**Plan de Gestión de**

**Configuración**

**V 1.0**

***Fecha: 29/04/2017***

ÍNDICE

[1. Introducción 3](#_Toc1404)

[1.1. Propósito 3](#_Toc2171)

[1.2. Alcance 3](#_Toc31978)

[1.3. Abreviaciones 3](#_Toc2311)

[1.4. Resumen Ejecutivo 4](#_Toc17250)

[2. Gestión de la SCM 5](#_Toc26475)

[2.1. Organización 5](#_Toc29810)

[2.2. Roles y responsabilidades 7](#_Toc19110)

[2.3. Políticas y directrices y procedimientos 8](#_Toc22048)

[2.4. Herramientas, entorno e infraestructura 12](#_Toc14242)

[2.4.1. Herramientas de control de versiones 12](#_Toc15552)

[2.4.2. Herramientas de entorno 14](#_Toc20520)

[2.5 Calendario 17](#_Toc3734)

[3. Actividades de la SCM](#_Toc1404) 17

[3.1. Identificacion de la configuracion](#_Toc2171) 17

[3.1.1. Items de los Items de la Configuracion](#_Toc31978) 17

[3.1.2. Definicion de la Nomenclatura de los Items](#_Toc2311) 18

[3.1.3. Lista de los Items con Nomenclatura](#_Toc2311) 19

# **Introducción**

## **Propósito**

El propósito de este documento es describir las actividades de gestión de configuración de software que deben ser llevadas a cabo durante el proceso de desarrollo del proyecto. Aquí se definen tanto los productos que se pondrán bajo control de configuración como los procedimientos que deben ser seguidos por los integrantes del equipo de trabajo.

## **Alcance**

El presente plan de configuración está realizando teniendo presente lo siguiente:

* El proyecto dura un aproximado de 12 semanas.
* El modelo de proceso de desarrollo de software a seguir es SCRUM
* Los proyectos tipo Front-end son desarrollados usando el Framework Angular 1.x.
* Los proyectos tipo Móvil son desarrollados en Java, usando el IDE Android Studio.
* Los proyectos tipo Back-end son desarrollados en Node.js, usando el framework Loopback.

## **Abreviaciones**

* SCMR: Software Configuration Management Responsible, Gestor de configuración del software.
* CCB: Configuration Contorl Board, Comité del Control de la Configuración.
* PL: Program Librarian, Bibliotecario

## **Resumen Ejecutivo**

Los siguientes puntos del presente documento van a presentar la siguiente estructura:

* Gestión del SCM: se muestra las responsabilidades y responsables del proyecto
* Actividades del SCM: se identifican las actividades y tareas que se requieren para la configuración del sistema
* Control:Configuración y control de cambios en la configuración del proyecto
* Estado:Registro y seguimiento de las actividades del SCM
* Auditoría: Como nos aseguramos de que los cambios efectuados se han hecho correctamente.
* Gestión de Release: se detalla cómo se hará la entrega final del proyecto al cliente.

# **Gestión de la SCM**

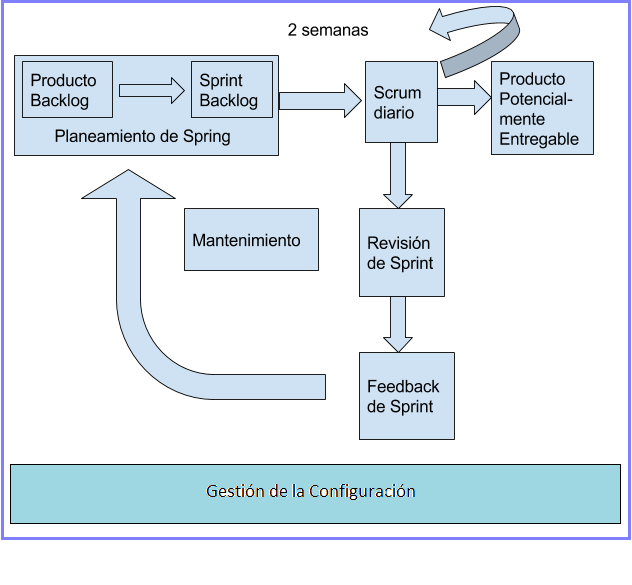
## **Organización**

SP11 es una empresa peruana que, desde su creación desarrolla software a la medida con personal especializado que reside en Perú, nuestra empresa fue fundada en el año 2015, nuestra propuesta de propuesta de valor se basa en la entrega de soluciones software personalizadas, capaces de soportara procesos clave y actividades de misión crítica, alineados con los objetivos estratégicos de nuestros clientes.

Para lograrlo utilizamos herramientas y metodologías de desarrollo, y de prueba que nos permiten entregar soluciones flexibles, escalables, con una gran facilidad de uso, multiplataforma, en línea y en tiempo real cuando sea requerido, fáciles de integrar y con los máximos niveles de seguridad.

Nuestro objetivo es que nuestros clientes nos consideren un socio estratégico a largo plazo, en todo lo que se refiera de entrega de soluciones tecnológicas que resuelven sus problemas de negocio.

