to build a template that allows to achieve a general solution.

In this research we define a Domain-Specific language that allows us to work with the JIRA process configuration schedule in a simpler way. It uses the model-driven development for the automatic generation of JIRA configurations that allow the management of a process.

A tool is created to provide a working environment for JIRA administrators, where the application configuration is based on a metamodel, contributing to the configuration of the processes, optimizing the productivity of the team that JIRA manages, ensuring important aspects such as quality, sustainability and reuse of elements.

Keywords

JIRA, Model-Driven Development (MDD), Domain-Specific Language (DSL), Domain-Specific Modeling (DSM), Abstraction, Automatic code generation, Software Engineering, metamodel, model.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABLAS

1. INTRODUCCIÓN

1.1. Ámbito del problema

La gestión de servicios de tecnologías de la información cada día está alcanzando mayor relevancia. Mientras que hasta los años 70 la mayor preocupación estaba en la mejora y desarrollo de nuevo hardware, hasta bien entrados los años 80, este interés era por el desarrollo del software. A partir de los 90 la preocupación se ha centrado en la gestión de los procesos y servicios.

La gestión de servicios ha sido siempre una continuación o extensión del desarrollo, pero en los últimos años se está cambiando esta situación, dado que, como corroboran estudios de Gartner Group, entre el 70% y 80% de los gastos en el ciclo de vida de los sistemas de información son en la fase de explotación. Esta situación se confirma viendo situaciones en las que el 60% del tiempo de los desarrolladores se dedica a tareas de mantenimiento, o que las actividades diarias de la plantilla de TI parecen centrarse en tareas de gestión. Con la gestión de servicios, se consigue una nueva actividad que está cada día más madura, y la prueba es la cantidad de marcos de trabajo teóricos que surgen cada día.

La competitividad es una condición para las organizaciones que buscan satisfacer las necesidades de sus clientes, en un medio globalizado y altamente dinámico. Las organizaciones basan su operación en procesos de negocio, y son las organizaciones que monitorizan, implementan mejora continua y gestionan adecuadamente sus procesos de negocio las que logran ser y mantenerse competitivas.

La identificación, descripción y automatización de procesos de negocio, son partes fundamentales del trabajo requerido para habilitar tareas de monitorización y mejora continua de procesos de negocio. Actualmente sistemas software de ticketing orientados a la gestión de procesos y servicios como JIRA, no sólo existen en una única versión que cubra las necesidades globales de un mercado objetivo, sino que también, existen muchas variantes, las cuales se especializan y diferencian por las necesidades divergentes de los clientes.

JIRA es una herramienta de ticketing en línea para la administración de tareas, el seguimiento de errores, incidencias o problemas y para la gestión operativa de proyectos. Todas estas funcionalidades las permite realizar mediante la configuración de roles, tipos de ticket, flujos de trabajo, pantallas, notificaciones y permisos que los administradores JIRA deben realizar manualmente dentro de la aplicación para cumplir con las necesidades del cliente. Frecuentemente, estas configuraciones son muy similares unas de otras, pudiendo reducir, considerablemente, el tiempo que los administradores JIRA dedican a realizar dichas configuraciones y aumentando su calidad automatizando estas tareas. Para lograrlo en el presente trabajo se realizará la implementación de un proceso de generación automática de configuraciones JIRA que, haciendo uso del API java que ofrece la aplicación, leerá un fichero de configuración

que seguirá la gramática de un DSL (Domain-Specific Language) que definiremos previamente para el modelado de configuraciones JIRA.

Desde un editor de texto se podrán producir modelos de configuraciones JIRA y a partir de esos modelos se generará el código fuente y las configuraciones en la aplicación. Las configuraciones finales se pueden producir a partir de un modelo siendo este el protagonista del proceso. De esta forma el proceso estará enfocado en el paradigma de modelado específico de dominio DSM (Domain-Specific Modeling).

1.2. Planteamiento del problema

Actualmente entre los objetivos de las organizaciones están la reducción de costes, el aumento de la calidad, el aumento de la productividad y la reducción del tiempo de desarrollo y mantenimiento de las aplicaciones. Estos objetivos se han demostrado que se pueden lograr con la aplicación de principios de enfoques como MDD (Model-Driven Development) [1]. Así mismo, hoy en día ha proliferado el uso de aplicaciones de gestión de procesos y servicios, tales como JIRA, que con un uso adecuado pueden mejorar la calidad, productividad y eficiencia de los procesos. A medida que los procesos se van adaptando a las necesidades, es necesario que los administradores JIRA vayan ajustando la configuración de cada uno de estos procesos en JIRA.

El conocimiento que los administradores JIRA tienen del proceso a configurar y el conocimiento que los gestores del proceso tienen de JIRA son bajos. En muchas ocasiones el personal encargado de gobernar los procesos no tienen el suficiente conocimiento de administración de la aplicación JIRA, lo que dificulta que en el momento que estos trasladan las necesidades del proceso a los administradores JIRA puedan surgir malentendidos, ya que los administradores JIRA no siempre tienen todo el conocimiento del proceso.

Siguiendo las buenas prácticas de los procesos de desarrollo de software, previamente a crear o modificar las configuraciones JIRA existentes en el entorno de producción, es necesario realizar dichas configuraciones en los entornos de integración y pre-producción para validar su correcto funcionamiento. Por lo tanto, se debe realizar las mismas tareas de configuración tantas veces como entornos tengamos, con el consiguiente incremento de los siguientes factores:

- La probabilidad de cometer errores se incrementa a medida que se incrementa el número de entornos.
- El tiempo de realización de las tareas de configuración será más elevado a medida que se incrementa el número de entornos.
- Los costes de configuración y mantenimiento serán mayores cuanto más entornos tengamos

1.3. Objetivo

El objetivo general de este trabajo es construir una herramienta para la generación automática de configuración de procesos en la herramienta JIRA bajo desarrollo dirigido por modelos textuales que aumente la productividad, calidad, la reducción del

tiempo de configuración y mantenimiento de procesos en JIRA, y en general reduzca los costes.

Para lograr el objetivo general se han planteado los siguientes objetivos específicos:

- Comparar las metodologías que usan MDD para implementar las mejores prácticas de ingeniería de software para la obtención de aplicaciones funcionales.
- Construir un DSL, que permita la generación automática de configuración de procesos en la herramienta JIRA.
- Evaluar la herramienta demostrando funcionalidad con pruebas de concepto.

Para la realización del proyecto se hará uso de diversas tecnologías dentro del marco del desarrollo web como JAVA, JSON, XML, HTML, CSS, JavaScript, así como el propio API de JIRA y otras tecnologías como Apache Velocity, Lucene, Apache Maven.

1.4. Estructura

La estructura de este documento se compone de ocho secciones.

En la primera sección se hace una introducción del ámbito y planteamiento del problema.

En la segunda sección se trata de describir la base teórica sobre la que se va a sustentar el proyecto que consiste en el desarrollo de un proceso de generación automático de configuración de procesos en JIRA bajo el marco de desarrollo dirigido por modelos.

En la sección de implementación del generador se describe el proceso de desarrollo del proyecto.

En la cuarta sección se trata de evaluar los resultados de capacidad y flexibilidad que ofrece el generador.

En conclusiones y trabajos futuros se exponen las impresiones obtenidas una vez concluido el proyecto sobre la realización del mismo y las áreas de trabajo futuras.

En la sección de anexos se exponen documentos de interés para la comprensión del proyecto.

2. ESTADO DEL ARTE

Este capítulo está compuesto del marco conceptual y de la descripción de los trabajos que abordan la misma temática del presente trabajo.

2.1. Proceso de negocio

Un proceso de negocio se define como una colección de actividades que toman una o más clases de entradas y crean una salida que es de valor para el cliente [2]. Las organizaciones basan su operación en los procesos de negocio, y es la clara definición, modelamiento, documentación y gestión de los procesos de negocio lo que brinda a las organizaciones la posibilidad de evaluar el desempeño de los procesos de negocio, y realizar los ajustes necesarios por medio de la reingeniería o rediseño de los procesos de negocio para mantener una mejora continua de la operación de la organización.

En JIRA podemos gestionar procesos de negocio personalizando los roles, permisos, flujos de trabajo, notificaciones y pantallas que intervienen en cada proceso.

El proceso de generación automática de configuración de procesos en la herramienta JIRA construido en este trabajo, tiene como objetivo facilitar la implementación y mantenimiento de la gestión de procesos con un alto nivel de calidad, en tiempos reducidos de implementación y a un bajo coste.

2.2. Abstracción y modelos

La abstracción es una característica preponderante en el desarrollo de software que proporciona la posibilidad de aislar los elementos de un contexto determinado para enfocar el propósito de donde se extrae, restando importancia en este punto a la forma de cómo hacerlo e invertir los esfuerzos en el qué hacer. Esta conceptualización se encuentra ligada al proceso implementado con la generación de modelos que se enfocan en aplicar una capa de abstracción superior que generaliza una solución aceptable dentro de un problema específico planteado. En la Figura 1 se muestra de forma gráfica este proceso de modelado elevando el nivel de abstracción de una solución puntual y tomando características conceptuales.

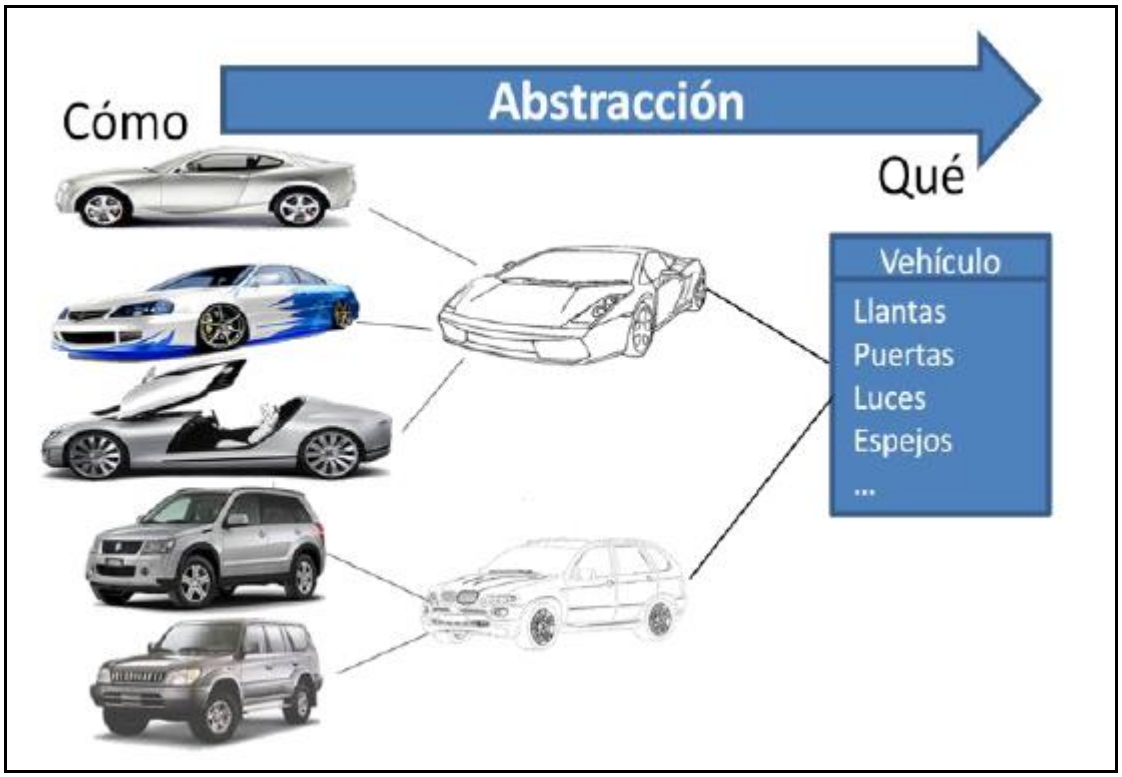


Figura 1: Proceso de abstracción de modelos

En el contexto de JIRA podemos abstraer la configuración de procesos en JIRA donde el proceso tendrá las siguientes características:

- Roles
- Tipos de ticket
- Flujos de trabajo
- Pantallas
- Permisos
- Notificaciones

2.3. Model-Driven Development (MDD)

El desarrollo de software dirigido por modelos (MDD), consiste en la producción de artefactos software a partir de una representación a escala de una solución general, realizándolo a través de la abstracción que se lleva implícito en el concepto de modelo y metamodelo, permitiendo ocultar características a cada nivel de abstracción superior para que la solución de un problema sea más sencillo para los actores; cuanto más complicado es el problema, surge la necesidad de producir más niveles de abstracción y al momento de la concretización de los mismos bajar cada uno de estos niveles para llegar a la generación de un producto terminado, que en este caso corresponde a la configuración de procesos en JIRA; en este entorno se encuentran metas a ser cumplidas por MDD, de las cuales se seleccionan a continuación las pertinentes a este trabajo [3].

2.3.1. Objetivos

A continuación se encuentran los principales objetivos o fines que se intenta conseguir con la metodología MDD [3]:

- **Velocidad en el desarrollo/configuración**: Este objetivo se plantea a través de la generación de configuraciones de procesos JIRA de forma automática, el cual reduce el tiempo en configurar un proceso.
- **Calidad**: Las configuraciones son generadas de forma automática a partir de modelos formales lo que disminuye los errores humanos.
- **Evitar la redundancia**: Se evita la configuración de procesos varias veces debido a que son generados y distribuidos para las implementaciones específicas.
- **Manejabilidad de los cambios tecnológicos**: El metamodelo y la herramienta siempre persisten, lo que cambian son las plantillas para la generación de configuraciones a una versión de JIRA específica.
- **Reusabilidad**: Las configuraciones que se generan pueden ser reutilizadas para la construcción de varios procesos.
- **Gestión de la complejidad con abstracción**: Se aplica la abstracción para que las soluciones obedezcan a metamodelos sencillos.
- **Entorno productivo**: Las configuraciones son escritas una sola vez. Es un solo punto de entrada.
- **Interoperabilidad**: Los metamodelos no obedecen a la tecnología sino a la lógica de negocio.
- **Portabilidad**: Las configuraciones pueden ser portadas de forma sencilla.

De los objetivos listados anteriormente en este proyecto tomamos como referencia la velocidad en la configuración, la calidad del software, gestión de la complejidad y el entorno productivo. Dentro de las buenas prácticas en la ingeniería de software se presenta la utilización del principio de abstracción, permitir que la computadora haga el trabajo y escribir programas para las personas y no para las computadoras.

2.3.2. Ventajas de trabajar con MDD

El no estar direccionado inicialmente a usar una plataforma, arquitectura de hardware y ninguna tecnología específica, hace de MDD un horizonte atractivo al desarrollador. La complejidad en las aplicaciones actuales hace del trabajo del desarrollador y de la ingeniería de software un campo de mucha investigación. Para alcanzar a suplir estos inconvenientes y producir a la vez software con calidad, se han tomado rumbos distintos para encontrar la solución, y todos han llegado a un punto común. Este punto en común es la elevación en el nivel de abstracción de los problemas para las soluciones informáticas, lo que redunda para proyectos actuales en la aplicación de MDD para la producción de sistemas informáticos con mantenibilidad, portabilidad y escalabilidad, lo que refleja calidad de software.

2.4. Metodologías basadas en MDD

En la actualidad los trabajos del desarrollo dirigido por modelos siguen dos enfoques de trabajo denominados metodologías, conocidas como Arquitectura Conducida por Modelos (MDA) y el Modelado sobre Dominios Específicos (DSM). La primera MDA, es una iniciativa liderada por la Object Management Group (OMG) que busca definir estándares que permitan la utilización de modelos para conducir el proceso de desarrollo de software en todas sus etapas. Su principal característica es la producción de artefactos basados en abstracción presentada gráficamente partiendo de elementos formales descritos con diagramas UML entre otros diagramas. DSM trabaja los conceptos de dominio basados en la utilización de modelos que hagan la representación de uno o varios dominios descritos en una misma organización, tecnología o arquitectura.

2.4.1. Model-Driven Architecture (MDA)

En la arquitectura dirigida por modelos es necesario tener en cuenta algunas etapas evolutivas o iteraciones básicas que se muestran en la Figura 2 [4].

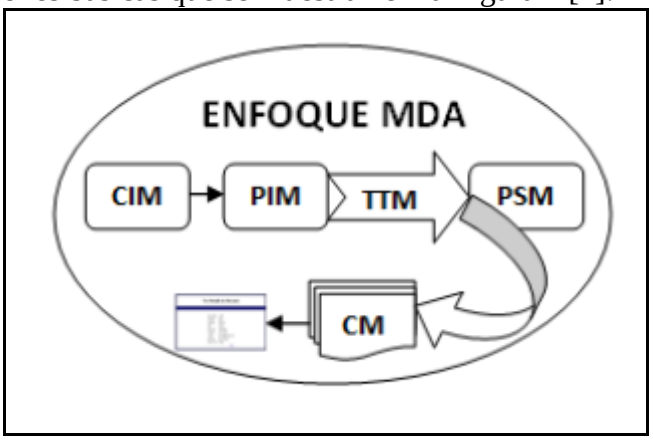


Figura 2: Pasos MDA en el proceso de desarrollo de software

Modelo Independiente de la Computación (CIM): Éste modelo expresa las necesidades del negocio de una manera general y poco técnica donde se presenta el conocimiento de la problemática que tiene el usuario final o dueño del producto [4].

Modelo Independiente de la Plataforma (PIM): En este punto el nivel de abstracción es alto, se constituye como la base para la implementación de una aplicación, debe estar construido de manera que defina la solución con una semántica precisa y formal, llevando la abstracción a tal punto que puedan ser implementados en varias plataformas [5], [6].

Tecnologías de transformación de modelos (TTM): Para pasar de un modelo a otro, es necesario la utilización de una tecnología capaz de convertirlo en un nuevo modelo, con un nivel de abstracción mayor o menor según el caso, al igual que el paso al código fuente; ésta labor la realizan las TTM dando al enfoque MDD la trazabilidad que necesita para bajar y subir en nivel abstracción en las soluciones planteadas [4].

Modelo Específico de la Plataforma (PSM): Las tareas técnicas en el manejo de lenguajes de programación: posibles máquinas virtuales, sistemas operativos, hardware e interconexiones entre otros se ven involucradas en estos modelos [4], [5], ellos pueden superar en número a los PIM, pero cada uno debe estar basado en por lo menos un PIM y trabajar de forma colaborativa para dar una solución completa.

Modelo del código (CM): Consiste en el resultado del ejercicio de la transformación de modelos más abstractos hasta llegar a la ausencia de modelos, es conocido también como el código generado [4].

En la actualidad no se ha llegado al grado de madurez de los modelos para descartar la importancia del código fuente [5], es por ello que se encuentra el proceso de depuración de posibles errores; aunque teóricamente la producción de éste código es de forma automática, debe presentar una compilación para llegar a un producto funcional que no debió necesitar la intervención humana para corregir, modificar o complementar el código fuente [4], [5].

Las características descritas anteriormente pertenecen a la primera metodología MDA, aunque alguna de ellas sean aplicables intrínsecamente también en la metodología DSM la cual concretiza su trabajo basado directamente en una especificación del dominio.

2.4.2. Domain-Specific Modeling (DSM)

El trabajo con DSM está enfocado en elevar en uno o varios niveles de abstracción una solución, presentando un metamodelo que describe las necesidades globales. El modelo con las respectivas transformaciones a través de las plantillas es capaz de generar artefactos de forma automática [3]. Puntualmente DSM se basa en la implementación de tres pasos que son:

- **Identificación y construcción de un metamodelo para el dominio específico***:* se realiza el estudio del dominio y se plantean los posibles casos generalizados, se describe la gramática del lenguaje y los elementos necesarios para obtener la generalidad de aplicaciones del mismo estilo que compartan el punto de entrada.
- **Construcción de las plantillas de generación de código***:* Estas plantillas permiten obtener el código para las plataformas elegidas según la información suministrada como punto de entrada en la herramienta.
- **Elaboración y configuración de las validaciones para el lenguaje a generar***:* Estas son las encargadas de asegurar que lo que se escribe en el lenguaje de dominio específico construido obedece a la gramática y a las reglas del negocio configuradas. Veamos la descripción de las características más relevantes de cada enfoque, de donde se obtiene referencia para la selección de uno de ellos.

2.4.3. Metodología seleccionada dentro de MDD

En el presente trabajo de investigación se encaminan los esfuerzos para la generación automática de configuraciones de procesos en JIRA mediante la metodología DSM dentro del desarrollo dirigido por modelos. Esta selección se hace basada en las siguientes características:

- **Mejorar la productividad de los desarrolladores mediante el trabajo en un dominio de problema conocido**: Apoyo a la industria del software, y a la apropiación de los conocimientos actuales.
- **El dominio del problema es estable con solución arquitectónica bien definida**: Se evitan errores en la codificación y se presenta un lenguaje limitado y estructurado.
- **Fortalecer el desarrollo de herramientas que trabajan bajo la metodología DSM**: Se desarrolla una nueva herramienta en esta metodología.
- **Se pueden aplicar esquemas para llegar a usuarios expertos, aprendices y solo con conocimientos del dominio**: Se hace más estrecha la relación software – experto en el negocio proporcionando soluciones más especializadas.

De las características anteriores se enfoca este trabajo en un lenguaje de dominio específico, para lo cual se describen las bondades presentes al desarrollar software bajo MDD en concordancia a la metodología DSM.

2.5. Domain-Specific Language (DSL)

Un Lenguaje Específico de Dominio (DSL) es un conjunto reducido de construcciones y operaciones que brindan una mayor expresividad y optimización para un dominio particular. Según [7] un DSL es “la última abstracción”, que captura precisamente la semántica de un dominio de aplicación. Algunos DSL bien conocidos incluyen SQL y expresiones regulares. Claramente cada uno es mejor que un lenguaje de propósito general para representar operaciones sobre, base de datos y cadenas respectivamente, pero no sucede lo mismo cuando se desea describir soluciones fuera de su dominio. Algunas industrias poseen también sus propios DSL. Por ejemplo, en telecomunicaciones, los lenguajes de descripción de llamadas son ampliamente utilizados para especificar la secuencia de estados en una llamada telefónica.

2.5.1. Utilización

Los usuarios de los DSL crean modelos que luego se compilan o se traducen a otros artefactos. Es común hallar DSL cuyo uso es generar código de programa en otro lenguaje o un ejecutable, pero también son utilizados para generar otros artefactos como un esquema de visualización para ciertos datos. Cuando se define un DSL, se pueden definir plantillas que lean un modelo del DSL y generen ejecutables, fuentes de otros lenguajes, archivos de texto u otros artefactos. Generalmente, los DSL son creados cuando un grupo de usuarios (no necesariamente desarrolladores) tienen que generar código similar para varios productos. En nuestro caso, diseñaremos un DSL para la

configuración de procesos en JIRA con el cual podremos generar automáticamente esas configuraciones en la herramienta. El principal beneficio de los DSL, es que pueden ser más sencillos de entender por los administradores JIRA y por los usuarios que trabajarán con el proceso generado por el DSL, ya que se manejan conceptos en términos del dominio. Además las configuraciones son más confiables, gracias a la automatización y refactorización. En un DSL, la representación es más sencilla, sólo se describe cómo resolver el problema en términos del dominio y no hay construcciones especiales de código para poder completar el algoritmo como ocurriría en un Lenguaje de propósito general GPL. Gracias a esto, es más sencillo también introducir cambios en la configuración si hay cambios en la especificación del proceso. A su vez, quizás no sea necesario un administrador JIRA para utilizar el DSL, ya que estará en términos del dominio y cualquier usuario del entorno debería manejar. La habilitación del experto en el dominio aumenta en gran medida la eficiencia del mantenimiento de una aplicación a medida que hay cambios en las especificaciones.

2.5.2. DSL dentro del marco de trabajo MDD

Trabajar los DSL enmarcados en MDD brinda como resultado ventajas para los desarrolladores de software, de las cuales se presentan las más relevantes a continuación:

- La mayoría de los problemas en la industria del software están dados en la comunicación existente entre el experto en el negocio y los encargados de las implementaciones técnicas; este proceso se lleva de forma manual donde se pueden presentar errores, si se usa un DSL que implemente la abstracción adecuada y el uso de los términos propios del dominio puede directamente interactuar con el experto del negocio.
- Cuando se trabaja con DSL estamos garantizando la portabilidad, mantenibilidad y reusabilidad de aplicaciones, lo que permite terminar con el caos que causan los avances y cambios en la tecnología que obligan a las aplicaciones a ser actualizadas o migradas generando altos costos en sistemas los cuales se ven reflejados en tiempo y dinero.
- El nivel de abstracción en MDD es alto lo que permite solucionar el problema una vez en el modelo más abstracto, y luego a partir de transformaciones se produce de forma automática el código, con mayor productividad del equipo que provee las soluciones.
- Trabajar con un DSL brinda seguridad para que el desarrollador realice sus tareas estando limitado a usar las características de su dominio y no tiene que manejar estructuras complejas de un lenguaje de propósito general, aumentando así la calidad de las aplicaciones debido a que no es validado un posible error incluyendo sentencias o especificaciones no permitidas.

2.6. JIRA

JIRA es una aplicación de ticketing desarrollada por la compañía Atlassian y lanzada al mercado en octubre de 2004. Facilita la comunicación entre varios interlocutores, que generalmente son empresas y sus clientes. Cuando el cliente quiere hacer llegar una consulta, sugerencia o incidencia a la empresa, el sistema de tickets JIRA le permite

crear un ticket con toda la información necesaria. Una vez que la empresa ha leído y respondido su petición o sugerencia se le comunica al cliente mediante una notificación vía email. Es una aplicación web que la empresa comercializa vendiendo los derechos de uso. El precio de los cuales depende del número de usuarios que vayan a usar la herramienta.

En JIRA podemos gestionar tareas que permiten a los equipos planificar, construir y finalizar proyectos. Además, permite capturar y organizar incidencias, asignar trabajo y hacer un seguimiento de la actividad del equipo.

Los principales tipos de entidades que maneja JIRA son:

- Project Categories
- Projects
- Components
- Issues
- Issue Types

En la siguiente figura observar el esquema de los principales tipos de entidades [12]:

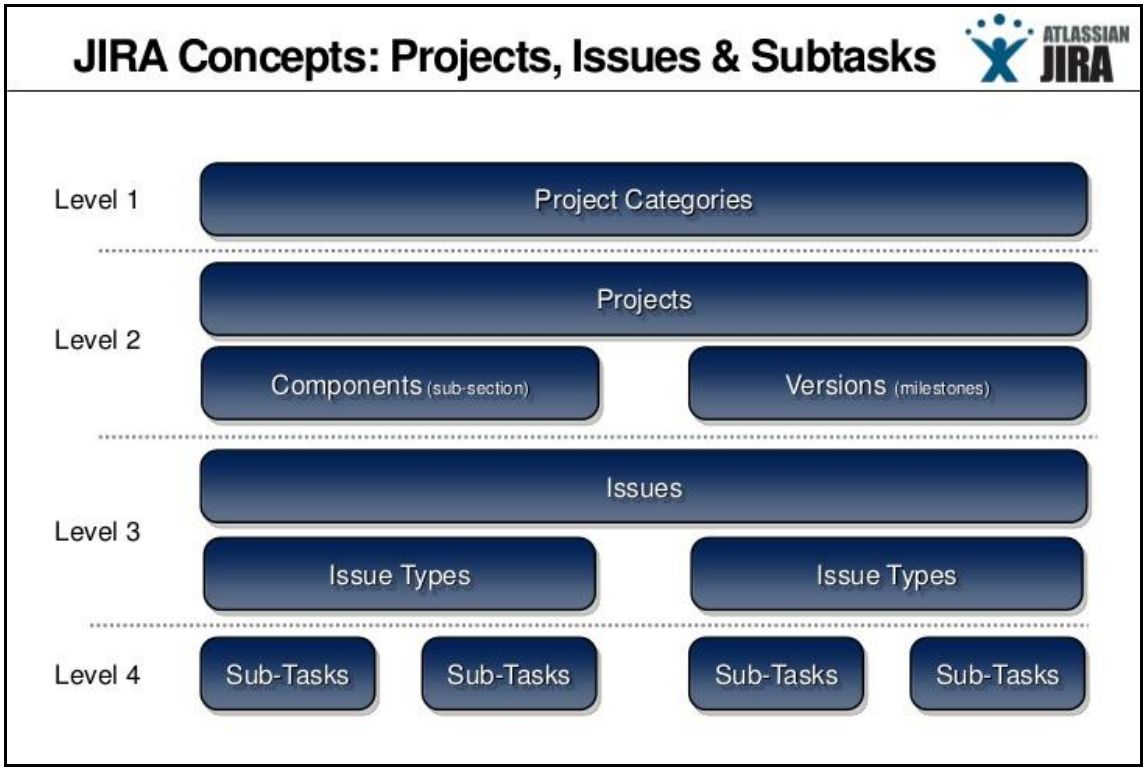


Figura 3: Principales conceptos y estructura JIRA

2.6.1. Funcionalidades

Las principales funcionalidades que ofrece la aplicación son:

- **Gestión de tareas**: los usuarios pueden crear tickets con toda la información necesaria, priorizar los tickets existentes, asignarlos entre los usuarios y

mantenerse al día de la evolución de los mismos. En la siguiente figura podemos observar el detalle de un ticket [11]:

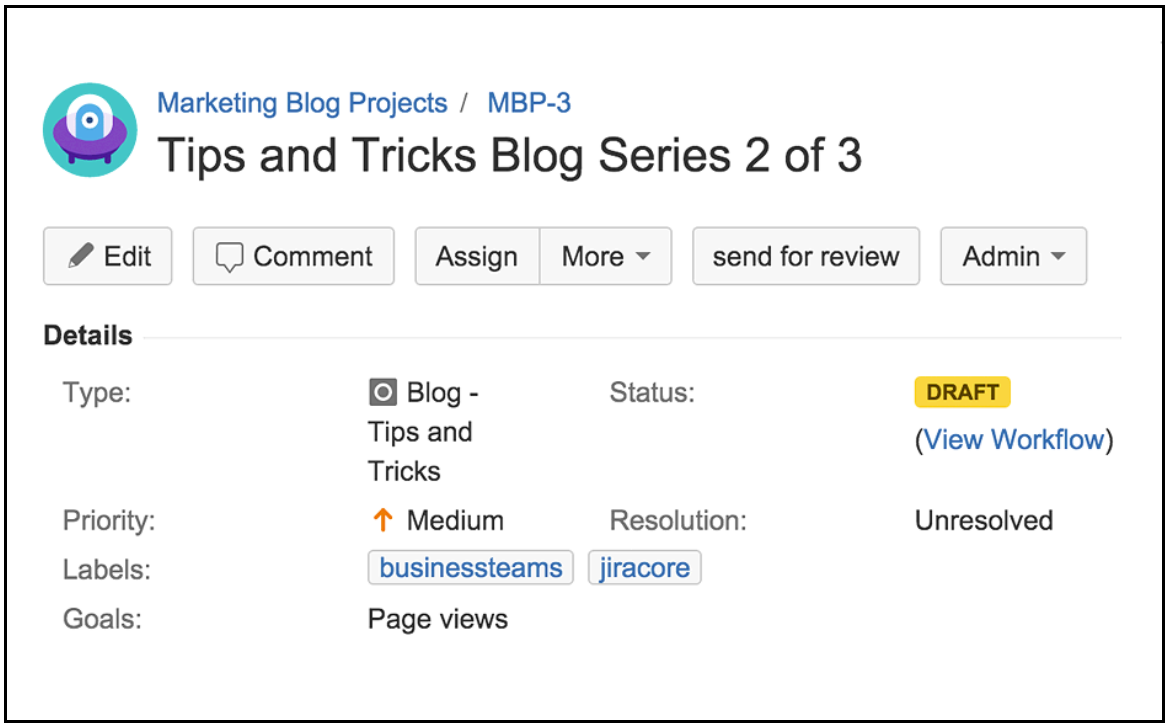


Figura 4: Pantalla JIRA de detalle de un ticket

- **Creación de flujos de trabajo**: JIRA proporciona flujos de trabajo predefinidos y permite crear nuevos para que los flujos de trabajo se adapten a las necesidades de cada organización [11]. A continuación podemos ver el esquema de un flujo de trabajo [11]:

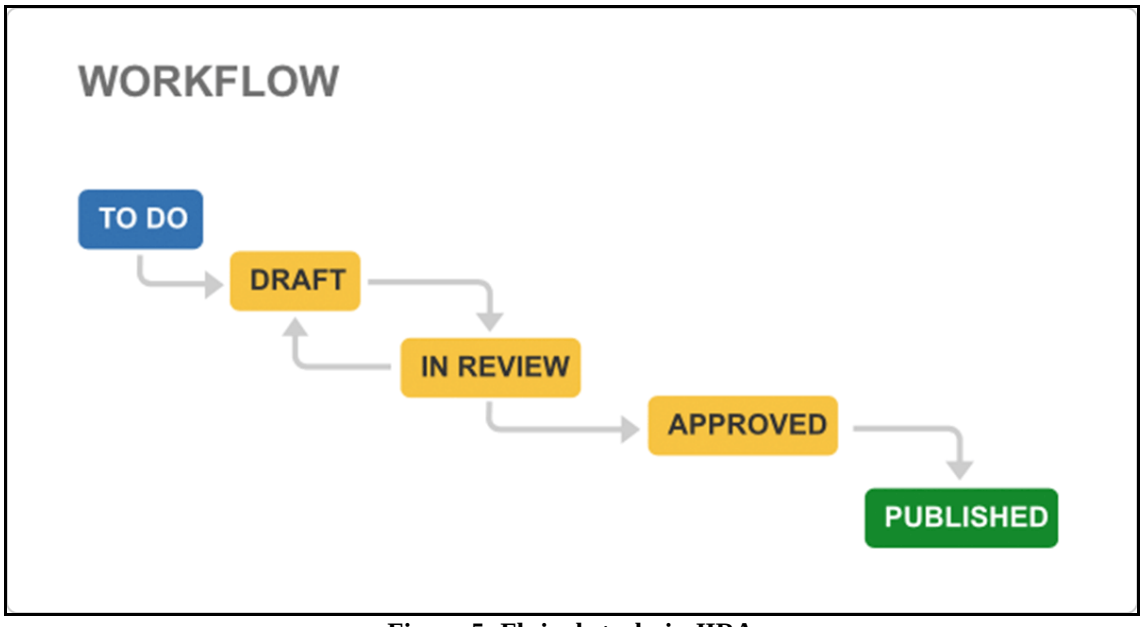


Figura 5: Flujo de trabajo JIRA

- **Planificación**: permite planificar el trabajo que se va a llevar a cabo, priorizando y asignando cada uno de los tickets. Además, se adapta a las metodologías ágiles de desarrollo [11].

- **Supervisa**: analiza y prioriza el trabajo del equipo en su contexto con una visibilidad completa [11].

- **Colaboración entre equipos y notificaciones**: facilita la colaboración entre los miembros de un equipo. Utiliza las menciones (mediante el símbolo @) para mantener a miembros concretos del equipo al tanto de lo que ocurre y mantente informado a través de notificaciones útiles y detalladas. Sabrás al instante si has recibido alguna asignación y cuándo hay que responder a los comentarios [11]. En la siguiente imagen podemos observar el detalle de comentarios de un ticket [11]:

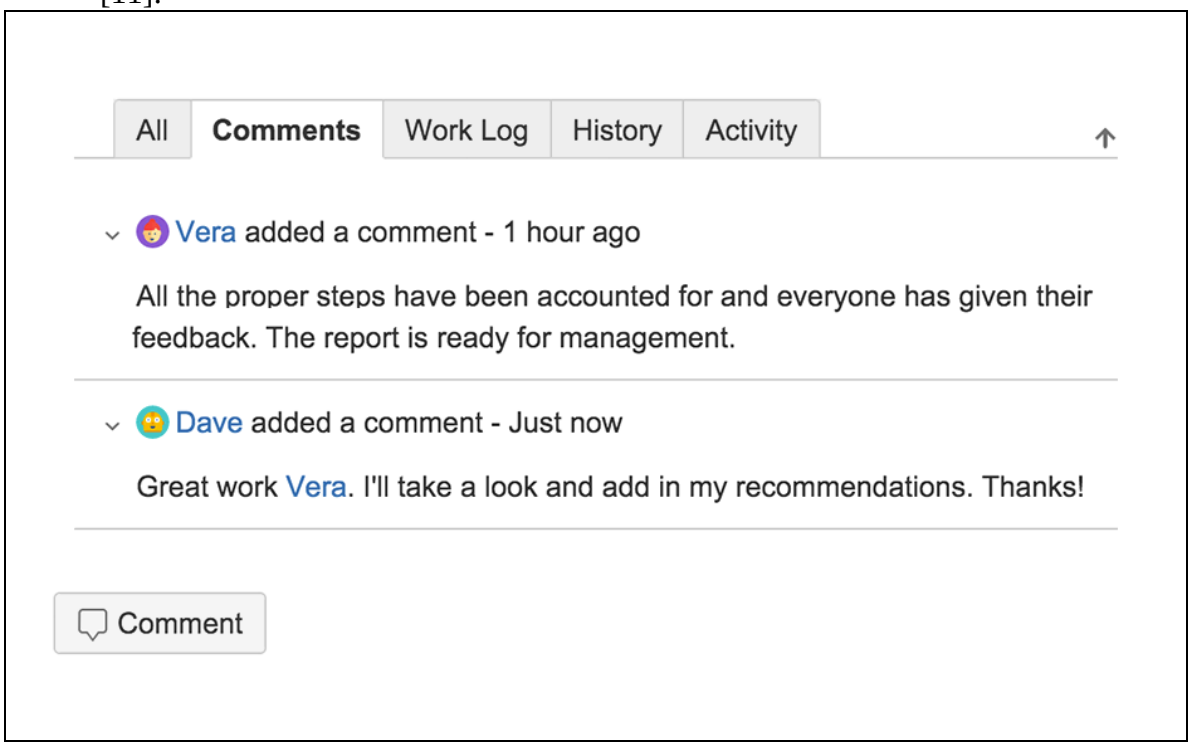


Figura 6: Pantalla de JIRA para añadir comentarios

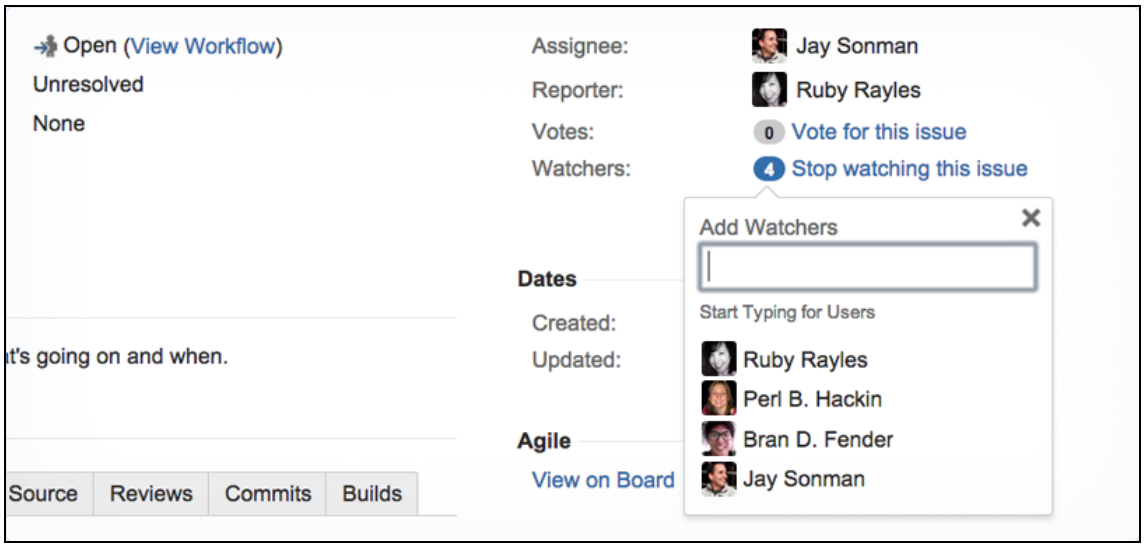


Figura 7: Pantalla de JIRA para gestionar la visibilidad y las notificaciones

- **Crea informes**: analiza absolutamente todos los datos con informes y cuadros de mandos que te ayudarán a entender cómo evoluciona tu equipo. Ajusta la configuración y haz que se reflejen los indicadores más relevantes [11].

- **Gestión de proyectos**: se puede gestionar todo lo relacionado con un proyecto. Además, existe la posibilidad de gestionar proyectos que quieran usar metodologías ágiles o de tipo Service Desk.

