**DESARROLLO DE UN MARCO DE TRABAJO SIGUIENDO EL ESTÁNDAR ISO/IEC 29110 PARA EL CASO DE ESTUDIO DE INTEGRACIÓN DE PROCESOS DEL PROYECTO SANAMBIENTE.**

Aarón Levi Grajales Gómez, aarongrajales@hotmail.com

Cristhian Fernando Balanta Pazú, cristianpazu2@gmail.com

Semillero ITMedia - Grupo de investigación GRINTIC

Trabajo de grado presentado para optar al título de Ingeniero de sistemas

Asesor(a): Beatriz Eugenia Marín, Ingeniera de sistemas



INSTITUCIÓN UNIVERSITARIA ANTONIO JOSÉ CAMACHO

FACULTAD DE INGENIERÍAS

INGENIERÍA DE SISTEMAS

SANTIAGO DE CALI - COLOMBIA

2020

**Notas de aceptación:**

**Aprobado por el Comité de Grado en cumplimiento de los requisitos exigidos por la Institución Universitaria Antonio José Camacho para optar al título de Ingeniero de Sistemas.**

**Jurado**

**Jurado**

**Santiago de Cali, 30 de junio de 2020**

**Resumen**

Actualmente se está desarrollando un ejercicio académico entre estudiantes de Ingeniería de Sistemas y Tecnología de Sistemas de la Institución Universitaria Antonio José Camacho con la empresa Sanambiente S.A. que busca construir una aplicación que permita gestionar los datos ambientales de las estaciones que la empresa tiene a lo largo del país, para ello se han conformado varios equipos que abordan el proyecto desde diferentes aspectos. Por la cantidad de integrantes se hace necesario aplicar un modelo de proceso de gestión que facilite el control y seguimiento de todos los recursos que intervienen, por esta razón el objetivo principal de este proyecto es establecer un marco de trabajo que permita administrar la construcción de la aplicación, siguiendo estándares de calidad y gestión, de tal manera que pueda ser replicado para futuros proyectos.

**Palabras clave:** ISO/IEC 29110, Gerencia de proyecto, Iconix.

**Abstract**

Currently an academic exercise is being developed between students of Systems Engineering and Systems Technology of the Antonio José Camacho University Institution with the company Sanambiente S.A. that seeks to build an application that allows managing the environmental data of the stations that the company has throughout the country, for this several teams have been formed that approach the project from different aspects. Due to the number of members it is necessary to apply a management process model that facilitates control and monitoring of all resources involved, for this reason the main objective of this project is to establish a framework to manage the construction of the application, following standards of quality and management, so that it can be replicated for future projects.

**Keywords:** ISO/IEC 29110, Project Management, Iconix.

**Tabla de contenido**

[Introducción 1](#_Toc40044834)

[1. Planteamiento del problema 2](#_Toc40044835)

[2. Objetivos 5](#_Toc40044836)

[2.1 Objetivo general 5](#_Toc40044837)

[2.2 Objetivos específicos 5](#_Toc40044838)

[3. Problema de investigación 6](#_Toc40044839)

[3.1 Formulación del problema 6](#_Toc40044840)

[3.2 Sistematización del problema 6](#_Toc40044841)

[3.3 Justificación 6](#_Toc40044842)

[4. Marco de referencia 8](#_Toc40044843)

[4.1 Antecedentes 8](#_Toc40044844)

[4.2 Marco conceptual 10](#_Toc40044845)

[4.2.1 Proceso de software 10](#_Toc40044846)

[4.2.2 Metodología de desarrollo de software 11](#_Toc40044847)

[4.2.3 Lenguaje Unificado de Modelado (Unified Modeling Language (UML)) 11](#_Toc40044848)

[4.2.4 Calidad de software 12](#_Toc40044849)

[4.2.5 Modelo de calidad de software 12](#_Toc40044850)

[4.3 Marco teórico 12](#_Toc40044851)

[4.3.1 Revisión de área 12](#_Toc40044852)

[4.3.1.1 PMBOK 14](#_Toc40044853)

[4.3.2 Comparación entre estándares para la gestión de proyectos 15](#_Toc40044854)

[4.3.3 Estándar ISO/IEC 29110. 16](#_Toc40044855)

[4.3.4 Comparación entre metodologías de desarrollo de software 22](#_Toc40044856)

[4.3.5 Metodología Iconix. 24](#_Toc40044857)

[4.3.6 Reconocimiento de las tecnologías utilizadas por los equipos de trabajo 29](#_Toc40044858)

[4.3.6.1 Angular Framework 29](#_Toc40044859)

[4.3.6.2 JavaScript 29](#_Toc40044860)

[4.3.6.3 TypeScript 29](#_Toc40044861)

[4.3.6.4 PostgreSQL 30](#_Toc40044862)

[4.3.6.5 NodeJS 30](#_Toc40044863)