Para cumplir con esta espectativas nuestra empresa como tal, cuenta con un esquema organizacional, donde cada integrante de nuestro equipo conoce sus obligaciones en nuestra organización, el siguiente esquema muestra la forma como está distribuido, manteniendo por supuesto una jerarquía como cualquier otra empresa:



*Relación del modelo de proceso usando, respecto a la gestión de configuraicón y mantenimiento*

La siguiente tabla muestra cómo se relacionan las actividades del proyecto con los roles en el proyecto.

|  |  |
| --- | --- |
| **Actividad** | **Responsable** |
| Planificar la Configuración del SCM | SCMR |
| Definición de la línea base del proyecto | SCMR, CCB, PL |
| Seguimiento del proyecto de la línea base | SCMR |
| Definición del ambiente | SCMR |
| Control de cambios | SCMR, CCB |
| Descripción de la versión | SCMR |
| Realizar informe final del SCM | SCMR |

Los integrantes usan google docs para estar vinculados a las actividades del SCM y posteriormente subidas al repositorio de github.

## **Roles y responsabilidades**

* **SCMR**
  + Debe proveer la infraestructura y el entorno de configuración para el proyecto. Debe preocuparse porque todos los integrantes del grupo entiendan y puedan ejecutar las actividades de SCM que el Plan les asigna, así como asegurar que éstas sean llevadas a cabo. Seguir la línea base, controlando las versiones y cambios de ella, son tareas correspondientes a el. Debe definir y construir el Ambiente Controlado e informar al resto del equipo sobre la manera de usarlo.
  + Personal necesario: 1
* **Bibliotecario (*Program Librarian*, PL)**
  + Encargado de respaldar con las tareas de almacenamiento de los *baselines*. Un *baseline* o línea base es uno o más documentos formalmente diseñados y corregidos en un tiempo específico del ciclo de vida de los ítems de configuración, estos a su vez, son el conjunto de versiones de un item aceptado por el cliente.
  + Personal necesario: 1
* **Comité de Control de Configuración (Configuration Control Board, CCB)**,
  + Máxima autoridad en la autorización de cambios.
  + Tiene la autoridad para aceptar o rechazar las propuestas de cambio a componentes de configuración. Como estos cambios tienen sentido controlarlos una vez que se crean las líneas base, el comité de control de cambios tiene la autoridad para gestionar las líneas base del producto y asegurar que los cambios son adecuadamente considerados y coordinados.
  + Personal necesario: 2

## **Políticas y directrices y procedimientos**

Respecto a los documentos en el repositorio de github, todos tienen derechos de administrador, osea, permiso de lectura, escritura y creación de nuevos documentos. Se tiene como política la total confianza en todos los integrantes del equipo.

**Políticas.-**

* Seguir los estándares de nomenclatura plasmados en el presente documento.
* Cada rol debe cumplir las actividades designadas en el documento.
* Cada miembro debe trabajar únicamente y exclusivamente sobre su branch.
* La filtración de información a agentes externos se encuentra extrictamente prohibida.
* Correcta comunicación entre los miembros encargados de la gestión de la configuración.
* Toda dependencia en los proyectos deberá ser estrictamente documentada.

**Directrices.-**

* Se encuentra estricamente prohibido el versionamiento en el nombre del item. Ejm: No se permite el nombre “SCAE\_AC\_v1.1.docx”
* Para los items de gestión de la configuración:
  + Los items deberán llevar el siguiente estándar de nombre:

<INICIALES DEL DOCUMENTO>.\*

Ejm: PGC.docx (Plan de gestión de la configuración)

* Para los items de gestión de proyectos:
  + Los items deberán llevar el siguiente estándar de nombre:

<INICIALES DEL PROYECTO>\_<INICIALES DEL DOCUMENTO>.\*

Ejm: SCAE\_AC.docx

* Los nombres de los branch deberá ser hecho en snake\_case de los nombres de los responsables en minúsculas. Ejm: Josafat Vara -> branch “josafat\_vara”
* Para los items de desarrollo de proyectos FrontEnd:
  + Todo folder o item que cuente con más de 2 palabras, deberá ser nombrado en Kebab-case. Ejm: dashboard home -> dashboard-home
  + Los componentes:
    - Los items que expongan los componentes deberán ser nombrados de la siguiente manera:

<NOMBRE DEL COMPONENT O FEATURE>.component.js

* + Las vistas o templates:
    - Deberán ir en el folder de su respectivo componente o feature (característica de la aplicación).
    - Deberán ser nombradas de la siguiente manera:

<NOMBRE DEL COMPONENT O FEATURE>.template.html

* + Los controladores
    - Deberán ir en el folder de su respectivo componente o feature (característica de la aplicación).
    - Deberán ser nombradas de la siguiente manera:

<NOMBRE DE COMPONENT O FEATURE>.controller.js

* + Los services(servicios)
    - Deberán ir en el folder src/app/core del proyecto.
    - Dependiendo de la utilidad del servicio, este deberá ir en la siguiente carpetas:
      * Utils.- servicios que utilizan componentes.
      * Helpers.- servicios para evitar la repetición de código o snippets.
      * Services.- servicios con lógica de procesos.
    - Los items que expongan los servicios, deberán ser nombrados de la siguiente manera:

<NOMBRE DEL SERVICIO>.service.js

* + Los resources(recursos de datos)
    - Deberán ir en el folder src/app/core/resources del proyecto.
    - Dependiendo de su fuente, deberán ir en las carpetas:
      * Local.- Extracción de datos almacenados en el navegador.
      * Test.- Para mock.
      * Remote.- Extracción de datos de fuentes remotas(APIs).
  + Los estilos de la aplicación
    - Deberán ir en src/assets.
    - Se deberá seguir el patrón de diseño 7-1. Ref: [https://sass-guidelin.es/es/#el-patron-7-1](https://sass-guidelin.es/es/" \l "el-patron-7-1)
  + Imágenes y documentos
    - Deberán ir en las carpetas src/images y src/docs respectivamente.
* Para los items de proyectos BackEnd:
  + Todo item con más de dos palabras deberá ser nombrado en Kebab-case.

Ejm: logging middleware -> logging-middleware

* + Será usado como estructura del proyecto el generado por el Framework loopback.
* Para los items de proyectos Móvil:
  + Todo folder el cual contenga más de 2 palabras como por ejemplo: android Test, será nombrado del siguiente modo: androidTest. En resumen del siguiente modo: <nombreDeLaCarpeta>
    - Solo en el caso de tratarse de las carpetas de los componentes gráficos tendrán esta estructura:

En resumen del siguiente modo: <drawable-hdpi>

* + Los componentes serán nombrados del siguiente modo:

Las clases hechas en JAVA las son expresadas del siguiente modo:

<NombreDeLaClase>.java

* + Los layouts y componentes gráficos o vistas se organizarán del siguiente modo:

Los archivos xml están con la siguiente nomenclatura: <activity\_main>.xml

Los archivos como fotos o imágenes tienen la siguiente nomenclatura: <ic\_add\_without\_circle>.png

* + Todo archivo que contenga configuración del mismo proyecto Android como el graddle o los services de Google API terndrán la siguiente nomenclatura:

<nombre-archivo>.<extension>

Ejm:

google-services.json

proguard-rules.pro

* + - * + Todas las imágenes se encuentran distribuidas en la carpeta <nombre- app>/<app>/<src>/<main>/<res>, y dependiendo su dimensión se pueden ubicar en drawable, drawable-nodpi, etc.

**Procedimientos.-**

* Respecto a la modificación de archivos pertenecientes a la gestión de la configuración.

1. Indicar a demás miembros encargados de los documentos qué documento está siendo manipulado, con el fin de evitar modificaciones en el mismo por parte de otro miembros, lo cual repercutiría en un sobreescribimiento del item.
2. Modificar el item apropiadamente.
3. Subir los cambios.
4. Hacer los merge necesarios para que el item modificado se encuentre disponible para todos.
5. Informar que el cambio fue realizado.

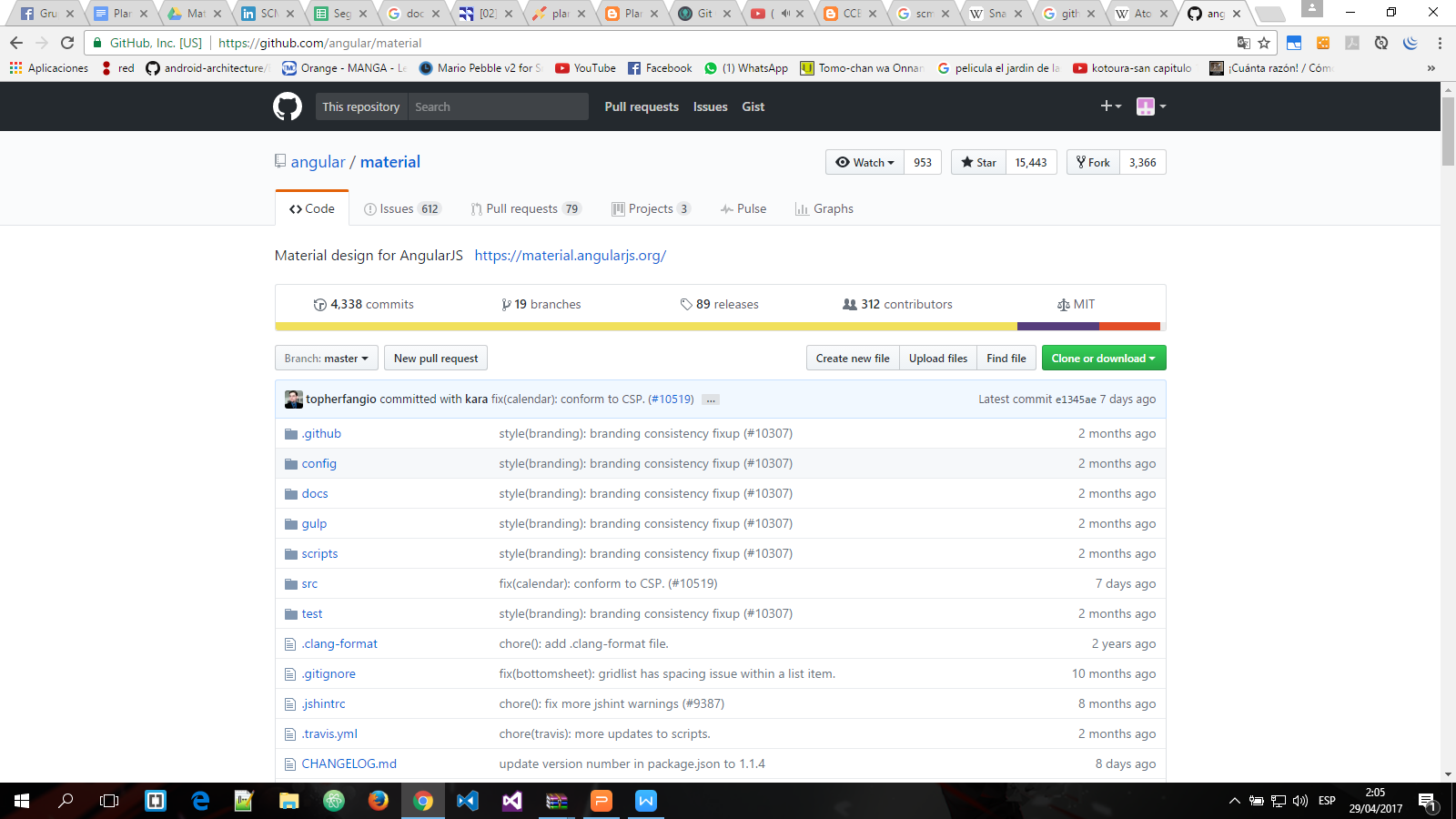
## **Herramientas, entorno e infraestructura**