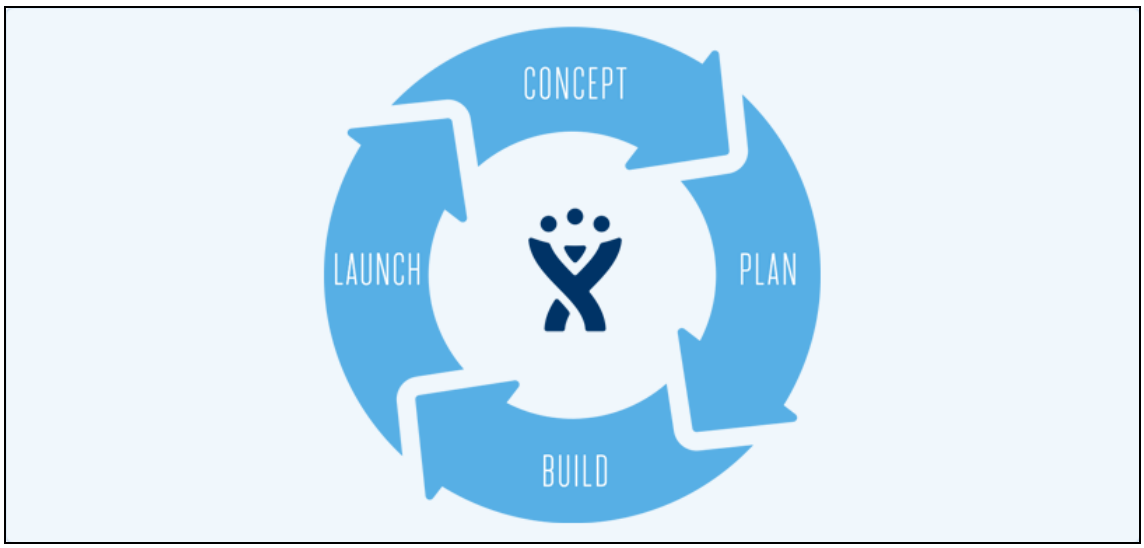


Figura 8: Esquema de gestión de proyectos según JIRA

- **Posibilidad de importar datos de otras herramientas de ticketing**: esta funcionalidad es útil si antes de implantar JIRA se ha usado otra herramienta de ticketing. De esta forma, la organización no pierde los datos anteriores [13].

Figura 9: Principales herramientas desde las que se pueden importar datos a JIRA

2.6.2. Administración de JIRA

Para adaptar las funcionalidades descritas en el apartado anterior a las necesidades de cada organización JIRA ofrece un menú de administración donde los administradores JIRA pueden moldear la configuración de cada proceso que se desee gestionar [14]. Es en este punto donde se centra el proceso de generación automática de configuraciones JIRA objeto de este trabajo. El proceso de generación automática generará cada una de las configuraciones necesarias para gestionar un proceso en JIRA. Estas configuraciones las describimos a continuación.

- **Administración de tipos de ticket:** En este apartado podemos dar de alta los tipos de ticket que se gestionarán en nuestro proceso.
- **Administración de flujos de trabajo:** En este apartado podemos configurar el flujo de trabajo que utilizará cada tipo de ticket. Podremos especificar los estados de este, así como sus transiciones (cambios de estado).
- **Administración de pantallas:** En este apartado podemos configurar la información que contendrá cada ticket de un determinado tipo de ticket. Esta información se almacena en campos del ticket que tendrán un nombre y un valor.
- **Administración de roles:** En este apartado podemos configurar los roles que intervendrán en el proceso.
- **Administración de permisos:** En este apartado podemos configurar los permisos de cada rol, grupo de usuarios o usuario que intervenga.
- **Administración de notificaciones:** En este apartado podemos configurar las notificaciones que se enviarán a cada usuario que intervenga en el proceso.

La abstracción de esta información nos permitirá crear el DSL que utilizaremos para poder modelar configuraciones JIRA.

3. IMPLEMENTACIÓN DEL GENERADOR

Después de estudiar los conceptos básicos del trabajo con MDD, DSL y JIRA, se procede a realizar la implementación de la teoría en un caso de estudio específico, conocido como implementación de referencia. Con el desarrollo de esta herramienta se busca una aproximación a la generalidad de configuración de procesos que se puedan modelar y realizar de forma automática.

3.1. Implementación de referencia

Para la implementación de referencia nos centraremos en un caso de estudio puntual que constituye un esquema general que puede ser entendido y seguido fácilmente, lo que permite que esta investigación sea base para nuevos trabajos. En este caso de estudio se gestionarán tickets de tipo tarea y sub-tarea que podrán ser dados de alta en la herramienta y seguirán el flujo, permisos y pantallas que especificaremos.

3.2. Metodología a seguir en la elaboración de la herramienta

Para iniciar con la herramienta debemos describir la metodología a utilizar basada en DSM de MDD. El procedimiento de elaboración de la herramienta hace uso de la metodología DSM la cual se presenta en forma específica para esta investigación como se muestra en la Figura 10, la cual se sigue paso a paso en el desarrollo.

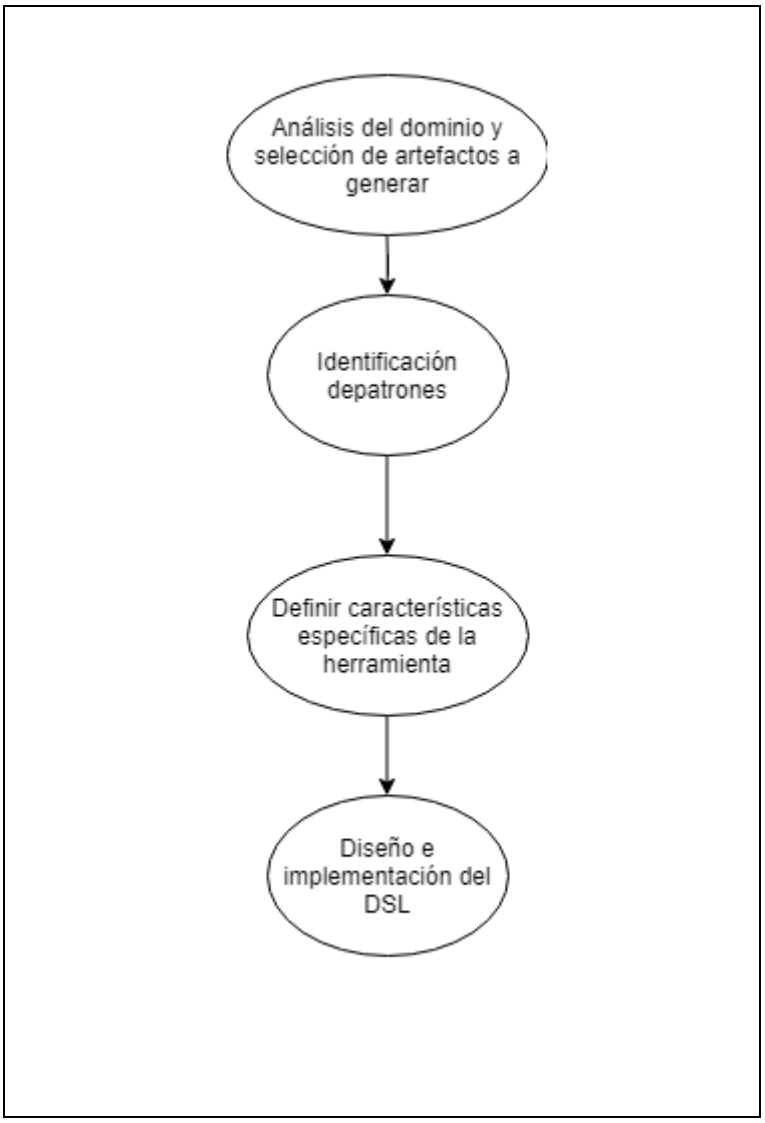


Figura 10: Metodología para la elaboración de la herramienta

3.2.1. Análisis del dominio del problema y selección de artefactos a generar

Para la implementación de referencia se pretenden generar los esquemas de configuración JIRA que corresponden a las características que se pueden modelar como una generalización del problema, para el cual se construye un metamodelo común donde cambien aspectos muy específicos para cada lógica de proceso a configurar. Para ello vamos a realizar una abstracción de las configuraciones de procesos en JIRA tal y como se ilustra en la Figura 11:

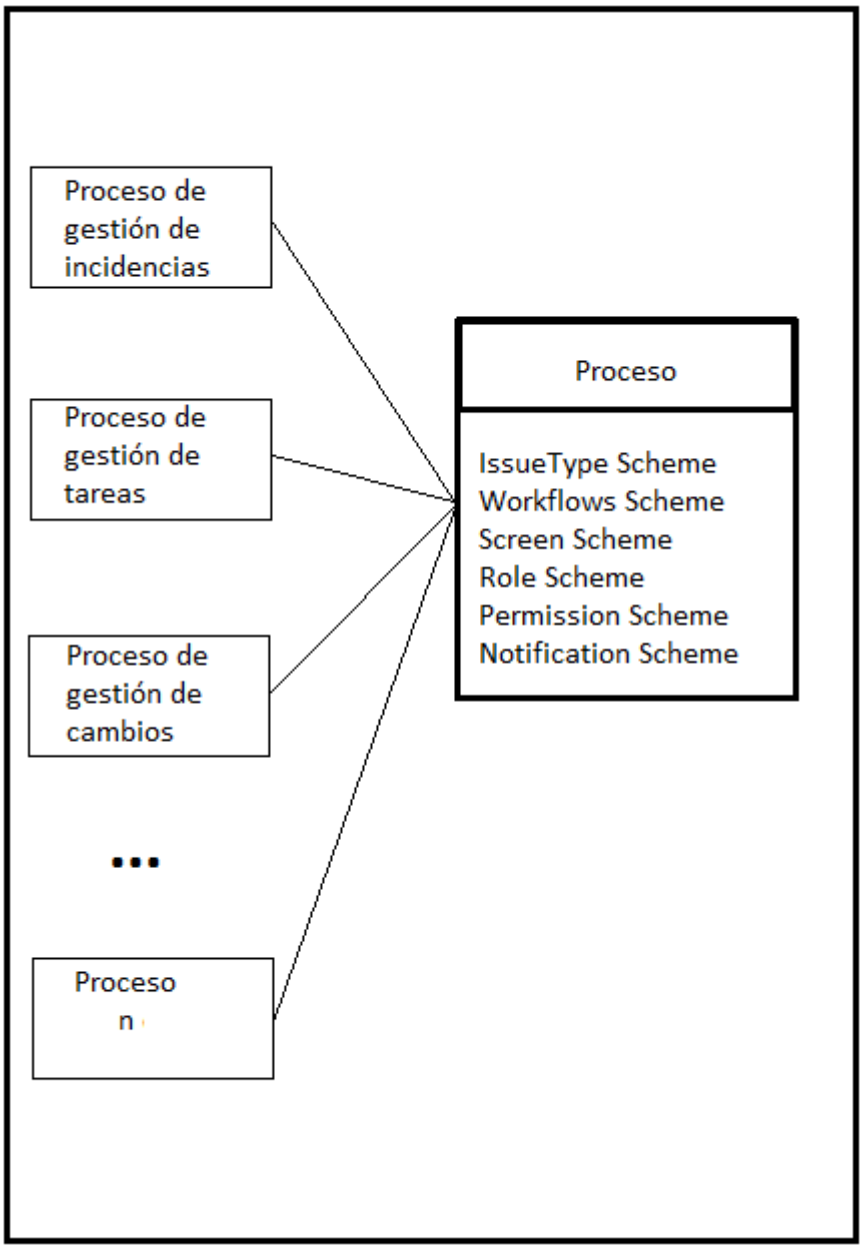


Figura 11: Abstracción de las configuraciones de procesos en JIRA

De la figura anterior podemos concluir que la configuración de un proceso en JIRA viene definida por los 6 esquemas siguientes:

- **IssueType Scheme**: este esquema define las tipologías de tickets que se podrán utilizar. Ejemplos de tipologías son Tareas, Incidencias, Peticiones…
- **Workflow Scheme**: en el esquema de flujos se define para cada tipología el flujo de trabajo que seguirá el proceso con un conjunto de transiciones. Estas transiciones detallan los pasos que puede realizar el proceso entre un estado y otro.
- **Screen Scheme**: este esquema define para cada tipología que campos, y de qué tipo, se mostrarán en los formularios de creación, edición y vista del ticket.
- **Role Scheme**: el esquema de roles define los perfiles que interactuarán con el proceso, los grupos y usuarios que forman parte de cada rol.

- **Permission Scheme**: el esquema de permisos permite indicar que roles, usuarios o grupos podrán realizar cada una de las funciones que la aplicación permite realizar sobre los tickets.
- **Notification Scheme**: Este esquema permite indicar para cada uno de los 17 eventos que se pueden producir mientras se trabaja en la aplicación a que destinatarios se enviará una notificación. Estos destinatarios se pueden definir en forma de rol, grupo o usuario.

3.2.2. Identificación de patrones dentro de la implementación de referencia

Esta identificación se realiza basada en lo que indica la metodología DSM, e identificamos el código esquemático repetitivo, el genérico y el individual.

- El código esquemático se puede generar a partir de una serie de elementos bien definidos, en la implementación de referencia lo encontramos en los tipos de ticket (Issuetypes), flujos de trabajo (workflows), campos (Fields y Customfields), roles, permisos, eventos y destinatarios, por ejemplo todos los tipos de ticket tienen un flujos y se define de forma general, lo que la diferencia son los estados y transiciones entre estados definidos por el usuario final al momento de darle su función. A continuación, podemos observar el código esquemático que genera la configuración donde se especifican los tipos de ticket que se podrá utilizar:

```
23  public List<String> create(JSONObject objJSON) {
24      //Leemos del JSON de configuracion los issueTypes a configurar
25      JSONArray jsonIssueTypes                = objJSON.getJSONArray("issueTypes");
26      DefaultIssueTypeManager defaultIssueTypeManager     = ComponentAccessor.getComponentOfType(com.atlassian.jira.config.DefaultIssueTypeManager.class);
27      List<String> lIssueTypeIds              = new ArrayList<String>();
28
29      //For cada Issuetype del JSON introducido
30      for(int i = 0; i < jsonIssueTypes.length(); i++) {
31          JSONObject jsonIssueType    = jsonIssueTypes.getJSONObject(i);
32          IssueType oIssueType        = null;
33          boolean b                   = false;
34          //Buscamos en todos los Issuetypes dados de alta en JIRA si hay uno con el mismo nombre
35          for(IssueType n : defaultIssueTypeManager.getIssueTypes()) {
36              if(n.getName().equals(jsonIssueType.get("name"))) {
37                  oIssueType  = n;
38                  b           = true;
39                  break;
40              }
41          }
42
43          //Si no existe el Issuetype con el nombre indicado
44          if(!b) {
45              if(jsonIssueType.getBoolean("subtask")) {
46                  oIssueType = defaultIssueTypeManager.createSubTaskIssueType(jsonIssueType.getString("name"), null, 10004L);
47              }else {
48                  oIssueType = defaultIssueTypeManager.createIssueType(jsonIssueType.getString("name"), null, 10004L);
49              }
50              lIssueTypeIds.add(oIssueType.getId());
51          }else {
52              lIssueTypeIds.add(oIssueType.getId());
53
54          }
55          //result += jsonIssueType.getString("name")+" : "+jsonIssueType.getBoolean("subtask")+"\n";
56      }
57
58      return lIssueTypeIds;
59  }
60
61  public void assignScheme(Project project, JSONObject objJSON, List<String> lIssueTypeIds) {
62      //Creamos el issueTypeScheme
63      FieldConfigScheme fieldConfigScheme = ComponentAccessor.getIssueTypeSchemeManager().create(objJSON.getString("key")+" IssueTypeScheme", null, lIssueTypeIds);
64      Long[] l2                           = new Long[1];
65      l2[0]                               = project.getId();
66      List<JiraContextNode> l             = CustomFieldUtils.buildJiraIssueContexts(false,  l2, ComponentAccessor.getProjectManager());
67      ComponentAccessor.getFieldConfigSchemeManager().removeSchemeAssociation(l, ComponentAccessor.getFieldManager().getIssueTypeField());
68
69      List<Long> lProjects = new ArrayList<Long>();
70      lProjects.add(project.getId());
71
72      ComponentAccessor.getFieldConfigSchemeManager().updateFieldConfigScheme(fieldConfigScheme, l, ComponentAccessor.getFieldManager().getIssueTypeField());
73      ComponentAccessor.getFieldManager().refresh();
74  }
```

Figura 12: Código esquemático generación issuetypes

- El código genérico es el que se identifica como base para la construcción de un archivo específico y no cambia siempre es el mismo para cualquier implementación, por ejemplo para crear una transición entre estados de flujo

siempre se deben incluir las 5 funciones de flujo que garantizan la consistencia de los datos en la aplicación. En la siguiente imagen podemos ver el código que genera esas 5 funciones:

```
    ResultDescriptor re2                    = DescriptorFactory.getFactory().createResultDescriptor();

    FunctionDescriptor function1            = DescriptorFactory.getFactory().createFunctionDescriptor();
    function1.setId(0);
    function1.setType("class");
    function1.setEntityId(0);
    function1.getArgs().put("full.module.key", "com.atlassian.jira.plugin.system.workflowupdateissuestatus-function");
    function1.getArgs().put("class.name", "UpdateIssueStatusFunction");
    re2.getPostFunctions().add(function1);

    FunctionDescriptor function2            = DescriptorFactory.getFactory().createFunctionDescriptor();
    function2.setId(0);
    function2.setType("class");
    function2.setEntityId(0);
    function2.getArgs().put("full.module.key", "com.atlassian.jira.plugin.system.workflowcreatecomment-function");
    function2.getArgs().put("class.name", "com.atlassian.jira.workflow.function.misc.CreateCommentFunction");
    re2.getPostFunctions().add(function2);

    FunctionDescriptor function3            = DescriptorFactory.getFactory().createFunctionDescriptor();
    function3.setId(0);
    function3.setType("class");
    function3.setEntityId(0);
    function3.getArgs().put("full.module.key", "com.atlassian.jira.plugin.system.workflowgeneratechangehistory-function");
    function3.getArgs().put("class.name", "com.atlassian.jira.workflow.function.issue.GenerateChangeHistoryFunction");
    re2.getPostFunctions().add(function3);

    FunctionDescriptor function4            = DescriptorFactory.getFactory().createFunctionDescriptor();
    function4.setId(0);
    function4.setType("class");
    function4.setEntityId(0);
    function4.getArgs().put("full.module.key", "com.atlassian.jira.plugin.system.workflowreindexissue-function");
    function4.getArgs().put("class.name", "com.atlassian.jira.workflow.function.issue.IssueReindexFunction");
    re2.getPostFunctions().add(function4);

    FunctionDescriptor function5            = DescriptorFactory.getFactory().createFunctionDescriptor();
    function5.setId(0);
    function5.setType("class");
    function5.setEntityId(0);
    function5.getArgs().put("full.module.key", "com.atlassian.jira.plugin.system.workflowfireevent-function");
    function5.getArgs().put("class.name", "com.atlassian.jira.workflow.function.event.FireIssueEventFunction");
    function5.getArgs().put("parameter", "13");
    re2.getPostFunctions().add(function5);
```

Figura 13: Código genérico que genera las 5 funciones de una transición

- El código individual es aquel que no puede ser generado fácilmente, en su mayor porcentaje está ligado a la lógica de negocio del proceso a gestionar y es el que debe ser incluido por un programador para lograr la completitud del código necesario en la ejecución correcta. En JIRA este código suele encontrarse en scripts Groovy [14] y [15] que permiten realizar automatizaciones personalizadas a las necesidades. El siguiente ejemplo podemos ver el código de un script que realiza la asignación automática de un ticket cuando este realiza una acción de flujo:

```
//Imports
import com.atlassian.jira.component.ComponentAccessor
import com.atlassian.jira.config.util.JiraHome
import com.atlassian.jira.event.type.EventDispatchOption
import com.atlassian.jira.issue.CustomFieldManager
import com.atlassian.jira.issue.IssueManager
import com.atlassian.jira.issue.MutableIssue
import com.atlassian.jira.issue.fields.CustomField
import com.atlassian.jira.user.ApplicationUser
import com.atlassian.jira.user.util.UserManager


IssueManager issueManager           = ComponentAccessor.getIssueManager();
UserManager userManager             = ComponentAccessor.getUserManager();
MutableIssue mIssue                 = (MutableIssue) issue;

try{


    CustomFieldManager cfManager    = ComponentAccessor.getCustomFieldManager();

    CustomField tecnico             = cfManager.getCustomFieldObject(new Long(12000)); //Campo Tecnico
    String tec                      = mIssue.getCustomFieldValue(tecnico).toString();

    ApplicationUser usr = userManager.getUserByName(tec);   //Recuperamos el usuario del tecnico
    if(usr != null){
        mIssue.setAssigneeId(userManager.getUserByName(tec).getKey());  //Asignamos al ticket el usuario del tecnico
    }
    //Actualizamos el ticket y reindexamos
    ComponentAccessor.getIssueManager().updateIssue(ComponentAccessor.getJiraAuthenticationContext().getLoggedInUser(), mIssue, EventI
}catch (Exception e){
    //
}
```

Figura 14: Código individual que asigna automáticamente un ticket

3.2.3. Características de la herramienta y arquitectura de la solución

En este punto ya se han identificado los artefactos susceptibles de ser generados. Ahora se pasa a la forma en cómo hacerlo. Ésta forma se conoce como la determinación del metamodelo informalmente, el cual plantea la arquitectura a seguir a grandes rasgos para la elaboración del generador. En la Figura 15 se muestran los pasos a realizar que no necesariamente deben ser idénticos a la solución final del metamodelo concreto, pero si una aproximación basada en el análisis de la implementación de referencia y lo que se desea lograr con el DSL.

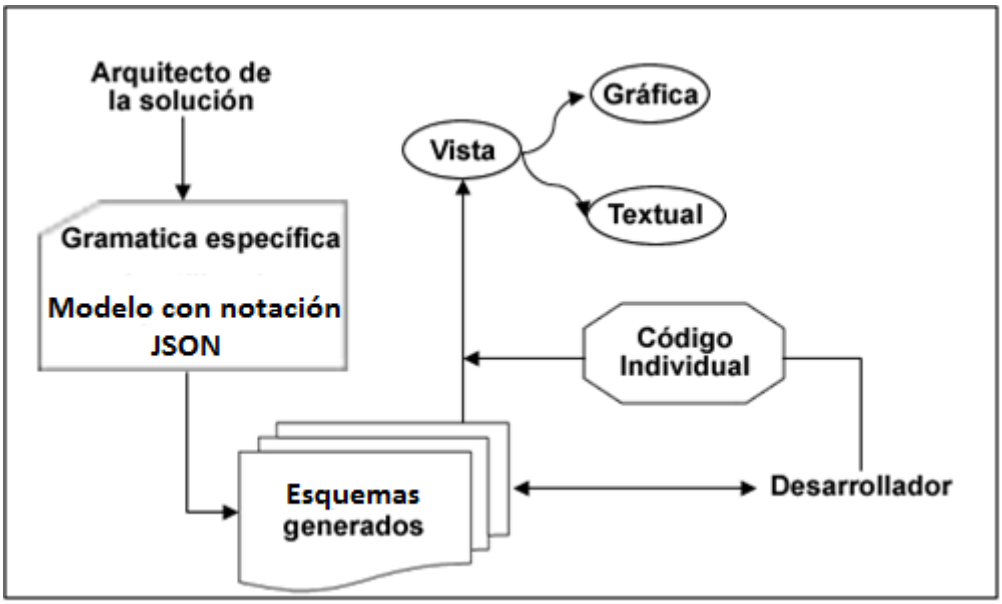


Figura 15: Solución planteada informalmente

Los pasos son los siguientes:

- Elaboración del fichero de configuración JSON
- Ejecución del generador de configuraciones JIRA
- Aplicación del código individual por parte de los desarrolladores
- Ver la configuración generada de forma gráfica o textual

En esta aproximación, partimos del análisis de los requerimientos de la organización que el arquitecto de la solución y el administrador JIRA plasmaran en notación JSON siguiendo la gramática especificada del DSL. Una vez construido el proceso de generación automática de configuraciones JIRA leerá dicho fichero y generará cada uno de los esquemas de los que está formado una configuración. Cuando la configuración esté generada, los desarrolladores podrán aplicar el código individual de las automatizaciones particulares del proceso. Finalmente, podremos ver en los menús de administración de JIRA, la configuración generada, de forma gráfica. Para poder obtener de forma instantánea y en formato textual la configuración de un proceso, el generador ofrecerá la opción de extraer dicha configuración en JSON.

3.2.4. Diseño e implementación del Lenguaje de Dominio Específico

Al determinar el metamodelo informal y la clasificación del código de la implementación de referencia, se procede a construir el metamodelo que se muestra en la Figura 16.

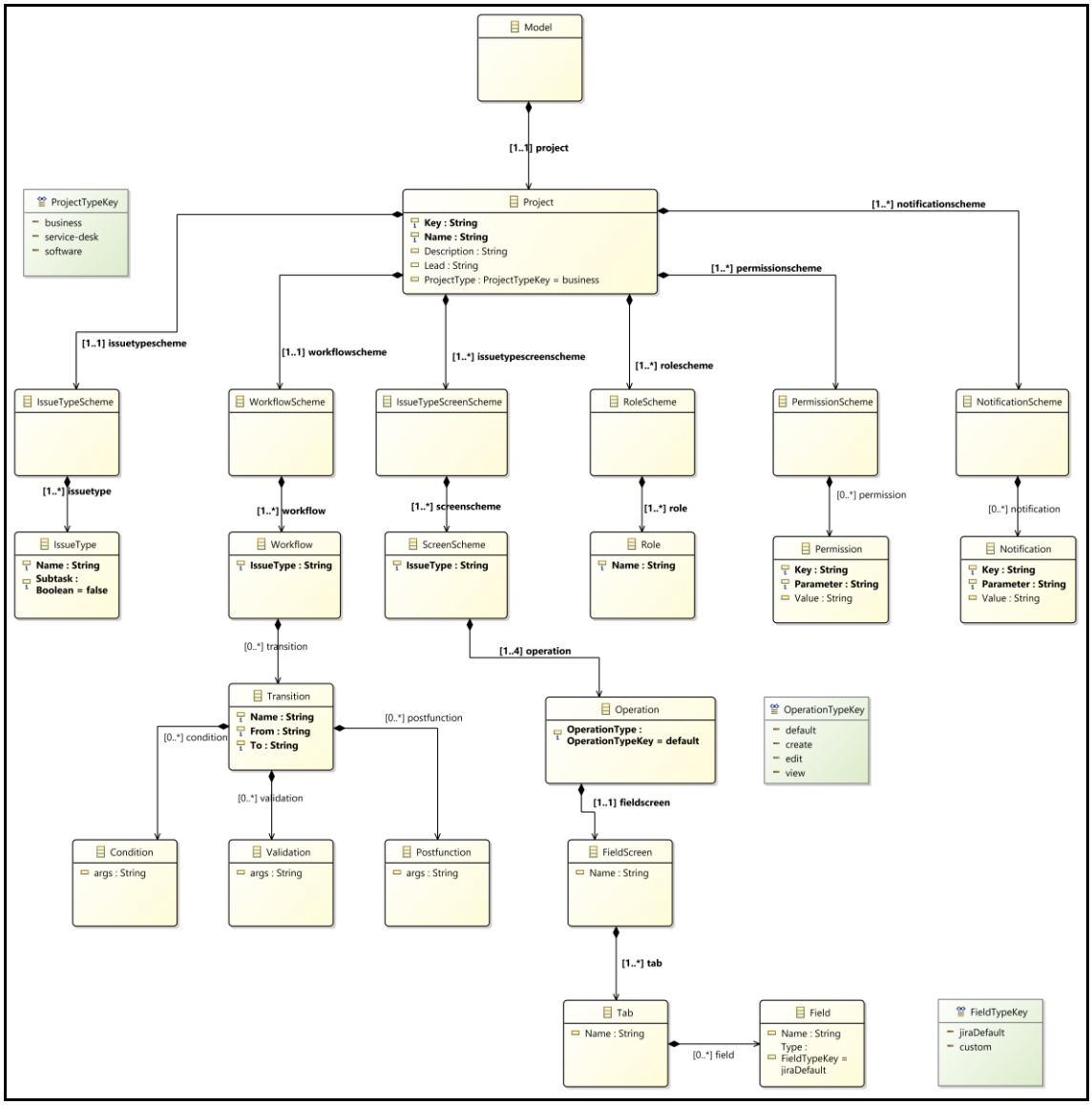


Figura 16: Metamodelo formalmente definido

Esta definición nos permite tener una gramática definida que seguirá la herramienta, el usuario del DSL puede crear un modelo que debe contener un proyecto y este debe contener cada uno de los 6 esquemas requeridos:

- **IssueTypeScheme**: este esquema define las tipologías de tickets que se podrán utilizar en la aplicación. Al menos una tipología debe estar definida. Ejemplos de tipologías son Tareas, Incidencias, Peticiones…
    - **IssueType**: una tipo de ticket tendrá un nombre e indicará si es o no una sub-tarea.

- **WorkflowScheme**: en el esquema de flujos se define para cada tipología el flujo de trabajo que seguirá el proceso con un conjunto de transiciones. Al menos un flujo de trabajo debe ser definido.
    - **Workflow**: los workflows se caracterizan por un issuetype y un 0 o más transiciones.
        - **Transition**: estas transiciones detallan los pasos que puede realizar el proceso entre un estado y otro. Además en cada transición podemos indicar condiciones, validaciones y post-funciones.
            - **Condiciones**: se ejecutan antes realizar la acción de cambio de estado y permite restringir la ejecución del cambio de estado a ciertos roles y grupos.
            - **Validaciones**: se ejecutan en el momento de realizar la acción de cambio de estado y permite validar que los valores de los campos del ticket cumplen unos criterios.
            - **Postfunciones**: se ejecutan en el momento que ya se ha realizado la acción de cambio de estado y permite informar los valores de los campos del ticket.
- **IssueTypeScreenScheme**: este esquema define para cada tipología que campos, y de qué tipo, se mostrarán en los formularios de creación, edición y vista del ticket mediante los ScreenScheme.
    - **ScreenScheme**: cada ScreenScheme tendrá asociado una tipología y un conjunto de operaciones.
        - **Operation**: se caracteriza por un tipo de operación, que puede ser default, create, edit y view, y una FieldScreen.
            - **FieldScreen**: lo especificará un nombre y un conjunto de Tabs.
                - **Tab**: lo especificará un nombre y un conjunto de campos.
                    - **Field**: lo especificará un nombre y un tipo que puede ser un campo propio de JIRA o personalizado.
- **RoleScheme**: el esquema de roles define los perfiles que interactuarán con el proceso, los grupos y usuarios que forman parte de cada rol.
    - **Role**: se caracteriza por un nombre y unos miembros.
- **PermissionScheme**: el esquema de permisos permite indicar que roles, usuarios o grupos podrán realizar cada una de las 34 acciones que la aplicación permite realizar sobre los tickets. Estas acciones son:
    - **Administer projects**: permite administrar configuraciones en el proyecto como los componentes y las versiones.
    - **Browse projects**: capacidad para navegar por los proyectos y los tickets de los mismos.
    - **Servicedesk agent**: permite a los usuarios interactuar con los clientes y acceder a las funciones de la mesa de servicio de un proyecto.
    - **Create issues**: capacidad para crear tickets.
    - **Add comments**: capacidad para añadir comentarios en los tickets.
    - **Create attachments**: capacidad para añadir documentos adjuntos en los tickets.
    - **Assign issues**: capacidad para asignar tickets a otros usuarios.

    - **Assignable user**: a usuarios con este permiso se les puede asignar un ticket.
    - **Resolve issues**: capacidad para resolver y reabrir incidencias. Esto incluye la capacidad para asignar una versión como versión con la solución del ticket.
    - **Link issues**: capacidad para enlazar tickets entre sí y crear tickets enlazados. Sólo es útil si está habilitado enlazar tickets.
    - **Edit issues**: capacidad para editar tickets.
    - **Delete issues**: capacidad para borrar tickets.
    - **Close issues**: capacidad de cerrar tickets. A menudo es útil que tus desarrolladores resuelvan tickets y un departamento de calidad sea el encargado de realizar el cierre.
    - **Move issues**: Capacidad de mover tickets entre proyectos o entre flujos de trabajo dentro del mismo proyecto (si procede). Ten en cuenta que el usuario sólo puede mover tickets a un proyecto en que tenga permisos de creación de tickets.
    - **Schedule issues**: la capacidad de ver o editar la fecha de entrega de un ticket.
    - **Modify reporter**: capacidad para cambiar al creador al crear o editar un ticket.
    - **Work on issues**: capacidad para crear informes del trabajo realizado respecto a un ticket. Sólo es útil si el control del tiempo está habilitado.
    - **Delete all worklogs**: capacidad para eliminar todos los registros de trabajo.
    - **Delete own worklogs**: capacidad para borrar tus informes de trabajo.
    - **Edit all worklogs**: capacidad para editar todos los registros de trabajo.
    - **Edit own worklogs**: capacidad para editar tus informes de trabajo.
    - **View voters and watchers**: capacidad para ver a los votantes y observadores de un ticket.
    - **Manage watchers**: capacidad de gestionar los observadores de un ticket.
    - **Edit all comments**: capacidad para editar todos los comentarios.
    - **Edit own comments**: capacidad para editar tus comentarios.
    - **Delete all comments**: capacidad para borrar todos los comentarios.
    - **Delete own comments**: capacidad para borrar tus comentarios.
    - **Delete all attachments**: los usuarios con este permiso pueden borrar todos los adjuntos.
    - **Delete own attachments**: los usuarios con este permiso pueden borrar sus propios adjuntos.
    - **View dev tools**: capacidad para visualizar las herramientas de desarrollo
    - **View readonly workflow**: los usuarios con este permiso pueden ver una versión de sólo lectura del flujo de trabajo.
    - **Set issue security**: capacidad de asignar el nivel de seguridad de un ticket para que sólo los usuarios con ese nivel de seguridad pueda ver el ticket.
    - **Transition issues**: capacidad para hacer cambios de estado en tickets.
    - **Manage sprints permission**: capacidad para gestionar los sprints del proyecto.

- **NotificationScheme**: Este esquema permite indicar para cada uno de los 17 eventos que se pueden producir en la aplicación a que destinatarios se enviará

una notificación. Estos destinatarios se pueden definir en forma de rol, grupo o usuario. Los eventos son:

- **Issue created**: evento que se lanza cuando un ticket es creado.
- **Issue updated**: evento que se lanza cuando un ticket es actualizado.
- **Issue assigned**: evento que se lanza cuando un ticket es asignado.
- **Issue resolved**: evento que se lanza cuando un ticket es resuelto.
- **Issue closed**: evento que se lanza cuando un ticket es cerrado.
- **Issue commented**: evento que se lanza cuando un ticket es comentado.
- **Issue comment edited**: evento que se lanza cuando se edita el comentario de un ticket.
- **Issue comment deleted**: evento que se lanza cuando se borra el comentario de un ticket.
- **Issue reopened**: evento que se lanza cuando un ticket es reabierto.
- **Issue deleted**: evento que se lanza cuando un ticket es borrado.
- **Issue moved**: evento que se lanza cuando un ticket es movido.
- **Work logged on issue**: evento que se lanza cuando se registra trabajo en un ticket.
- **Work started on issue**: evento que se lanza cuando un ticket se ha comenzado a trabajar.
- **Work stopped on issue**: evento que se lanza cuando se completa el trabajo en un ticket.
- **Issue worklog updated**: evento que se lanza cuando un registro de trabajo se edita en un ticket.
- **Issue worklog deleted**: evento que se lanza cuando un registro de trabajo se borra de un ticket.
- **Generic event**: evento que se lanza cuando un ticket cambia de estado.

Este es el punto de partida para que la herramienta pueda producir configuraciones JIRA. A partir de esta definición básica se pueden generar los artefactos para la gestión de procesos en JIRA.

3.2.5. Gramática del Lenguaje de Dominio Específico

Con la ayuda del metamodelo definido, y con el objetivo de facilitar el uso, la notación que utilizaremos para escribir el DSL es JSON porque para las personas es sencillo de leer y escribir y para las computadoras es fácil de parsear y generar. En los siguientes apartados definiremos la gramática poniendo como ejemplo el caso de estudio de gestión de tareas y sub-tareas. En el Anexo I se podrá ver el ejemplo completo.

3.2.5.1. project

En JIRA los proyectos son las estructuras donde se almacenan los tickets. Sus características son:

```
{
	"key": "CASO1",
	"name": "CASO1: Gestion de tareas",
```

```
        "description": " CASO1: Gestion de tareas",
        "projectTypeKey": "business",
        "lead": "admin",
        "issueTypes": [],
        "assigneeType": "unassigned",
        "roles": [],
        "workflowScheme": [],
        "issueTypeScreenScheme": [],
        "notificationScheme": {},
        "issueSecurityScheme": "",
        "permissionScheme": {}
}
```

Dentro del objeto anterior, se podrá utilizar la gramática indicada en la siguiente tabla, pudiendo emplearse los elementos en el orden que se prefiera.

Nombre atributo	**Tipo atributo**	**Descripción**
key	Texto libre	Corresponde a la clave del proyecto
name	Texto libre	Corresponde al nombre del proyecto
description	Texto libre	Corresponde a la descripción del proyecto
projectTypeKey	Texto con las siguientes opciones: - business -serviceDesk -software	Corresponde a uno de los tres tipo de proyecto
lead	Texto correspondiente a un nombre de usuario dado de alta en la aplicación	Corresponde a un nombre de usuario dado de alta en la aplicación
assigneeType	Texto con las siguientes opciones:	Corresponde a uno de los tres tipos de asignación automática de tickets

	- project_default - component_lead - project_lead - unassigned	en el proyecto.
issueTypes	Array de Objeto	Corresponde al esquema de peticiones del proyecto.
roles	Array de Objeto	Corresponde al esquema de roles del proyecto.
workflowScheme	Array de Objeto	Corresponde al esquema de flujos del proyecto.
issueTypeScreenScheme	Array de Objeto	Corresponde al esquema de pantallas del proyecto.
notificationScheme	Objeto	Corresponde al esquema de notificaciones del proyecto.
permissionScheme	Objeto	Corresponde al esquema de permisos del proyecto.

Tabla 1: Gramática general de un proyecto

3.2.5.2. issueTypes

En este array se definen los tipos de ticket que se podrán trabajar en el proyecto. El siguiente ejemplo muestra dos tipos de ticket, Tarea y Subtarea:

```
"issueTypes": [
        {
          "name": "Tarea",
          "subtask": false
        },
        {
          "name": "Subtarea",
          "subtask": true
        }
]
```

La gramática que admite cada issueType es la siguiente:

Nombre atributo	Tipo atributo	Descripción
name	Texto libre	Corresponde al nombre del tipo de ticket
subtask	Booleano	Se indica con true que el tipo de ticket es una sub-tarea. Se indica con false que el tipo de ticket no es una sub-tarea

Tabla 2: Gramática de issueTypes para un proyecto

3.2.5.3. roles

En este array se definen los roles del proyecto y sus miembros. El siguiente ejemplo muestra dos roles, Administrators, cuyos miembros se definen como los integrantes del grupo jira-administrators, y Manager, cuyo miembro será el usuario admin:

```
"roles": [
	{
		"name": "Administrators",
		"actors": [
			{
			  "name": "jira-administrators",
			  "type": "Group"
			}
		]
	},
	{
		"name": "Managers",
		"actors": [
			{
			  "name": "admin",
			  "type": "User"
			}
		]
	}
]
```

La gramática que admite es la siguiente:

Nombre atributo	Tipo atributo	Descripción
name	Texto libre	Corresponde al nombre del rol

actors	Array de Objeto	Corresponde al array de objetos de define los miembros del rol.