[4.3.6.6 GitHub 30](#_Toc40044864)

[4.3.6.7 GitHub Desktop 30](#_Toc40044865)

[4.3.6.8 Visual Studio Code 30](#_Toc40044866)

[4.3.6.9 Jest 31](#_Toc40044867)

[4.3.6.10 Jmeter 31](#_Toc40044868)

[4.4 Marco legal 31](#_Toc40044869)

[5. Resultados 33](#_Toc40044870)

[5.1 Fase 1: Planificación del proyecto 33](#_Toc40044871)

[5.1.1 Crear el plan del proyecto. 33](#_Toc40044872)

[5.1.2 Crear repositorio del proyecto 86](#_Toc40044873)

[5.1.3 Registrar los resultados de la verificación 88](#_Toc40044874)

[5.2 Fase 2: Ejecución del plan del proyecto 89](#_Toc40044875)

[5.2.1 Diligenciar un acta de reunión 89](#_Toc40044876)

[5.2.2 Diligenciar el registro de estado de progreso 90](#_Toc40044877)

[5.2.3 Diligenciar una solicitud de cambio 91](#_Toc40044878)

[5.3 Fase 3: Evaluación y control del proyecto 92](#_Toc40044879)

[5.3.1 Diligenciar el registro de corrección 93](#_Toc40044880)

[5.4 Fase 4: Cierre del proyecto 93](#_Toc40044881)

[5.4.1 Diligenciar un registro de aceptación. 94](#_Toc40044882)

[6. Conclusiones 95](#_Toc40044883)

[7. Referencias 96](#_Toc40044884)

**Lista de tablas**

[Tabla 1. Diagnóstico y pronóstico del proyecto. 4](#_Toc40044808)

[Tabla 2. Diagramas de UML por categoría. 12](#_Toc40044809)

[Tabla 3. Estándares para la gestión de proyectos. 13](#_Toc40044810)

[Tabla 4. Comparación entre estándares para la gestión de proyectos. 15](#_Toc40044811)

[Tabla 5. Público objetivo de la ISO/IEC 29110. 17](#_Toc40044812)

[Tabla 6. Productos o artefactos según la ISO/IEC 29110. 21](#_Toc40044813)

[Tabla 7. Comparación entre metodologías de desarrollo de software. 23](#_Toc40044814)

**Lista de figuras**

[Figura 1. Diagrama Causa y efecto. 4](#_Toc40044795)

[Figura 2. Procesos básicos de la guía de perfiles. 18](#_Toc40044796)

[Figura 3. Diagrama del proceso de gestión del proyecto. 19](#_Toc40044797)

[Figura 4. Diagrama del proceso de implementación de software. 20](#_Toc40044798)

[Figura 5. Procesos de ICONIX. 25](#_Toc40044799)

[Figura 6. Captura de imagen de referencia de la aplicación de escritorio de GitHub: GitHub Desktop. 87](#_Toc40044800)

[Figura 7. Repositorio del proyecto, visto desde la aplicación web de GitHub. 88](#_Toc40044801)

[Figura 8. Ejemplo de una plantilla de resultados de la verificación del plan de proyecto. 89](#_Toc40044802)

[Figura 9. Ejemplo de un acta de reunión lograda con el equipo de trabajo. 90](#_Toc40044803)

[Figura 10. Ejemplo de un registro de estado de progreso sobre los casos de uso 04 y 08 pertenecientes al equipo de desarrollo. 91](#_Toc40044804)

[Figura 11. Ejemplo de una solicitud de cambio iniciada. 92](#_Toc40044805)

[Figura 12. Ejemplo de un registro de corrección en estado pendiente. 93](#_Toc40044806)

**Índice de acrónimos**

**ISO**: International Organization for Standardization.

**IEC:** International Electrotechnical Commission**.**

**PMI:** Project Management Institute.

**PMBOK:** Project Management Body of Knowledge.

**APM:** Association for Project Management.

**APMBOK:** Association for Project Management Body of Knowledge.

**CMMI-DEV:** Capability Maturity Model Integration for Development.

# Introducción

El siguiente proyecto describe el cómo implementar un marco de trabajo siguiendo estándares y metodologías para facilitar el control y seguimiento de todos los recursos que intervienen en el desarrollo de un producto de software. El estándar ISO/IEC 29110 es el que se tomará como base para llevar a cabo el desarrollo de este proyecto, éste se divide en dos procesos, el primero es la gestión de proyectos y el segundo es la implementación de software. Éste estándar se enfoca en dar lineamientos para empresas pequeñas puesto que dichas empresas están empezando a surgir y no tienen una guía de apoyo para competir de manera equilibrada en el mercado con las empresas macro.

Para complementar la aplicación del marco de trabajo, se implementó la metodología ICONIX, modelo de trabajo útil en la orientación de procesos de implementación de software que permitió una mayor velocidad en el desarrollo del proyecto, propiciando resultados de alta calidad en un tiempo corto.