### **Herramientas de control de versiones**

* Github

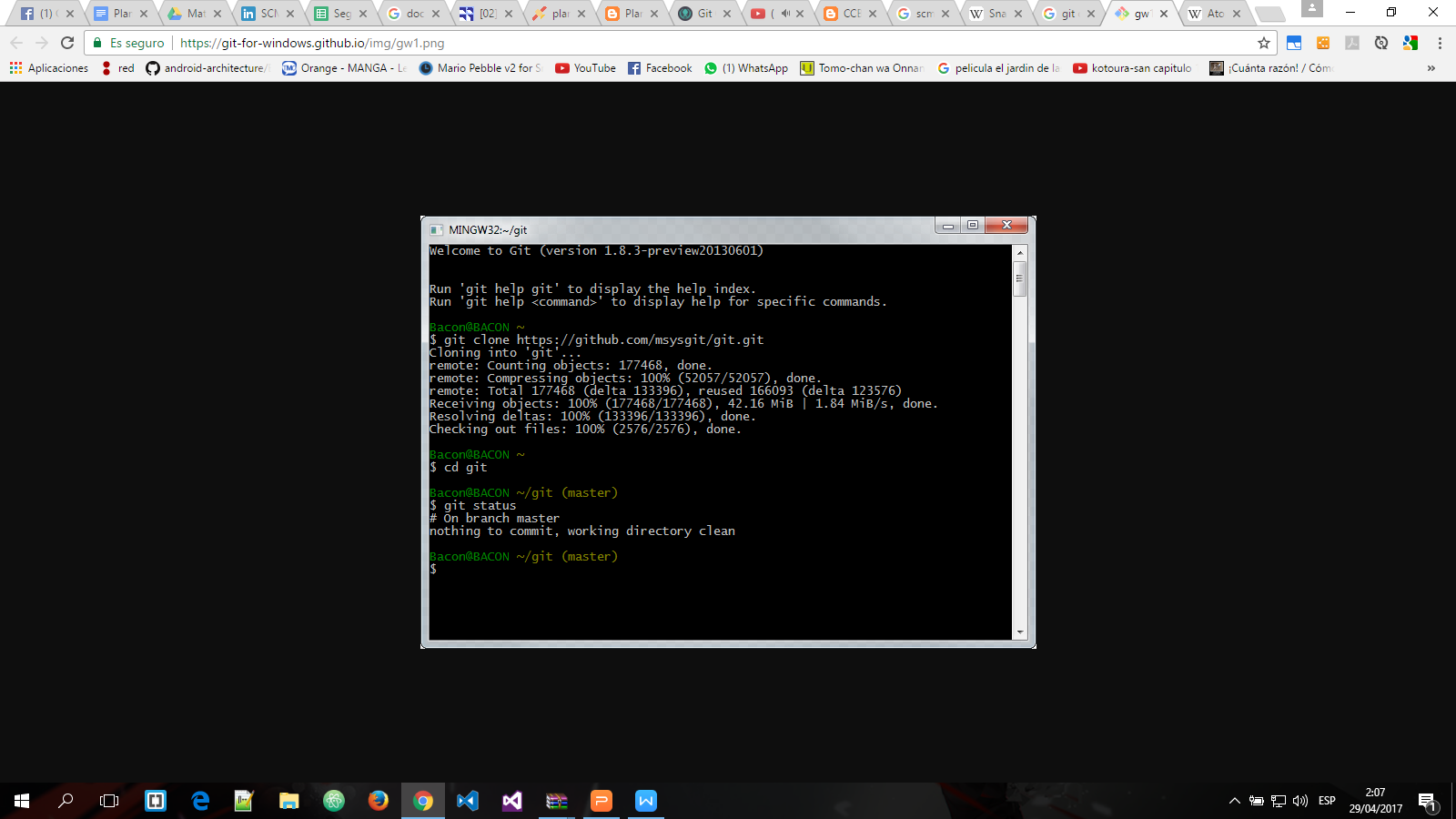
Un hosting online para nuestros repositorios que utiliza git para el mantenimiento y versionado del código fuente, añadiendo una serie de servicios extras para la gestión del proyecto y el código fuente. La parte gratuita de este hosting permite alojar nuestro código en repositorios públicos, si queremos repositorios privados entramos a la parte “premium”. En la actualidad ofrecen varias herramientas útiles para el trabajo en equipo, entre ellos cabe destacar