Tabla 3: Gramática de roles de un proyecto

Para definir los miembros del rol lo hacemos mediante el atributo actors. Este es un array de objetos, la gramática del cual es la siguiente:

Nombre atributo	Tipo atributo	Descripción
name	Texto libre	Corresponde al nombre del miembro del rol
type	Texto con las siguientes opciones: - User - Group	Corresponde al tipo del miembro del rol que puede ser User o Group

Tabla 4: Gramática de los actors de un rol

3.2.5.4. workflowScheme

En este array se definen los flujos de trabajo del proyecto y sus transiciones. El siguiente ejemplo muestra un esquema de flujos de trabajo donde se utilizará un único flujo de trabajo para todos los tipos de ticket. Este flujo de trabajo únicamente tendrá dos estados, el estado Open y el estado Closed. Al crearse el ticket se establece en el estado Open y desde el estado Open se podrá transicionar al estado Closed mediante la transición Close:

```
"workflowScheme": [
	{
		"issueType": "DEFAULT",
		"initialAction": {
			"validators": [
				{
					"parameter": "Create Issue",
					"name": "PermissionValidator"
				}
			],
			"postfunctions": [
				{
					"name": "IssueCreateFunction"
				},
				{
					"name": "IssueReindexFunction"
				},
				{
```

```
                                "parameter": "1",
                                "name": "FireIssueEventFunction"
                        }
                ],
                "name": "Create",
                "to": "Open",
                "conditions": []
        },
        "transitions": [
                {
                        "validators": [],
                        "postfunctions": [
                          {
                                "name": "UpdateIssueStatusFunction"
                          },
                          {
                                "name": "AssignToCurrentUserFunction"
                          },
                          {
                                "name": "CreateCommentFunction"
                          },
                          {
                                "name": "GenerateChangeHistoryFunction"
                          },
                          {
                                "name": "IssueReindexFunction"
                          },
                          {
                                "name": "FireIssueEventFunction",
                                "parameter": "13"
                          }
                        ],
                        "name": "Close",
                        "from": "Open",
                        "to": "Closed",
                        "conditions": []
                }
        ]
    }
]
```

La gramática que admite cada objeto del array es la siguiente:

Nombre atributo	**Tipo atributo**	**Descripción**
issueType	Texto con el nombre de un tipo de ticket existente o el literal DEFAULT	Corresponde al nombre del tipo de ticket al que se le definen las pantallas y campos
initialAction	Objeto	Corresponde a la primera transición

		que se realiza al crear el ticket
transitions	Array de Objeto	Corresponde a cada una de las transiciones que se pueden realizar en cada uno de los estados del del ticket

Tabla 5: Gramática de los flujos de cada ticket

Tanto initialAction como Transitions definirán transiciones, con la diferencia que initialAction solo definirá una transición y Transitions puede definir más de una. Para definir cada transición seguiremos la siguiente gramática con la excepción que para initialAction no hace falta definir el atributo from ni el atributo conditions:

Nombre atributo	**Tipo atributo**	**Descripción**
name	Texto libre	Corresponde al nombre la transición
from	Texto libre	Corresponde al nombre del estado origen desde el que se iniciará la transición
to	Texto libre	Corresponde al nombre del estado destino desde el que se acabará la transición
validators	Array de Objeto	
postfunctions	Array de Objeto	
conditions	Array de Objeto	

Tabla 6: Gramática de las transiciones de cada flujo

Cada una de los métodos que forman parte de las acciones de transición validators, postfunctions, conditions que seguirán la misma gramática:

Nombre atributo	**Tipo atributo**	**Descripción**
name	Texto libre	Corresponde al nombre de la clase de validador, condición o función

		que se ejecutará
parameter	Texto libre	Corresponde al parámetro de la clase en caso de ser necesario

Tabla 7: Gramática de los métodos de cada transición

3.2.5.5. issueTypeScreenScheme

En este array se define para cada tipo de ticket las pantallas del proyecto y sus campos, de tal manera que cada objeto del array corresponde a un tipo de ticket. El siguiente ejemplo muestra un esquema de pantallas donde por defecto todos los tipos de ticket tendrán una pantalla que se mostrará en todas las operaciones que se pueden realizar con el ticket (create, edit y view). Esta pantalla contendrá una única pestaña (tab) con los campos Summary, Description y Priority:

```
"issueTypeScreenScheme": [
	{
		"issueType": "DEFAULT",
		"operations": {
			"default": [
				{
					"tab": "Pestaña de campo",
					"fields": [
						{
							"name": "Summary",
							"type": "jiraDefault"
						},
						{
							"name": "Description",
							"type": "jiraDefault"
						},
						{
							"name": "Priority",
							"type": "jiraDefault"
						}
					]
				}
			],
			"view": [],
			"edit": [],
			"create": []
		}
	}
]
```

La gramática que admite cada objeto del array es la siguiente:

Nombre atributo	Tipo atributo	Descripción
issueType	Texto con el nombre de un tipo de ticket existente o el literal DEFAULT	Corresponde al nombre del tipo de ticket al que se le definen las pantallas y campos
operations	Objeto	Corresponde a la operación, que se puede realizar en el ticket, a la cual se está configurando las pantallas y campos

Tabla 8: Gramática de las operaciones de cada tipo de ticket

Para definir las pantallas de cada operación utilizamos el atributo operations. Este es un array de objetos, la gramática del cual es la siguiente:

Nombre atributo	Tipo atributo	Descripción
default	Array de Objeto	Corresponde a la pantaña, que se mostrará por defecto, representada en un array de pestañas
view	Array de Objeto	Corresponde a la pantaña, que se mostrará en la vista del ticket, representada en un array de pestañas
edit	Array de Objeto	Corresponde a la pantaña, que se mostrará al editar el ticket, representada en un array de pestañas
create	Array de Objeto	Corresponde a la pantalla, que se

		mostrará al crear el ticket, representada en un array de pestañas

Tabla 9: Gramática de las pantallas de cada operación del ticket

Cada una de las pestañas que forman parte de las operaciones default, view, edit y create seguirán la misma gramática:

Nombre atributo	Tipo atributo	Descripción
tab	Texto libre	Corresponde al nombre de la pestaña
fields	Array de Objeto	Corresponde a los campos que formarán parte de la pestaña

Tabla 10: Gramática de las pestañas de una pantalla

La gramática que seguirá cada campo es la siguiente:

Nombre atributo	Tipo atributo	Descripción
name	Texto libre	Corresponde al nombre del campo
type	Texto con las siguientes opciones: - jiraDefault - custom	Corresponde a tipo del campo

Tabla 11: Gramática de los campos

3.2.5.6. notificationScheme

En este objeto se define para cada evento que se lanza en la aplicación los destinatarios a los que se enviará una notificación. El siguiente ejemplo muestra un esquema de notificaciones en el que se enviará al crear, editar o transicionar un ticket, una notificación al asignado actual del ticket, al usuario que ha creado el ticket y a todos los observadores del mismo:

```
"notificationScheme": {
      "issue_deleted": [],
```

```
        "issue_comment_edited": [],
        "issue_worklog_deleted": [],
        "issue_reopened": [],
        "issue_closed": [],
        "issue_worklog_updated": [],
        "issue_assigned": [],
        "issue_comment_deleted": [],
        "work_logged_on_issue": [],
        "issue_resolved": [],
        "work_started_on_issue": [],
        "work_stopped_on_issue": [],
        "issue_commented": [],
        "issue_moved": [],
        "issue_created": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "issue_updated": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "generic_event": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ]
}
```


La gramática que define las notificaciones del proyecto es:

Nombre atributo	Tipo atributo	Descripción
issue_deleted	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_comment_edited	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_worklog_deleted	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_reopened	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_closed	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_worklog_updated	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_assigned	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_comment_deleted	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
work_logged_on_issue	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_resolved	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
work_started_on_issue	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
work_stopped_on_issue	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_commented	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_moved	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_created	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento
issue_updated	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento

generic_event	Array de Objeto	Corresponde al listado de destinatarios que se enviarán en dicho evento

Tabla 12: Gramática de las notificaciones del proyecto

Cada uno de los objetos que definen los destinatarios de las notificaciones seguirá la siguiente gramática:

Nombre atributo	**Tipo atributo**	**Descripción**
parameter	Texto libre	Corresponde al parámetro que se puede establecer para los tipos de destinatarios que no son autodefinidos: - Single_User - Group_Dropdown - Project_Role - Single_Email_Address - User_Custom_Field_Value - Group_Custom_Field_Value
type	Texto con las siguientes opciones: - Current_Assignee - Current_Reporter - Remote_User - Project_Lead - Component_Lead - Single_User - Group_Dropdown - Project_Role	Corresponde a tipo de destinatario

	- Single_Email_Address - All_Watchers - User_Custom_Field_Value - Group_Custom_Field_Value	

Tabla 13: Gramática de una notificación

3.2.5.7. permissionScheme

En este objeto se definen los permisos que podrán realizar cada uno de los usuarios, grupos o roles del proyecto. El siguiente ejemplo muestra un esquema de permisos en el que los miembros con rol Administrators podrán crear tickets y los miembros con rol Managers podrán transicionar, editar y ver tickets:

```
"permissionScheme": {
	"add_comments": [],
	"manage_sprints_permission": [],
	"delete_all_comments": [],
	"edit_all_comments": [],
	"create_issues": [
		{
			"parameter": "Administrators",
			"type": "projectrole"
		}
	],
	"schedule_issues": [],
	"set_issue_security": [],
	"administer_projects": [],
	"delete_all_attachments": [],
	"transition_issues": [
		{
			"parameter": "Managers",
			"type": "projectrole"
		}
	],
	"edit_all_worklogs": [],
	"view_dev_tools": [],
	"link_issues": [],
	"delete_all_worklogs": [],
	"assignable_user": [],
	"resolve_issues": [],
	"work_on_issues": [],
	"edit_own_comments": [],
	"view_readonly_workflow": [],
	"delete_own_attachments": [],
	"delete_own_comments": [],
	"manage_watchers": [],
	"edit_issues": [
		{
			"parameter": "Managers",
			"type": "projectrole"
		}
```

```
        ],
        "close_issues": [],
        "move_issues": [],
        "edit_own_worklogs": [],
        "delete_own_worklogs": [],
        "modify_reporter": [],
        "view_voters_and_watchers": [],
        "delete_issues": [],
        "browse_projects": [
              {
                    "parameter": "Managers",
                    "type": "projectrole"
              }
        ],
        "assign_issues": [],
        "servicedesk_agent": [],
        "create_attachments": []
}
```

La gramática que define los permisos del proyecto es:

Nombre atributo	**Tipo atributo**	**Descripción**
add_comments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
manage_sprints_permission	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_all_comments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
edit_all_comments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
create_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
schedule_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
set_issue_security	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
administer_projects	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_all_attachments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
transition_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción

edit_all_worklogs	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
view_dev_tools	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
link_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_all_worklogs	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
assignable_user	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
resolve_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
work_on_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
edit_own_comments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
view_readonly_workflow	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_own_attachments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_own_comments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
manage_watchers	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
edit_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
close_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
move_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
edit_own_worklogs	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_own_worklogs	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta

		acción
modify_reporter	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
view_voters_and_watchers	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
delete_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
browse_projects	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
assign_issues	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
servicedesk_agent	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción
create_attachments	Array de Objeto	Corresponde al listado de miembros que podrán realizar esta acción

Tabla 14: Gramática del esquema de permisos

Cada uno de los objetos que definen los miembros que podrá realizar cada acción seguirá la siguiente gramática:

Nombre atributo	**Tipo atributo**	**Descripción**
parameter	Texto libre	Corresponde al parámetro que se puede establecer para los tipos de miembros que no son autodefinidos: - projectrole - group - userCF - user - groupCF
type	Texto con las siguientes	Corresponde al tipo de

	opciones: - projectrole - group - userCF - reporter - assignee - user - lead - groupCF	miembro

Tabla 15: Gramática de un permiso

3.3. Proceso de generación

Para ejecutar la tarea de generación, se proporciona el complemento JIRA que hemos desarrollado para tal efecto siguiendo la guía del Anexo IV. Este complemento ofrece, en un Gadget [16] que podemos insertar en un cuadro de mandos [17], la funcionalidad que consiste en el generador que pasándole como entrada el fichero JSON, lo parseará, validará teniendo en cuenta la gramática establecida y creará las configuraciones especificadas.

Una vez instalado dicho complemento, añadiremos el gadget llamado Config As Code a un cuadro de mandos. El aspecto que tiene es el siguiente:

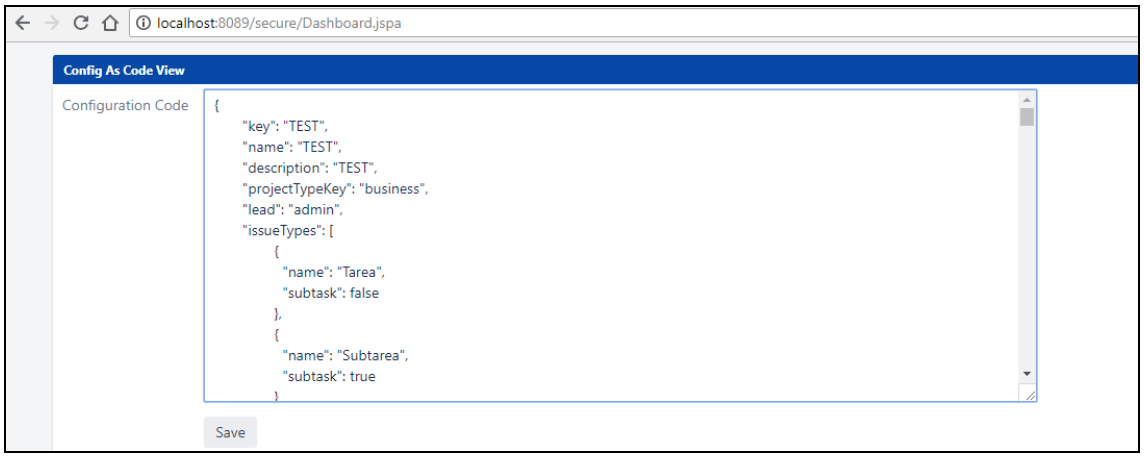


Figura 17: Gadget generador de configuraciones JIRA

En la caja de texto con etiqueta Configuration Code debemos insertar el código JSON con las configuraciones especificadas y hacer clic en el botón Save. Cuando empiece el proceso, se leerá y parseará el JSON comprobando que es un JSON válido. En caso

afirmativo, el siguiente paso es recorrer cada elemento para comprobar si cumple la gramática definida. Una vez hecha esta validación el proceso ya está listo para empezar a generar los elementos. Para su generación, se hará uso de la API Java de JIRA [10].

Esto nos generará automáticamente la configuración especificada en el JSON.

En caso que el código JSON insertado no supere las validaciones, aparecerá un mensaje y se parará el proceso.

4. EVALUACIÓN DEL GENERADOR

Para la evaluación del generador se tendrá en cuenta su capacidad de generación y flexibilidad para resolver configuraciones JIRA diferentes. Para ello se estudiarán tres casos que van aumentando el grado de complejidad.

4.1. Selección de casos

En este punto se tiene en cuenta la complejidad de las pruebas y la versatilidad de las mismas, se inicia con el caso de menor grado de complejidad y a medida que se pasa a otra prueba se aumenta la complejidad. Es de anotar que las pruebas seleccionadas deben ser aplicadas en el DSL construido. Las pruebas presentadas para la evaluación de la herramienta no son pruebas convencionales sobre un software desarrollado, son pruebas de generación de código y artefactos que deben ser validados bajo una arquitectura y una metodología, este código además debe ser validado en la herramienta. Por ello que se presentan los siguientes tres casos de pruebas.

4.2. Caso 1: Gestión de tareas

Para este caso se plantea un proceso de gestión de tareas el cual obedece a las siguientes especificaciones.

4.2.1. Especificaciones

Las características del proceso de gestión de tareas se describen a continuación:

- **Tipos de ticket**: Para la gestión de tareas se podrá dar de alta los siguientes tipos de ticket:
  - Tarea
  - Subtarea
- **Roles**: Los roles que intervendrán serán:
  - Administradores
  - Gestores
- **Flujos de trabajo**: Tanto las tareas como las sub-tareas tendrán el siguiente flujo de trabajo:

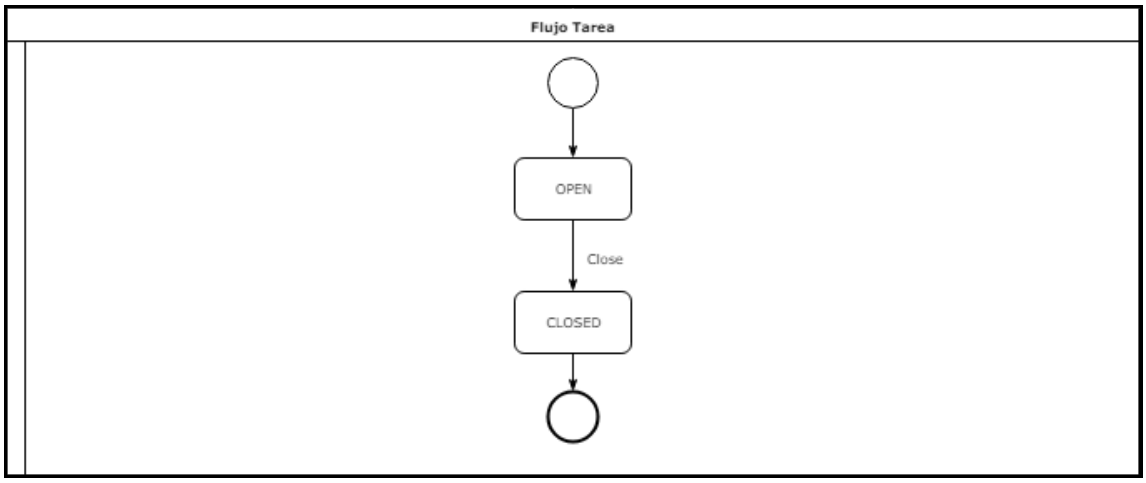


Figura 18: Flujo de trabajo Tareas

- **Pantallas**: Los campos que se podrán informar tanto en el alta y modificación como en la visualización serán:
    - Sumario
    - Descripción
    - Prioridad
- **Notificaciones**: Las notificaciones que se enviarán son:
    - En la creación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En la modificación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En el cambio de estado del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
- **Permisos**: Los permisos que tendrán cada involucrado serán:
    - **Crear tickets**: personas con rol administrador
    - **Ver tickets**: personas con rol administrador o gestor
    - **Cambiar estados**: personas con rol gestor
    - **Editar tickets**: personas con rol gestor

Todas estas especificaciones las plasmamos en el JSON del Anexo I.

4.2.2. Comprobación elementos generados

Una vez ejecutado el proceso podemos acceder a la administración de la aplicación dentro del apartado Proyectos para ver el proyecto generado:

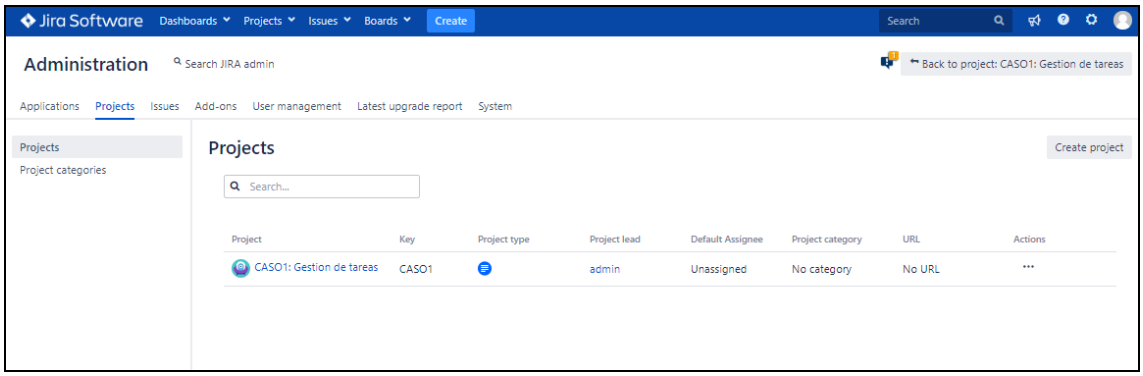


Figura 19: Menú administración proyectos JIRA

En dicho menú podemos observar el proyecto CASO1 con su información básica. Si hacemos clic en el accederemos a la configuración de los esquemas generados:

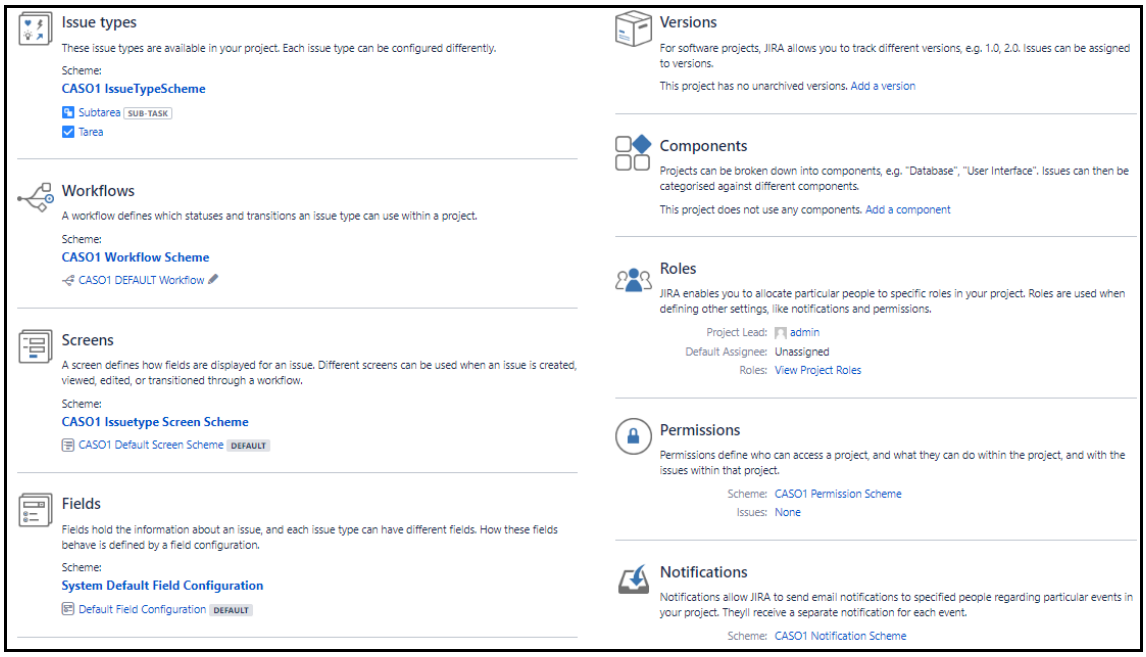


Figura 20: Detalle de los esquemas generados en el caso 1

- **Comprobación IssueType Scheme**

Si accedemos al IssueTypeScheme generado para el proyecto CASO1 podemos observar cómo se han asignado correctamente los 2 tipos de ticket Tarea y Subtarea:

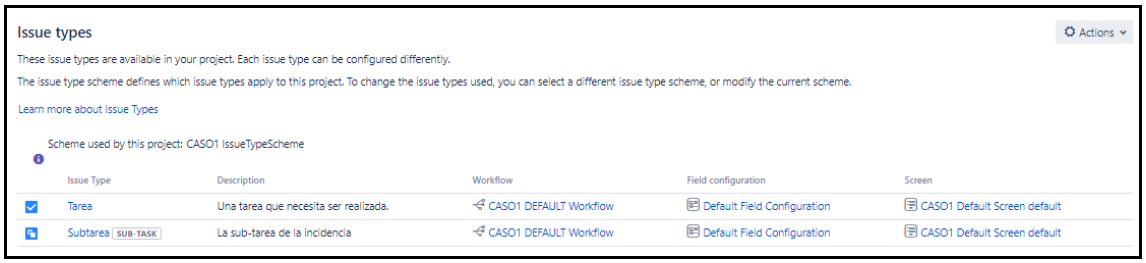


Figura 21: Detalle IssueTypeScheme del caso 1

- **Comprobación Role Scheme**

Si accedemos los roles generado para el proyecto CASO1 podemos observar cómo se ha asignado el grupo jira-administrators al rol Administrators y se ha asignado el usuario admin al rol Managers:

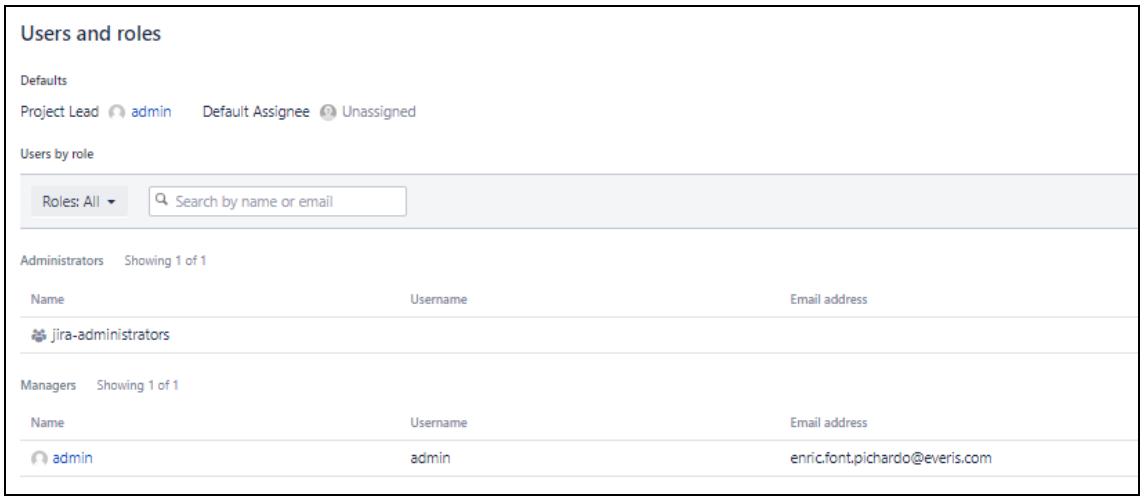


Figura 22: Detalle Roles del caso 1

- **Comprobación Workflow Scheme**

Si accedemos al WorkflowScheme generado para el proyecto CASO1 podemos observar cómo se ha asignado correctamente el flujo CASO1 DEFAULT Workflow a los 2 tipos de ticket Tarea y Subtarea:

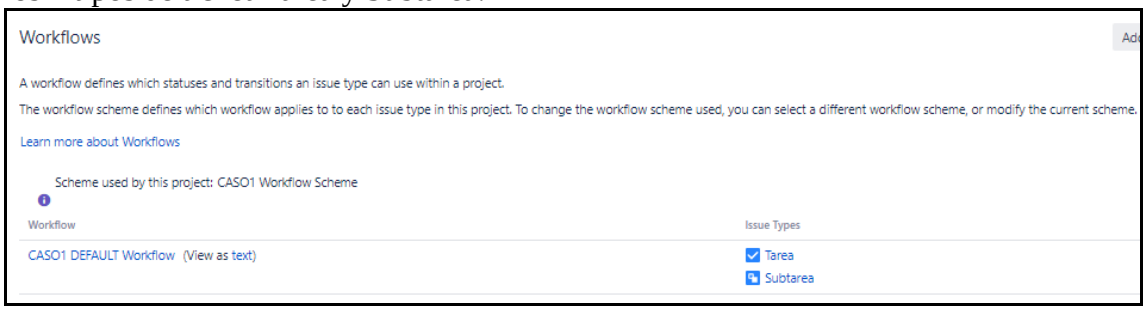


Figura 23: Detalle WorkflowScheme del caso 1

Además podemos ver el diagrama del flujo CASO1 DEFAULT Workflow:

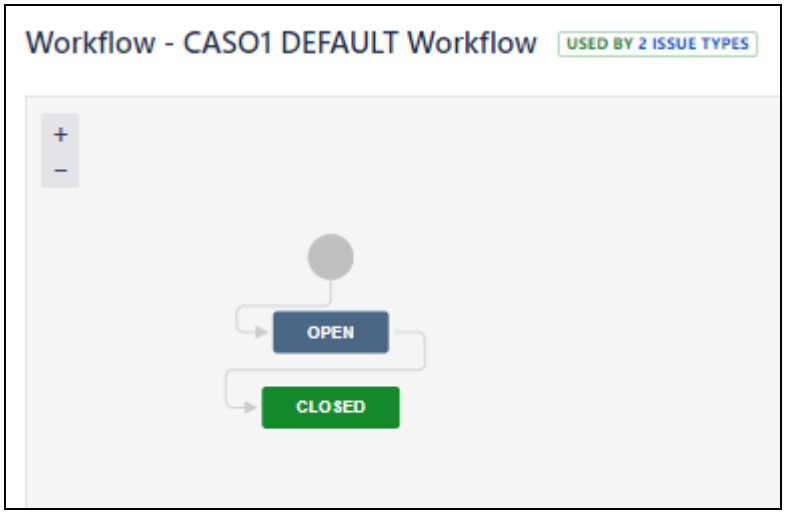


Figura 24: Detalle del Workflow del caso 1

- **Comprobación IssueType Screen Scheme**

Si accedemos al IssueTypeScreenScheme generado para el proyecto CASO1 podemos observar cómo se ha asignado correctamente el esquema de pantallas CASO1 DEFAULT Screen Scheme a los 2 tipos de ticket Tarea y Subtarea donde en las operaciones de creación, edición y vista tendremos la misma pantalla CASO1 DEFAULT Screen Default:

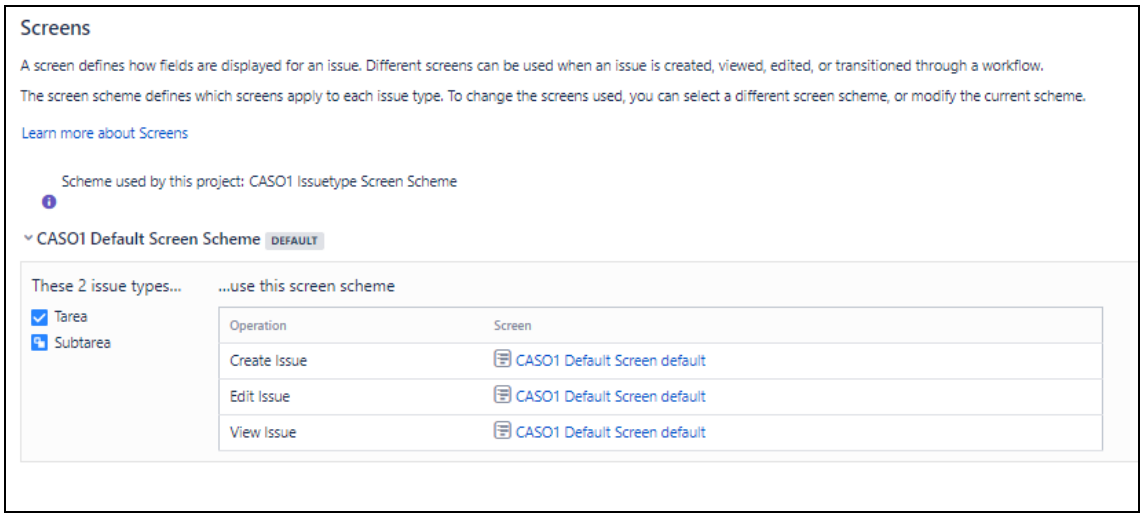


Figura 25: Detalle del IssueTypeScreenScheme del caso 1

Además podemos acceder a la pantalla CASO1 DEFAULT Screen Default y ver sus campos:

Figura 26: Detalle de la Screen del caso 1

- **Comprobación Permission Scheme**

Si accedemos al PermissionScheme generado para el proyecto CASO1 podemos observar los permisos otorgados:

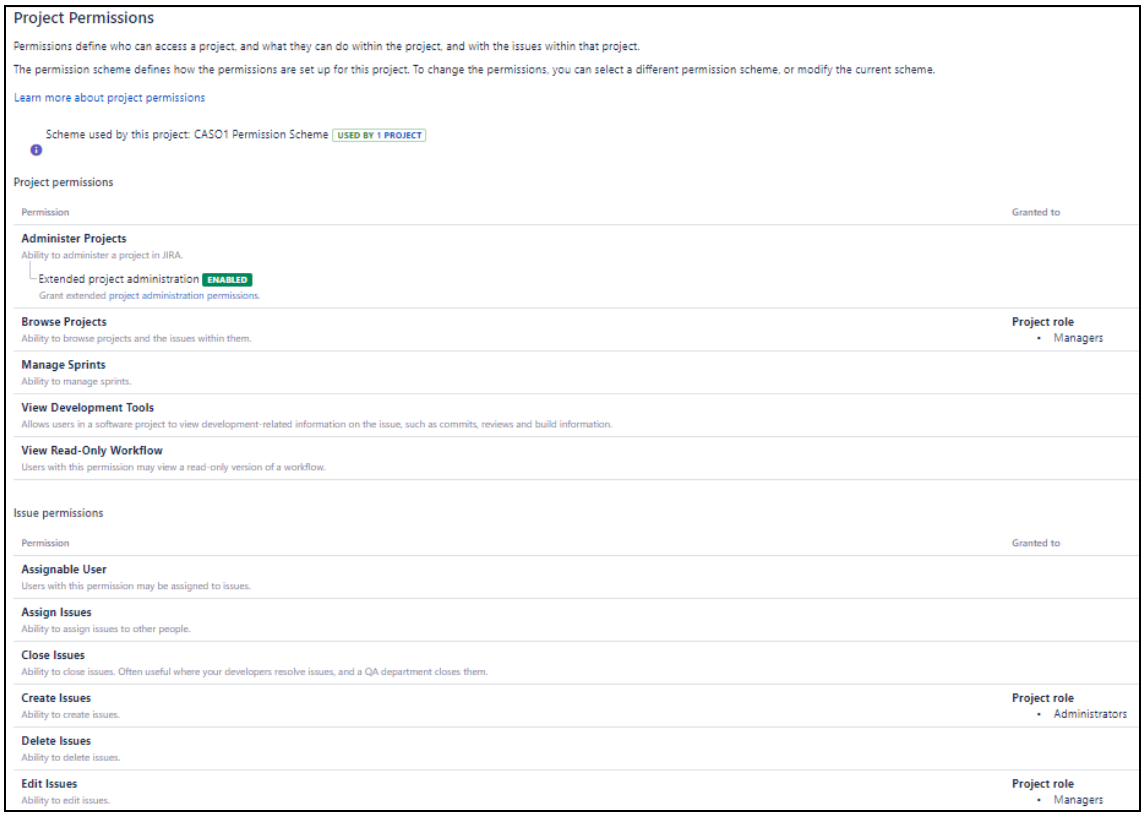


Figura 27: Detalle del PermissionScheme del caso 1

- **Comprobación Notification Scheme**

Si accedemos al NotificationScheme generado para el proyecto CASO1 podemos observar las notificaciones activadas:

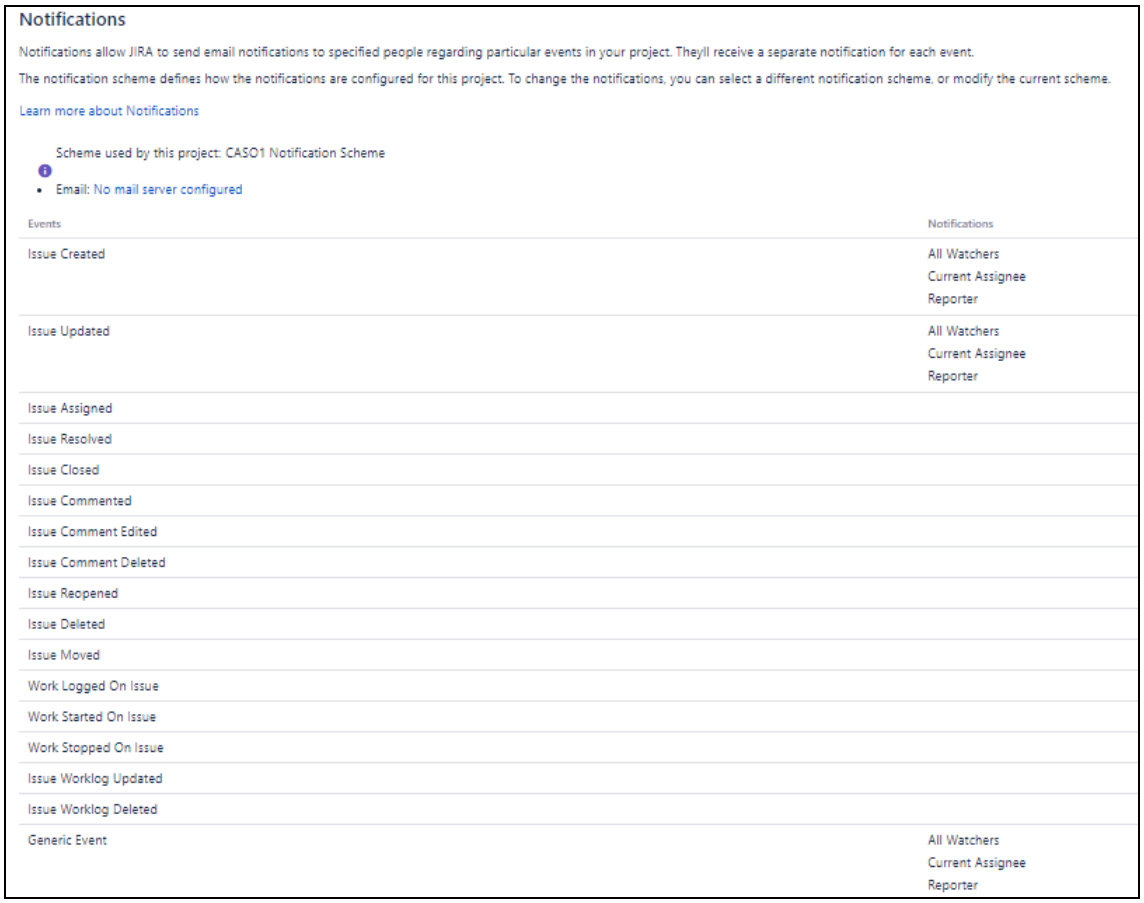


Figura 28: Detalle del NotificationScheme del caso 1

4.2.3. Pruebas de funcionamiento

Para realizar las pruebas de funcionamiento vamos a realizar acciones con tickets. Estas acciones van a ser las siguientes:

- Creación de ticket
- Edición de ticket
- Asignación de ticket
- Añadir comentarios en un ticket
- Añadir documentos adjuntos en un ticket
- Cambiar el estado de un ticket

4.2.3.1. Creación de tickets

Para la creación de un ticket hacemos clic en el botón Create y seleccionamos el Project CASO1. Posteriormente seleccionamos el issuetype Tarea y rellenamos el resto de información necesaria:

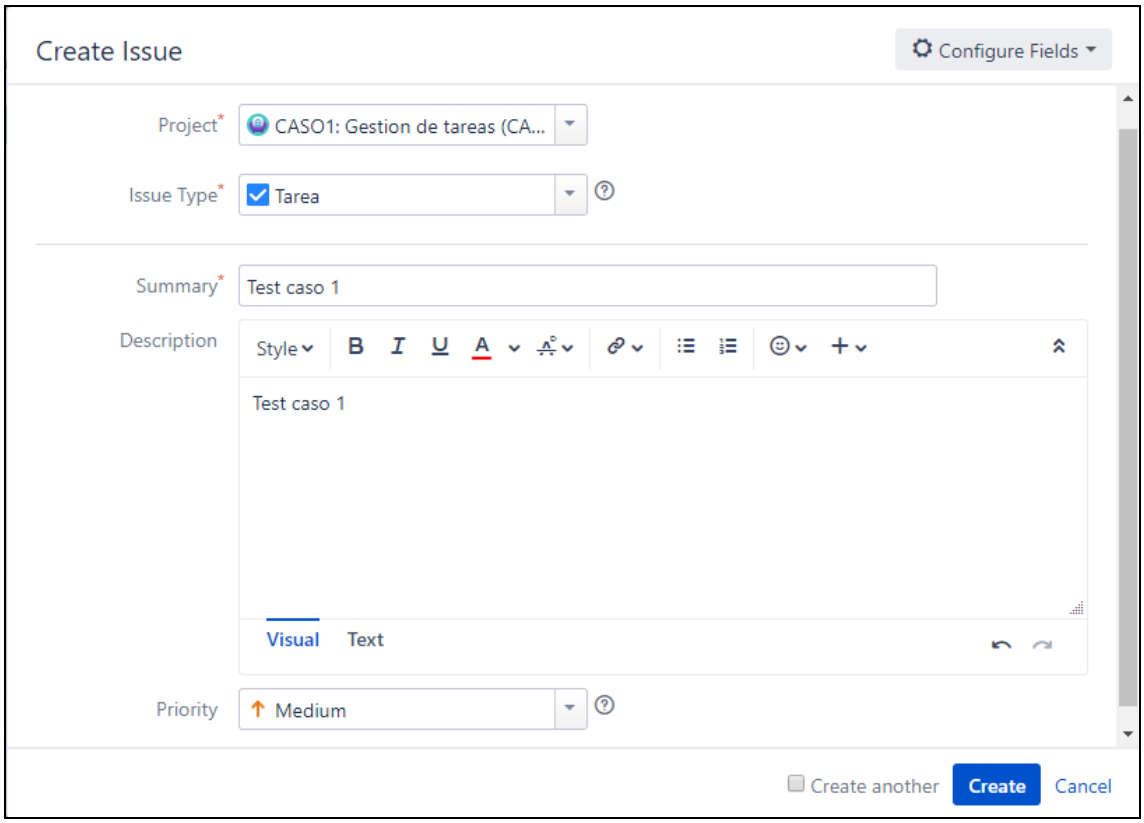


Figura 29: Creación de ticket CASO1

Una vez creado podemos visualizar su información accediendo a el:

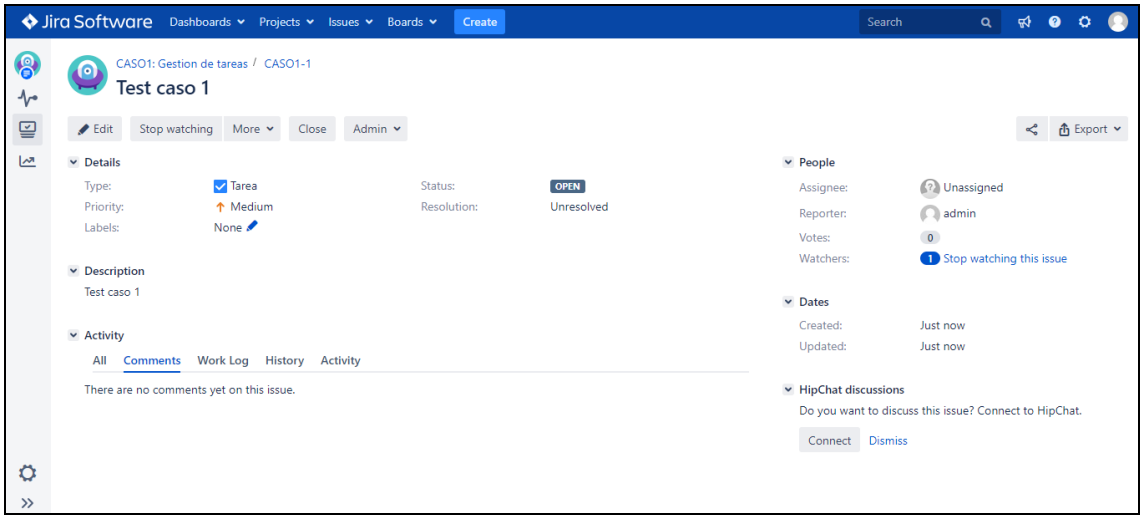


Figura 30: Visualización ticket CASO1

4.2.3.2. Edición de tickets

Para la edición de un ticket, en la pantalla de visualización del ticket, hacemos clic en el botón Edit y modificamos los campos que queremos editar:

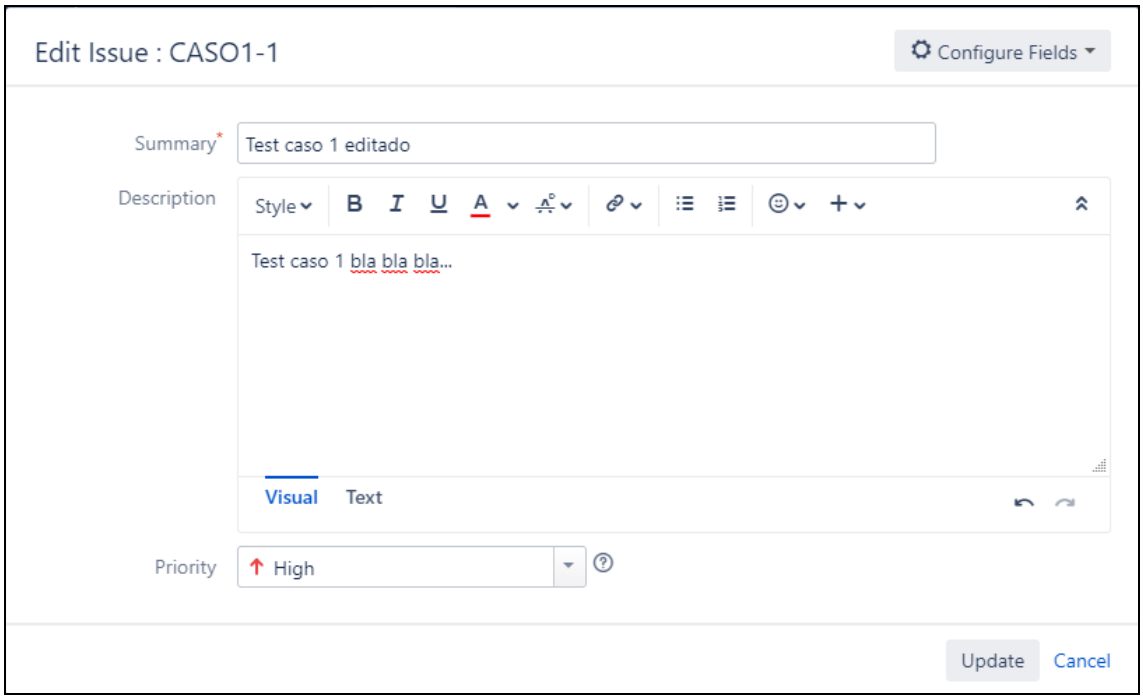


Figura 31: Edición de ticket CASO1

4.2.3.3. Asignación de tickets

La asignación de tickets no está activada en el CASO1 porque no hemos otorgado permisos para asignar ni para ser asignado a ningún miembro.