# Planteamiento del problema

Sanambiente S.A.S es una empresa especializada en ofrecer soluciones integrales en productos y/o servicios que aportan al cuidado y mejoramiento de las condiciones ambientales en un medio ambiente específico.

La estructura organizacional de la empresa se encuentra conformada brevemente de la siguiente manera: en el puesto más alto se encuentra la junta directiva y el gerente general, luego se encuentran las gerencias de ventas, estrategia, técnica y administrativa (ésta última posee las oficinas de tesorería, logística y contabilidad), seguidas cada una por su respectiva jefatura (ejemplo: la gerencia técnica posee jefe técnico, jefe de tecnología, coordinador de servicios integrales y coordinador de proyectos), y adicionalmente por un líder jurídico que poseen.

Los datos que maneja la empresa son de tipo ambiental, todos estos dependiendo del producto y/o servicio y la información específica que tiene como labor transferir de los datalogger de las estaciones concretas para brindar al cliente, entre los que se encuentran: datos de muestreo continuo de agua por diferentes metodologías, como son: muestreo continuo en intervalos de tiempo, muestreo continuo o simple y muestreo por eventos (caudal, calidad de agua, lluvia) ; medidas en forma instantánea de la humedad del suelo que se especifique, conductividad eléctrica/salinidad y temperatura; muestreo de aguas superficiales, agua potable, aguas residuales y no residuales con toma de muestra puntual, compuesta e integrada en cuerpo lótico (ecosistema de aguas fluyentes) y medición en sitio de caudal, pH, temperatura, conductividad, oxígeno disuelto; mediciones continuas de partículas PM10 o PM2.5 o PM10-2.5; medición del ruido ambiental (ruido que existe en un determinado lugar y en un momento dado); medición de la calidad del aire, particularmente, la medición de CO (monóxido de carbono) y O3 (ozono), pero éstas se llevan a cabo en un área delimitada, dependiendo en donde se encuentre ubicado el producto y/o servicio encargado de realizar esta labor; medición y transmisión de información meteorológica, nivel o hidrometeorológica gracias a la implementación de comunicación satelital GOES.

La empresa Sanambiente maneja actualmente un sistema para realizar la transferencia de datos ambientales de los datalogger de las estaciones gracias a los protocolos de comunicación que utilizan, FTP y Modbus.

Para realizar la transferencia de datos por el protocolo Modbus, el sistema manda una ID, el puerto, la IP y el mensaje solicitando los datos ambientales, de igual forma la estación donde se encuentra el datalogger responde con un mensaje al sistema, con los datos ambientales requeridos, luego, lo que hace el sistema es capturarlos y almacenarlos en la base de datos. Para realizar la transferencia de datos por el protocolo FTP, es un poco diferente, ya que inicialmente se debe configurar un pequeño servidor FTP que generalmente se configura con el Internet Information Services propio de Windows, en el cual se encontrarán almacenados en una carpeta los datos ambientales transferidos del datalogger de la estación específica, luego, con otra aplicación llamada FTP Import/Export se leen estos datos de la carpeta, y la forma en que esta aplicación realiza la lectura es de acuerdo a una plantilla ya creada que está estructurada por canales (ej. SO2, NH3), luego de leerlos la aplicación procede a realizar el almacenamientos de los datos en la base de datos.

El inconveniente que posee la empresa Sanambiente está relacionado con la desarticulación de aplicativos, que requiere intervención manual de los desarrolladores para que la comunicación y transferencia de los datos ambientales se lleve a cabo sin problemas.

Para dar solución a este inconveniente se generará un aplicativo que integre los diferentes procesos de la gestión de datos. Para ello se han propuesto dos proyectos de grado que abordan diferentes perspectivas: calidad y desarrollo, pero se requiere un marco de trabajo que permita integrarlas correctamente.

Tabla 1. Diagnóstico y pronóstico del proyecto.

|  |  |  |  |
| --- | --- | --- | --- |
| **Síntoma** | **Causa** | **Efecto** | **Consecuencia** |
| Los equipos de calidad y desarrollo abordan el proyecto por separado. | No hay integración de los equipos. | Esto conlleva que el proyecto pierda la orientación. | Productos pequeños sin conexión. |
| Falta de comunicación de los equipos: calidad y desarrollo. | Falta de coordinación con los integrantes de los equipos. | Que cada equipo realice el proyecto por su cuenta. | Productos incompatibles que dificultan su integración necesaria para su correcto funcionamiento. |
| **Diagnóstico** | | **Pronóstico** | |

*Nota:* Elaboración propia.

Figura 1. Diagrama Causa y efecto.



*Nota:* Elaboración propia.