* Una wiki: para el mantenimiento de distintas versiones de las páginas.
* Un sistema de seguimiento de problemas, que permite a un miembro de tu equipo detallar el problema con tu software.
* Herramienta de versión de código, donde puedes añadir anotaciones en cualquier punto de un fichero.
* Un visor de ramas: donde se puede comparar los progresos realizados en las distintas ramas de nuestro repositorio.



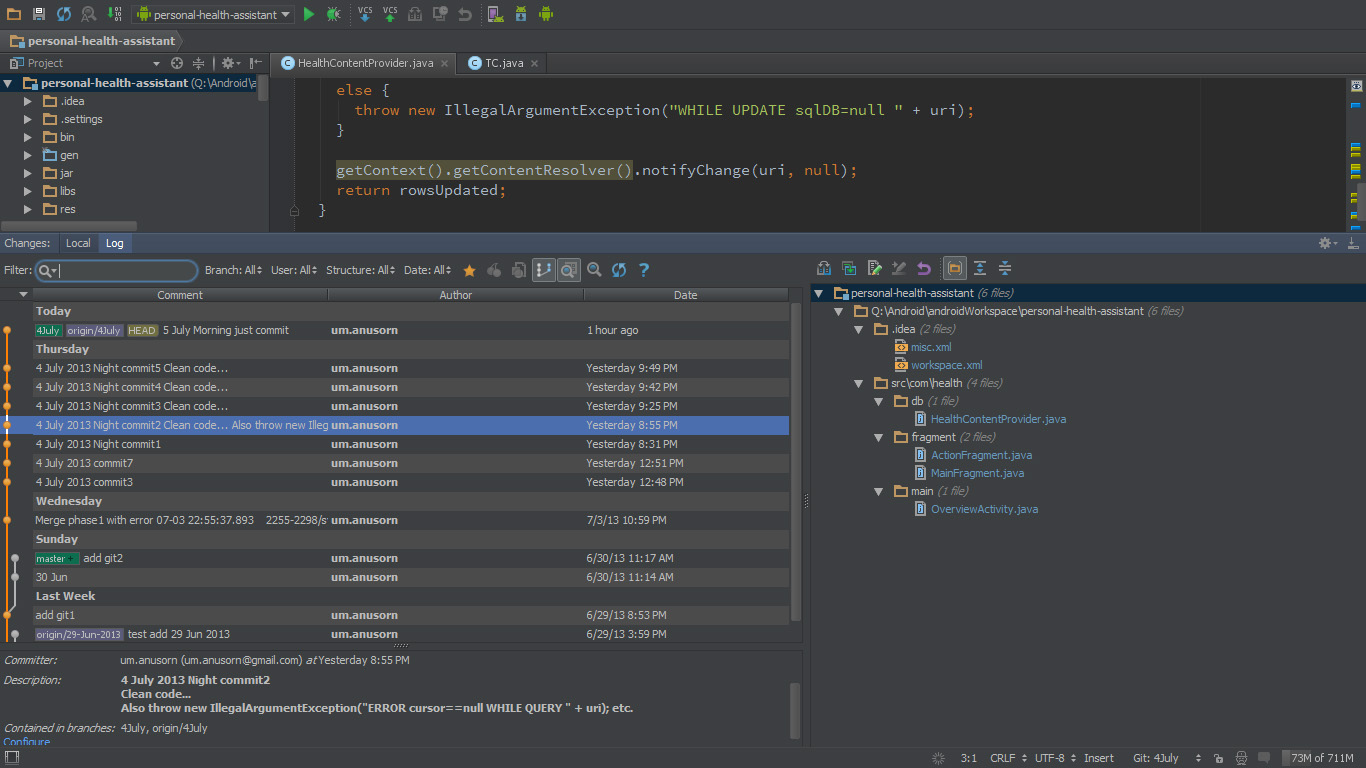
* Git CLI

Git es un sistema rápido, escalable, distribuido de control de versiones con un número muy grande de comandos que proporciona las operacione de alto nivel y acceso completo al nivel inferior. La herramienta a usar para el caso de proyectos Front-end y Back-end será Git CLI(Command Line).