4.2.3.4. Añadir comentarios en un ticket

La acción de añadir comentarios en tickets no está activada en el CASO1 porque no hemos otorgado permisos para comentar a ningún miembro.

4.2.3.5. Añadir documentos adjuntos en un ticket

La acción de añadir documentos adjuntos en tickets no está activada en el CASO1 porque no hemos otorgado permisos para adjuntar documentos a ningún miembro.

4.2.3.6. Cambiar el estado de un ticket

En el momento de la creación el ticket permanece en el estado OPEN tal y como hemos especificado. En este estado la única transición disponible es la Close que cambiará el estado del ticket a CLOSED. Para realizar el cambio de estado hacemos clic en el botón Close:

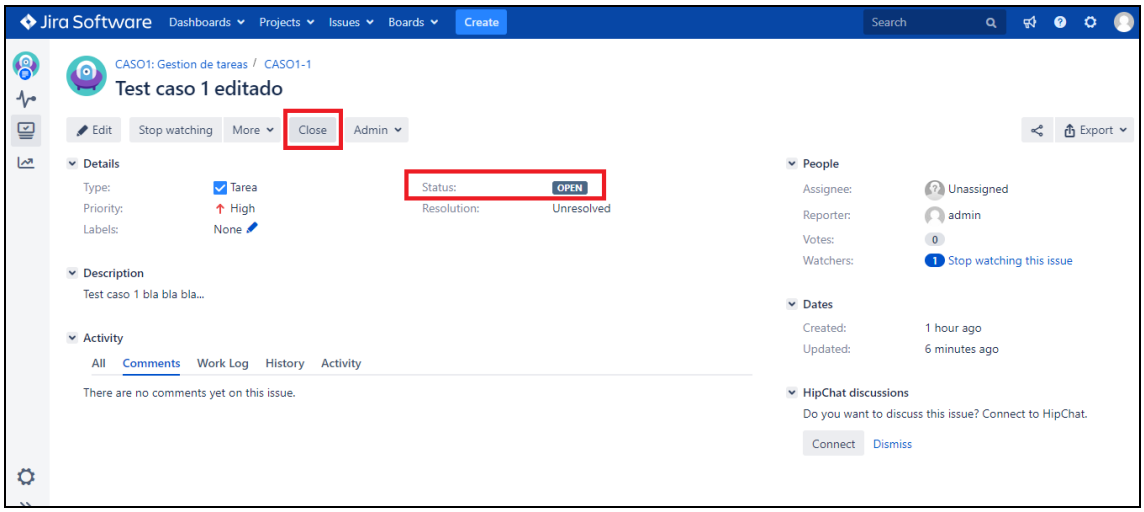


Figura 32: Cambio de estado de ticket CASO1

Al finalizar la transición podemos ver como el ticket ha cambiado al estado CLOSED:

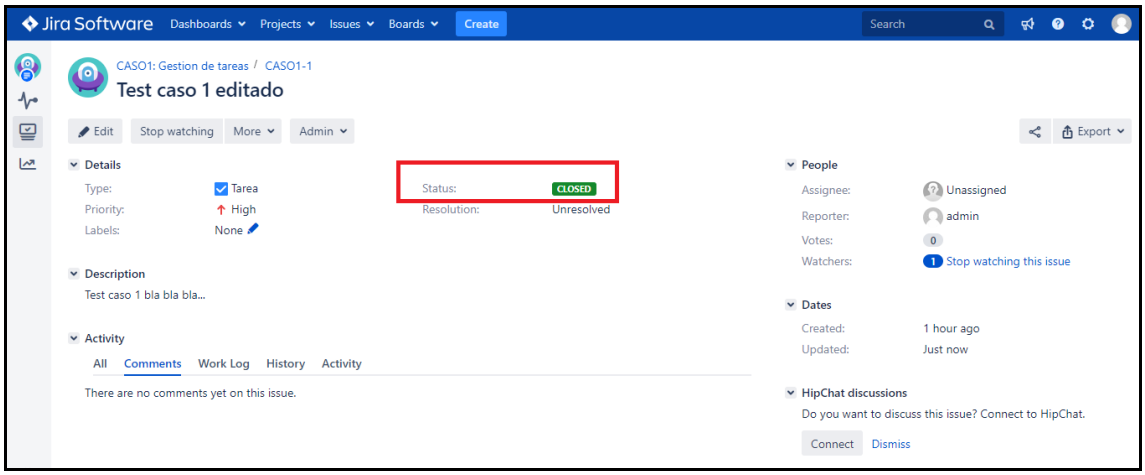


Figura 33: Visualización del cambio de estado de ticket CASO1

4.3. Caso 2: Gestión de Incidencias

Para este caso se plantea un proceso de gestión de incidencias el cual obedece a las siguientes especificaciones.

4.3.1. Especificaciones

Las características del proceso de gestión de incidencias se describen a continuación:

- **Tipos de ticket**: Para la gestión de incidencias se podrá dar de alta los siguientes tipos de ticket:
  - Incidencia
- **Roles**: Los roles que intervendrán serán:
  - Requester
  - Dispatcher
  - Level 1 Support

    - Level 2 Support
- **Flujos de trabajo**: Las incidencias tendrán el siguiente flujo de trabajo:

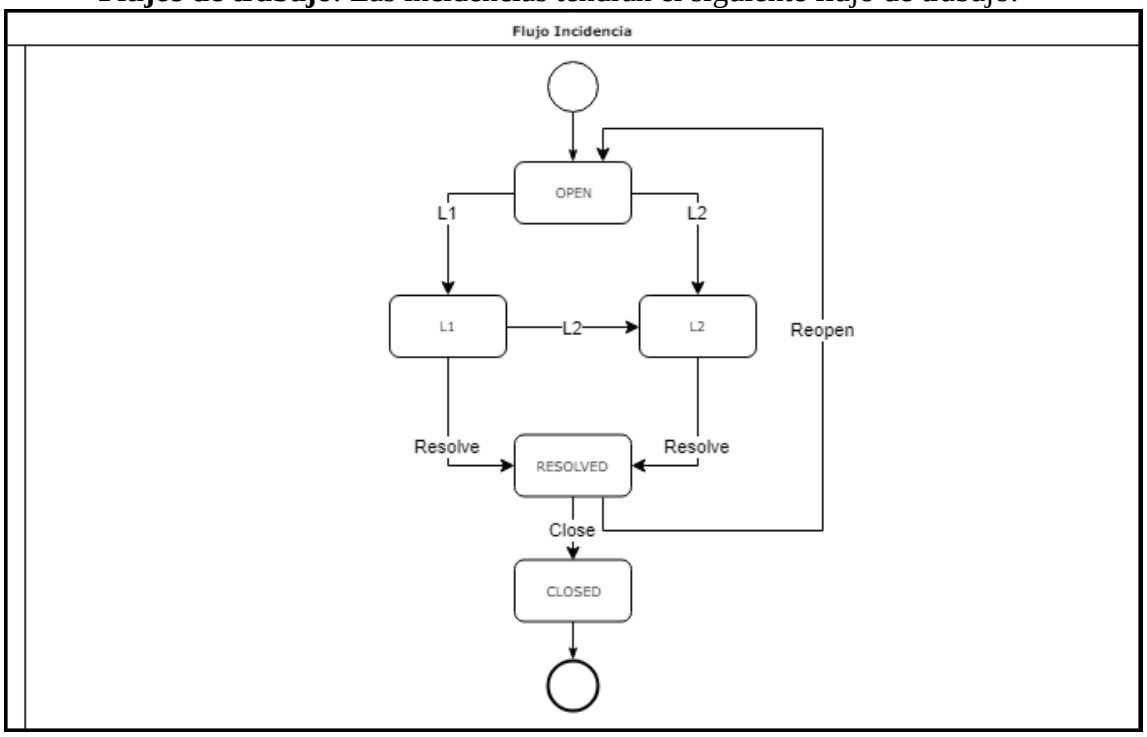


Figura 34: Flujo de trabajo de Incidencias

- **Pantallas**: Los campos que se podrán informar tanto en el alta y modificación como en la visualización serán:
    - Sumario
    - Descripción
    - Prioridad
    - Situación correcta de la incidencia
    - Mensaje de error exacto
    - Datos para reproducir el caso
    - Código de transacción / Ruta de menú
- **Notificaciones**: Las notificaciones que se enviarán son:
    - En la creación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En la modificación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En el cambio de estado del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - Al resolver el ticket:
        - Al creador del ticket
    - Al asignar el ticket:
        - Al responsable del ticket
    - Al cerrar el ticket:

        - Al responsable del ticket
    - Al reabrir el ticket:
        - Al responsable del ticket
- **Permisos**: Los permisos que tendrán cada involucrado serán:
    - **Crear tickets**: personas con rol requester
    - **Ver tickets**: personas con rol requester, dispatcher, level1 o level2
    - **Cambiar estados**: personas con rol requester, dispatcher, level1 o level2
    - **Editar tickets**: personas con rol requester, dispatcher, level1 o level2
    - **Añadir comentarios**: personas con rol requester, dispatcher, level1 o level2
    - **Usuario asignable**: personas con rol level1 o level2
    - **Resolver tickets**: personas con rol level1 o level2
    - **Cerrar tickets**: personas con rol level1 o level2
    - **Asignar tickets**: personas con rol dispatcher
    - **Añadir documentos adjuntos**: personas con rol requester, dispatcher, level1 o level2

Todas estas especificaciones las plasmamos en el JSON del Anexo II.

4.3.2. Comprobación elementos generados

Una vez ejecutado el proceso podemos acceder a la administración de la aplicación dentro del apartado Proyectos para ver el proyecto generado:

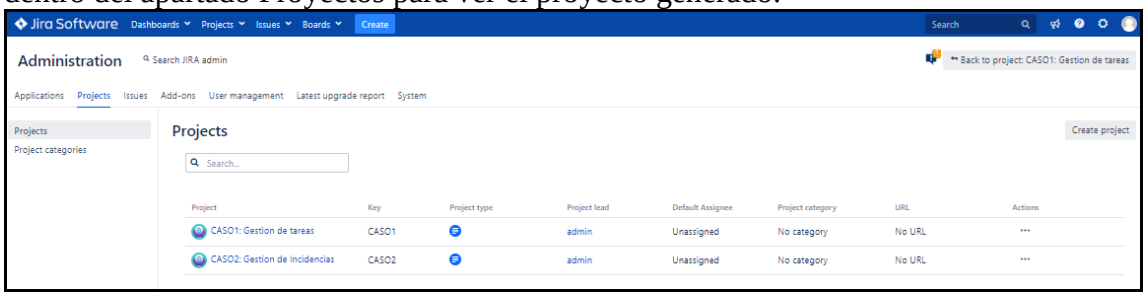


Figura 35: Menú administración proyectos JIRA

En dicho menú podemos observar el proyecto CASO2 con su información básica. Si hacemos clic en el accederemos a la configuración de los esquemas generados:

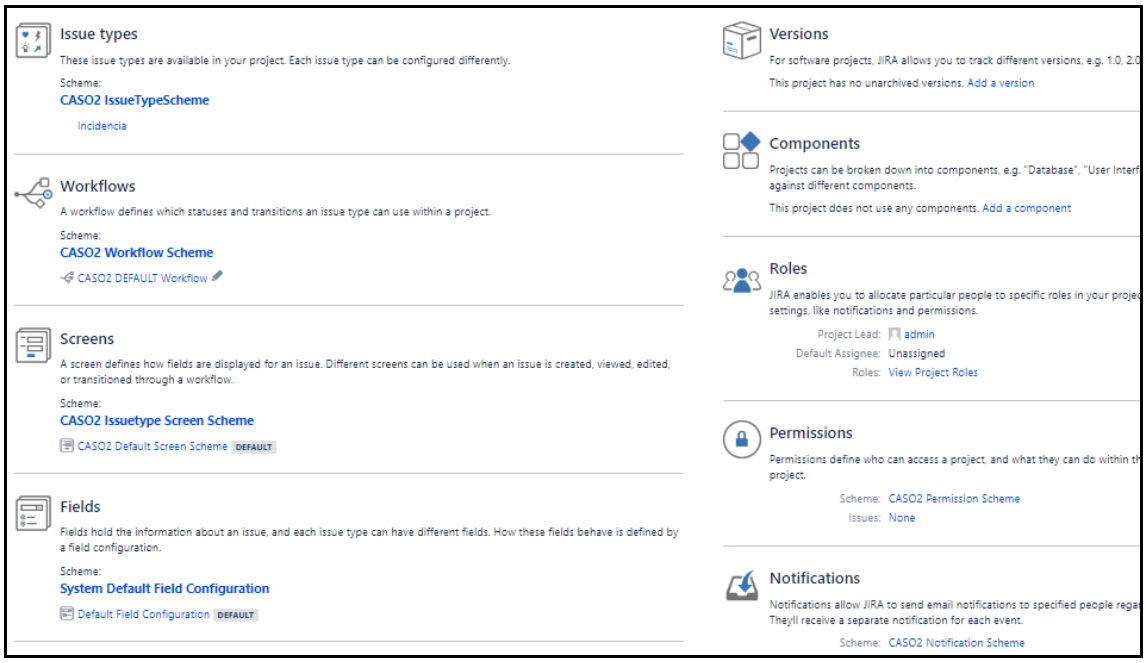


Figura 36: Detalle de los esquemas generados en el caso 2

- **Comprobación IssueType Scheme**

Si accedemos al IssueTypeScheme generado para el proyecto CASO2 podemos observar cómo se han asignado correctamente el tipo de ticket Incidencia:

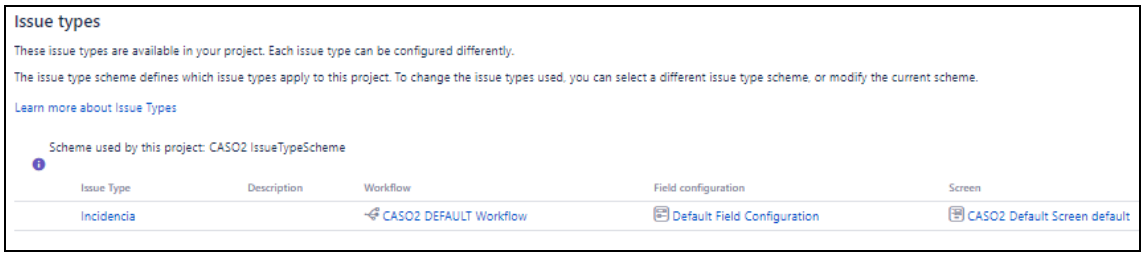


Figura 37: Detalle IssueTypeScheme del caso 2

- **Comprobación Role Scheme**

Si accedemos los roles generado para el proyecto CASO2 podemos observar cómo se ha asignado el grupo jira-administrators al rol Administrators y se ha asignado el usuario admin al rol Managers:

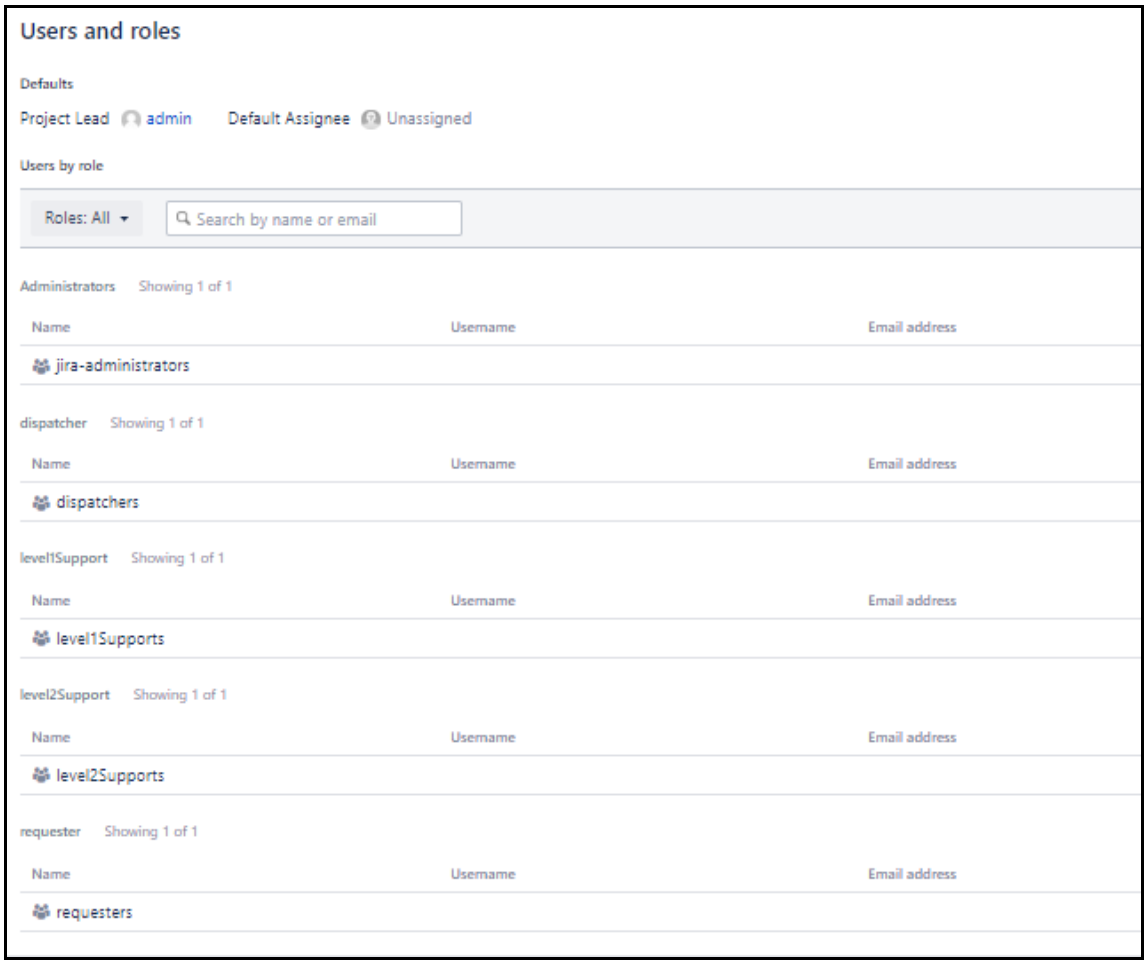


Figura 38: Detalle Roles del caso 2

- **Comprobación Workflow Scheme**

Si accedemos al WorkflowScheme generado para el proyecto CASO2 podemos observar cómo se ha asignado correctamente el flujo CASO2 DEFAULT Workflow al tipo de ticket Incidencia:

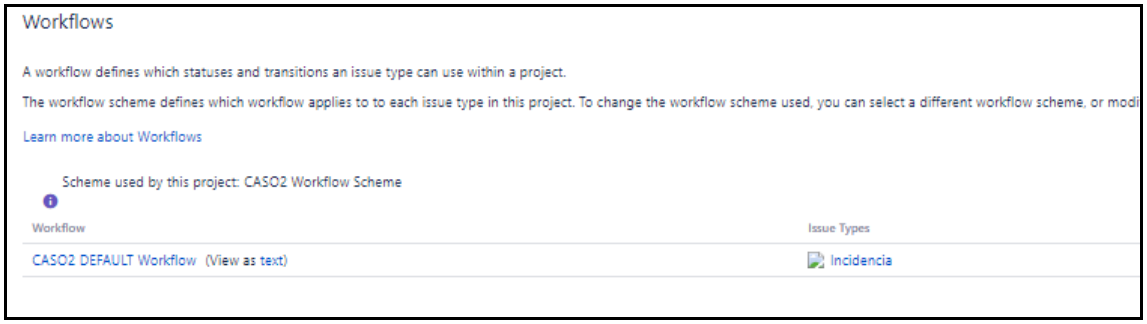


Figura 39: Detalle WorkflowScheme del caso 2

Además podemos ver el diagrama del flujo CASO2 DEFAULT Workflow:

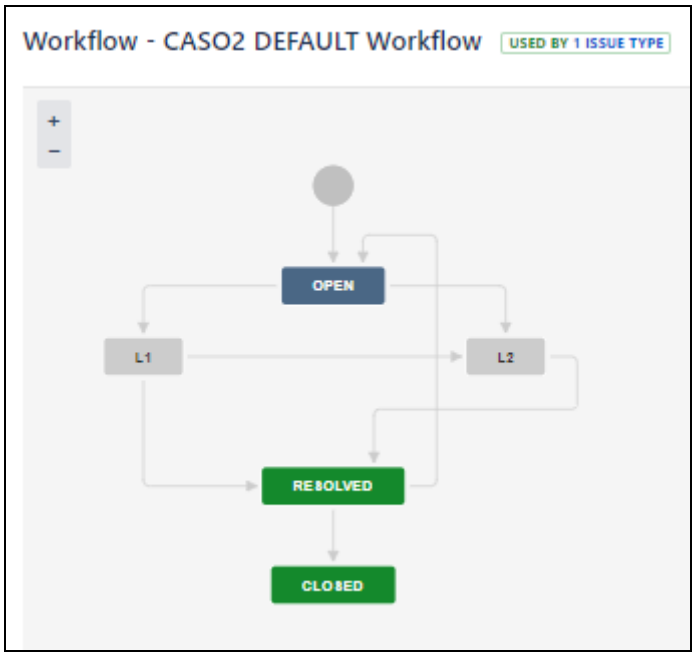


Figura 40: Detalle del Workflow del caso 2

- **Comprobación IssueType Screen Scheme**

Si accedemos al IssueTypeScreenScheme generado para el proyecto CASO2 podemos observar cómo se ha asignado correctamente el esquema de pantallas CASO2 DEFAULT Screen Scheme al tipo de ticket Incidencia donde en las operaciones de creación, edición y vista tendremos la misma pantalla CASO2 DEFAULT Screen Default:

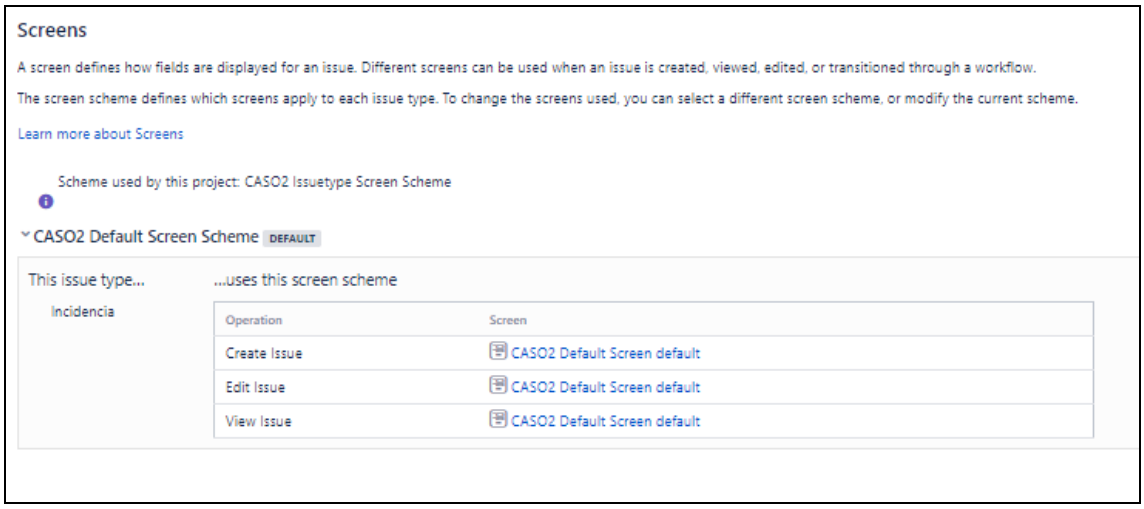


Figura 41: Detalle del IssueTypeScreenScheme del caso 2

Además podemos acceder a la pantalla CASO2 DEFAULT Screen Default y ver sus campos:

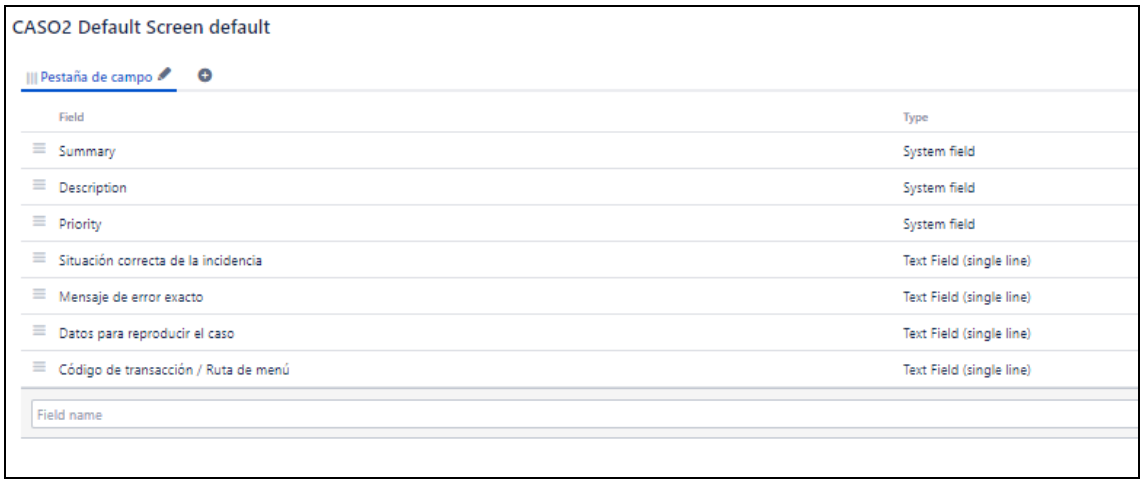


Figura 42: Detalle de la Screen del caso 2

- **Comprobación Permission Scheme**

Si accedemos al PermissionScheme generado para el proyecto CASO2 podemos observar los permisos otorgados:

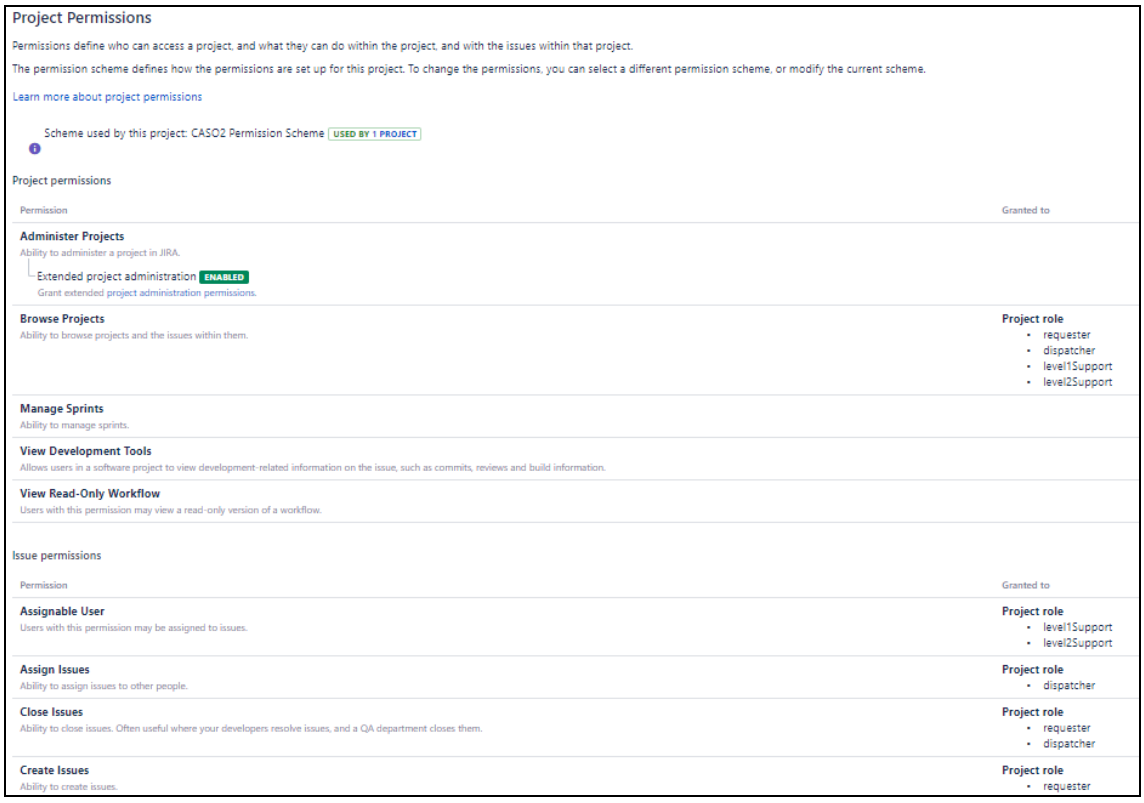
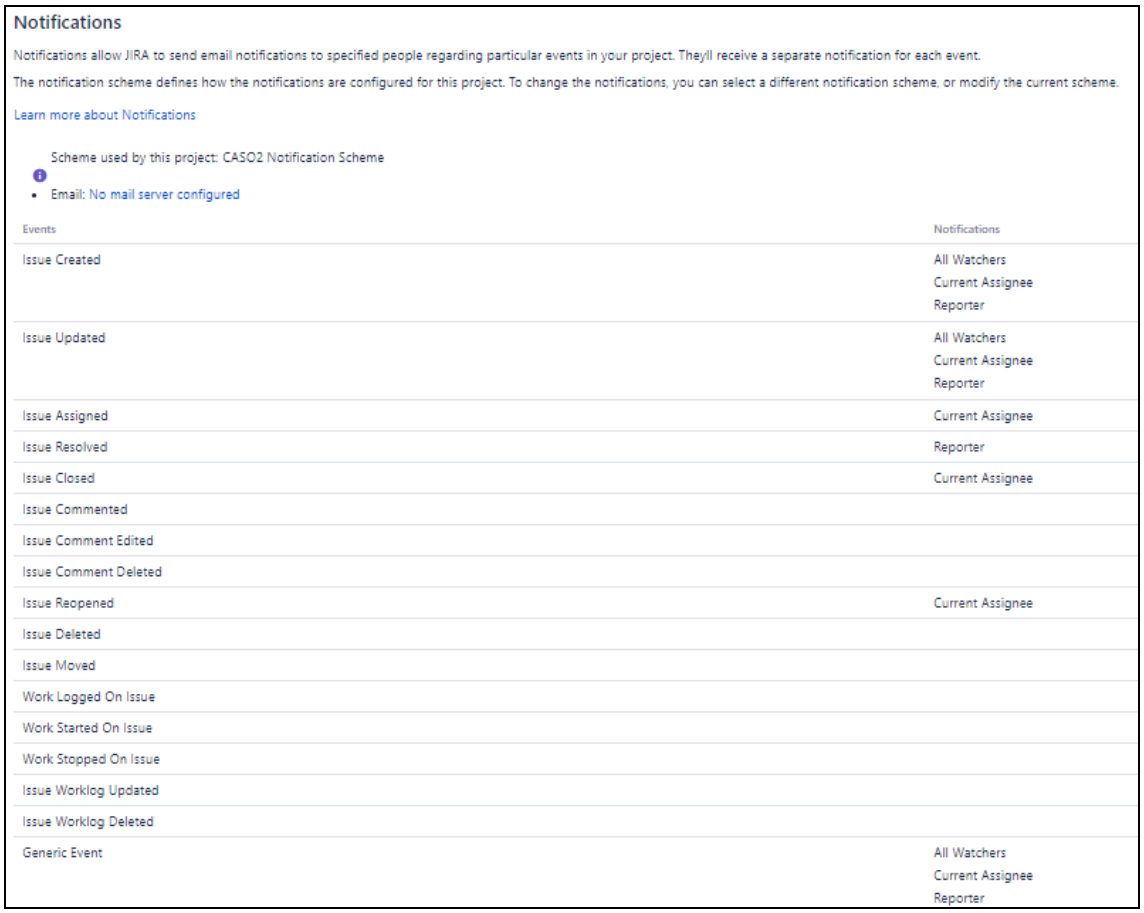


Figura 43: Detalle del PermissionScheme del caso 2

- **Comprobación Notification Scheme**

Si accedemos al NotificationScheme generado para el proyecto CASO2 podemos observar las notificaciones activadas:



Notifications

Notifications allow JIRA to send email notifications to specified people regarding particular events in your project. They'll receive a separate notification for each event.

The notification scheme defines how the notifications are configured for this project. To change the notifications, you can select a different notification scheme, or modify the current scheme.