# Objetivos

## Objetivo general

Establecer un marco de trabajo siguiendo los lineamientos dados por la ISO/IEC 29110 utilizando como caso de estudio la integración de procesos del proyecto Sanambiente.

## Objetivos específicos

* Identificar los procesos y medidas propuestas por la ISO/IEC 29110 que se adecúen a las características del gestor de datos ambientales de la empresa Sanambiente.
* Establecer la ruta de procesos y herramientas que permitan la integración de los equipos del proyecto de Sanambiente de acuerdo a la ISO/IEC 29110.
* Aplicar el marco de trabajo identificado en la ISO/IEC 29110 al proyecto Sanambiente.
* Evaluar los resultados de la aplicación del marco de trabajo de acuerdo a la ISO/IEC 29110.

# Problema de investigación

## Formulación del problema

¿Cómo integrar los equipos del proyecto de gestión de datos de Sanambiente para que cumplan con criterios de calidad y robustez hacia el futuro?

## Sistematización del problema

* ¿Cuáles elementos de la ISO/IEC 29110 se pueden ajustar a las características del proyecto Sanambiente?
* ¿Cuáles son las herramientas propuestas por la ISO/IEC 29110 que permitan la integración de los diferentes equipos de trabajo del proyecto Sanambiente?
* ¿Cómo asegurar que el proyecto Sanambiente cumple adecuadamente un marco de calidad?
* ¿Cómo asegurar que el marco propuesto cumple los estándares seleccionados para enmarcar el proyecto?

## Justificación

(Hay que ponerla en tiempo presente. \*Aún no se corrige)

En el presente proyecto se realizó el aporte de un marco de trabajo para la integración de los equipos (calidad y desarrollo), el cual se llevó a cabo tomando como base la norma ISO/IEC 29110, ya que determina una serie de pasos a cumplir; adicional a esto, la investigación otorgó unos conocimientos para ser empleados en el futuro, en un área de trabajo que requiera la aplicación de un marco con unos lineamientos a seguir para realizar la integración efectiva de todos los equipos implicados en un proyecto. Además, la integración de los equipos conllevó a tener una mejor comunicación en el proceso de la realización del aplicativo obteniendo avances significativos gracias a la participación de todos en conjunto, y dando como resultados el desarrollo de software, y un marco de trabajo para ser base de implementación en proyectos futuros.

El proyecto realizado está vinculado particularmente al programa académico de Ingeniería de sistemas, por lo cual se afirma que, se puso en práctica lo aprendido a lo largo del programa.

Este proyecto posee escalabilidad, esto quiere decir que seguirá en constante evolución a lo largo del tiempo, añadiendo nuevas funcionalidades y mejoras, sin degradar su desempeño. (justificar)

# Marco de referencia

## Antecedentes

Para el presente proyecto se analizaron diferentes proyectos que tuvieron el mismo o similar objeto de estudio que el presente proyecto tiene, estos fueron tomados como base para llevar a cabo el desarrollo de este proyecto, estos proyectos brindan una guía para saber cómo proceder y de qué forma iniciar la elaboración del presente proyecto, los pasos a seguir, cuáles pueden ser de utilidad para el presente proyecto y poderlos adecuar de acuerdo a las necesidades que éste tiene determinadas. La información que será mostrada en este marco es de aquellos proyectos que tienen relación con los temas que el presente proyecto aborda: ISO/IEC 29110 y Metodología ICONIX. El tipo de revisión que se hizo a estos proyectos es meramente enfocado en la parte donde tocan los temas mencionados anteriormente, qué uso hicieron de estos temas en sus proyectos, cómo lo ejecutaron y qué resultados les dieron. Para la realización de este marco se hizo el análisis y estudio de cinco documentos, tesis, concretamente; pero entre ellos, cuatro fueron los que más tenían mejor dominio del tema y mostraban de forma clara la aplicación de estos temas al detalle.

Según (Madruñero Padilla, 2018) cuyo proyecto denominado “Implementación del estándar ISO/IEC 29110 en el proceso de desarrollo de software de la dirección de desarrollo tecnológico e informático de la universidad técnica del norte” este trabajo se manejó la ISO/IEC 29110 donde se llevó a cabo la gestión de procesos de desarrollo de software en la universidad técnica del norte, de la ciudad Ibarra-Ecuador, donde se buscaba la formalidad en todas las etapas del proceso de desarrollo mejorando la gestión del desarrollo permitiendo incorporar hitos de control en el proceso y su gestión para lograr una toma de decisiones oportuna en el proyecto.

Como resultado que se obtuvo en esa investigación fue la implementación del estándar ISO/IEC 29110 ya que pudo establecer una relación con la metodología SCRUM en la cual con llevo aportar a la institución una metodología para la elaboración del desarrollo de software.