* Android Studio Git

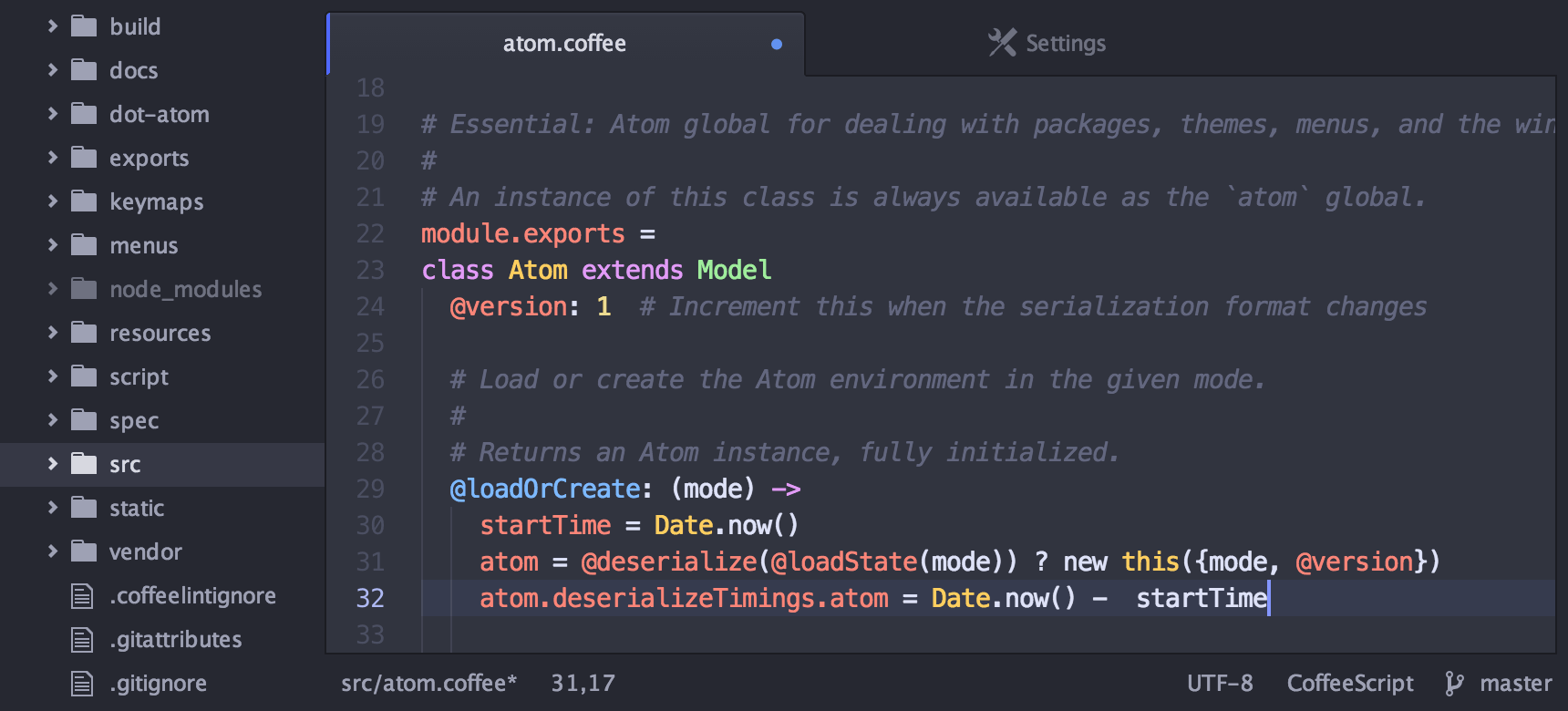
Plugin del IDE Android Studio para gestionar branchs y cambios en el mismo entorno de desarrollo.



### **Herramientas de entorno**

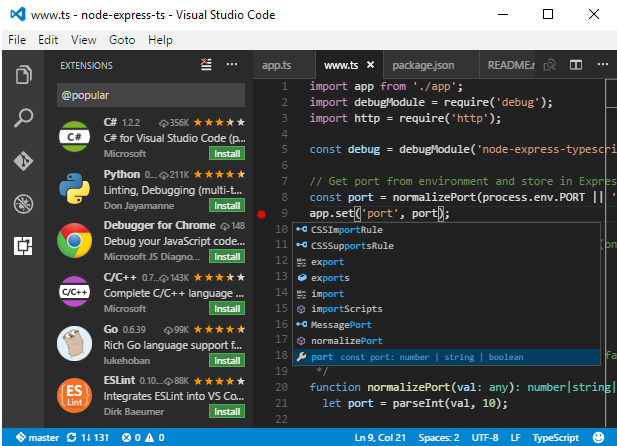
* Atom(Backend)

Es un editor de código de fuente abierta para macOS, Linux, y Windows1con soporte para plug-ins escrito en Node.js, Incrustando Git Control, desarrollado por GitHub. Atom es una aplicación de escritorio construida utilizando tecnologías web. La mayor parte de los paquetes tienen licencias de software libre y es construido y mantenido por su comunidad. Atom está basado en Electrón (Anteriormente conocido como Atom Shell), Un framework que permite aplicaciones de escritorio multiplataforma usando Chromium y Node.js. Está escrito en CoffeeScript y Less.