Learn more about Notifications

Scheme used by this project: CASO2 Notification Scheme

- Email: No mail server configured

Events	Notifications
Issue Created	All Watchers Current Assignee Reporter
Issue Updated	All Watchers Current Assignee Reporter
Issue Assigned	Current Assignee
Issue Resolved	Reporter
Issue Closed	Current Assignee
Issue Commented	
Issue Comment Edited	
Issue Comment Deleted	
Issue Reopened	Current Assignee
Issue Deleted	
Issue Moved	
Work Logged On Issue	
Work Started On Issue	
Work Stopped On Issue	
Issue Worklog Updated	
Issue Worklog Deleted	
Generic Event	All Watchers Current Assignee Reporter

Figura 44: Detalle del NotificationScheme del caso 2

4.3.3. Pruebas de funcionamiento

Para realizar las pruebas de funcionamiento vamos a realizar acciones con tickets. Estas acciones van a ser las siguientes:

- Creación de ticket
- Edición de ticket
- Asignación de ticket
- Añadir comentarios en un ticket
- Añadir documentos adjuntos en un ticket
- Cambiar el estado de un ticket

4.3.3.1. Creación de tickets

Para la creación de un ticket hacemos clic en el botón create y seleccionamos el Project CASO2. Posteriormente seleccionamos el issuetype Incidencia y rellenamos el resto de información necesaria:

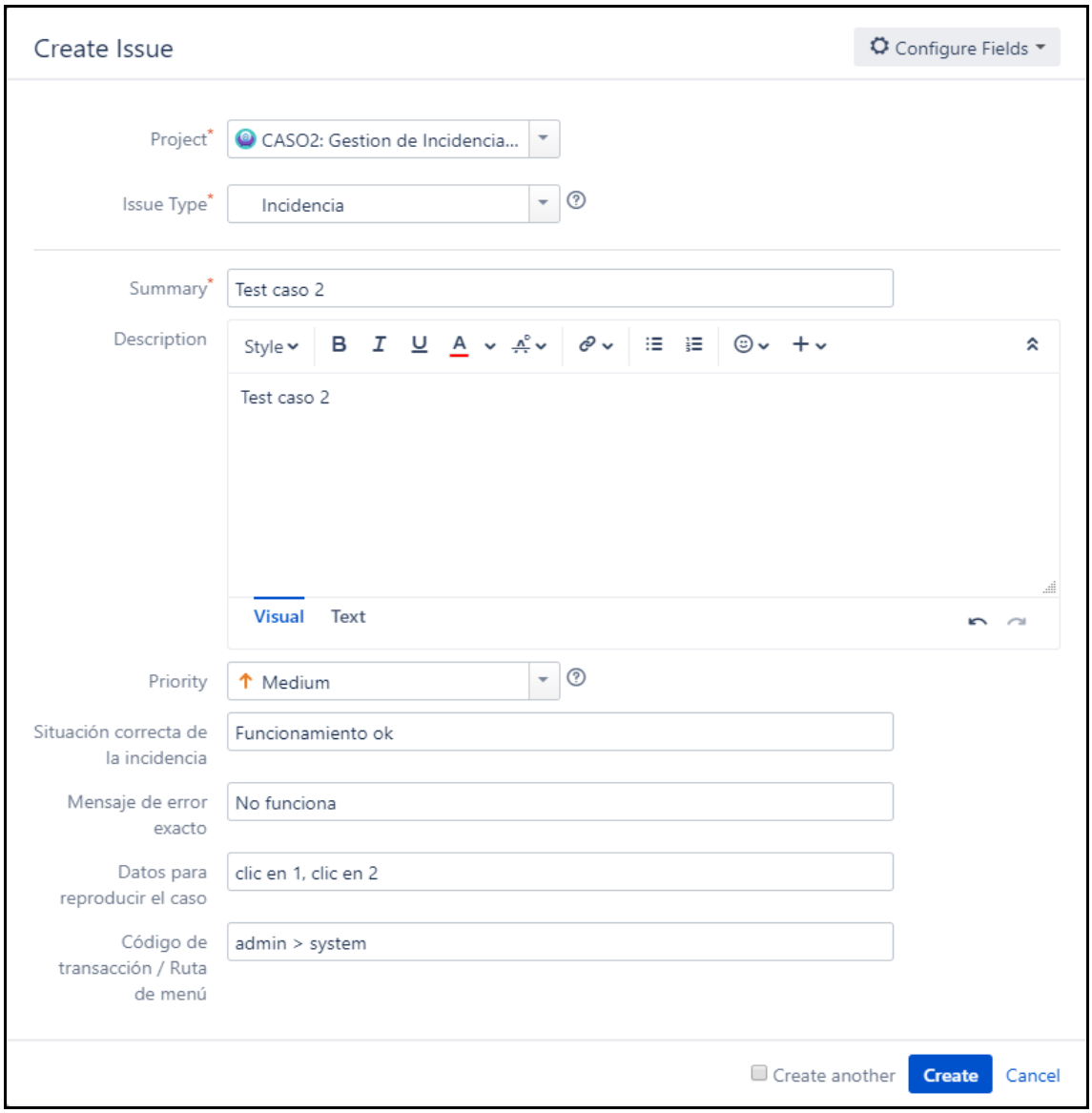


Figura 45: Creación de ticket CASO2

Una vez creado podemos visualizar su información accediendo a el:

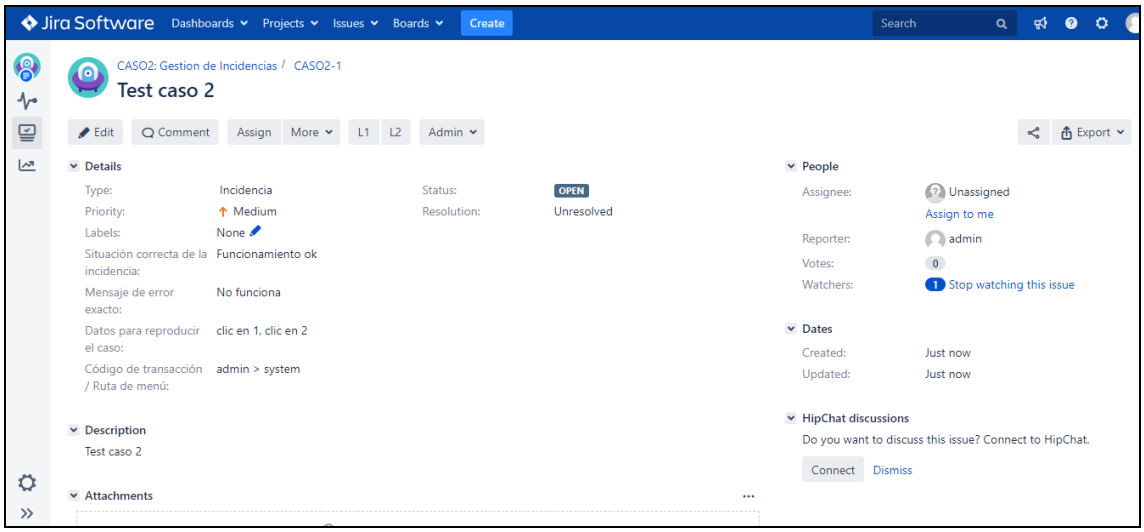


Figura 46: Visualización ticket CASO2

4.3.3.2. Edición de tickets

Para la edición de un ticket, en la pantalla de visualización del ticket, hacemos clic en el botón Edit y modificamos los campos que queremos editar:

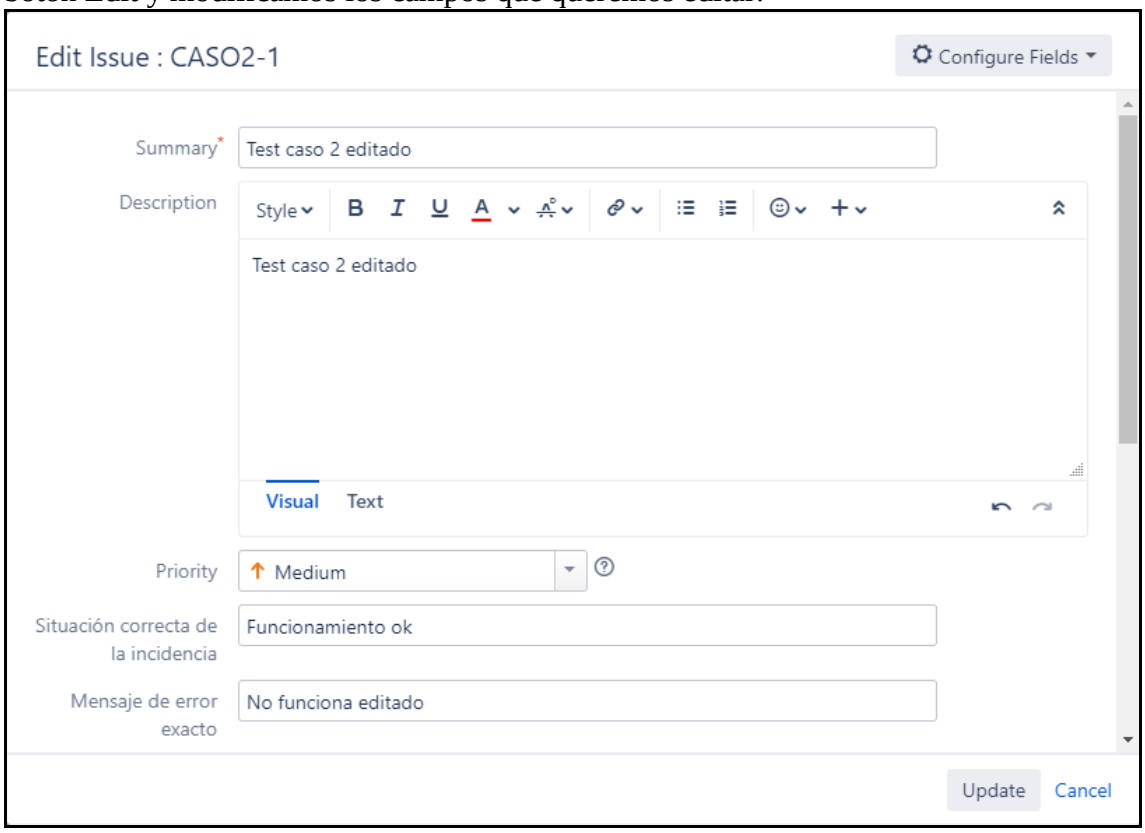


Figura 47: Edición de ticket CASO2

4.3.3.3. Asignación de tickets

Para asignar un ticket del CASO2 hacemos clic en el botón Assign y seleccionamos uno de los usuarios que pueden ser asignados:

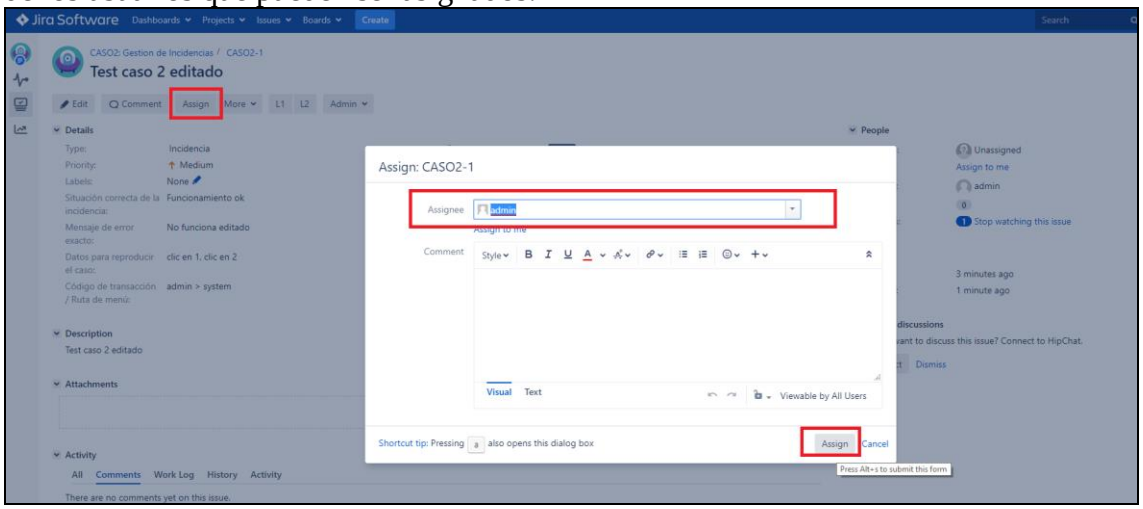


Figura 48: Asignación de ticket CASO 2

4.3.3.4. Añadir comentarios en un ticket

Para añadir un comentario en un ticket del CASO2 hacemos clic en el botón Comment e introducimos el texto del comentario:

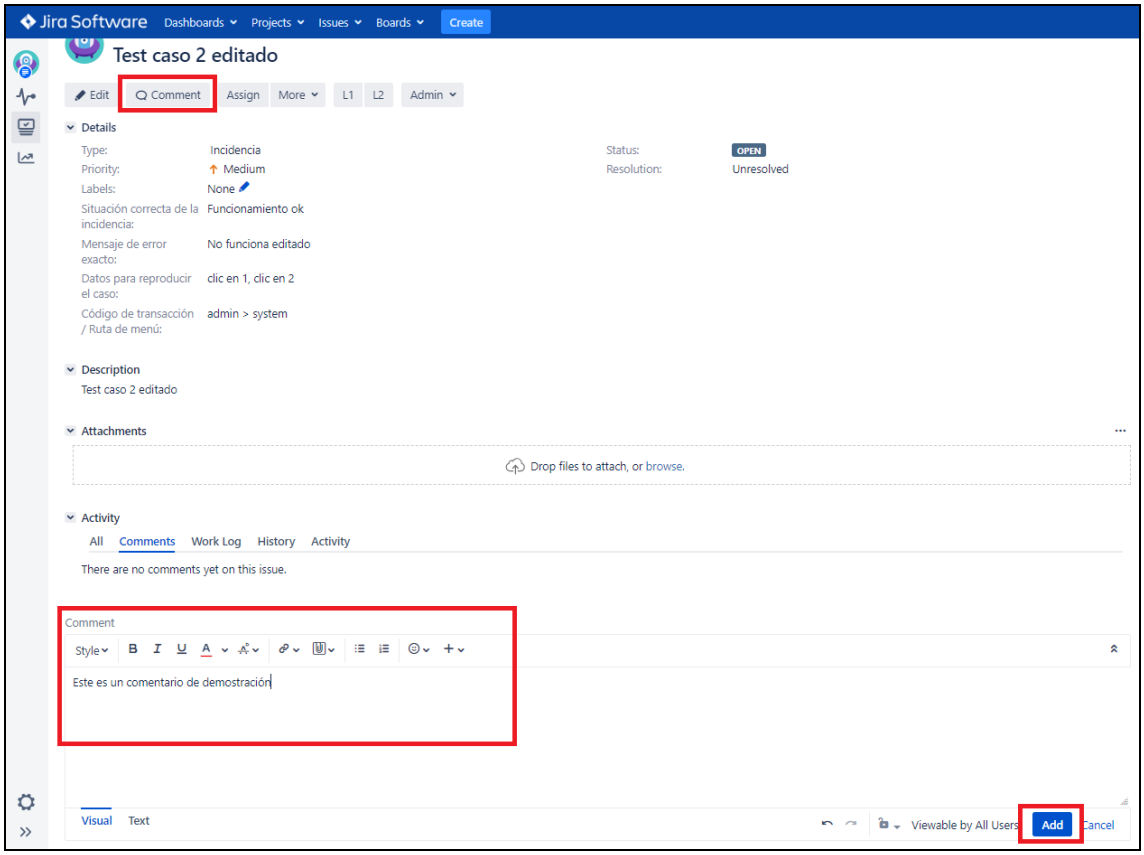


Figura 49: Añadir comentario en ticket CASO2

4.3.3.5. Añadir documentos adjuntos en un ticket

Para añadir un documento adjunto en un ticket del CASO2 arrastramos al ticket el documento desde el Escritorio:

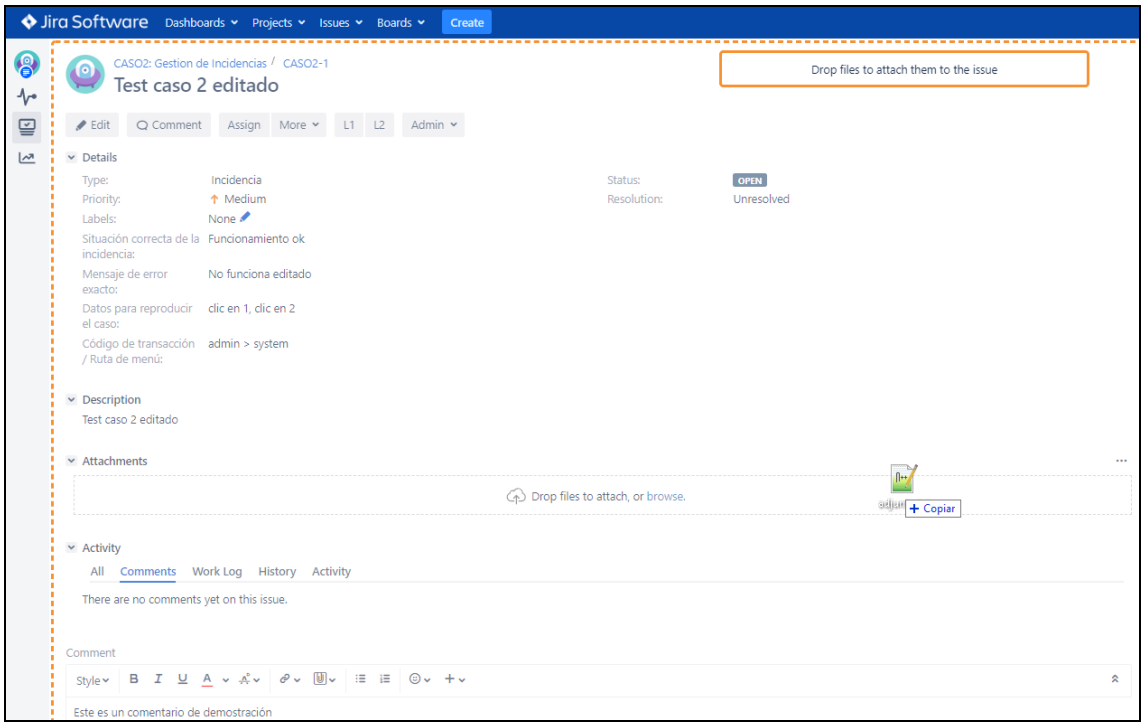


Figura 50: Añadir documento adjunto en ticket CASO2

4.3.3.6. Cambiar el estado de un ticket

En el momento de la creación el ticket permanece en el estado OPEN tal y como hemos especificado. En este estado las únicas transiciones disponibles son L1 que cambiará el estado del ticket a L1 y L2 que cambiará el estado del ticket a L2. Para realizar el cambio de estado hacemos clic en uno de los dos botones. En este caso hacemos clic en L1:

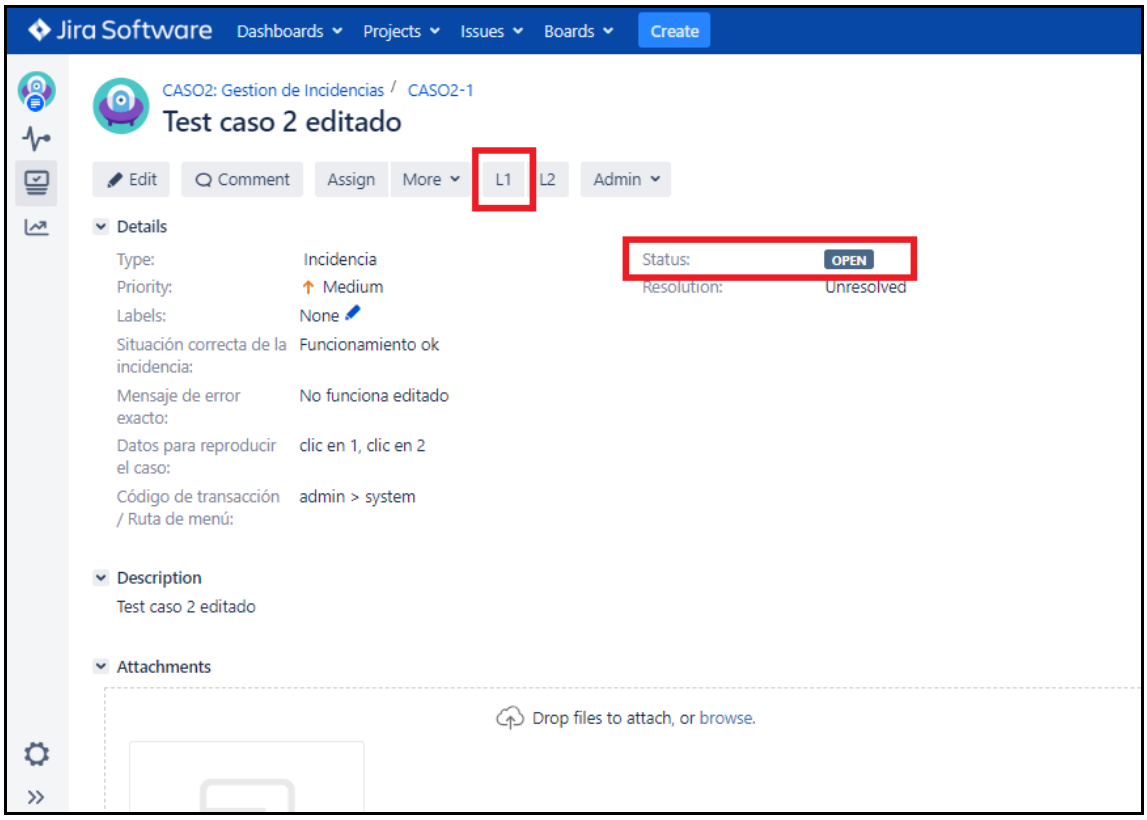


Figura 51: Cambio de estado de ticket CASO2

Al finalizar la transición podemos ver como el ticket ha cambiado al estado L1:

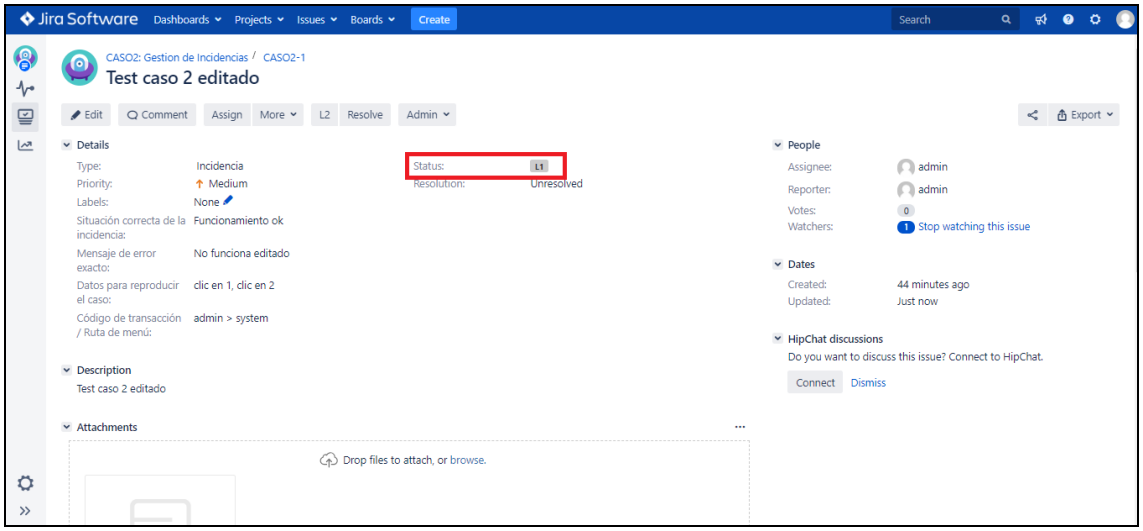


Figura 52: Visualización del cambio de estado de ticket CASO2

Cuando el ticket se encuentra en el estado L1 podemos pasarlo al estado L2 o Resolverlo.

4.4. Caso 3: Gestión de Peticiones con aprobación

Para este caso se plantea un proceso de gestión de peticiones con aprobación el cual obedece a las siguientes especificaciones.

4.4.1. Especificaciones

Las características del proceso de gestión de peticiones se describen a continuación:

- **Tipos de ticket**: Para la gestión de peticiones se podrá dar de alta los siguientes tipos de ticket:
    - Petición
- **Roles**: Los roles que intervendrán serán:
    - Requester
    - Approver
    - Dispatcher
    - Level 1 Support
    - Level 2 Support
- **Flujos de trabajo**: Las peticiones tendrán el siguiente flujo de trabajo:

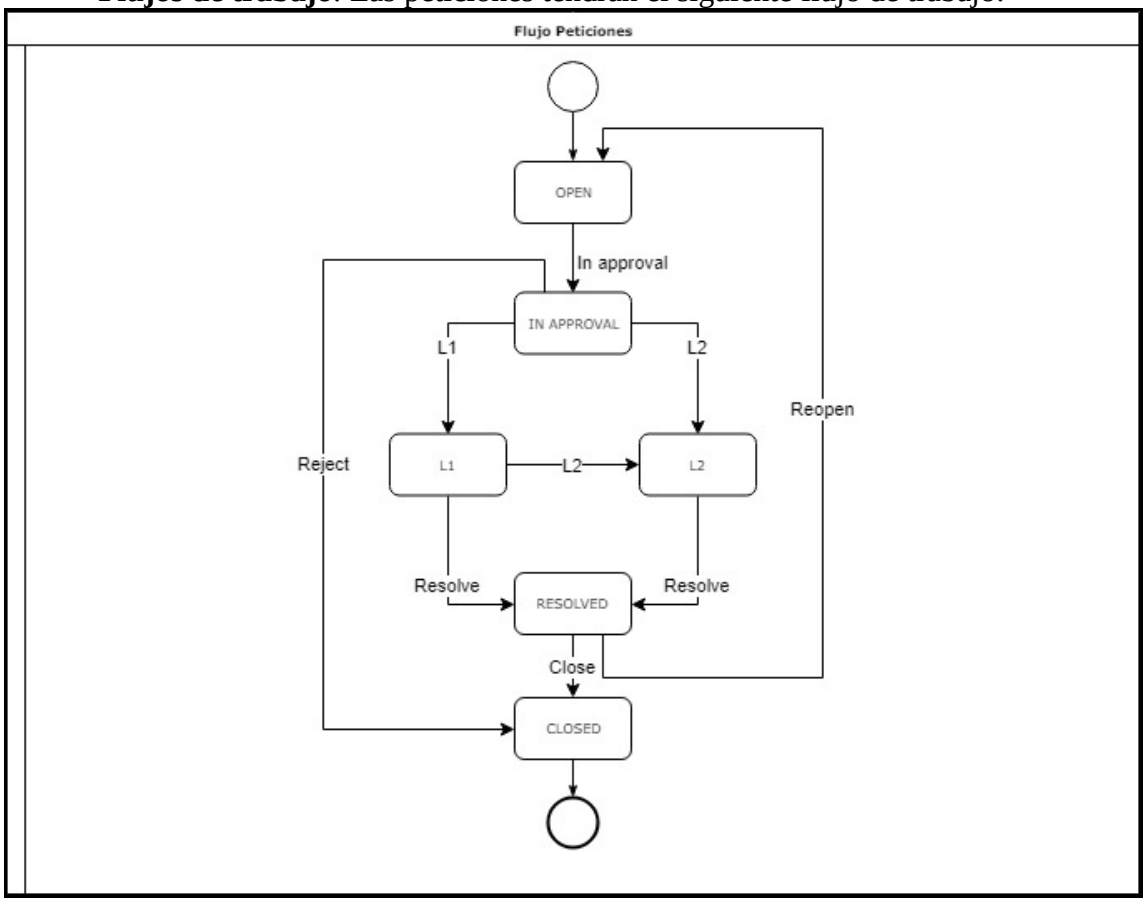


Figura 53: Flujo de trabajo de Peticiones

    - **Condiciones de flujo**: Desde el estado IN APPROVAL solo los usuarios con rol approver podrán realizar las acciones L1 y L2.
    - **Validaciones de flujo**: Desde el estado IN APPROVAL se deberá añadir un comentario para realizar las acciones L1 y L2.

- **Pantallas**: Los campos que se podrán informar tanto en el alta y modificación como en la visualización serán:
    - Sumario
    - Descripción

    - Prioridad
    - Área
    - Categoría
- **Notificaciones**: Las notificaciones que se enviarán son:
    - En la creación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En la modificación del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - En el cambio de estado del ticket:
        - Al creador del ticket
        - Al responsable del ticket
        - A todos los involucrados del ticket
    - Al resolver el ticket:
        - Al creador del ticket
    - Al asignar el ticket:
        - Al responsable del ticket
    - Al cerrar el ticket:
        - Al responsable del ticket
    - Al reabrir el ticket:
        - Al responsable del ticket
- **Permisos**: Los permisos que tendrán cada involucrado serán:
    - **Crear tickets**: personas con rol requester
    - **Ver tickets**: personas con rol requester, approver, dispatcher, level1 o level2
    - **Cambiar estados**: personas con rol requester, approver, dispatcher, level1 o level2
    - **Editar tickets**: personas con rol requester, approver, dispatcher, level1 o level2
    - **Añadir comentarios**: personas con rol requester, approver, dispatcher, level1 o level2
    - **Usuario asignable**: personas con rol level1 o level2
    - **Resolver tickets**: personas con rol level1 o level2
    - **Cerrar tickets**: personas con rol level1 o level2
    - **Asignar tickets**: personas con rol dispatcher
    - **Añadir documentos adjuntos**: personas con rol requester, approver, dispatcher, level1 o level2

Todas estas especificaciones las plasmamos en el JSON del Anexo III.

4.4.2. Comprobación elementos generados

Una vez ejecutado el proceso podemos acceder a la administración de la aplicación dentro del apartado Proyectos para ver el proyecto generado:

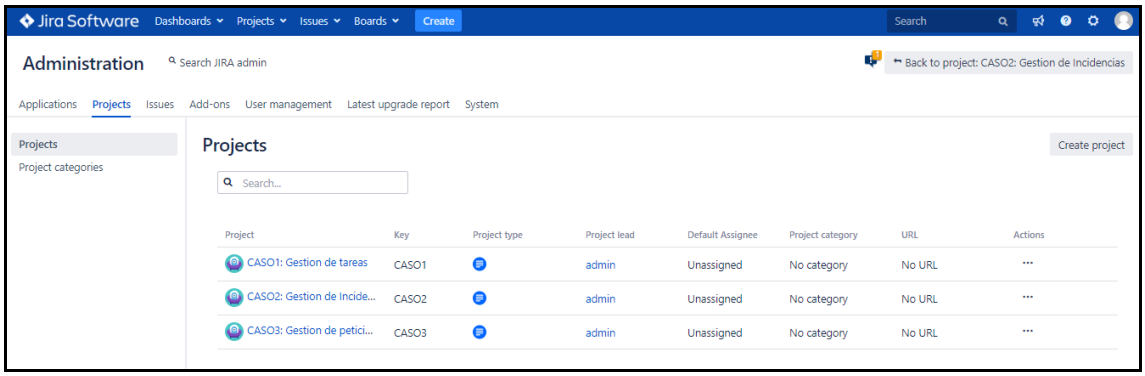


Figura 54: Menú administración proyectos JIRA

En dicho menú podemos observar el proyecto CASO3 con su información básica. Si hacemos clic en el accederemos a la configuración de los esquemas generados:

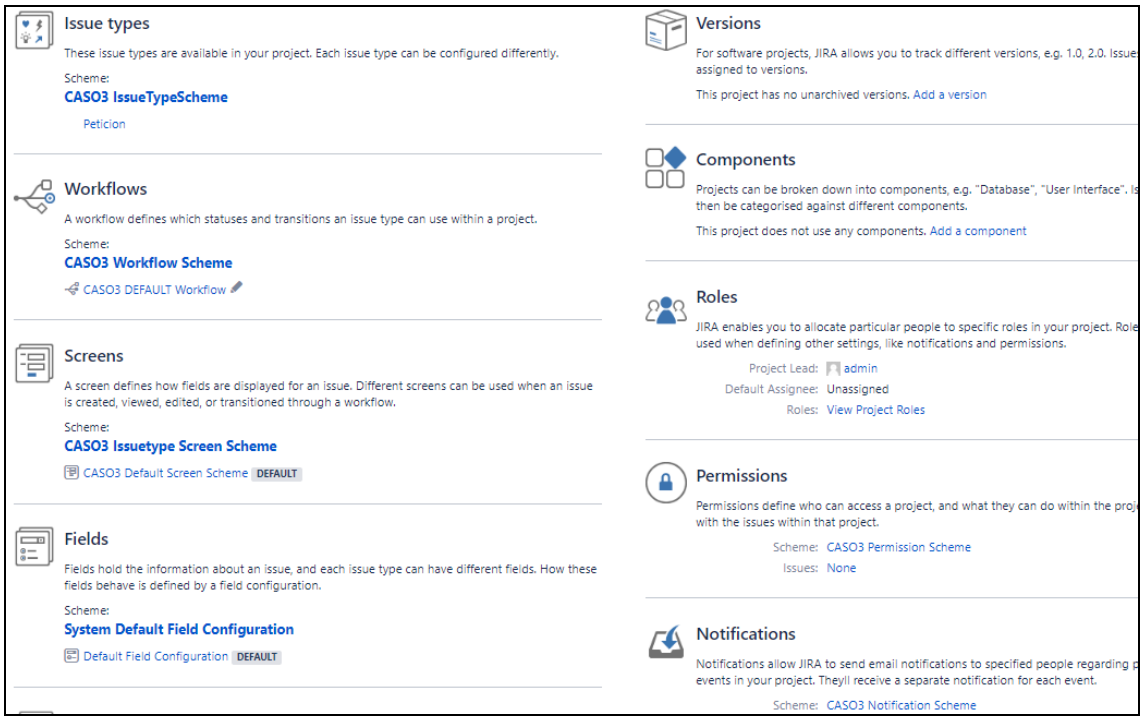


Figura 55: Detalle de los esquemas generados en el caso 3

- **Comprobación IssueType Scheme**

Si accedemos al IssueTypeScheme generado para el proyecto CASO3 podemos observar cómo se han asignado correctamente el tipo de ticket Peticion:

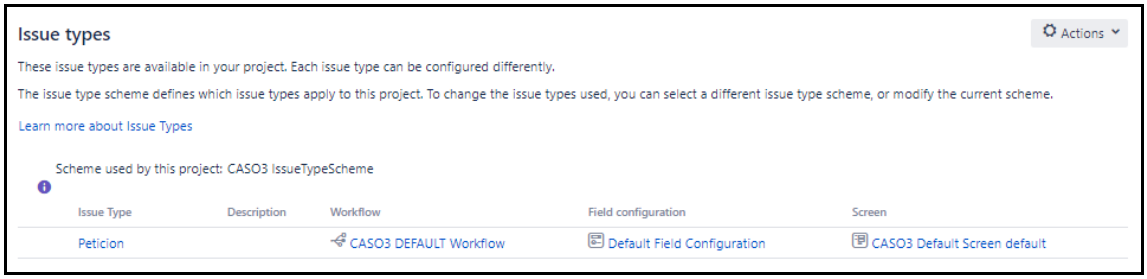


Figura 56: Detalle IssueTypeScheme del caso 3

- **Comprobación Role Scheme**

Si accedemos los roles generado para el proyecto CASO3 podemos observar cómo se ha asignado el grupo jira-administrators al rol Administrators y se ha asignado el usuario admin al rol Managers:

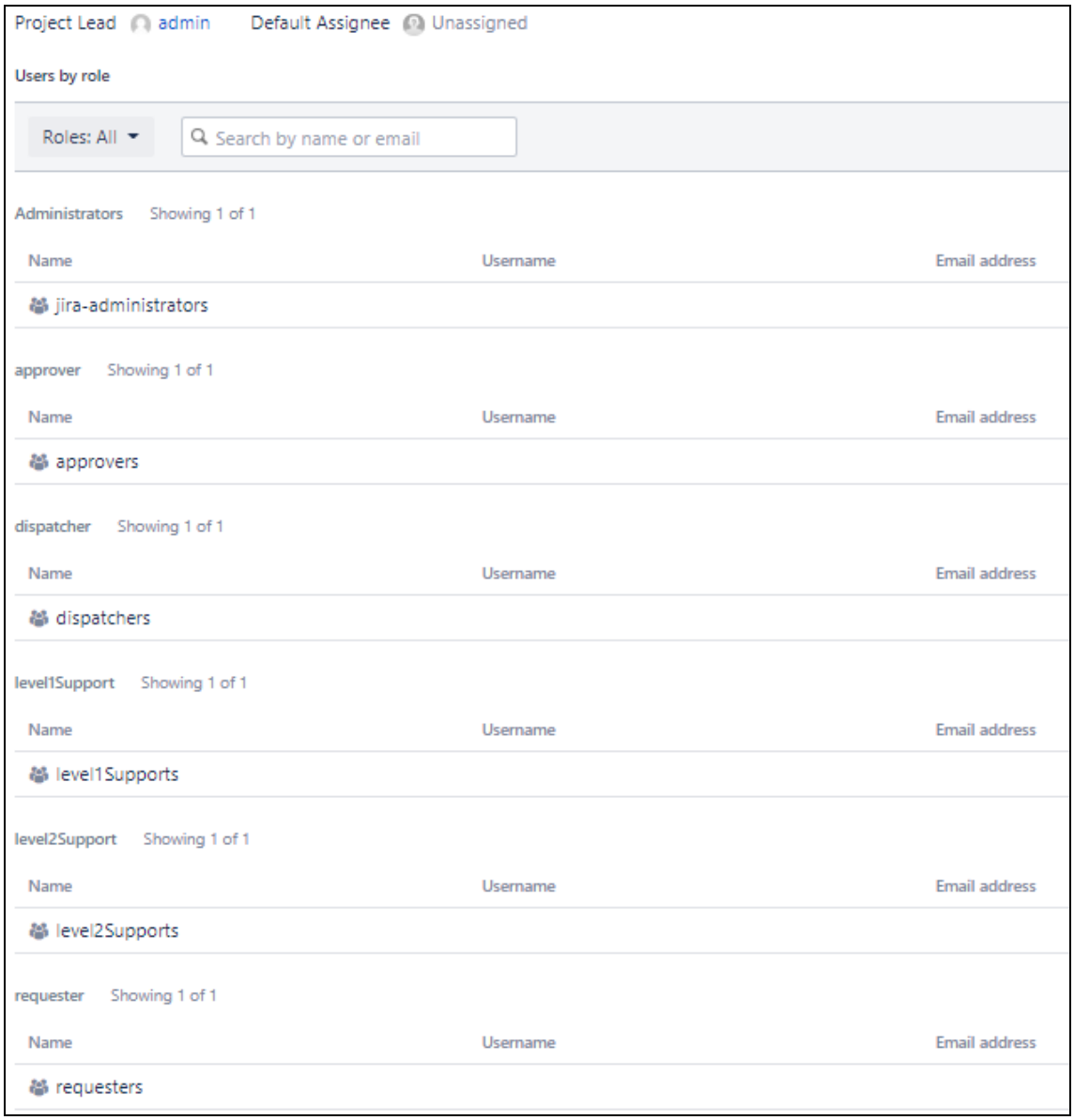


Figura 57: Detalle Roles del caso 3

- **Comprobación Workflow Scheme**

Si accedemos al WorkflowScheme generado para el proyecto CASO3 podemos observar cómo se ha asignado correctamente el flujo CASO3 DEFAULT Workflow al tipo de ticket Peticion:

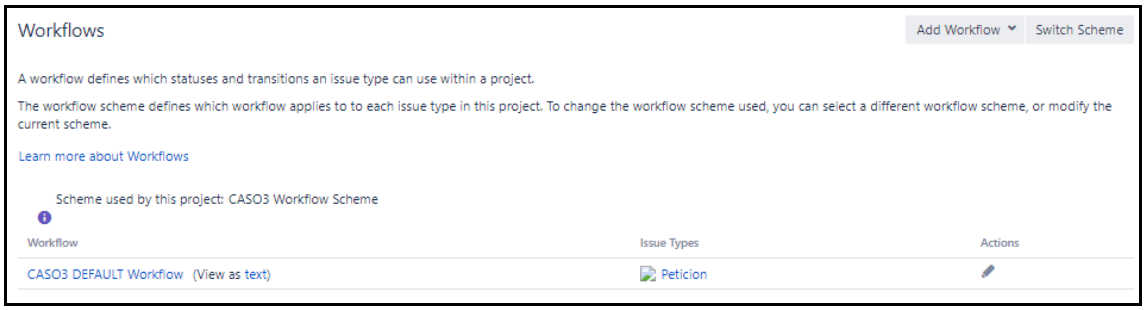


Figura 58: Detalle WorkflowScheme del caso 3

Además podemos ver el diagrama del flujo CASO3 DEFAULT Workflow:

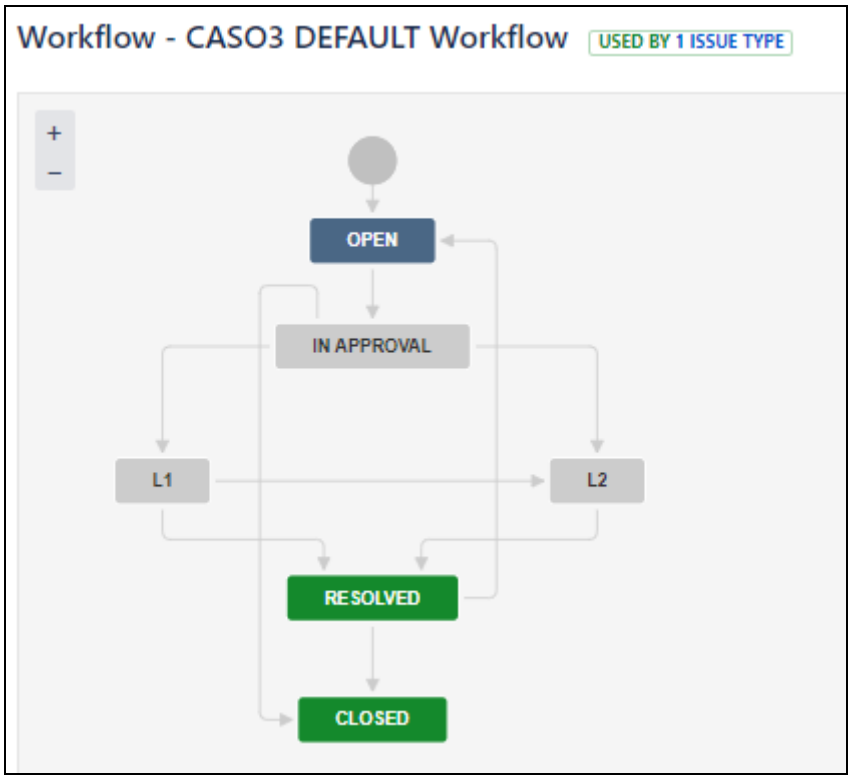


Figura 59: Detalle del Workflow del caso 3

- **Comprobación IssueType Screen Scheme**

Si accedemos al IssueTypeScreenScheme generado para el proyecto CASO3 podemos observar cómo se ha asignado correctamente el esquema de pantallas CASO3 DEFAULT Screen Scheme al tipo de ticket Peticion donde en las operaciones de creación, edición y vista tendremos la misma pantalla CASO3 DEFAULT Screen Default:

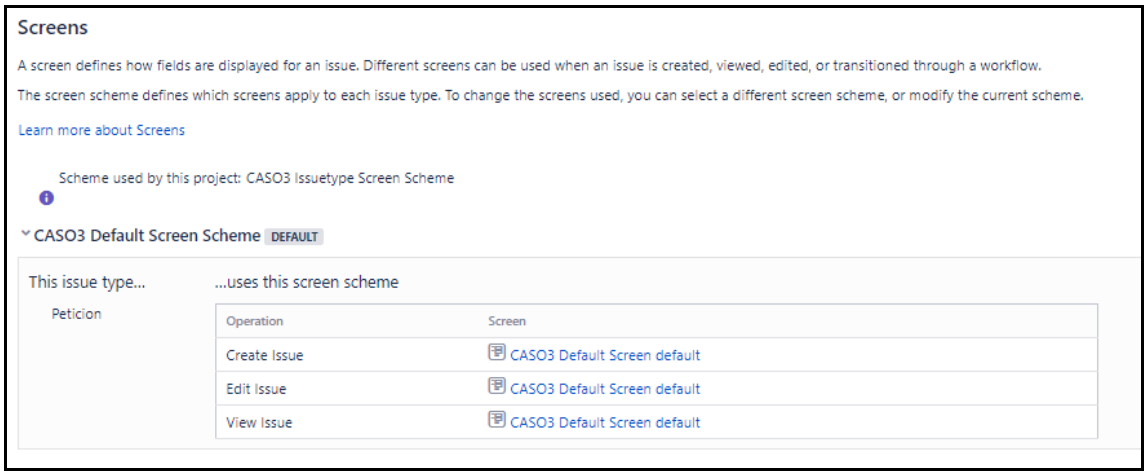


Figura 60: Detalle del IssueTypeScreenScheme del caso 3

Además podemos acceder a la pantalla CASO3 DEFAULT Screen Default y ver sus campos:

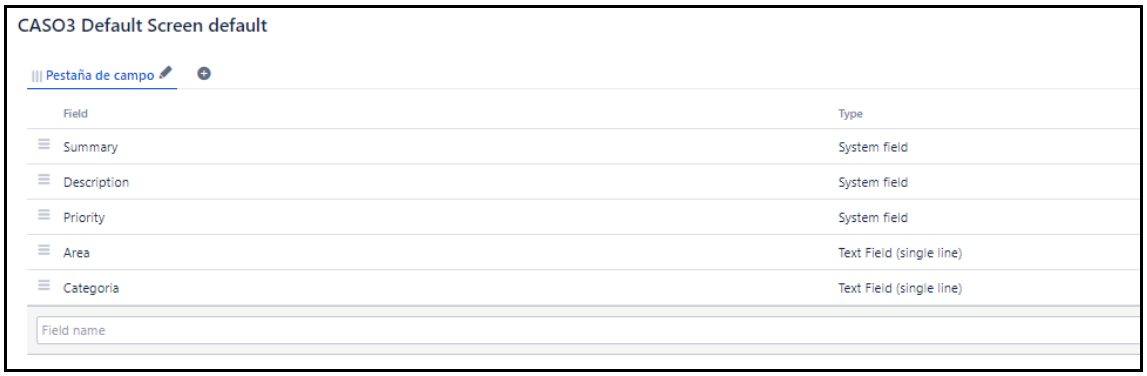
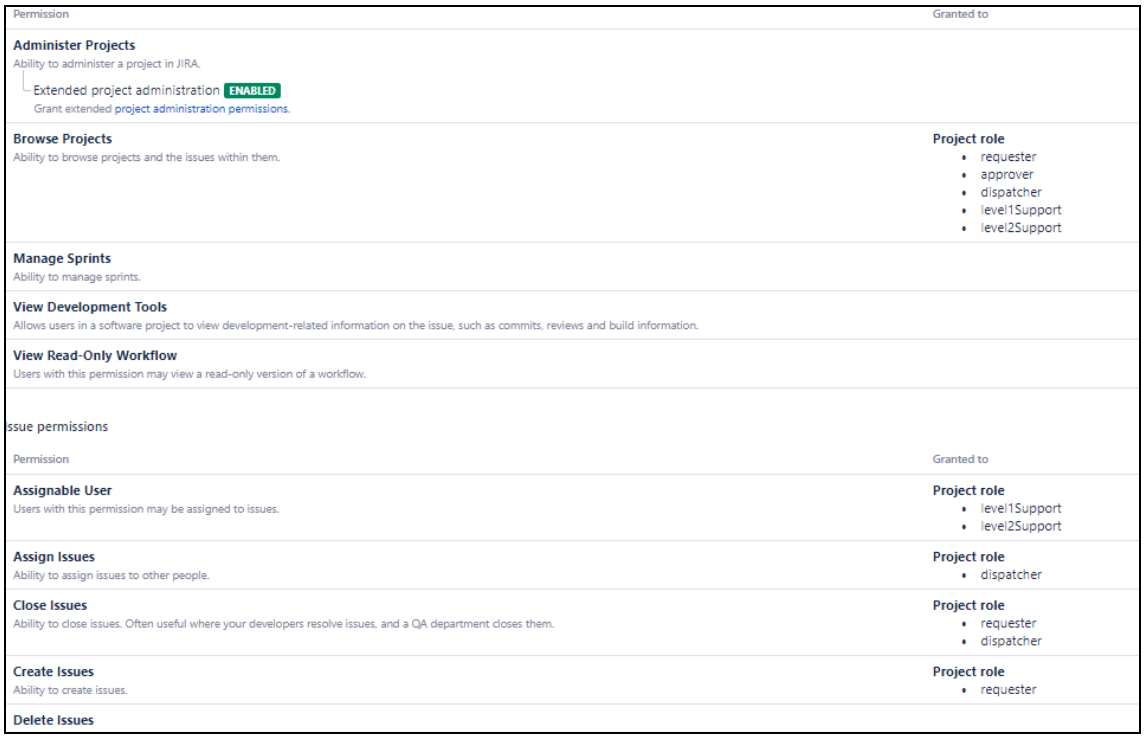


Figura 61: Detalle de la Screen del caso 3

- **Comprobación Permission Scheme**

Si accedemos al PermissionScheme generado para el proyecto CASO3 podemos observar los permisos otorgados:



Permission	Granted to
Administer Projects Ability to administer a project in JIRA. Extended project administration ENABLED Grant extended project administration permissions.	
Browse Projects Ability to browse projects and the issues within them.	**Project role** • requester • approver • dispatcher • level1Support • level2Support
Manage Sprints Ability to manage sprints.	
View Development Tools Allows users in a software project to view development-related information on the issue, such as commits, reviews and build information.	
View Read-Only Workflow Users with this permission may view a read-only version of a workflow.	

ssue permissions

Permission	Granted to
Assignable User Users with this permission may be assigned to issues.	**Project role** • level1Support • level2Support
Assign Issues Ability to assign issues to other people.	**Project role** • dispatcher
Close Issues Ability to close issues. Often useful where your developers resolve issues, and a QA department closes them.	**Project role** • requester • dispatcher
Create Issues Ability to create issues.	**Project role** • requester
Delete Issues	

Figura 62: Detalle del PermissionScheme del caso 3

- **Comprobación Notification Scheme**

Si accedemos al NotificationScheme generado para el proyecto CASO3 podemos observar las notificaciones activadas:

Events	Notifications
Issue Created	All Watchers Current Assignee Reporter
Issue Updated	All Watchers Current Assignee Reporter
Issue Assigned	Current Assignee
Issue Resolved	Reporter
Issue Closed	Current Assignee
Issue Commented	
Issue Comment Edited	
Issue Comment Deleted	
Issue Reopened	Current Assignee
Issue Deleted	
Issue Moved	
Work Logged On Issue	
Work Started On Issue	
Work Stopped On Issue	
Issue Worklog Updated	
Issue Worklog Deleted	
Generic Event	All Watchers Current Assignee Reporter

Figura 63: Detalle del NotificationScheme del caso 3

4.4.3. Código individual

Para aplicar la validación de flujo que valida la introducción de un comentario en la transición que va del estado IN APPROVAL al estado L1 o L2 debemos realizar los siguientes pasos:

- Accedemos al flujo CASO3 DEFAULT Workflow:

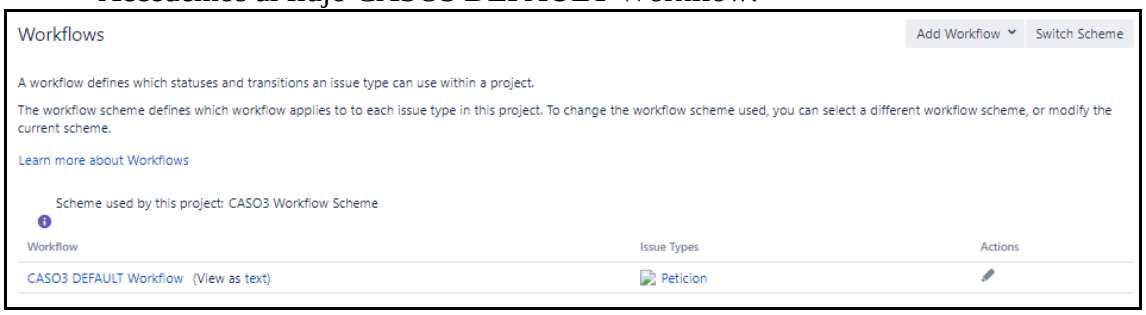


Figura 64: Detalle del flujo del caso 3

- Editamos el flujo y seleccionamos las dos transiciones afectadas (primero una y luego la otra) y acedemos a la sección de validaciones:

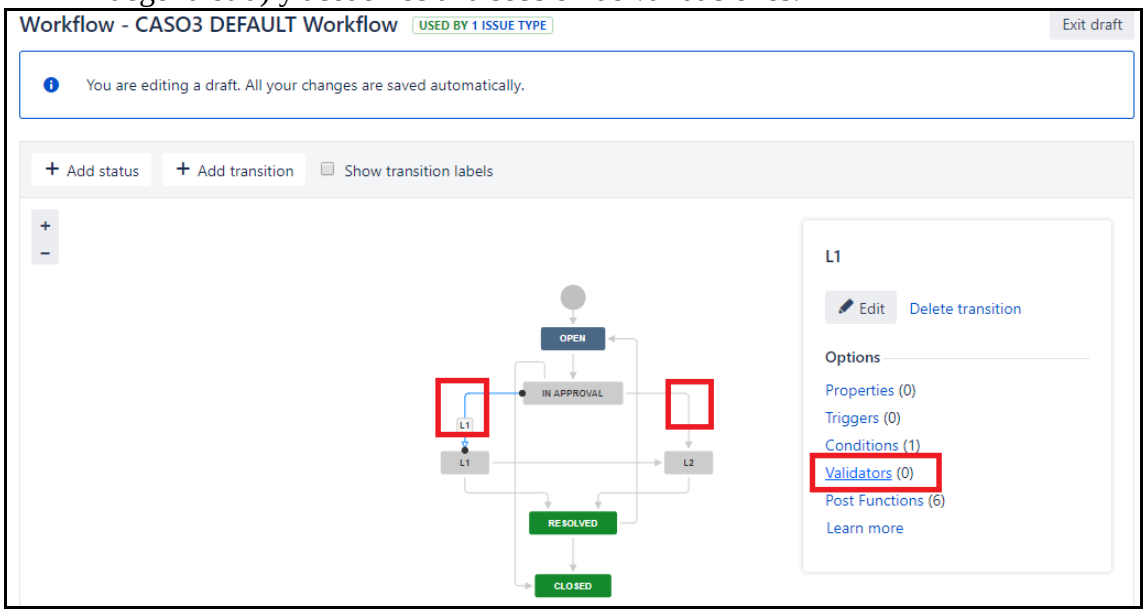


Figura 65: Edición del flujo del caso 3

- Añadimos el siguiente código como Script validator:

```
import com.opensymphony.workflow.InvalidInputException


if (lastComment == null) {
      invalidInputException = new InvalidInputException("Se debe informar un
comentario");
}
```

- Publicamos el flujo

4.4.4. Pruebas de funcionamiento

Para realizar las pruebas de funcionamiento vamos a realizar acciones con tickets. Estas acciones van a ser las siguientes:

- Creación de ticket
- Edición de ticket
- Asignación de ticket
- Añadir comentarios en un ticket
- Añadir documentos adjuntos en un ticket
- Cambiar el estado de un ticket

4.4.4.1. Creación de tickets

Para la creación de un ticket hacemos clic en el botón create y seleccionamos el Project CASO3. Posteriormente seleccionamos el issuetype Peticion y rellenamos el resto de información necesaria:

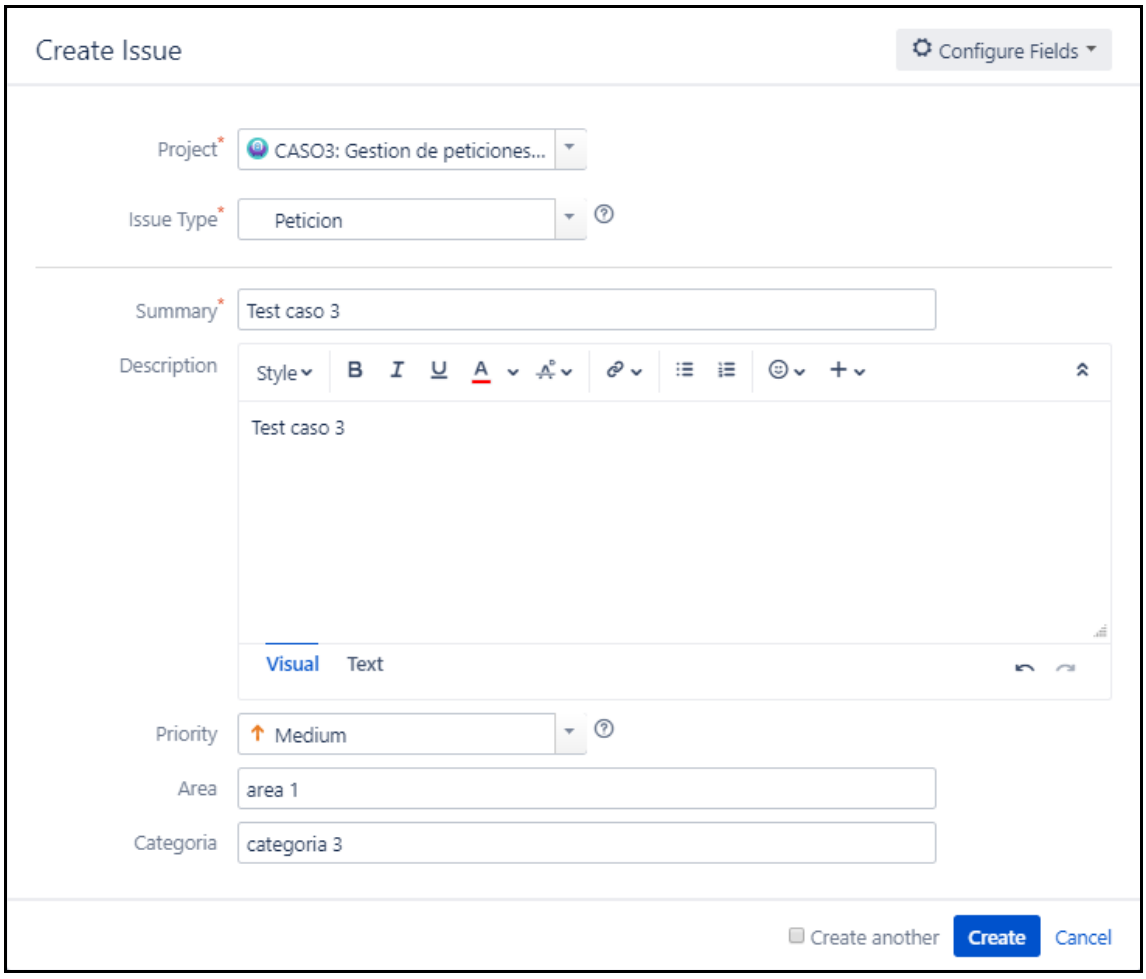


Figura 66: Creación de ticket CASO3

Una vez creado podemos visualizar su información accediendo a el:

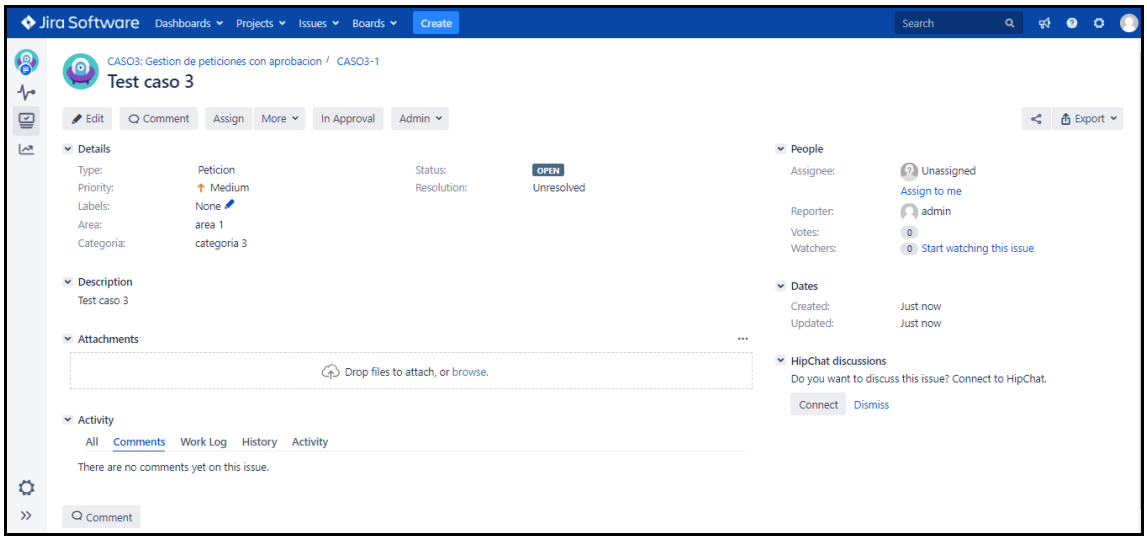


Figura 67: Visualización ticket CASO3

4.4.4.2. Edición de tickets

Para la edición de un ticket, en la pantalla de visualización del ticket, hacemos clic en el botón Edit y modificamos los campos que queremos editar:

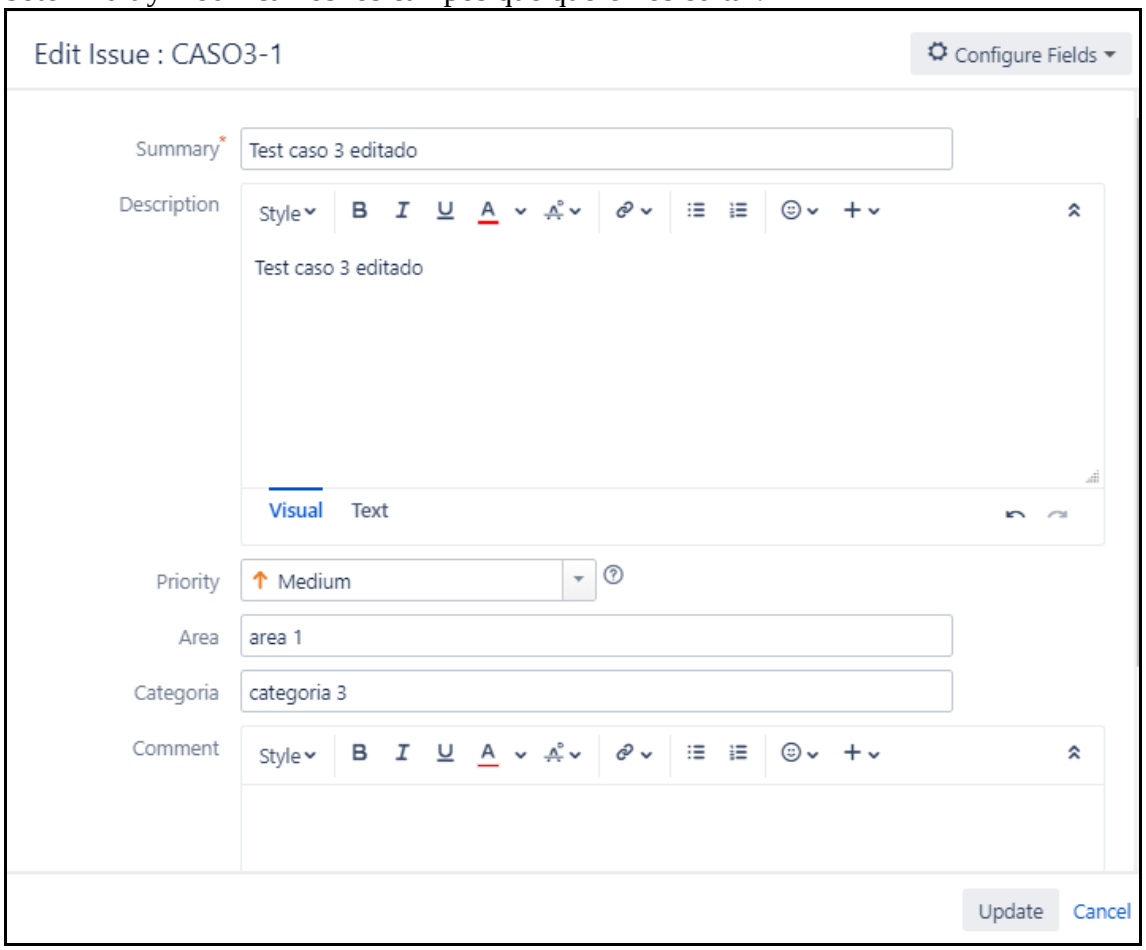


Figura 68: Edición de ticket CASO3

4.4.4.3. Asignación de tickets

Para asignar un ticket del CASO3 hacemos clic en el botón Assign y seleccionamos uno de los usuarios que pueden ser asignados:

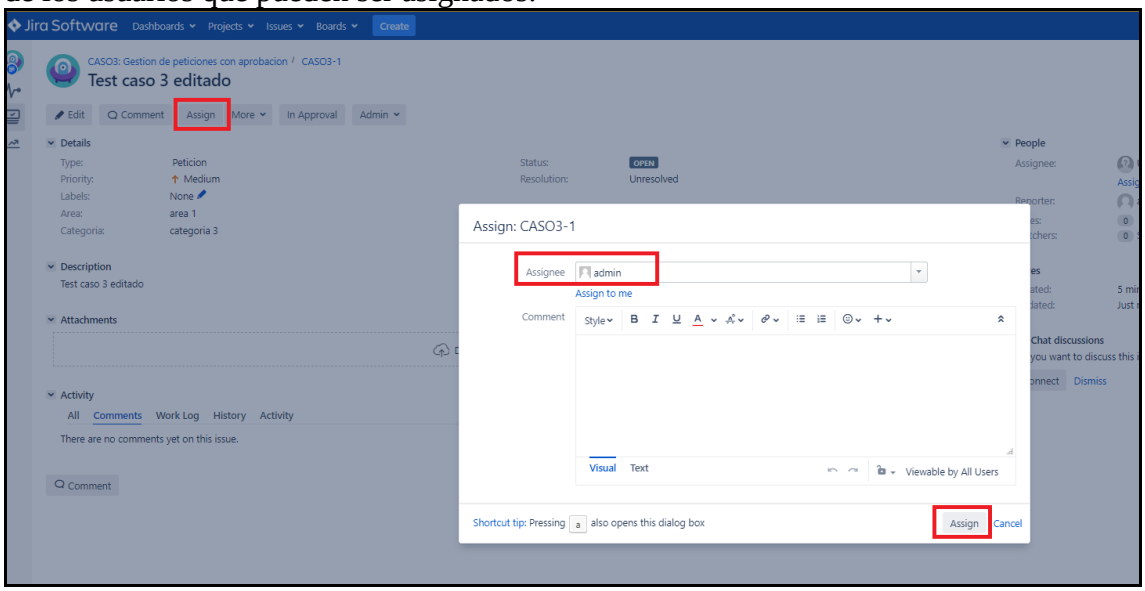


Figura 69: Asignación de ticket CASO 3

4.4.4.4. Añadir comentarios en un ticket

Para añadir un comentario en un ticket del CASO3 hacemos clic en el botón Comment e introducimos el texto del comentario:

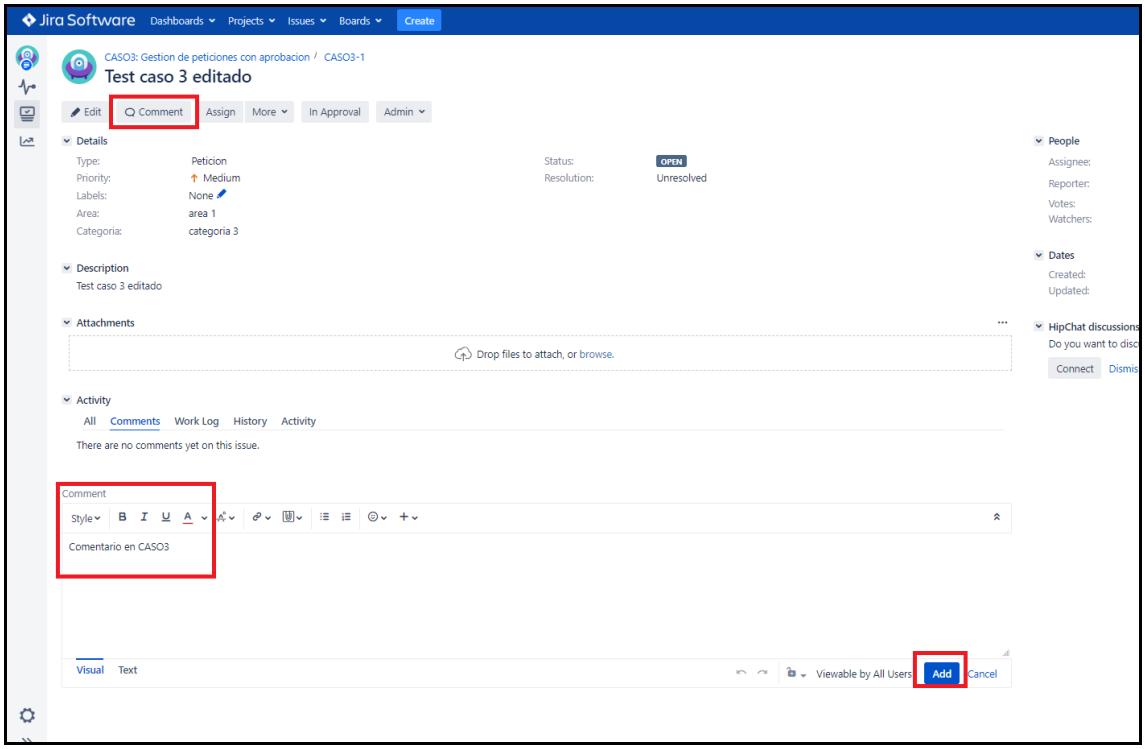


Figura 70: Añadir comentario en ticket CASO3

4.4.4.5. Añadir documentos adjuntos en un ticket

Para añadir un documento adjunto en un ticket del CASO3 arrastramos al ticket el documento desde el Escritorio:

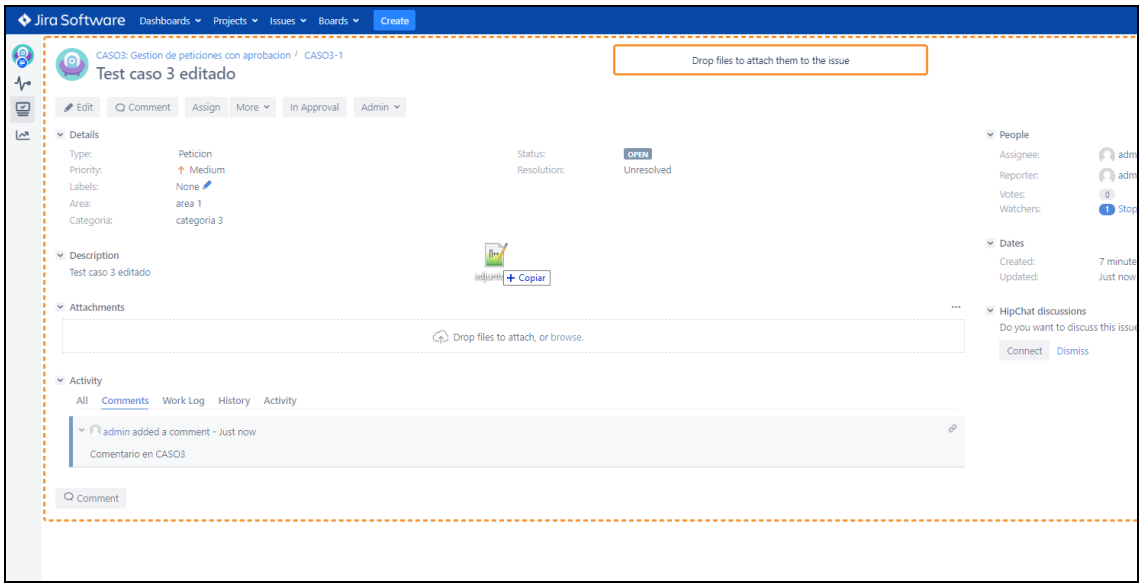


Figura 71: Añadir documento adjunto en ticket CASO3

4.4.4.6. Cambiar el estado de un ticket

En el momento de la creación el ticket permanece en el estado OPEN tal y como hemos especificado. En este estado la única transición disponible es IN APPROVAL que cambiará el estado del ticket a IN APPROVAL. Para realizar el cambio de estado hacemos clic en el botón IN APPROVAL:

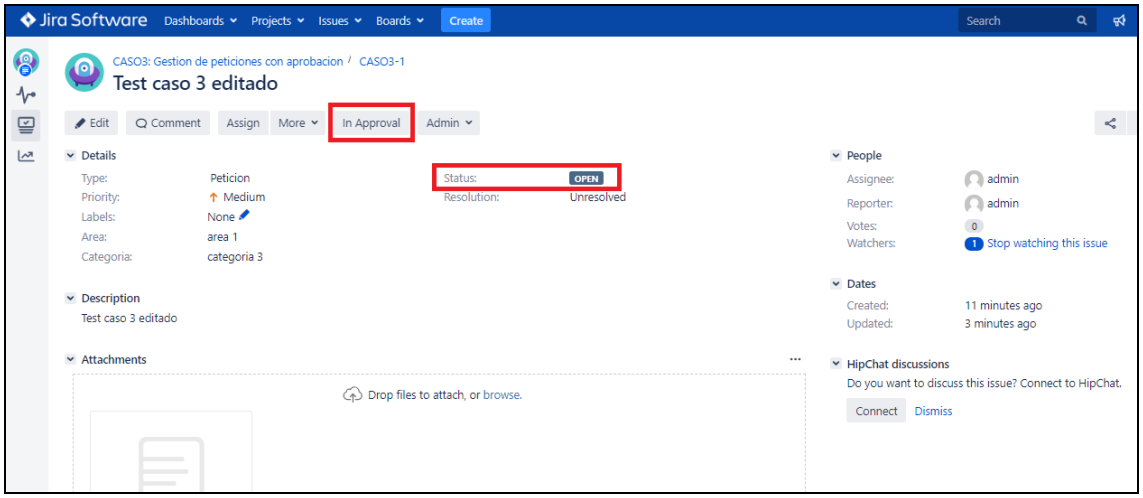


Figura 72: Cambio de estado de ticket CASO3

Al finalizar la transición podemos ver como el ticket ha cambiado al estado IN APPROVAL. En este punto el ticket podrá realizar 3 acciones. Reject que cambiará directamente el etado a CLOSED, L1 que cambiará el estado del ticket a L1 o L2 que

cambiará el estado del ticket a L2. Estas acciones las puede realizar un miembro con rol approver. Si no pertenece a approver no tiene disponibles estas acciones.

A partir de aquí el flujo sigue igual que en el CASO2.

4.5. Análisis de las configuraciones generadas

Las pruebas aplicadas se analizan frente a la siguiente tabla donde se muestra para cada caso el número de elementos generados.

Detalle	**Caso 1**	**Caso 2**	**Caso 3**
Tipos de ticket generados	2	1	1
Roles generados	2	4	5
Estados del flujo generado	2	5	6
Transiciones del flujo generado	2	8	10
Condiciones de flujo generadas	0	0	2
Validaciones de flujo generados	0	0	0
Campos generados	3	7	5
Notificaciones generadas	3	7	7
Permisos generados	4	10	10
Código individual introducido manualmente	0	0	1

Tabla 16: Análisis de las configuraciones generadas

Observando los resultados podemos ver que, como norma general, cada caso genera un número mayor de elementos a excepción de los tipos de ticket y campos generados. En relación a los tipos de ticket podemos ver que en el Caso 1 generamos un tipo de ticket más, Subtarea. En relación a los campos vemos que en el Caso 2 generamos 2 campos más que en el Caso 3.

Por último podemos observar que la complejidad de las validaciones de aprobación del Caso 3 nos obliga a introducir manualmente código individual para poder cumplir con las especificaciones del proceso.

5. CONCLUSIONES Y TRABAJOS FUTUROS

Este trabajo pretendía construir una herramienta para la generación automática de configuración de procesos en JIRA mediante desarrollo dirigido por modelos. Con el uso de DSM se evidencia la generación automática de configuraciones JIRA donde cada vez es menos necesario el conocimiento de un administrador JIRA. Las configuraciones JIRA son más entendibles y modificables por terceros, llegando a ser entendidas por los expertos del proceso y negocio.

El proyecto genera configuraciones JIRA para varios procesos, presentando una buena opción para el futuro de la administración de JIRA.

La herramienta construida proporciona un lenguaje de dominio específico que, en relación al uso, la curva de aprendizaje aumenta para programadores, porque aprender un nuevo lenguaje es de mayor inversión de tiempo que aplicar uno conocido, pero disminuye para programadores que se inician en este campo. La herramienta posee una gramática sencilla y fácil de usar.

El nuevo lenguaje construido permite a los administradores JIRA ahorro de tiempo en sus configuraciones y contribuye en la calidad ya que la gramática definida no da lugar para que el administrador JIRA se equivoque.

Los resultados obtenidos respecto a los objetivos de trabajar con metodología de desarrollo dirigido por modelos vistos en la sección 2.3.1 son los siguientes:

Velocidad en el desarrollo/configuración: Este objetivo lo vemos cumplido dado que las configuraciones manuales realizadas son mínimas frente a las generadas automáticamente. Tanto en el caso 1 como en el caso 2 no es necesario realizar configuraciones manualmente. En el caso 3, únicamente es necesario insertar de forma manual el código Groovy de la validación que valida la inserción de un comentario en la transición desde el estado IN APPROVAL hacia los estados L1 y L2.

Calidad: Este objetivo se obtiene en el momento de la generación de configuraciones JIRA estandarizadas y no tienen intervención humana, lo que connota un alto grado de calidad. Las validaciones dentro de la herramienta evitan que el punto de entrada tenga posibles errores que el usuario pueda introducir escribiendo el DSL.

Gestión de la complejidad: Este objetivo se cumple teniendo en cuenta que la complejidad de la configuración del proceso está modelada directamente en la herramienta, y las características que no están ligadas a la lógica del proceso no se deben trabajar. De esa parte se encarga el DSL, interpretando y generando las configuraciones que se requieren.

Entorno productivo: Este objetivo lo vemos cumplido si comparamos la forma manual de realizar configuraciones en JIRA y la forma de generarlas automáticamente mediante el modelo. De la forma manual tenemos que ir uno por uno a todos los esquemas de configuración y construir dicha configuración. Si quisiéramos replicar dicha configuración en otra instancia de la aplicación deberíamos volver a realizar la misma configuración por duplicado. Con la herramienta construida, escribiríamos el modelo de la configuración y lanzaríamos el generador para generar la configuración

automáticamente. Si quisiéramos replicar la misma configuración en otra instancia, simplemente debemos lanzar el generador con el mismo modelo.

A corto plazo la herramienta puede ser evolucionada para contemplar la generación de los nuevos esquemas de configuración que han surgido en las nuevas versiones de JIRA como por ejemplo los esquemas de prioridad que permiten tener diferentes valores de prioridad para cada proceso.

Para permitir que la herramienta permita la configuración de procesos de tipo Service Desk que permiten la gestión de SLA, Colas de trabajo, y Portales de usuario personalizados deberemos modificar el DSL para incluir estos elementos y construir su generación. Lo mismo ocurre con los elementos de procesos de Software que trabajan con metodologías agiles y permiten la visualización de tableros Kanban o usar Scrum.

Como trabajo a medio/largo plazo está la generación de las configuraciones de procesos en otras herramientas del mismo ámbito que JIRA como pueden ser ServiceNow, Remedy o EasyVista de manera que con un mismo modelo podamos configurar aplicaciones diferentes.

6. BIBLIOGRAFIA

[1] Wim Bast, Anneke Kleppe, Jos Warmer. 2003. MDA Explained, The Model Driven Architecture: Practice and Promise. Addison-Wesley.

[2] Hammer, M. and Champy, J. 1993. Reengineering the Corporation: A Manifesto for Business Revolution. Harper Business, New York, NY.

[3] Völter, M. and Stahl, T. 2006. Model-Driven Software Development, John Wiley & Sons Ltd.

[4] Cañadas, J., Palma, J., & Túnez, S. 2011. Defining the semantics of rule-based Web applications through model-driven development. International Journal of Applied Mathematics and Computer Science.

[5] Kraus, A., Knapp, A., & Koch, N. 2007. Model-driven generation of web applications in UWE. Model-Driven Web Engineering

[6] Ceri, S., & Fraternali, P. 2002. Conceptual Modeling of Data-Intensive Web Applications. *IEEE INTERNET COMPUTING*

[7] P. Hudak. 1996. Building domain-specific embedded languages. ACM Computing Surveys.

[8] A. van Deursen, P. Klint. 2002. Domain-Specific Language Design Requires Feature Description. CIT Journal of Computing and Information Technology, Special Issue on Domain-Specific Languages, Part II, Eds. R. Laemmel, M. Mernik.

[9] Igor, D., Gordana, M., Branko, P., & Maja, T. 2010. Computer Science and Information Systems. A domain-specific language for defining static structure of database applications.

[10] Atlassian, «Atlassian JIRA - Server API,» 2018. [En línea]. Available: https://docs.atlassian.com/software/jira/docs/api/latest/.

[11] Atlassian, «JIRA Core: Funciones,» Atlassian, 2018. [En línea]. Available: https://es.atlassian.com/software/jira/core/features.

[12] Atlassian, «JIRA Core Server Documentation,» Atlassian, 2018. [En línea]. Available: https://confluence.atlassian.com/jiracoreserver/jira-core-server-7-12-documentation-938846148.html.

[13] Atlassian, «Migrating from other issue trackers,» 2016. [En línea]. Available: https://confluence.atlassian.com/adminjiraserver073/migrating-from-other-issue-trackers-861253679.html#Migratingfromotherissuetrackers-Built-inimporters.

[14] R. Sagar, Mastering JIRA, Packt Publishing, 2015.

[15] D. König, P. King, G. Laforge, H. D'Arcy, C. Champeau, E. Pragt y J. Skeet, Groovy in Action, Second Edition, Manning Publications, 2015.

[16] Atlassian, «Gadget developer documentation,» Atlassian, 2017. [En línea]. Available: https://developer.atlassian.com/server/framework/gadgets/.

[17] Atlassian, «Configuring dashboards,» Atlassian, 2018. [En línea]. Available: https://confluence.atlassian.com/jiracoreserver079/configuring-dashboards-950290117.html.