En la tesis (Laporte et al., 2016) bajo el título general de “ THE IMPLEMENTATION OF ISO/IEC 29110 SOFTWARE ENGINEERING STANDARDS AND GUIDES IN VERY SMALL ENTITIES” se demuestra que el ISO/IEC 29110 tiene perfiles de ciclo de vida para empresas muy pequeñas (de desarrollo de software) que deben seguir para que puedan elaborar software con altos estándares de calidad, así mismo, este estándar permite que las empresas pequeñas puedan hacer gestión a sus proyectos que permite tener un seguimiento a los proceso que se realizan, dichos procesos ayudan al mejoramiento y el rendimiento de la empresa.

El resultado que se puede obtener por este proyecto es que las empresas de desarrollo de software pueden hacer software que cumplan con los estándares de calidad ya que la ISO/IEC 29110 ayuda a las pequeñas empresas poder desarrollar software de calidad, como resultado de la implementación de esta ISO a cualquier empresa pequeña contribuye a ser competitiva en el mercado nacional o internacional.

Según la tesis de (LEGARIA, 2018) con el título de “IMPLEMENTACION DE PROCESO ORGANIZACIONAL DE GESTION DE PROYECTOS EN DEVELOPIT” donde nos habla de cómo la ISO/IEC 29110 cumple con los proceso de la gestión de proyecto, DevelopIT como empresa necesita una norma que se adecue a la compañía para eso se utiliza la ISO/IEC 29110 ya que tiene menos procesos respecto a la gestión de proyectos, estos procesos que tiene la ISO/IEC 29110 se adaptara para los proyectos de infraestructura tecnológica que implementa DevelopIT esto permita a la organización pueda ejecutar proyectos considerados complejos y no complejos.

Se puede concluir que, para implementar un proyecto de gestión de proyectos en una pequeña empresa es necesario que esta sea adaptada a las necesidades de la organización, En ese sentido la norma ISO/IEC 29110 cumple con las características para ser adaptada a pequeñas organizaciones. También es posible que otras organizaciones con características similares a DevelopIT puedan hacer uso de este proceso de gestión de proyectos, ya que, al basarse en una norma, y al diseñarse con la intención de adaptarse a las características de pequeñas organizaciones, debiera ser por lo tanto útil y usable por organizaciones con similares características, es decir, pequeñas organizaciones, que no tienen personal especializado en gestión de proyectos, sus recursos son escasos y que los roles de los integrantes son multitarea.

De acuerdo (PORRAS, 2019) en el estudio “METODOLOGÍA ÁGIL ICONIX EN LA CALIDAD DEL PRODUCTO SOFTWARE, LIMA, 2017” tiene como objetivo de que la metodología ágil ICONIX brinda proceso para el desarrollo de software mediante la aplicación de técnicas probadas y aceptadas de ingeniería de software para empresas micro, pequeñas y medianas esto permitirá que este tipo de empresas tenga un mejoramiento de desarrollo de software, además puede tener productos aceptables y sostenible.

El resultado que se obtuvo de este proyecto fue la implementación de la metodología ICONIX que ayudo mejorar la calidad del desarrollo del software que se estaba buscando, gracias a esta metodología que brinda pasos a seguir donde se enfoca en cómo se debe desarrollar un sistema, Este proyecto ayudara a la empresa que brinda el servicio de desarrollo de software nuevas formas de hacer aplicaciones con agilidad que tiene beneficios como: disminuir el tiempo y los costos.

## Marco conceptual

### Proceso de software

Es un marco de trabajo de las tareas que se necesitan para construir software de alta calidad. Un proceso de software establece la perspectiva que se debe elegir cuando el software es manipulado por la ingeniería (Pressman, 2002, p. 13). Es el conjunto de actividades que se relacionan y enfocan en la creación de un producto de software. Este software elaborado puede ser un producto nuevo, la modificación o una software ya existente (Ramos & Mendoza, 2014, p. 13).

### **Metodología de desarrollo de software**

Es aquella que posee la función de realizar la organización, planificación y el control del proceso de desarrollo de software (Hugues et al., 2010).

Una metodología de desarrollo de software especifica cómo hay que dividir el proyecto por etapas o fases, las tareas que se deben llevar a cabo en cada una, las salidas que se deben producir y cuando producir, las herramientas que hay que emplear, las restricciones apropiadas que se deben aplicar, y la manera en que se debe administrar y llevar el control pertinente sobre un proyecto (Cataldi, 2000, p. 40).

### **Lenguaje Unificado de Modelado (Unified Modeling Language (UML))**

El Lenguaje Unificado de Modelado, es un lenguaje estándar de modelado para productos de software, que proporciona la visualización, definición, elaboración y documentación de aquellos artefactos de sistemas en los que un producto de software posee una labor o función significativa. En esencia, UML permite a los desarrolladores ver los logros de su trabajo representados en esquemas o diagramas definidos (Jacobson et al., 2000, p. 407). Este lenguaje fue creado por un grupo de personas con amplio conocimiento en la ingeniería de software: Ivar Jacobson, Grady Booch y James Rumbaugh en el año 1995 (González Toste et al., 2017, p. 7).