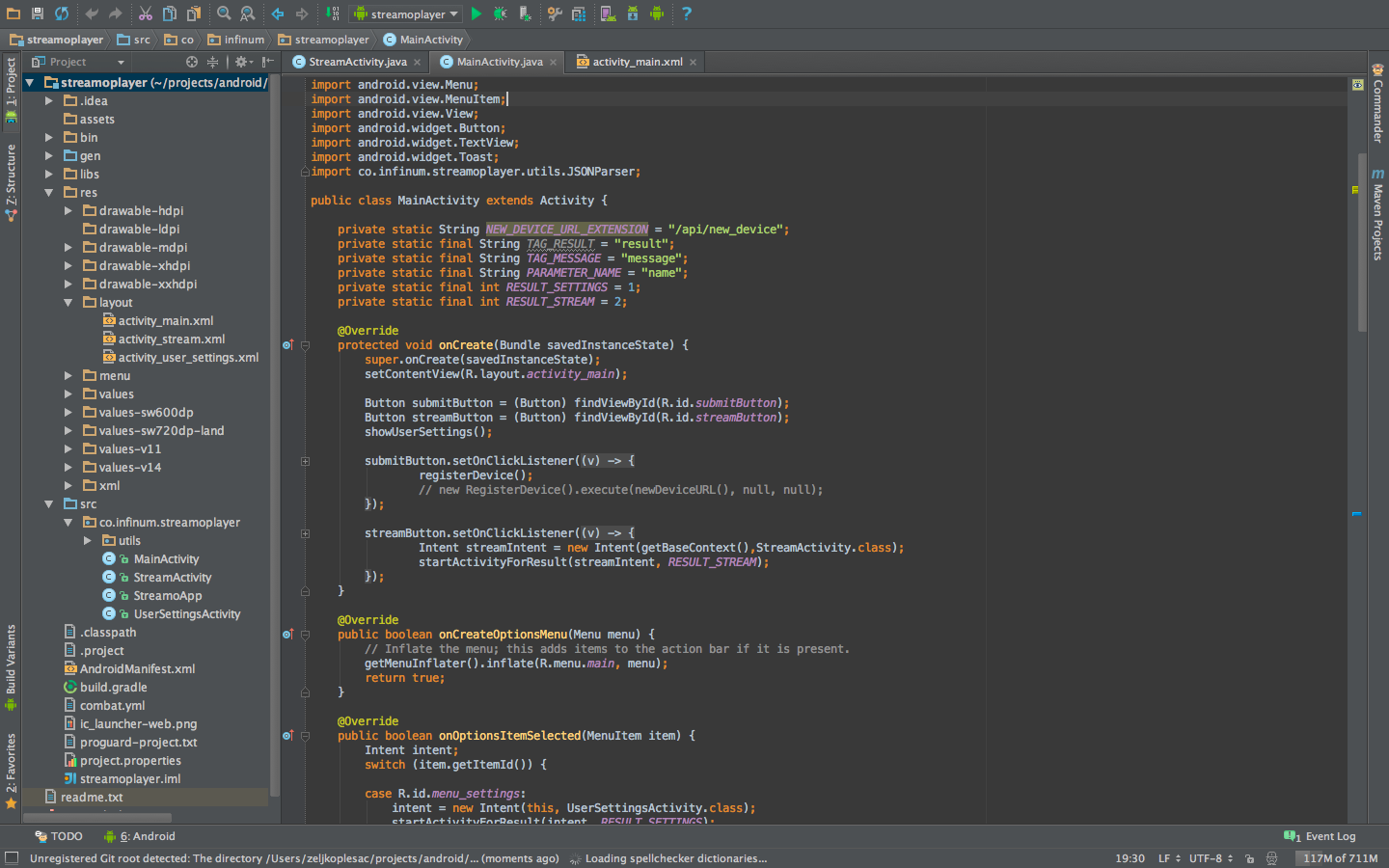
* Visual Studio Code(Frontend)

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS . Incluye soporte para depuración, control Git incorporado, resaltado de sintaxis, terminación de código inteligente, fragmentos y refactorización de código . También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, atajos de teclado y preferencias. Es libre y de código abierto, aunque la descarga oficial está bajo una licencia de propiedad . Visual Studio Code se basa en Electron, un framework que se utiliza para desplegar aplicaciones Node.js para el escritorio que se ejecuta en el motor de diseño Blink . Aunque utiliza el marco de Electron, el software no utiliza Atom y en su lugar emplea el mismo componente de editor (codificado "Monaco") utilizado en Visual Studio Team Services (antes llamado Visual Studio Online).



* Android Studio(Móvil)

Es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014. Está basado en el software IntelliJ IDEA de JetBrains, y es publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux.