7. SIGLAS

API	Application Programming Interface
CIM	Computer Independent Model
CM	Code Model
CSS	Cascading Style Sheets
DSL	Domain-Specific Language
DSM	Domain-Specific Modeling
GPL	General Purpose Language
HTML	Hypertext Markup Language
JSON	JavaScript Object Notation
MDA	Model-Driven Architecture
MDD	Model-Driven Development
OMG	Object Management Group
PIM	Platform Independent Model
PSM	Platform Specific Model
SQL	Structured Query Language
TTM	Transformation Technology Models
UML	Unified Modeling Language
XML	eXtensible Markup Language

8. ANEXOS

8.1. Anexo I: JSON Caso 1 (Gestión de tareas)


```
{
	"key": "CASO1",
	"name": "CASO1: Gestion de tareas",
	"description": "CASO1: Gestion de tareas",
	"projectTypeKey": "business",
	"lead": "admin",
	"issueTypes": [
		{
		  "name": "Tarea",
		  "subtask": false
		},
		{
		  "name": "Subtarea",
		  "subtask": true
		}
	],
	"assigneeType": "unassigned",
	"roles": [
		{
			"name": "Administrators",
			"actors": [
				{
				  "name": "jira-administrators",
				  "type": "Group"
				}
			]
		},
		{
			"name": "Managers",
			"actors": [
				{
				  "name": "admin",
				  "type": "User"
				}
			]
		}
	],
	"workflowScheme": [
		{
			"issueType": "DEFAULT",
			"initialAction": {
				"validators": [
					{
						"parameter": "Create Issue",
						"name": "PermissionValidator"
					}
				],
				"postfunctions": [
					{
						"name": "IssueCreateFunction"
					},
```

```
                                {
                                        "name": "IssueReindexFunction"
                                },
                                {
                                        "parameter": "1",
                                        "name": "FireIssueEventFunction"
                                }
                        ],
                        "name": "Create",
                        "to": "Open",
                        "conditions": []
                },
                "transitions": [
                        {
                                "validators": [],
                                "postfunctions": [
                                  {
                                        "name": "UpdateIssueStatusFunction"
                                  },
                                  {
                                        "name": "AssignToCurrentUserFunction"
                                  },
                                  {
                                        "name": "CreateCommentFunction"
                                  },
                                  {
                                        "name":"GenerateChangeHistoryFunction"
                                  },
                                  {
                                        "name": "IssueReindexFunction"
                                  },
                                  {
                                        "name": "FireIssueEventFunction",
                                        "parameter": "13"

                                  }
                                ],
                                "name": "Close",
                                "from": "Open",
                                "to": "Closed",
                                "conditions": []
                        }
                ]
        }
],
"issueTypeScreenScheme": [
        {
                "issueType": "DEFAULT",
                "operations": {
                        "default": [
                                {
                                        "tab": "Pestaña de campo",
                                        "fields": [
                                                {
                                                        "name": "Summary",
                                                        "type": "jiraDefault"
                                                },
                                                {
                                                        "name": "Description",
```

```
                                                "type": "jiraDefault"
                                        },
                                        {
                                                "name": "Priority",
                                                "type": "jiraDefault"
                                        }
                                ]
                        }
                ],
                "view": [],
                "edit": [],
                "create": []
        }
    }
],
"notificationScheme": {
    "issue_deleted": [],
    "issue_comment_edited": [],
    "issue_worklog_deleted": [],
    "issue_reopened": [],
    "issue_closed": [],
    "issue_worklog_updated": [],
    "issue_assigned": [],
    "issue_comment_deleted": [],
    "work_logged_on_issue": [],
    "issue_resolved": [],
    "work_started_on_issue": [],
    "work_stopped_on_issue": [],
    "issue_commented": [],
    "issue_moved": [],
    "issue_created": [
        {
            "parameter": "",
            "type": "Current_Assignee"
        },
        {
            "parameter": "",
            "type": "Current_Reporter"
        },
        {
            "parameter": "",
            "type": "All_Watchers"
        }
    ],
    "issue_updated": [
        {
            "parameter": "",
            "type": "Current_Assignee"
        },
        {
            "parameter": "",
            "type": "Current_Reporter"
        },
        {
            "parameter": "",
            "type": "All_Watchers"
        }
    ],
    "generic_event": [
```

```
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ]
},
"issueSecurityScheme": "",
"permissionScheme": {
        "add_comments": [],
        "manage_sprints_permission": [],
        "delete_all_comments": [],
        "edit_all_comments": [],
        "create_issues": [
                {
                        "parameter": "Administrators",
                        "type": " projectrole"
                }
        ],
        "schedule_issues": [],
        "set_issue_security": [],
        "administer_projects": [],
        "delete_all_attachments": [],
        "transition_issues": [
                {
                        "parameter": "Managers",
                        "type": " projectrole"
                }
        ],
        "edit_all_worklogs": [],
        "view_dev_tools": [],
        "link_issues": [],
        "delete_all_worklogs": [],
        "assignable_user": [],
        "resolve_issues": [],
        "work_on_issues": [],
        "edit_own_comments": [],
        "view_readonly_workflow": [],
        "delete_own_attachments": [],
        "delete_own_comments": [],
        "manage_watchers": [],
        "edit_issues": [
                {
                        "parameter": "Managers",
                        "type": " projectrole"
                }
        ],
        "close_issues": [],
        "move_issues": [],
        "edit_own_worklogs": [],
        "delete_own_worklogs": [],
        "modify_reporter": [],
```

```
                "view_voters_and_watchers": [],
                "delete_issues": [],
                "browse_projects": [
                        {
                                "parameter": "Managers",
                                "type": " projectrole"
                        }
                ],
                "assign_issues": [],
                "servicedesk_agent": [],
                "create_attachments": []
        }
}
```

8.2. Anexo II: JSON Caso 2 (Gestión de incidencias)


```
{
	"key": "CASO2",
	"name": "CASO2: Gestion de Incidencias",
	"description": "CASO2: Gestion de Incidencias",
	"projectTypeKey": "business",
	"lead": "admin",
	"issueTypes": [
		{
		  "name": "Incidencia",
		  "subtask": false
		}
	],
	"assigneeType": "unassigned",
	"roles": [
		{
			"name": "requester",
			"actors": [
				{
				  "name": "requesters",
				  "type": "Group"
				}
			]
		},
		{
			"name": "dispatcher",
			"actors": [
				{
				  "name": "dispatchers",
				  "type": "Group"
				}
			]
		},
		{
			"name": "level1Support",
			"actors": [
				{
				  "name": "level1Supports",
				  "type": "Group"
				}
			]
		},
		{
			"name": "level2Support",
			"actors": [
				{
				  "name": "level2Supports",
				  "type": "Group"
				}
			]
		}
	],
	"workflowScheme": [
		{
			"issueType": "DEFAULT",
			"initialAction": {
```

```
                "validators": [
                        {
                                "parameter": "Create Issue",
                                "name": "PermissionValidator"
                        }
                ],
                "postfunctions": [
                        {
                                "name": "IssueCreateFunction"
                        },
                        {
                                "name": "IssueReindexFunction"
                        },
                        {
                                "parameter": "1",
                                "name": "FireIssueEventFunction"
                        }
                ],
                "name": "Create",
                "to": "Open",
                "conditions": []
        },
        "transitions": [
                {
                        "validators": [],
                        "postfunctions": [
                          {
                                "name": "UpdateIssueStatusFunction"
                          },
                          {
                                "name": "AssignToCurrentUserFunction"
                          },
                          {
                                "name": "CreateCommentFunction"
                          },
                          {
                                "name":"GenerateChangeHistoryFunction"
                          },
                          {
                                "name": "IssueReindexFunction"
                          },
                          {
                                "name": "FireIssueEventFunction",
                                "parameter": "13"

                          }
                        ],
                        "name": "L1",
                        "from": "Open",
                        "to": "L1",
                        "conditions": []
                },
                {
                        "validators": [],
                        "postfunctions": [
                          {
                                "name": "UpdateIssueStatusFunction"
                          },
                          {
```

```
                    "name": "AssignToCurrentUserFunction"
                },
                {
                    "name": "CreateCommentFunction"
                },
                {
                    "name":"GenerateChangeHistoryFunction"
                },
                {
                    "name": "IssueReindexFunction"
                },
                {
                    "name": "FireIssueEventFunction",
                    "parameter": "13"

                }
            ],
            "name": "L2",
            "from": "Open",
            "to": "L2",
            "conditions": []
        },
        {
            "validators": [],
            "postfunctions": [
                {
                    "name": "UpdateIssueStatusFunction"
                },
                {
                    "name": "AssignToCurrentUserFunction"
                },
                {
                    "name": "CreateCommentFunction"
                },
                {
                    "name":"GenerateChangeHistoryFunction"
                },
                {
                    "name": "IssueReindexFunction"
                },
                {
                    "name": "FireIssueEventFunction",
                    "parameter": "13"

                }
            ],
            "name": "L2",
            "from": "L1",
            "to": "L2",
            "conditions": []
        },
        {
            "validators": [],
            "postfunctions": [
                {
                    "name": "UpdateIssueStatusFunction"
                },
                {
                    "name": "AssignToCurrentUserFunction"
```

```
            },
            {
                "name": "CreateCommentFunction"
            },
            {
                "name":"GenerateChangeHistoryFunction"
            },
            {
                "name": "IssueReindexFunction"
            },
            {
                "name": "FireIssueEventFunction",
                "parameter": "13"

            }
        ],
        "name": "Resolve",
        "from": "L1",
        "to": "Resolved",
        "conditions": []
    },
    {
        "validators": [],
        "postfunctions": [
            {
                "name": "UpdateIssueStatusFunction"
            },
            {
                "name": "AssignToCurrentUserFunction"
            },
            {
                "name": "CreateCommentFunction"
            },
            {
                "name":"GenerateChangeHistoryFunction"
            },
            {
                "name": "IssueReindexFunction"
            },
            {
                "name": "FireIssueEventFunction",
                "parameter": "13"

            }
        ],
        "name": "Resolve",
        "from": "L2",
        "to": "Resolved",
        "conditions": []
    },
    {
        "validators": [],
        "postfunctions": [
            {
                "name": "UpdateIssueStatusFunction"
            },
            {
                "name": "AssignToCurrentUserFunction"
            },
```

```
                                {
                                    "name": "CreateCommentFunction"
                                },
                                {
                                    "name":"GenerateChangeHistoryFunction"
                                },
                                {
                                    "name": "IssueReindexFunction"
                                },
                                {
                                    "name": "FireIssueEventFunction",
                                    "parameter": "13"

                                }
                            ],
                            "name": "Close",
                            "from": "Resolved",
                            "to": "Closed",
                            "conditions": []
                        },
                        {
                            "validators": [],
                            "postfunctions": [
                                {
                                    "name": "UpdateIssueStatusFunction"
                                },
                                {
                                    "name": "AssignToCurrentUserFunction"
                                },
                                {
                                    "name": "CreateCommentFunction"
                                },
                                {
                                    "name":"GenerateChangeHistoryFunction"
                                },
                                {
                                    "name": "IssueReindexFunction"
                                },
                                {
                                    "name": "FireIssueEventFunction",
                                    "parameter": "13"

                                }
                            ],
                            "name": "Reopen",
                            "from": "Resolved",
                            "to": "Open",
                            "conditions": []
                        }
                    ]
            }
    ],
    "issueTypeScreenScheme": [
            {
                    "issueType": "DEFAULT",
                    "operations": {
                            "default": [
                                    {
                                            "tab": "Pestaña de campo",
```

```
                                    "fields": [
                                          {
                                                "name": "Summary",
                                                "type": "jiraDefault"
                                          },
                                          {
                                                "name": "Description",
                                                "type": "jiraDefault"
                                          },
                                          {
                                                "name": "Priority",
                                                "type": "jiraDefault"
                                          },
                                          {
                              "name": "Situación correcta de la incidencia",
                              "type": "custom"
                                          },
                                          {
                              "name": "Mensaje de error exacto",
                              "type": "custom"
                                          },
                                          {
                              "name": "Datos para reproducir el caso",
                              "type": "custom"
                                          },
                                          {
                              "name": "Código de transacción / Ruta de menú",
                              "type": "custom"
                                          }
                                    ]
                              }
                        ],
                        "view": [],
                        "edit": [],
                        "create": []
                  }
            }
      ],
      "notificationScheme": {
            "issue_deleted": [],
            "issue_comment_edited": [],
            "issue_worklog_deleted": [],
            "issue_reopened": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
                  }
            ],
            "issue_closed": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
                  }
            ],
            "issue_worklog_updated": [],
            "issue_assigned": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
```

```
                }
        ],
        "issue_comment_deleted": [],
        "work_logged_on_issue": [],
        "issue_resolved": [
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                }
        ],
        "work_started_on_issue": [],
        "work_stopped_on_issue": [],
        "issue_commented": [],
        "issue_moved": [],
        "issue_created": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "issue_updated": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "generic_event": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ]
},
"issueSecurityScheme": "",
"permissionScheme": {
```

```
"add_comments": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        },
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"manage_sprints_permission": [],
"delete_all_comments": [],
"edit_all_comments": [],
"create_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        }
],
"schedule_issues": [],
"set_issue_security": [],
"administer_projects": [],
"delete_all_attachments": [],
"transition_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        },
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"edit_all_worklogs": [],
"view_dev_tools": [],
"link_issues": [],
"delete_all_worklogs": [],
"assignable_user": [
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
```

```
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"resolve_issues": [
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"work_on_issues": [],
"edit_own_comments": [],
"view_readonly_workflow": [],
"delete_own_attachments": [],
"delete_own_comments": [],
"manage_watchers": [],
"edit_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "approver",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        },
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"close_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        }
],
"move_issues": [],
"edit_own_worklogs": [],
"delete_own_worklogs": [],
"modify_reporter": [],
"view_voters_and_watchers": [],
"delete_issues": [],
"browse_projects": [
```

```
                        {
                                "parameter": "requester",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "approver",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level1Support",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level2Support",
                                "type": " projectrole"
                        }
                ],
                "assign_issues": [
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        }
                ],
                "servicedesk_agent": [],
                "create_attachments": [
                        {
                                "parameter": "requester",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "approver",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level1Support",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level2Support",
                                "type": " projectrole"
                        }
                ]
        }
}
```

8.3. Anexo III: JSON Caso 3 (Gestión de peticiones)


```
{
	"key": "CASO3",
	"name": "CASO3: Gestion de peticiones con aprobacion",
	"description": "CASO3: Gestion de peticiones con aprobacion",
	"projectTypeKey": "business",
	"lead": "admin",
	"issueTypes": [
		{
		  "name": "Peticion",
		  "subtask": false
		}
	],
	"assigneeType": "unassigned",
	"roles": [
		{
			"name": "requester",
			"actors": [
				{
				  "name": "requesters",
				  "type": "Group"
				}
			]
		},
		{
			"name": "approver",
			"actors": [
				{
				  "name": "approvers",
				  "type": "Group"
				}
			]
		},
		{
			"name": "dispatcher",
			"actors": [
				{
				  "name": "dispatchers",
				  "type": "Group"
				}
			]
		},
		{
			"name": "level1Support",
			"actors": [
				{
				  "name": "level1Supports",
				  "type": "Group"
				}
			]
		},
		{
			"name": "level2Support",
			"actors": [
				{
```

```
                        "name": "level2Supports",
                        "type": "Group"
                    }
                ]
        }
],
"workflowScheme": [
        {
                "issueType": "DEFAULT",
                "initialAction": {
                        "validators": [
                                {
                                        "parameter": "Create Issue",
                                        "name": "PermissionValidator"
                                }
                        ],
                        "postfunctions": [
                                {
                                        "name": "IssueCreateFunction"
                                },
                                {
                                        "name": "IssueReindexFunction"
                                },
                                {
                                        "parameter": "1",
                                        "name": "FireIssueEventFunction"
                                }
                        ],
                        "name": "Create",
                        "to": "Open",
                        "conditions": []
                },
                "transitions": [
                        {
                                "validators": [],
                                "postfunctions": [
                                  {
                                        "name": "UpdateIssueStatusFunction"
                                  },
                                  {
                                        "name": "AssignToCurrentUserFunction"
                                  },
                                  {
                                        "name": "CreateCommentFunction"
                                  },
                                  {
                                        "name":"GenerateChangeHistoryFunction"
                                  },
                                  {
                                        "name": "IssueReindexFunction"
                                  },
                                  {
                                        "name": "FireIssueEventFunction",
                                        "parameter": "13"

                                  }
                                ],
                                "name": "In Approval",
                                "from": "Open",
```

```
                "to": "In Approval",
                "conditions": []
            },
            {
                "validators": [],
                "postfunctions": [
                  {
                        "name": "UpdateIssueStatusFunction"
                  },
                  {
                        "name": "AssignToCurrentUserFunction"
                  },
                  {
                        "name": "CreateCommentFunction"
                  },
                  {
                        "name":"GenerateChangeHistoryFunction"
                  },
                  {
                        "name": "IssueReindexFunction"
                  },
                  {
                        "name": "FireIssueEventFunction",
                        "parameter": "13"

                  }
                ],
                "name": "L1",
                "from": "In Approval",
                "to": "L1",
                "conditions": [
                  {
                        "name": "InProjectRoleCondition",
                        "parameter": "approver"

                  }
                ]
            },
            {
                "validators": [],
                "postfunctions": [
                  {
                        "name": "UpdateIssueStatusFunction"
                  },
                  {
                        "name": "AssignToCurrentUserFunction"
                  },
                  {
                        "name": "CreateCommentFunction"
                  },
                  {
                        "name":"GenerateChangeHistoryFunction"
                  },
                  {
                        "name": "IssueReindexFunction"
                  },
                  {
                        "name": "FireIssueEventFunction",
```

```
                    "parameter": "13"

                }
            ],
            "name": "L2",
            "from": "In Approval",
            "to": "L2",
            "conditions": [
                {
                    "name": "InProjectRoleCondition",
                    "parameter": "approver"

                }
            ]
        },
        {
            "validators": [],
            "postfunctions": [
                {
                    "name": "UpdateIssueStatusFunction"
                },
                {
                    "name": "AssignToCurrentUserFunction"
                },
                {
                    "name": "CreateCommentFunction"
                },
                {
                    "name":"GenerateChangeHistoryFunction"
                },
                {
                    "name": "IssueReindexFunction"
                },
                {
                    "name": "FireIssueEventFunction",
                    "parameter": "13"

                }
            ],
            "name": "L2",
            "from": "L1",
            "to": "L2",
            "conditions": []
        },
        {
            "validators": [],
            "postfunctions": [
                {
                    "name": "UpdateIssueStatusFunction"
                },
                {
                    "name": "AssignToCurrentUserFunction"
                },
                {
                    "name": "CreateCommentFunction"
                },
                {
                    "name":"GenerateChangeHistoryFunction"
                },
```

```
                    {
                        "name": "IssueReindexFunction"
                    },
                    {
                        "name": "FireIssueEventFunction",
                        "parameter": "13"

                    }
                ],
                "name": "Resolve",
                "from": "L1",
                "to": "Resolved",
                "conditions": []
            },
            {
                "validators": [],
                "postfunctions": [
                    {
                        "name": "UpdateIssueStatusFunction"
                    },
                    {
                        "name": "AssignToCurrentUserFunction"
                    },
                    {
                        "name": "CreateCommentFunction"
                    },
                    {
                        "name":"GenerateChangeHistoryFunction"
                    },
                    {
                        "name": "IssueReindexFunction"
                    },
                    {
                        "name": "FireIssueEventFunction",
                        "parameter": "13"

                    }
                ],
                "name": "Resolve",
                "from": "L2",
                "to": "Resolved",
                "conditions": []
            },
            {
                "validators": [],
                "postfunctions": [
                    {
                        "name": "UpdateIssueStatusFunction"
                    },
                    {
                        "name": "AssignToCurrentUserFunction"
                    },
                    {
                        "name": "CreateCommentFunction"
                    },
                    {
                        "name":"GenerateChangeHistoryFunction"
                    },
                    {
```

```
                                    "name": "IssueReindexFunction"
                                },
                                {
                                    "name": "FireIssueEventFunction",
                                    "parameter": "13"

                                }
                            ],
                            "name": "Close",
                            "from": "Resolved",
                            "to": "Closed",
                            "conditions": []
                        },
                        {
                            "validators": [],
                            "postfunctions": [
                                {
                                    "name": "UpdateIssueStatusFunction"
                                },
                                {
                                    "name": "AssignToCurrentUserFunction"
                                },
                                {
                                    "name": "CreateCommentFunction"
                                },
                                {
                                    "name":"GenerateChangeHistoryFunction"
                                },
                                {
                                    "name": "IssueReindexFunction"
                                },
                                {
                                    "name": "FireIssueEventFunction",
                                    "parameter": "13"

                                }
                            ],
                            "name": "Reopen",
                            "from": "Resolved",
                            "to": "Open",
                            "conditions": []
                        }
                    ]
            }
    ],
    "issueTypeScreenScheme": [
            {
                    "issueType": "DEFAULT",
                    "operations": {
                            "default": [
                                    {
                                            "tab": "Pestaña de campo",
                                            "fields": [
                                                    {
                                                            "name": "Summary",
                                                            "type": "jiraDefault"
                                                    },
                                                    {
                                                            "name": "Description",
```

```
                                                "type": "jiraDefault"
                                          },
                                          {
                                                "name": "Priority",
                                                "type": "jiraDefault"
                                          },
                                          {
                                                "name": "Area",
                                                "type": "custom"
                                          },
                                          {
                                                "name": "Categoria",
                                                "type": "custom"
                                          }
                                    ]
                              }
                        ],
                        "view": [],
                        "edit": [],
                        "create": []
                  }
            }
      ],
      "notificationScheme": {
            "issue_deleted": [],
            "issue_comment_edited": [],
            "issue_worklog_deleted": [],
            "issue_reopened": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
                  }
            ],
            "issue_closed": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
                  }
            ],
            "issue_worklog_updated": [],
            "issue_assigned": [
                  {
                        "parameter": "",
                        "type": "Current_Assignee"
                  }
            ],
            "issue_comment_deleted": [],
            "work_logged_on_issue": [],
            "issue_resolved": [
                  {
                        "parameter": "",
                        "type": "Current_Reporter"
                  }
            ],
            "work_started_on_issue": [],
            "work_stopped_on_issue": [],
            "issue_commented": [],
            "issue_moved": [],
            "issue_created": [
```

```
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "issue_updated": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ],
        "generic_event": [
                {
                        "parameter": "",
                        "type": "Current_Assignee"
                },
                {
                        "parameter": "",
                        "type": "Current_Reporter"
                },
                {
                        "parameter": "",
                        "type": "All_Watchers"
                }
        ]
},
"issueSecurityScheme": "",
"permissionScheme": {
        "add_comments": [
                {
                        "parameter": "requester",
                        "type": " projectrole"
                },
                {
                        "parameter": "approver",
                        "type": " projectrole"
                },
                {
                        "parameter": "dispatcher",
                        "type": " projectrole"
                },
                {
                        "parameter": "level1Support",
```

```
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"manage_sprints_permission": [],
"delete_all_comments": [],
"edit_all_comments": [],
"create_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        }
],
"schedule_issues": [],
"set_issue_security": [],
"administer_projects": [],
"delete_all_attachments": [],
"transition_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "approver",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        },
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"edit_all_worklogs": [],
"view_dev_tools": [],
"link_issues": [],
"delete_all_worklogs": [],
"assignable_user": [
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"resolve_issues": [
        {
                "parameter": "level1Support",
```

```
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"work_on_issues": [],
"edit_own_comments": [],
"view_readonly_workflow": [],
"delete_own_attachments": [],
"delete_own_comments": [],
"manage_watchers": [],
"edit_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "approver",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        },
        {
                "parameter": "level1Support",
                "type": " projectrole"
        },
        {
                "parameter": "level2Support",
                "type": " projectrole"
        }
],
"close_issues": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "dispatcher",
                "type": " projectrole"
        }
],
"move_issues": [],
"edit_own_worklogs": [],
"delete_own_worklogs": [],
"modify_reporter": [],
"view_voters_and_watchers": [],
"delete_issues": [],
"browse_projects": [
        {
                "parameter": "requester",
                "type": " projectrole"
        },
        {
                "parameter": "approver",
                "type": " projectrole"
```

```
                        },
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level1Support",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level2Support",
                                "type": " projectrole"
                        }
                ],
                "assign_issues": [
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        }
                ],
                "servicedesk_agent": [],
                "create_attachments": [
                        {
                                "parameter": "requester",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "approver",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "dispatcher",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level1Support",
                                "type": " projectrole"
                        },
                        {
                                "parameter": "level2Support",
                                "type": " projectrole"
                        }
                ]
        }
}
```

8.4. Anexo IV: Desarrollo de complementos Gadget JIRA

El presente apartado tiene como objetivo explicar cómo se desarrolla un Gadget JIRA mediante un ejemplo práctico que muestre los tickets de un filtro predefinido. Con este ejemplo tendremos las bases para realizar este tipo de complementos y a partir de él realizar Gadgets más complejos añadiéndole nuevas funcionalidades.

8.4.1. Funcionalidades del Gadget a Implementar

El Gadget que se implementará en este desarrollo corresponde a un panel que muestre todas las peticiones abiertas ordenadas por menor tiempo restante. La información que se debe mostrar son los siguientes campos de las peticiones:

- Portal
- Pkey
- Descripción
- Responsable
- Prioridad
- Estado
- Fecha de creación
- Fecha de entrega
- SLA
- Tempo restante
- Porcentaje de evolución(% de tiempo que se gastó de SLA)

Además si hacemos clic en la pkey de un ticket seremos redirigidos al ticket en JIRA.
Las peticiones estarán ordenadas por menor tempo restante.
Las peticiones en el panel se mostrarán de diferentes colores según el porcentaje de evolución que tengan:

- <= 50% → verde
- 50% < x <= 75% → amarilla
- 75% < x <= 99% → roja
- = 100% → negra

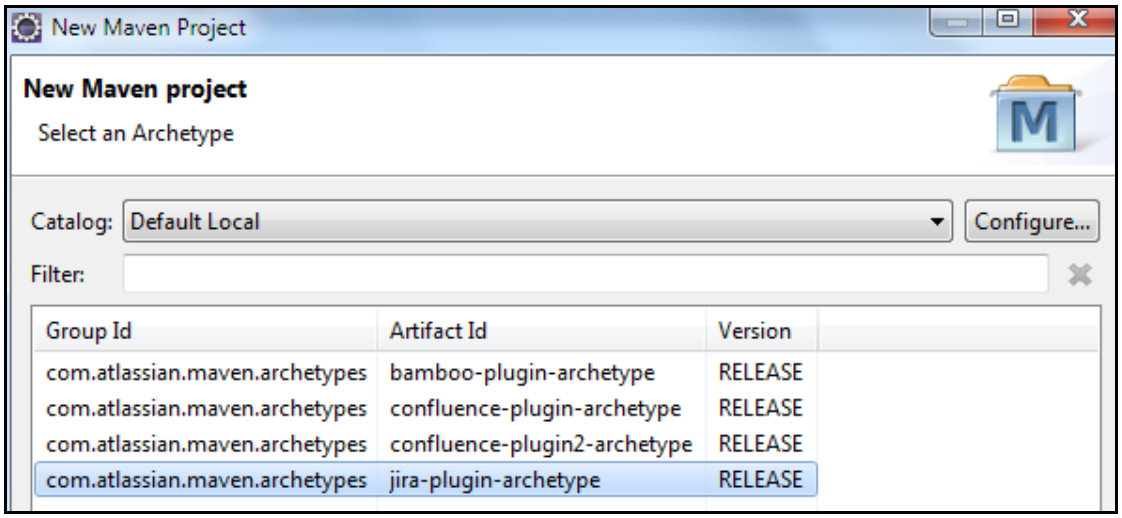
En la siguiente imagen se muestra como deberá quedar el resultado final del Gadget.

Portal	Jira	Descripción	Responsable	Prioridad	Status	Fecha de apertura	Fecha de entrega	SLA	Tempo restante	Porcentaje de evolución
Portonet	PRTSEG - 1312	MGMT - Erro criar destaque	Francisco Matos	Média	Em análise	28/2/2013 16:46:00	5/3/2013 09:46:00	20:00	00:00	100%
Portal de Cliente	PRTSEG-1317	Base de Conhecimento: Top 20 e Últimas	Jhonatas	Média	Em resolução	28/2/2013 16:46:00	5/3/2013 09:46:00	20:00	04:00	76%
Portal Institucional	PRTSEG-1326	Base de Conhecimento: Duplicidade na pergunta ao editar	William Ferreira	Alta	Em resolução	28/2/2013 16:46:00	5/3/2013 09:46:00	20:00	09:00	51%
Portal de Cliente	PRTSEG-1022	Alterações no Jira	Francisco Matos	Baixa	Em análise	28/2/2013 16:46:00	5/3/2013 09:46:00	20:00	14:00	30%
Portonet	PRTSEG-1303	Erro ao efetuar o login e acesso ao Portal do Cliente	Francisco Matos	Baixa	Em análise	28/2/2013 16:46:00	5/3/2013 09:46:00	20:00	18:00	10%

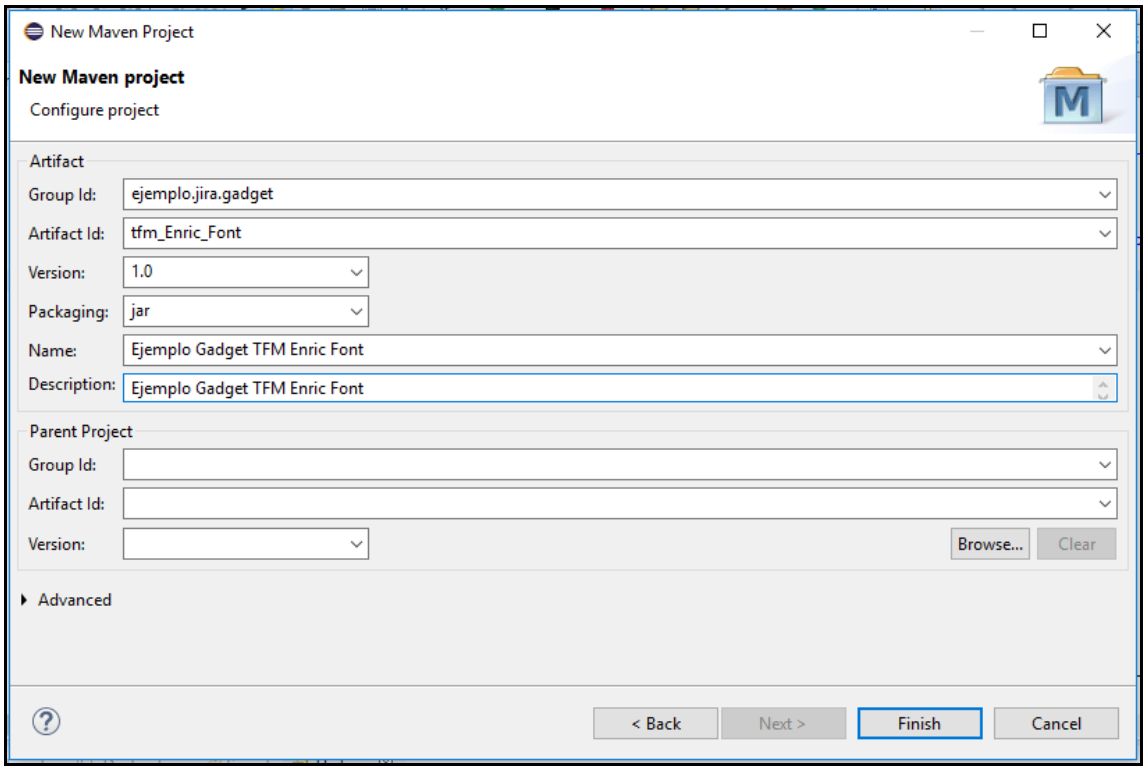
8.4.2. Creación estructura Gadget

8.4.2.1. Creación Proyecto Eclipse

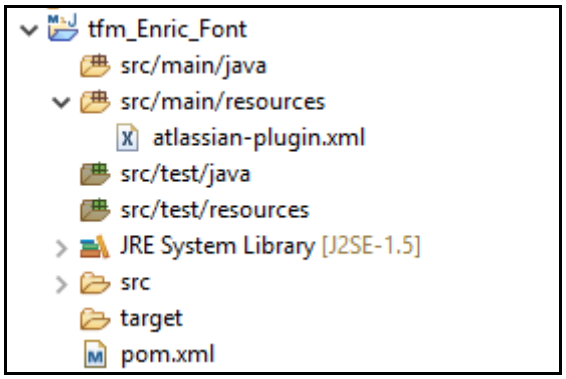
En el Eclipse hacemos File > New > Other > Maven Project.
Seleccionamos el workspace que deseemos y como Archetype escogemos:
jira-plugin-archetype



Posteriormente rellenamos la información para el pom.xml y hacemos clic en “Finish” para crearlo:



Una vez hecho esto Eclipse nos ha creado el siguiente proyecto:



8.4.2.2. Modificación pom.xml

Abrimos el fichero pom.xml y realizamos los cambios dejándolo como está en la siguiente imagen:

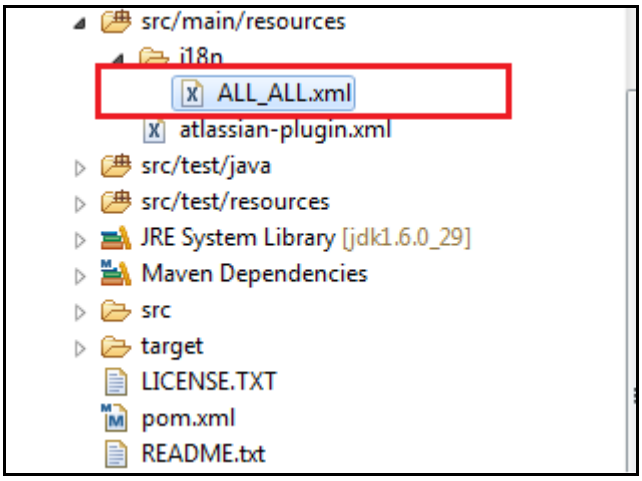
```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>
  <groupId>ejemplo.jira.gadget</groupId>
  <artifactId>tfm_Enric_Font</artifactId>
  <version>1.0</version>
  <name>Ejemplo Gadget TFM Enric Font</name>
  <description>Ejemplo Gadget TFM Enric Font</description>
  <packaging>atlassian-plugin</packaging>

    <properties>
        <jira.version>7.2.2</jira.version>
        <amps.version>6.3.15</amps.version>
        <plugin.testrunner.version>1.2.3</plugin.testrunner.version>
        <atlassian.spring.scanner.version>1.2.13</atlassian.spring.scanner.version>
        <!-- This key is used to keep the consistency between the key in atlassian-plugin.xml and the key to generate bundle. -->
        <atlassian.plugin.key>${project.groupId}.${project.artifactId}</atlassian.plugin.key>
        <!-- TestKit version 6.x for JIRA 6.x -->
        <testkit.version>6.3.11</testkit.version>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
    </properties>
</project>
```

8.4.2.3. Modificación atlassian-plugin.xml

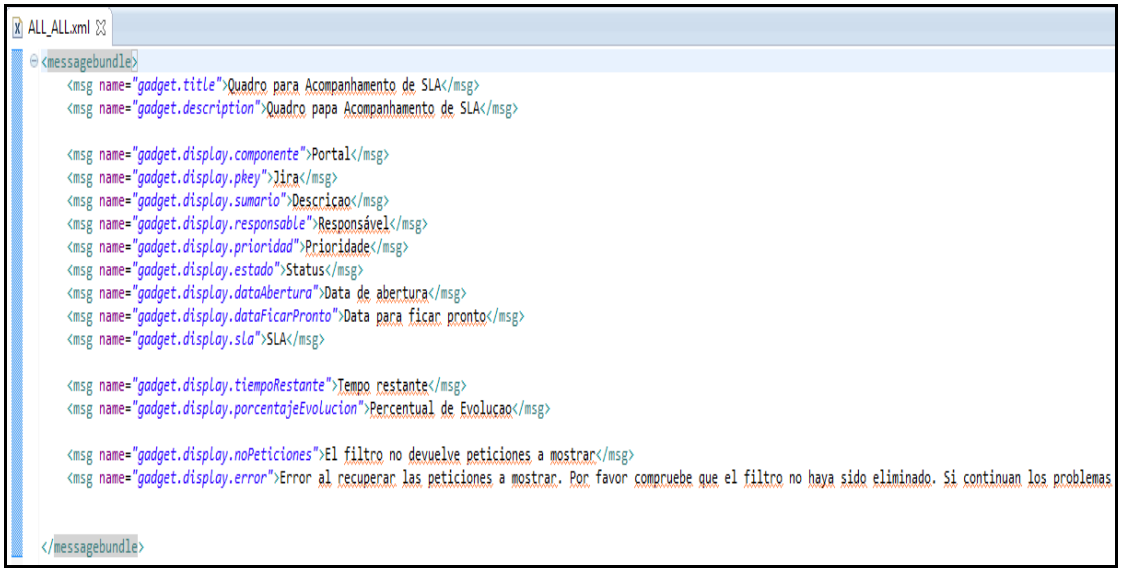
Abrimos el fichero atlassian-plugin.xml y realizamos los cambios dejándolo como está en la siguiente imagen:

8.4.2.4. Crear fichero ALL_ALL.xml

En src/main/resources creamos la carpeta i18n y dentro de esta carpeta creamos el XML ALL_ALL.xml:

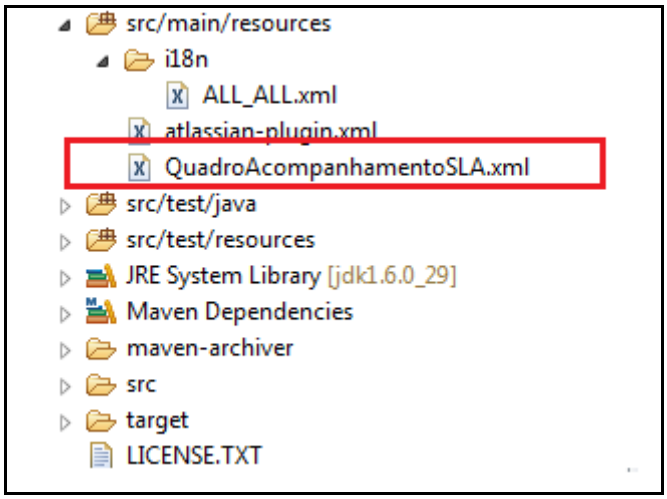


Dentro de este fichero XML se pondrán los literales de la cabecera a mostrar en el panel. El contenido será el siguiente:

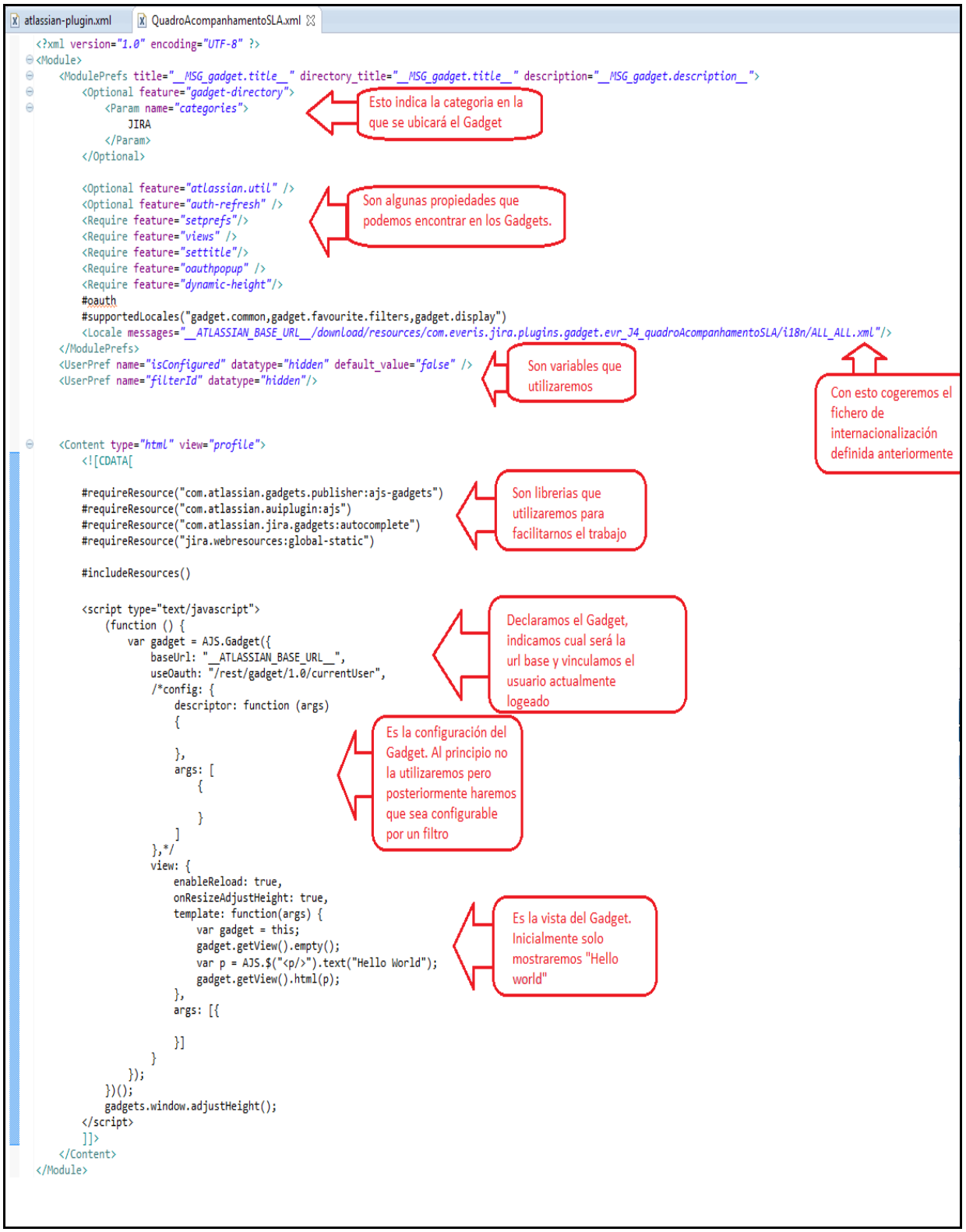


8.4.2.5. Crear fichero QuadroAcompanhamentoSLA.xml

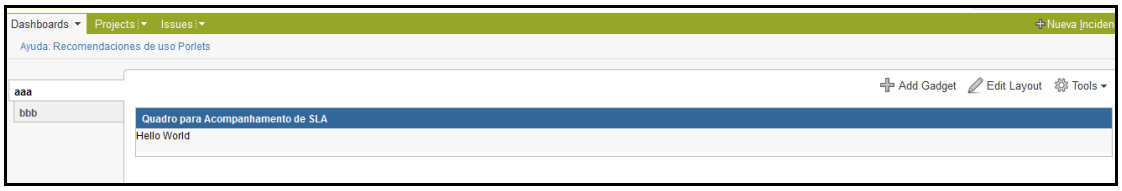
En src/main/resources creamos el XML del Gadget QuadroAcompanhamentoSLA.xml:



Este es el fichero que contiene la vista del Gadget. Primero crearemos el “Hello World” y en posteriores apartados lo modificaremos. El contenido por ahora será:



Llegados a este punto podemos probar el plugin que hemos desarrollado hasta ahora. Compilamos el Gadget haciendo “Run as” > “Maven package” y el jar generado lo copiamos en la carpeta jira-home/plugins/installed-plugins de nuestra instalación local de JIRA. Iniciamos JIRA. En el Dashboard añadimos nuestro Gadget y vemos que se muestra así:



8.4.3. Creación Modelo de Datos

Como modelo de datos necesitaremos la clase JAVA que utilizaremos para representar un ticket JIRA con la información que nos interesa. También necesitaremos la clase que represente una lista de peticiones.