Como UML posee su enfoque en la descripción de métodos y procesos, se facilita la definición de las interacciones que deben llevarse a cabo por parte de los programadores y los usuarios con el software que se está desarrollando, a través de diferentes tipos de diagramas. Estos se encuentran categorizados de la siguiente manera (ver tabla 2) (Matla Cruz, 2014, pp. 24-26):

Tabla 2. Diagramas de UML por categoría.

|  |  |  |
| --- | --- | --- |
| **De estructura** | **De comportamiento** | **De interacción** |
| Diagrama de clases | Diagrama de estado | Diagrama de secuencia |
| Diagrama de objetos | Casos de uso | Diagrama de colaboración |
| Diagrama de componentes |  |  |
| Diagrama de paquetes |  |  |

*Nota:* Elaboración propia con base en Matla Cruz, E. O. (2014). *Desarrollo de software guiado por la norma ISO/IEC 29110 y Scrum: SIDEP v.2.0*. Universidad Nacional Autónoma de México, Ciudad de México.

### **Calidad de software**

El concepto de calidad planteado en la norma (UNE, 1995), expresa que “la calidad es el conjunto de propiedades y características de un producto o servicio que le confieren su aptitud para satisfacer unas necesidades explicitas o implícitas”. En el campo de la ingeniería de software, de acuerdo a la (IEEE, 1991), la calidad de software es el nivel con el que un sistema o proceso acata correctamente con los requerimientos definidos y satisface con certeza las necesidades que el cliente expresó.

### **Modelo de calidad de software**

Según (Scalone, 2006), los modelos de calidad son documentaciones que: abarcan en gran medida las mejores prácticas para dar soporte a los procesos que conlleva un producto de software, ofrecen apoyo en temas de gestión en los que cada organización hay que tener presentes y, permiten medir los avances del producto en materia de calidad.

## Marco teórico

### Revisión de área

El objetivo de este subcapítulo es dar a conocer las principales tendencias en modelos y estándares que en este momento están siendo utilizadas para la gestión de los proyectos de software en las organizaciones.

La “crisis del software” del año 1969 perdura hoy en día, llegando al punto en que aún están presentes los problemas asociados al fracaso de los proyectos. El Standish Group y Dove confirman que el fracaso de un proyecto es por la carencia de gestión en este (Calvo Manzano & García, 2008, p. 8).

La gestión de proyectos ha tenido un alto crecimiento en las últimas dos décadas, debido a la necesidad de muchas organizaciones de mejorar sus procesos, y tanto las herramientas como técnicas se están estableciendo cada vez más como aplicaciones estándares para la gestión de los proyectos, no sólo de software, sino también de las otras áreas (Roberts & Wallace, 2002, p. 43).

Es tanto el crecimiento, que las organizaciones líderes en modelos y estándares para la gestión, han tenido que ir creando o mejorando estos para su aplicación en los proyectos de alta complejidad de la actualidad. A continuación, se muestran algunos de los principales modelos y estándares que, a día de hoy, son los más utilizados por las organizaciones para la gestión de sus proyectos (ver tabla 3).

Tabla 3. Estándares para la gestión de proyectos.

|  |  |  |  |
| --- | --- | --- | --- |
| **Modelo/Estándar** | **Organización** | **Primera publicación** | **Última publicación** |
| PMBOK | PMI | 1996 | 6ta. Edición, 09/2017 |
| APMBOK | APM | 1992 | 7ta. Edición, 05/2019 |
| ISO 10006 | ISO | 12/1997 | 3ra. Edición, 11/2017 |
| ISO 12207 | ISO | 1995 | 3ra. Edición, 11/2017 |
| ISO 21500 | ISO | 09/2012 | 1ra. Edición, 09/2012 |
| PRINCE2 | OGC | 1996 | 5ta. Edición, 06/2009 |
| CMMI-DEV | CMMI Institute | 2002 | Versión 2.0, 2018 |

*Nota:* Elaboración propia con información recopilada de: Estrada Reyes, J. N. (2015). Análisis de la gestión de proyectos a nivel mundial. *Palermo Business Review*, *12*, 38**,** ISO. (s. f.). *International Organization for Standardization* **y** OGC. (2009). *Éxito en la gestión de proyectos con PRINCE2*. The Stationery Office.

El uso de estos modelos y estándares cada vez está cobrando más fuerza en las organizaciones, dado que al aplicarlos en sus proyectos, están proporcionando consigo beneficios de mejora continua y satisfacción al cliente (Lugo García & Marín Sánchez, 2016).