**2.5 Calendario**

|  |  |  |
| --- | --- | --- |
| **Actividad** | **Depende de** | **Comienzo** |
| Realizar el plan de gestión de la configuración | Nada | Semana 1: 29/04/2017 |
|  |  |  |
|  |  |  |
|  |  |  |

# **Actividades de la SCM**

En dicha sección se realizarán diferentes actividades requeridas cuyos objetivos son:

* Asegurar que en el plan del proyecto se haya contemplado recursos a un apropiado para estructurar la gestión de la configuración del proyecto
* Delegación explicita de actividades de gestión de la configuración a líderes apropiados
* Establecer el Comité de control de cambios (CCC) y establecer un procedimiento para su gestión
* Definir un esquema de versionamiento que satisfaga el ciclo de vida del proyecto.
* Crear una estructura organizacional intuitiva y de fácil uso para almacena la información del proyecto

## **Identificación de la configuración**

## **Ítems de configuración:**

Para este proyecto se ha clasificado los ítems de configuración en:

* **Ítems en evolución**: Son los documentos que están sujetos a una o más revisiones y nuevas liberaciones durante el ciclo de vida.
* **Ítems fuente:** Son generalmente código fuente y archivos objeto utilizados para compilar una aplicación de software para ambiente de producción, los cuales son generalmente números y cambian frecuentemente.
* **Ítems de soporte:** como sistemas operativos y software base, de los cuales el proyecto requiere ciertas versiones para su operación exitosa
* **Ítem de Archivo:** Son las revisiones SQA (Aseguramiento de la Calidad del Software), las cuales generalmente sirvieron de soporte de decisiones durante el ciclo de vida del software, son almacenadas normalmente en formato electrónico para referencia futura

En el siguiente cuadro se señalarán los diferentes ítems a realizar en este proyecto con su respectiva clasificación tomada:

|  |  |  |
| --- | --- | --- |
| **TIPO** | **NOMBRE DEL ITEM** | **FUENTE** |
| **Evolución** | Plan de la Gestión de la Configuración | Empresa |
| **Evolución** | Documento de Acta del Proyecto | Empresa |
| **Evolución** | Modelo de Datos | Proyecto |
| **Evolución** | Manuales de Usuario | Proyecto |
| **Evolución** | Documento de Requerimientos | Proyecto |
| **Fuente** | Reporte de Asistencias | Proyecto |
| **Fuente** | Pantalla del Administrador | Proyecto |
| **Fuente** | Apk del Proyecto móvil | Proyecto |
| **Soporte** | Frameworks empleados | Proyecto |
| **Soporte** | Android Studio(Version Utilizada) | Proyecto |
| **Archivo** | Plan de calidad | Empresa |
| **Archivo** | Documento de Evaluacion y Ajuste del Plan de Calidad | Empresa |
| **Archivo** | Informe de la Revision Tecnica Formal (RTF) | Empresa |

## **Definiciones de nomenclatura de los ítems:**

Teniendo en cuenta la clasificación tomada a los ítems de configuración se tomará la siguiente nomenclatura:

* **Ítems en evolución**:
  + Sea el caso de ítems de empresa → [Iniciales del documento].[docx|\*]
  + Sea el caso de ítems de proyecto → [Iniciales del proyecto]\_[Iniciales del documento].[docx|\*]
* **Ítems fuente:**
  + Para los items de desarrollo de proyectos FrontEnd:
    - Todo item que cuente con más de 2 palabras, deberá ser nombrado en Kebab-case.
    - Los componentes: <NOMBRE DEL COMPONENT O FEATURE>.component.js
    - Las vistas o templates: <NOMBRE DEL COMPONENT O FEATURE>.template.html
    - Los controladores <NOMBRE DE COMPONENT O FEATURE>.controller.js
    - Los services(servicios) <NOMBRE DEL SERVICIO>.service.js
  + Para los items de proyectos BackEnd:
    - Todo item con más de dos palabras deberá ser nombrado: [nombre del ítem en kebab-case].{js|json}
  + Para los items de proyectos Móvil:
    - Las clases hechas en JAVA las son expresadas del siguiente modo: [Nombre de la clase en PascalCase].java
    - Los layouts y componentes gráficos o vistas serán nombrados en <nombre del ítem en snake\_case>.{png|xml}
* **Ítems de soporte:**
  + Para los proyectos de Front-end ,Back-end y Android,los archivos de configuración serán nombrados: [nombre del archivo en kebab-case].[extesion]
* **Ítem de Archivo:**
  + Serán nombrados: [Iniciales del proyecto]\_[Iniciales del documento].docx

## **Lista de los Ítems con Nomenclatura:**

En el siguiente cuadro se dara la nomenclatura de algunos ítems durante el ciclo de vida del software (se tomará de ejemplo a “Sistema de Control de Asistencia de Estudiantes”):

|  |  |  |
| --- | --- | --- |
| **NOMENCLATURA** | **ENTREGABLE** | **TIPO** |
| **SCAE\_DCC** | Modelo de Casos de Uso | Requerimientos |
| **SCAE\_AS** | Alcance del Sistema | Requerimientos |
| **SCAE\_MD** | Modelo de Diseño | Diseño |
| **SCAE\_DA** | Diseño de la Arquitectura | Diseño |
| **SCAE\_II** | Informe de Integración | Implementación |
| **SCAE\_AC** | Informe de Consolidación | Implementación |
| **SCAE\_MCP** | Modelo de Casos de Prueba | Verificación |
| **SCAE\_IVD** | Informe de Verificación de Documento | Verificación |
| **PGC** | Plan de gestión de configuración | SCM |
| **PCC** | Plan de Control de Cambios | SCM |
| **SCAE\_PC** | Plan de calidad | SQA |
| **SCAE\_DEAPC** | Documento de Evaluación y Ajuste del Plan de Calidad | SQA |