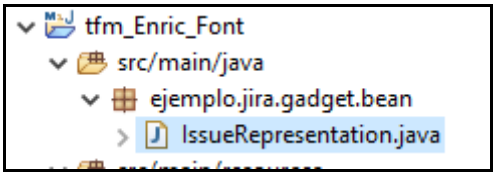
8.4.3.1. Crear fichero QuadroAcompanhamentoSLA.xml

Primero de todo eliminamos la clase MyPlugin.java ya que no es necesaria para este plugin.

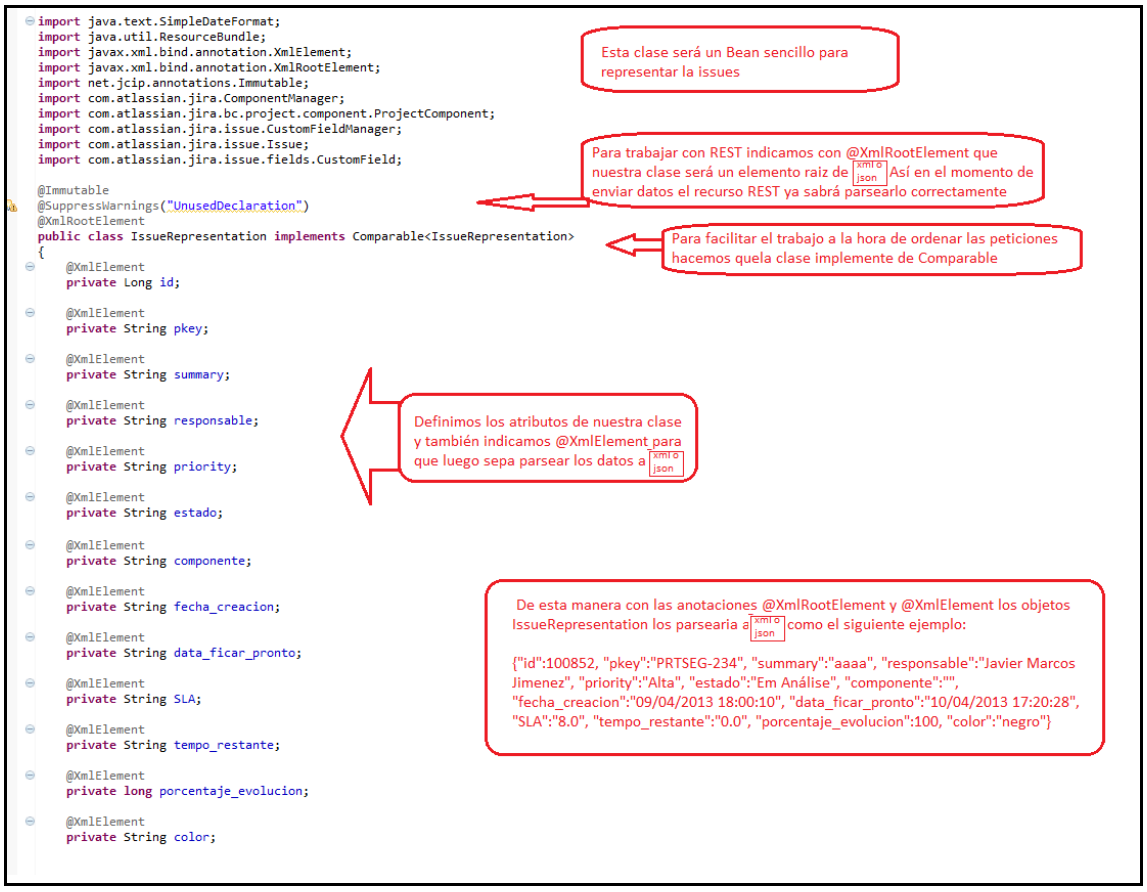
Para tener el código ordenado crearemos un nuevo package:

ejemplo.jira.gadget.bean

Dentro del package creamos la clase IssueRepresentation.java y quedará de la siguiente manera:



El contenido de IssueRepresentation.java es el siguiente:




```java
import java.text.SimpleDateFormat;
import java.util.ResourceBundle;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import net.jcip.annotations.Immutable;
import com.atlassian.jira.ComponentManager;
import com.atlassian.jira.bc.project.component.ProjectComponent;
import com.atlassian.jira.issue.CustomFieldManager;
import com.atlassian.jira.issue.Issue;
import com.atlassian.jira.issue.fields.CustomField;

@Immutable
@SuppressWarnings("UnusedDeclaration")
@XmlRootElement
public class IssueRepresentation implements Comparable<IssueRepresentation>
{
    @XmlElement
    private Long id;

    @XmlElement
    private String pkey;

    @XmlElement
    private String summary;

    @XmlElement
    private String responsable;

    @XmlElement
    private String priority;

    @XmlElement
    private String estado;

    @XmlElement
    private String componente;

    @XmlElement
    private String fecha_creacion;

    @XmlElement
    private String data_ficar_pronto;

    @XmlElement
    private String SLA;

    @XmlElement
    private String tempo_restante;

    @XmlElement
    private long porcentaje_evolucion;

    @XmlElement
    private String color;
```

El código sigue en la siguiente página:

```
public IssueRepresentation(Issue issue)
{
    SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

    ResourceBundle p = ResourceBundle.getBundle("com.everis.jira.plugins.gadget.quadroAcompanhamentoSLA.bean.parametros");
    String s_data_ficar_pronto = p.getString("cf_data_ficar_pronto").toString().trim();
    String s_SLA = p.getString("cf_SLA").toString().trim();
    String s_tempo_restante = p.getString("cf_tempo_restante").toString().trim();

    CustomFieldManager cfManager = ComponentManager.getInstance().getCustomFieldManager();

    this.id = issue.getId();
    this.pkey = issue.getKey();
    this.summary = issue.getSummary();
    this.responsable = (issue.getAssigneeUser()!=null)? issue.getAssigneeUser().getDisplayName() : "";
    this.priority = (issue.getPriorityObject()!=null)? issue.getPriorityObject().getNameTranslation() : "";
    this.estado = issue.getStatusObject().getNameTranslation();

    if(issue.getComponentObjects()!=null){
        Object[] componentes = issue.getComponentObjects().toArray();
        if(componentes!=null && componentes.length>0){
            this.componente = ((ProjectComponent)componentes[0]).getName();
        }else{
            this.componente = "";
        }

    }else{
        this.componente = "";
    }

    this.fecha_creacion = (issue.getCreated()!=null)? sdf.format(issue.getCreated()) : "";

    CustomField cf_data_ficar_pronto = cfManager.getCustomFieldObject(s_data_ficar_pronto);
    if(cf_data_ficar_pronto!=null){
        this.data_ficar_pronto = (issue.getCustomFieldValue(cf_data_ficar_pronto)!=null)? sdf.format(issue.getCustomFieldValue(cf_data_ficar_pronto)) : "";
    }else{
        this.data_ficar_pronto = "";
    }

    CustomField cf_SLA = cfManager.getCustomFieldObject(s_SLA);
    if(cf_SLA!=null){
        this.SLA = (issue.getCustomFieldValue(cf_SLA)!=null)? ((Double)issue.getCustomFieldValue(cf_SLA)).toString() : "";
    }else{
        this.SLA = "";
    }

    CustomField cf_tempo_restante = cfManager.getCustomFieldObject(s_tempo_restante);
    if(cf_tempo_restante!=null){
        this.tempo_restante = (issue.getCustomFieldValue(cf_tempo_restante)!=null)? ((Double)issue.getCustomFieldValue(cf_tempo_restante)).toString() : "";
    }else{
        this.tempo_restante = "";
    }

    Double porcentaje = 0.0;
    if(!"".equals(this.SLA) && !"".equals(this.tempo_restante)){
        Double num_sla = Double.parseDouble(this.SLA);
        Double num_tempo_restante = Double.parseDouble(this.tempo_restante);

        Double num_tempo_gastado = num_sla - num_tempo_restante;
        if(num_tempo_gastado<0.0){
            num_tempo_gastado = 0.0;
        }


        if(num_tempo_gastado>0.0){

            porcentaje =  (num_tempo_gastado*100)/num_sla;
        }


        this.porcentaje_evolucion = Math.round(porcentaje);

    }


    this.color = "verde";
    if(50.0 < porcentaje && porcentaje <= 75.0){
        this.color = "amarillo";
    }else if(75.0 < porcentaje && porcentaje <= 99.0){
        this.color = "rojo";
    }else if(porcentaje == 100.0){
        this.color = "negro";
    }

}


public int compareTo(IssueRepresentation i) {
    if(this.porcentaje_evolucion<i.porcentaje_evolucion){
        return -1;
    }else if(this.porcentaje_evolucion==i.porcentaje_evolucion){
        return 0;
    }else{
        return 1;
    }
}

}
```

En el constructor primero leemos el fichero de configuración para saber que id tienen lo customfields que utilizaremos

Despues a cada atributo vamos asignando el valor correspondiente

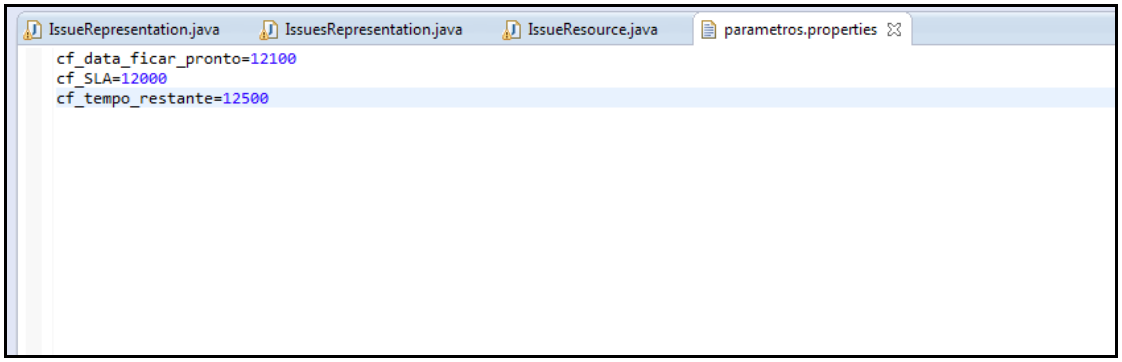
Calculamos elporcentaje de tiempo que hemos gastado de SLA

Segun el porcentaje indicamos el color de la petición que mostraremos para la vista

Este método debe ser implementado al implementar Comparable. En el devolvemos -1 si nuestro objeto es menor, 0 si es igual i 1 si es mayor

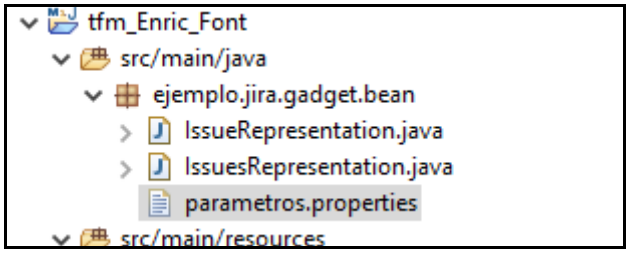
8.4.3.2. Crear fichero parametros.properties

Dentro del package ejemplo.jira.gadget.bean creamos el fichero parametros.properties que contendrá los identificadores de los customfields que hemos utilizado en IssueRepresentation.java:



8.4.3.3. Crear fichero IssuesRepresentation.java

Creamos la clase IssuesRepresentation.java también dentro del package ejemplo.jira.gadget.bean



El contenido de IssuesRepresentation.java contiene lo siguiente:

```java
import java.util.ArrayList;

@Immutable
@SuppressWarnings("UnusedDeclaration")
@XmlRootElement
public class IssuesRepresentation
{
    @XmlElement
    private List<IssueRepresentation> issues;

    public IssuesRepresentation(Iterable<IssueRepresentation> issues)
    {
        this.issues = new ArrayList<IssueRepresentation>();

        for (IssueRepresentation representation : issues)
        {
            this.issues.add(representation);
        }
    }

}
```

Esta clase será una lista de IssueRepresentation. También añadimos las anotaciones @XmlRootElement

Solo contendrá un atributo que será la lista. Anotamos con @XmlElement

En el constructor vamos añadiendo peticiones a la lista

De esta manera con las anotaciones @XmlRootElement y @XmlElement junto con las anotaciones que también se pusieron en IssueRepresentation la aplicación mandará a la vista los datos de las peticiones parseados de la siguiente manera:


```
{"issues":[{"id":100852,"pkey":"PRTSEG-234","summary":"aaaa","responsable":"Javier Marcos
Jimenez","priority":"Alta","estado":"Em Análise","componente":"","fecha_creacion":"09/04/2013
18:00:10","data_ficar_pronto":"10/04/2013
17:20:28","SLA":"8.0","tempo_restante":"0.0","porcentaje_evolucion":100,"color":"negro"},
{"id":100950,"pkey":"PRTSEG-235","summary":"dsffghdrhb rthrvehrv hteyhevhtyhb etyh ehey
53ey","responsable":"Javier Marcos Jimenez","priority":"Bloqueado","estado":"Aguardando
informação","componente":"SIACOM","fecha_creacion":"09/04/2013 18:06:43","data_ficar_pronto":"09/04/2013
19:22:02","SLA":"1.0","tempo_restante":"0.19","porcentaje_evolucion":81,"color":"rojo"}]}
```


8.4.4. Creación Modelo de Datos

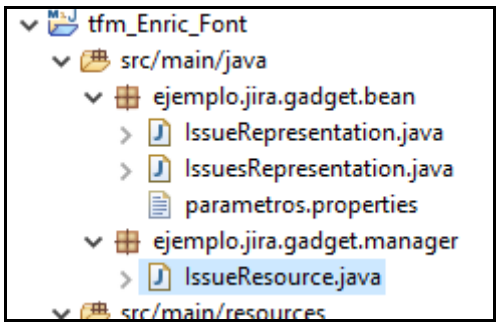
El manager será la clase JAVA encargada de recibir las solicitudes web haciéndose servir de un recurso REST que definiremos en apartado posterior.

8.4.4.1. Crear fichero IssueResource.java

Para tener el código ordenado crearemos un nuevo package:

ejemplo.jira.gadget.manager

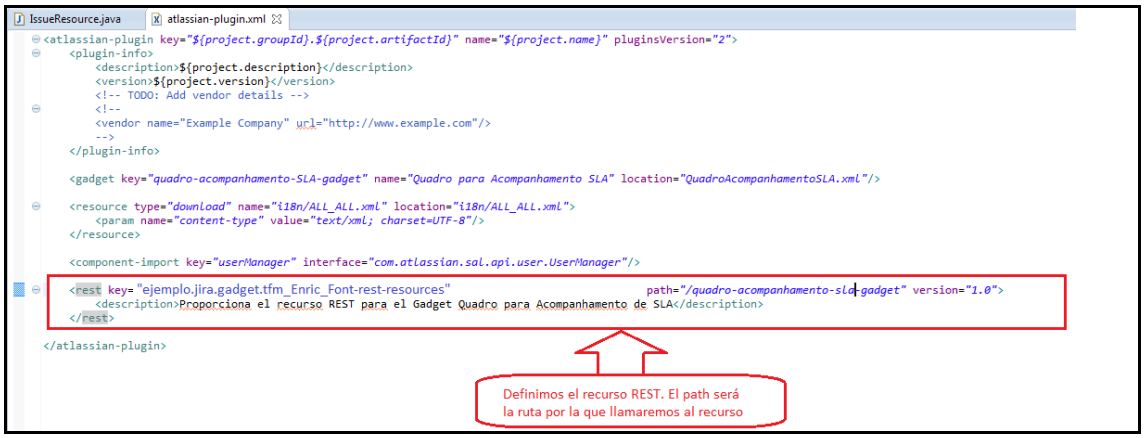
Dentro del package creamos la clase IssueResource.java que será nuestro manager y quedará de la siguiente manera:



El contenido de IssueResource.java:

8.4.5. Creación Recurso REST

En las clases java ya hemos ido preparando el desarrollo para que nuestro recurso REST funcione correctamente pero falta definirlo en nuestro plugin y así lo reconozca JIRA. Simplemente se debe de añadir lo siguiente en el fichero atlassian-plugin.xml dejándolo como se muestra a continuación:

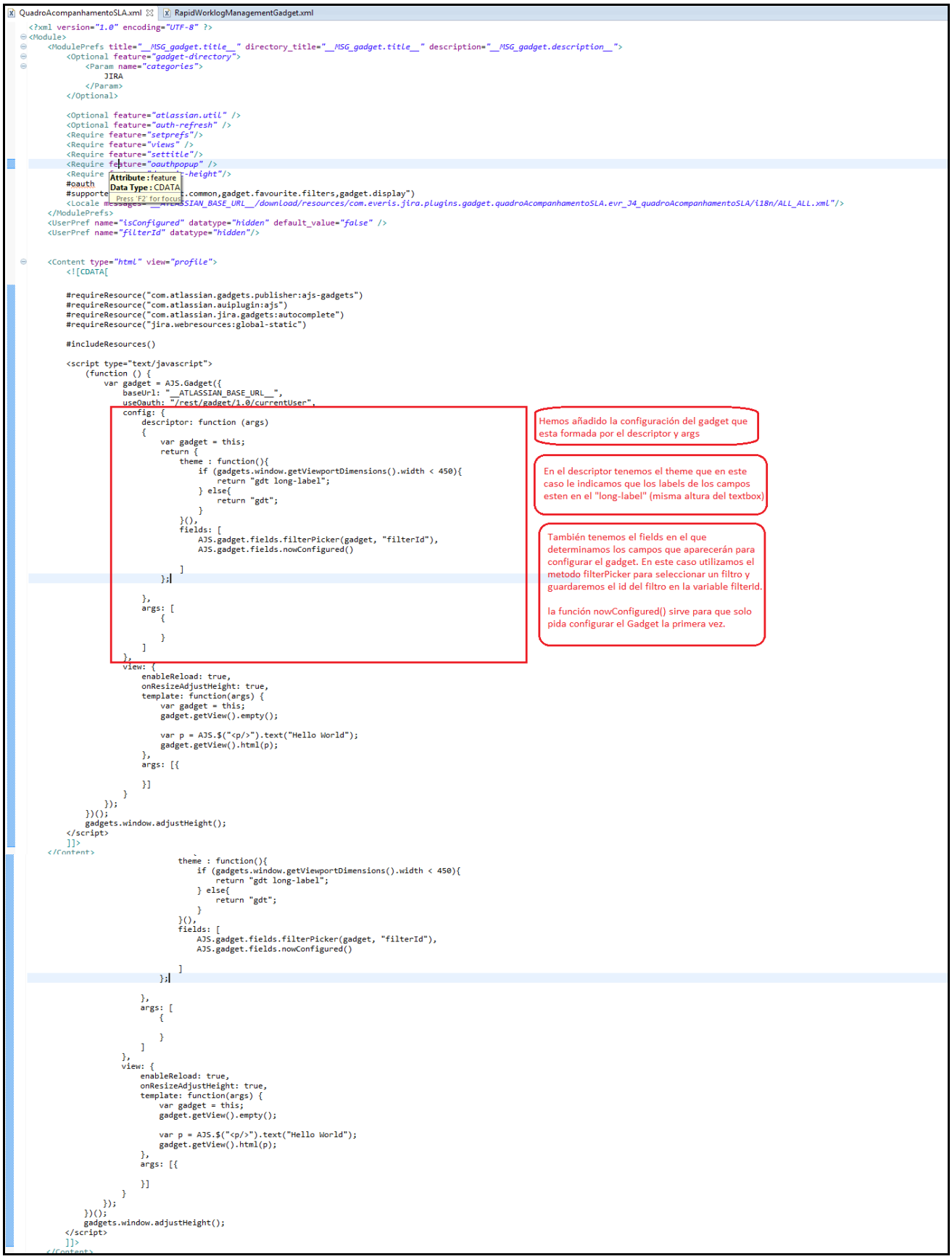


8.4.6. Creación Vista del Gadget

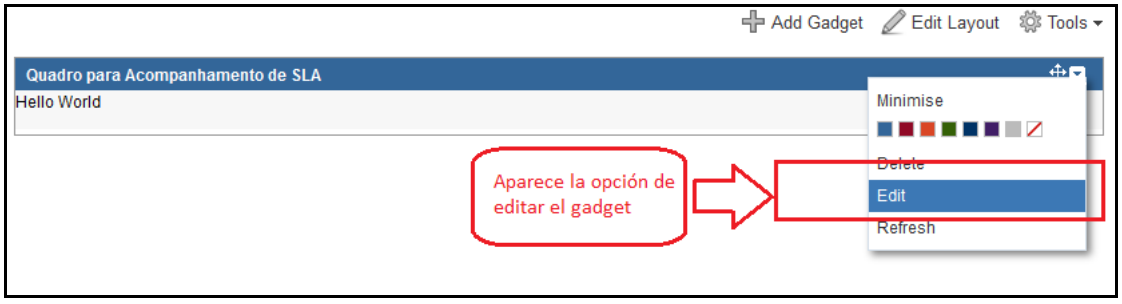
En este punto solo nos queda que desde la vista del gadget se haga una llamada rest para obtener las peticiones de un filtro y pintar una tabla HTML con el resultado.

8.4.6.1. Habilitar Gadget para configurarlo por filtro

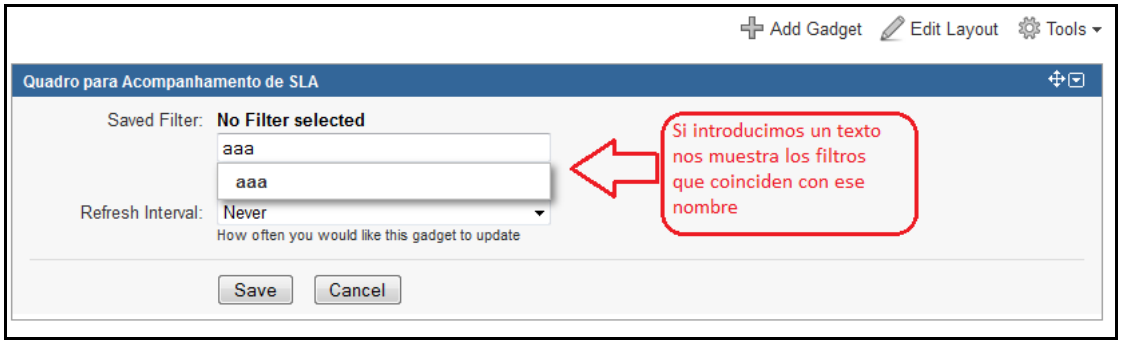
En el fichero QuadroAcompanhamentoSLA.xml añadimos lo siguiente:



Si volvemos a generar el JAR y arrancamos la aplicación vemos lo siguiente:



Si hacemos click en edit nos aparece el menú de configuración y podemos seleccionar un filtro:



8.4.6.2. Obtener Peticiones en vista

Para obtener las peticiones en la vista hacemos una llamada a nuestro recurso REST en el momento de mostrar la vista del Gadget. Para ello añadimos lo siguiente en QuadroAcompanhamentoSLA.xml:

QuadroAcompanhamentoSLA.xml | RapidWorklogManagementGadget.xml

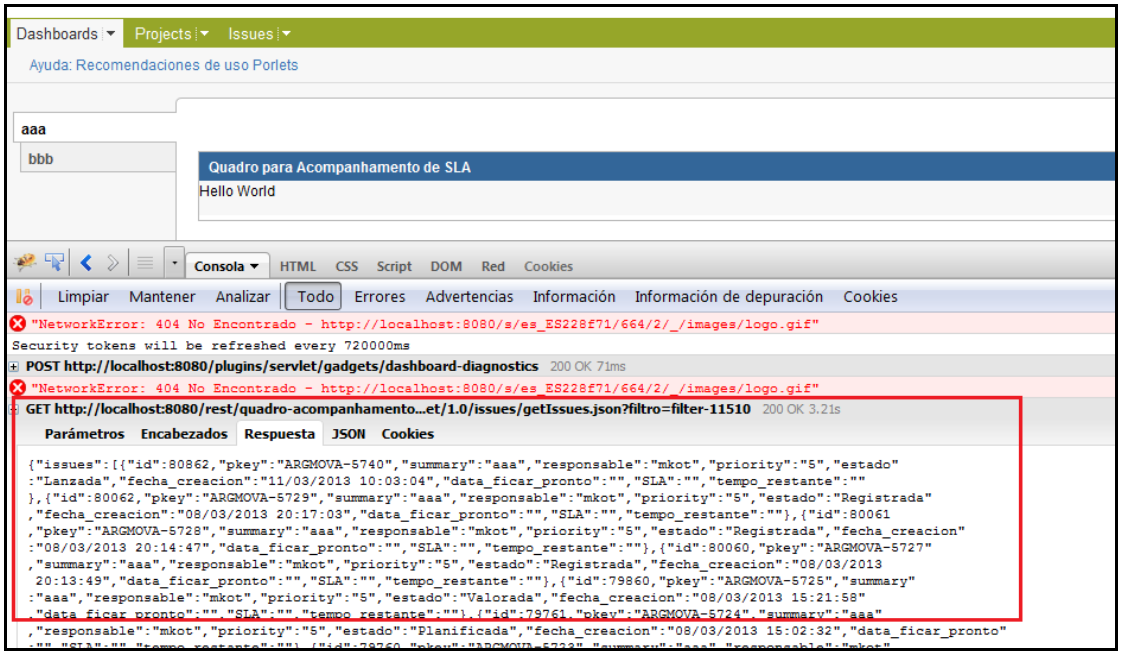
```
<?xml version="1.0" encoding="UTF-8" ?>
<Module>
    <ModulePrefs title="__MSG_gadget.title__" directory_title="__MSG_gadget.title__" description="__MSG_gadget.description__">
        <Optional feature="gadget-directory">
            <Param name="categories">
                JIRA
            </Param>
        </Optional>

        <Optional feature="atlassian.util" />
        <Optional feature="auth-refresh" />
        <Require feature="setprefs"/>
        <Require feature="views" />
        <Require feature="settitle"/>
        <Require feature="oauthpopup" />
        <Require feature="dynamic-height"/>
        #oauth
        #supportedLocales("gadget.common,gadget.favourite.filters,gadget.display")
        <Locale messages="__ATLASSIAN_BASE_URL__/download/resources/com.everis.jira.plugins.gadget.quadroAcompanhamentoSLA.evr_J4_quadroAcompanha
    </ModulePrefs>
    <UserPref name="isConfigured" datatype="hidden" default_value="false" />
    <UserPref name="filterId" datatype="hidden"/>


    <Content type="html" view="profile">
        <![CDATA[

        #requireResource("com.atlassian.gadgets.publisher:ajs-gadgets")
        #requireResource("com.atlassian.auiplugin:ajs")
        #requireResource("com.atlassian.jira.gadgets:autocomplete")
        #requireResource("jira.webresources:global-static")

        #includeResources()

        <script type="text/javascript">
            (function () {
                var gadget = AJS.Gadget({
                    baseUrl: "__ATLASSIAN_BASE_URL__",
                    useOauth: "/rest/gadget/1.0/currentUser",
                    config: {
                        descriptor: function (args)
                        {
                            var gadget = this;
                            return {
                                theme : function(){
                                    if (gadgets.window.getViewportDimensions().width < 450){
                                        return "gdt long-label";
                                    } else{
                                        return "gdt";
                                    }
                                }(),
                                fields: [
                                    AJS.gadget.fields.filterPicker(gadget, "filterId"),
                                    AJS.gadget.fields.nowConfigured()

                                ]
                            };

                        },
                        args: [
                            {

                            }
                        ]
                    },
                    view: {
                        enableReload: true,
                        onResizeAdjustHeight: true,
                        template: function(args) {
                            var gadget = this;
                            gadget.getView().empty();

                            var p = AJS.$("<p/>").text("Hello World");
                            gadget.getView().html(p);
                        },
                        args: [{
                            key: "issueData",
                            ajaxOptions: function() {
                                return {
                                    url: "/rest/quadro-acompanhamento-sla-gadget/1.0/issues/getIssues.json",
                                    data:
                                    {
                                        filtro : gadgets.util.unescapeString(this.getPref("filterId"))
                                    }
                                };
                            }

                        }]
                    }
                });
            })();
            gadgets.window.adjustHeight();
        </script>
        ]]>
    </Content>
</Module>
```

Si nos fijamos la llamada es /rest/ruta de nuestro recurso rest/version de nuestro recurso res/ruta definida en la clase manager/ruta definida en el metodo.

Al indicarle .json le forzamos a que nos genere el resultado en json

En el args de la view añadimos para que en la variable issueData guarde el resultado que devuelva la llamada Ajax a nuestro recurso Rest pasandole como parametro el filtro

Si volvemos a compilar el JAR y a reiniciar la aplicación, en el momento de mostrar la vista del Gadget automáticamente podemos ver en el Firebug como se hace la llamada al REST y su respuesta:



8.4.6.3. Pintar tabla HTML con las peticiones

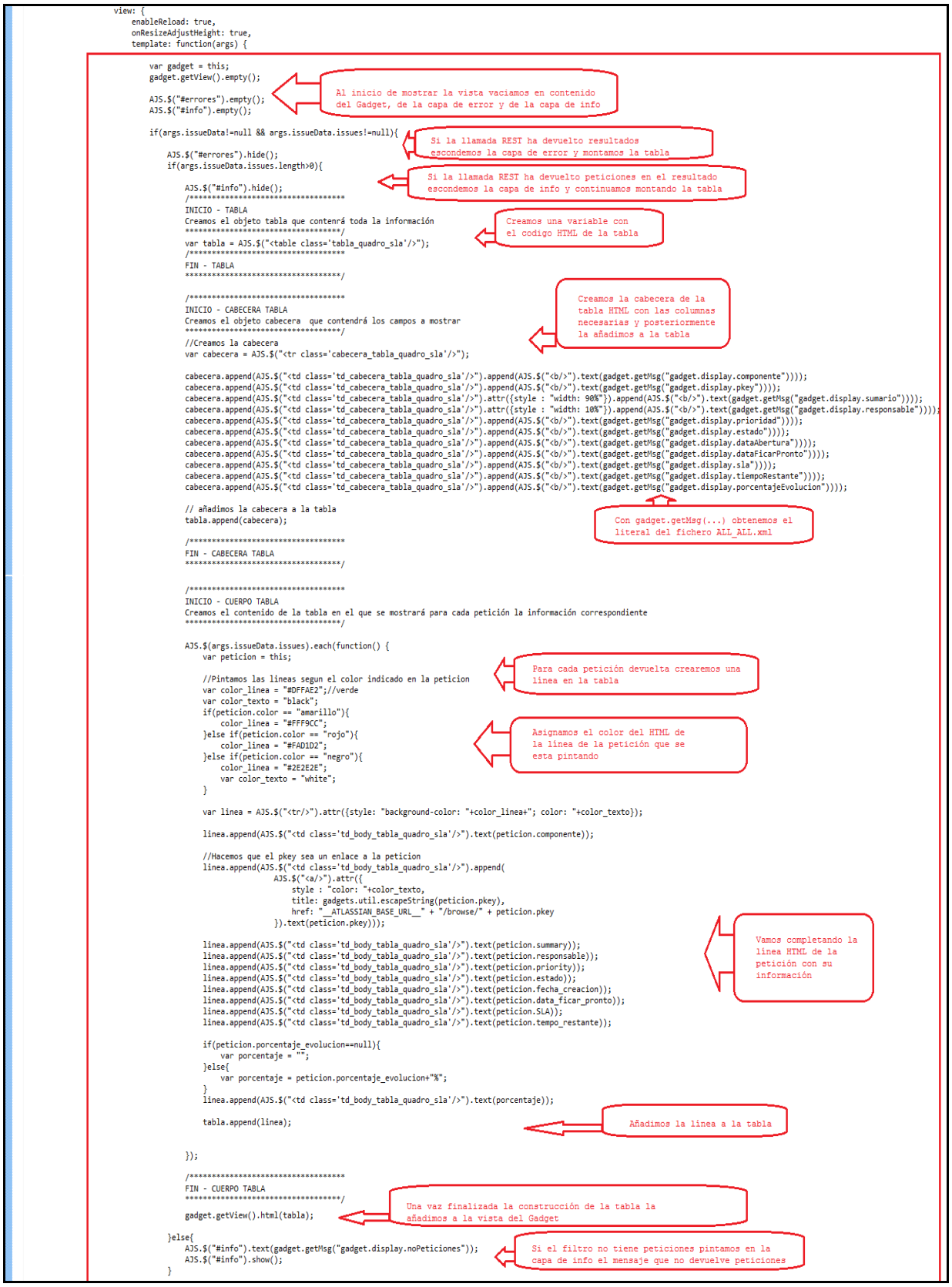
En este momento solo tenemos que coger los datos que devuelve la llamada REST y montar una tabla HTML con esa información. Para modificamos el fichero QuadroAcompanhamentoSLA.xml añadimos lo siguiente:

IssueResource.java | atlassian-plugin.xml | QuadroAcompanhamentoSLA.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<Module>
    <ModulePrefs title="__MSG_gadget.title__" directory_title="__MSG_gadget.title__" description="__MSG_gadget.description__">
        <Optional feature="gadget-directory">
            <Param name="categories">
                JIRA
            </Param>
        </Optional>

        <Optional feature="atlassian.util" />
        <Optional feature="auth-refresh" />
        <Require feature="setprefs"/>
        <Require feature="views" />
        <Require feature="settitle"/>
        <Require feature="oauthpopup" />
        <Require feature="dynamic-height"/>
        #oauth
        #supportedLocales("gadget.common,gadget.favourite.filters,gadget.display")
        <Locale messages="__ATLASSIAN_BASE_URL__/download/resources/com.everis.jira.plugins.gadget.quadroAcompanhamentoSLA.evr_J4_quadroAcompanhamentoSLA/i18n/ALL_ALL.xml"/>
    </ModulePrefs>
    <UserPref name="isConfigured" datatype="hidden" default_value="false" />
    <UserPref name="filterId" datatype="hidden"/>


    <Content type="html" view="profile">
        <![CDATA[

        #requireResource("com.atlassian.gadgets.publisher:ajs-gadgets")
        #requireResource("com.atlassian.auiplugin:ajs")
        #requireResource("com.atlassian.jira.gadgets:autocomplete")
        #requireResource("jira.webresources:global-static")

        #includeResources()

        <div id="errores" class="aui-message error" style="display: none;"></div>
        <div id="info" class="aui-message info" style="display: none;"></div>

        <script type="text/javascript">
            (function () {
                var gadget = AJS.Gadget({
                    baseUrl: "__ATLASSIAN_BASE_URL__",
                    useOauth: "/rest/gadget/1.0/currentUser",
                    config: {
                        descriptor: function (args)
                        {
                            var gadget = this;
                            return {
                                theme : function(){
                                    if (gadgets.window.getViewportDimensions().width < 450){
                                        return "gdt long-label";
                                    } else{
                                        return "gdt";
                                    }
                                }(),
                                fields: [
                                    AJS.gadget.fields.filterPicker(gadget, "filterId"),
                                    AJS.gadget.fields.nowConfigured()

                                ]
                            };

                        },
                        args: [
                            {

                            }
                        ]
                    },
                    view: {
```

Añadimos las capas para mostrar los mensajes de información y error

El código continúa en la siguiente página:



El código continúa en la siguiente página:

```
                    /***********************************
                    FIN - CUERPO TABLA
                    ***********************************/
                    gadget.getView().html(tabla);

                }else{
                    AJS.$("#info").text(gadget.getMsg("gadget.display.noPeticiones"));
                    AJS.$("#info").show();
                }

            }else{
                AJS.$("#errores").text(gadget.getMsg("gadget.display.error"));
                AJS.$("#errores").show();
            }

            gadget.resize();
            gadget.hideLoading();

        },
        args: [{
            key: "issueData",
            ajaxOptions: function() {
                return {
                    url: "/rest/quadro-acompanhamento-sla-gadget/1.0/issues/getIssues.json",
                    data:
                    {
                        filtro : gadgets.util.unescapeString(this.getPref("filterId"))
                    }
                };
            }

        }]
    }
});
})();
gadgets.window.adjustHeight();
</script>

<style type="text/css">
    .tabla_quadro_sla { border: 1px solid #D2D2D2; text-align: center; width: 100%;  }
    .cabecera_tabla_quadro_sla { background-color: #96A629; color: white; height: 50px; }
    .td_cabecera_tabla_quadro_sla { border: 1px solid #D2D2D2; text-align: center }
    .td_body_tabla_quadro_sla { border: 1px solid #D2D2D2; padding : 2px; }
</style>
]]>
</Content>
</Module>
```

Una vaz finalizada la construcción de la tabla la añadimos a la vista del Gadget

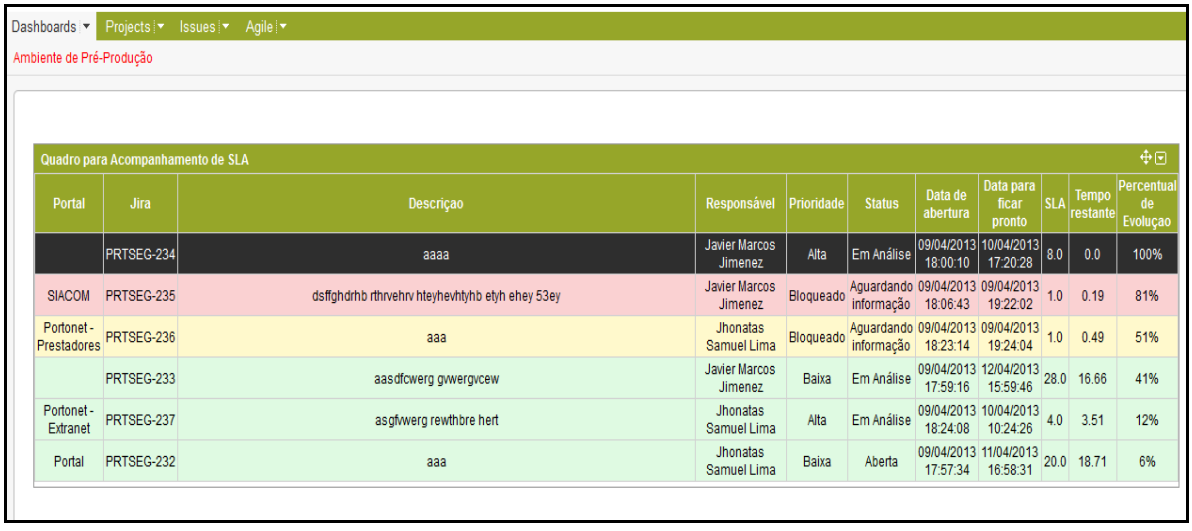
Si el filtro no tiene peticiones pintamos en la capa de info el mensaje que no devuelve peticiones

Si la llamada Rest devuelve respuesta sin datos pintamos el mensaje en la capa de error

Esto es para que cuando finalice la construción de la vista se quite la barra de "cargando" y muestre la vista del Gadget

Añadimos el css de la vista

Si volvemos a compilar el JAR y a reiniciar la aplicación, vemos el resultado final de nuestro Gadget:



Portal	Jira	Descriçao	Responsável	Prioridade	Status	Data de abertura	Data para ficar pronto	SLA	Tempo restante	Percentual de Evoluçao
	PRTSEG-234	aaaa	Javier Marcos Jimenez	Alta	Em Análise	09/04/2013 18:00:10	10/04/2013 17:20:28	8.0	0.0	100%
SIACOM	PRTSEG-235	dsffghdrhb rthrvehrv hteyhevhtyhb etyh ehey 53ey	Javier Marcos Jimenez	Bloqueado	Aguardando informaçao	09/04/2013 18:06:43	09/04/2013 19:22:02	1.0	0.19	81%
Portonet - Prestadores	PRTSEG-236	aaa	Jhonatas Samuel Lima	Bloqueado	Aguardando informaçao	09/04/2013 18:23:14	09/04/2013 19:24:04	1.0	0.49	51%
	PRTSEG-233	aasdfcwerg gvwergvcew	Javier Marcos Jimenez	Baixa	Em Análise	09/04/2013 17:59:16	12/04/2013 15:59:46	28.0	16.66	41%
Portonet - Extranet	PRTSEG-237	asgfwerg rewthbre hert	Jhonatas Samuel Lima	Alta	Em Análise	09/04/2013 18:24:08	10/04/2013 10:24:26	4.0	3.51	12%
Portal	PRTSEG-232	aaa	Jhonatas Samuel Lima	Baixa	Aberta	09/04/2013 17:57:34	11/04/2013 16:58:31	20.0	18.71	6%