La gestión de proyectos como toda disciplina está en constante evolución, debido a que el mercado de software así lo está demandando. Los proyectos requieren más tiempo por su complejidad, por sus grandes volúmenes de actividades y roles; y los modelos o estándares de gestión deben estar a la altura brindando las herramientas y elementos adecuados para encargarse de ellos de la mejor manera.

Cabe mencionar que esta disciplina no está completamente estandarizada, dicho de otro modo, no dispone de una única manera para ser aplicada, debido a la existencia de diferentes puntos de vista (PMI, ISO, CMMI Institute, etc.) (Vásquez González, 2007, p. 12). No obstante, el auge que ha tenido la gestión de proyectos a lo largo del tiempo es alto, y ha logrado que se internacionalice y posicione hoy en día como uno de los líderes del camino hacia el desarrollo de prácticas profesionales (Roberts & Wallace, 2002, p. 44).

Para terminar, sería propicio conocer un poco acerca de la guía más conocida y usada por las organizaciones para la gestión de sus proyectos: PMBOK. Por lo tanto, a continuación, se dará una breve explicación de esta.

#### 4.3.1.1 PMBOK

Es una guía de apoyo que proporciona los pasos necesarios para realizar proyectos de forma exitosa y explica a detalle aquellas variables que deben ser tenidas en cuenta para la ejecución de un proyecto (Estrada Reyes, 2015, p. 80).

PMBOK define formalmente un total de 5 grupos de procesos los cuales son: Inicio, Planeación; Ejecución; Monitoreo y Control; y Cierre.

Estos grupos son la base de partida para comprender la gestión de proyectos, cada uno de estos posee los pasos necesarios para que sean aplicados correctamente, de manera ordenada y con la probabilidad de que aumente el éxito (Estrada Reyes, 2015, p. 81).

### Comparación entre estándares para la gestión de proyectos

Se llevó a cabo la comparación entre dos estándares para la gestión de proyectos, con el fin de determinar cuál de estos era el ideal para ser aplicado en este caso.

Existe un amplio número de estándares que se pueden aplicar para la gestión de un proyecto, por lo cual, es muy difícil seleccionar uno de ellos; no obstante, sería propicio determinar cuáles de estos estándares pueden asegurar que el proyecto dé resultados con alta probabilidad de éxito.

Con el fin de seleccionar qué estándar era el adecuado para aplicar en este proyecto, se hizo la comparación entre la guía PMBOK y el estándar ISO/IEC 29110. En la tabla 4, se compararon estos dos estándares en cuanto a las características que cada uno posee.

Tabla 4. Comparación entre estándares para la gestión de proyectos.

|  |  |
| --- | --- |
| **ISO/IEC 29110** | **PMBOK** |
| Guía para los procesos de Gestión de proyectos e Implementación de software. | Guía para el proceso de Gestión de proyectos. |
| Las fases del proceso de gestión de proyectos son: Planificación del proyecto; ejecución del Plan del proyecto; evaluación y control del proyecto; y cierre del proyecto. | Las fases del proceso de gestión de proyectos son: Inicio, Planificación; Ejecución; Monitorización y control; y Cierre. |
| Estándar dirigido a empresas pequeñas. | Estándar dirigido a empresas de cualquier tamaño interesadas en mejorar la gestión de sus proyectos. |
| No define áreas de conocimientos para las fases. | Defina áreas de conocimientos y esta las asocia a las fases. |
| Es impulsada por los requisitos del cliente. | Es impulsada por los requisitos del cliente. |
| Está dirigida a empresas de desarrollo de software. | Está dirigida a empresas de cualquier área. |
| No es muy amplia la documentación que proporciona, y la que se encuentra, es destinada para el área de desarrollo de software. | Como apoyo, tiene una amplia documentación general de proyectos de distintas áreas. |
| **ISO/IEC 29110** | **PMBOK** |
| Posibilidad de certificación ISO/IEC 29110. | Tiene dos niveles de certificación: PMP (Project Management Professional) y CAMP (Certificate Associate in Project Mangement). |
| Estándar creado por la ISO. | Estándar creado por el PMI. |

*Nota:* Elaboración propia con información recopilada de: Assaff, R. (s. f.). *PMBOK - El cuerpo de conocimientos de la gestión de proyectos*. 73-82**,** Fernández Parra, K., Garrido Saroza, A., Ramírez Martínez, Y., & Perdomo Bello, I. (s. f.). PMBOK y PRINCE 2 similitudes y diferencias. *Revista Científica*, *23*, 117-130. https://doi.org/10.14483/udistrital.jour.RC.2015.23.a9 **y** Puerta Marín, H. A., & Espinosa Bedoya, A. (2015). UNA ADAPTACIÓN DE PMBOK AL CICLO DE VIDA DE DESARROLLO DE PROYECTOS SOFTWARE EN PEQUEÑAS EMPRESAS. *VI Congreso Iberoamericano de Ingeniería de Proyectos - CIIP*.

Luego de haber realizado la comparación entre los dos estándares, se decidió aplicar la ISO/IEC 29110 en este caso, debido a que su enfoque son los proyectos de desarrollo de software; brinda todas las herramientas necesarias para llevar a cabo el seguimiento y control adecuado de todos los recursos que intervienen en estos; y está dirigida a las empresas pequeñas, como lo es Sanambiente.

### **Estándar ISO/IEC 29110.**

Es una norma creada por la ISO bajo el título *Ingeniería de Software – Perfiles del ciclo de vida para entidades muy pequeñas* (Very Small Enterprises (VSEs)) y se basa en subconjuntos de elementos normativos apropiados, conocidos como perfiles VSE. El propósito de los perfiles VSE es definir un subconjunto de normas internacionales para el contexto de VSE (ISO/IEC, 2011, p. vi).

La norma se desarrolló para mejorar la calidad del software ayudando en el rendimiento de los productos en pymes, donde puedan contar con los mismos niveles de competitividad en mercado de las grandes industrias. La ISO/IEC 29110 se encuentra divido en 5 partes de acuerdo a un público objetivo (ver tabla 5):

Tabla 5. Público objetivo de la ISO/IEC 29110.

|  |  |  |
| --- | --- | --- |
| **ISO/IEC 29110** | **Título** | **Público objetivo** |
| Parte 1 | Panorama General | VSEs, Evaluadores, productores de estándares, vendedores de herramientas y vendedores de metodología |
| Parte 2 | Marco y Taxonomía | Productores de normas, proveedores de herramientas de metodología. No está diseñado para VSEs. |
| Parte 3 | Guía de evaluación | Asesores y VSEs |
| Parte 4 | Especificaciones del perfil | Productores de normas, proveedores de herramientas de metodología. No está diseñado para VSEs. |
| Parte 5 | Guía de gestión e ingeniería | VSEs |

*Nota:* ISO/IEC. (2011). *Software engineering - Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*, p.vi.

El estándar define los procesos de gestión de proyectos y desarrollo de software (los cuales están interrelacionados), y las prácticas que integran están basadas en la selección de la ISO/IEC 12207:2008, bajo el título *Sistemas e ingeniería de software – Procesos del ciclo de vida del software* e ISO/IEC 15289:2006, bajo el título *Sistemas e ingeniería de software – Contenido de los sistemas y productos de información del proceso del ciclo de vida del software (Documentación),* presentando los productos y documentos requeridos en cada etapa (ver figura 2) (ISO/IEC, 2011, p. 3).

Figura 2. Procesos básicos de la guía de perfiles.

Declaración de trabajo

**Proceso:** Gestión del proyecto

**Proceso:** Implementación del

Software

Configuración de Software

*Nota:* ISO/IEC. (2011). *Software engineering - Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*, p.4.

A continuación, se realizará una breve descripción de cada uno de los procesos y las actividades que conllevan.

* **Proceso de gestión del proyecto (PM):**

Tiene como propósito administrar de manera jerárquica las tareas del proyecto de implementación de software, permitiendo cumplir con los objetivos del proyecto en la calidad, tiempo y costos esperados. Para este proceso se tienen las siguientes actividades: planificación del proyecto, ejecución del proyecto, evaluación y control del proyecto y por último cierre del proyecto.

El siguiente diagrama muestra de manera clara el flujo entre las actividades del proceso de gestión del proyecto, incluyendo sus productos más importantes y su relación (ver figura 3).

Figura 3. Diagrama del proceso de gestión del proyecto.



*Nota:* ISO/IEC. (2011). *Software engineering - Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*, p.9.

* **Proceso de implementación de software (SI):**

Tiene como propósito la realización sistemática de las actividades de análisis, construcción, integración y pruebas de software nuevos o modificados según los requisitos (ISO/IEC, 2011, p. 16). Para este proceso se tienen las siguientes actividades: inicio de la implementación del software, análisis de los requerimientos de software, diseño arquitectónico y detallado de programas informáticos, construcción de software, integración de software y pruebas, y por último la entrega de productos.

El siguiente diagrama muestra de manera clara el flujo entre las actividades del proceso de implementación de software, incluyendo sus productos más importantes y su relación (ver figura 4).

Figura 4. Diagrama del proceso de implementación de software.



*Nota:* ISO/IEC. (2011). *Software engineering - Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*.

* **Productos o artefactos:**

La ISO/IEC 29110 define unos productos o artefactos obligatorios para su implementación. En la siguiente tabla se listan junto con sus descripciones y el proceso de donde provienen (ver tabla 6).

Tabla 6. Productos o artefactos según la ISO/IEC 29110.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Descripción** | **Proceso de donde proviene** |
| 1. | Acceptance Record (Registro de aceptación) | Documenta la aceptación del cliente de los entregables del proyecto. | Gestión del proyecto. |
| 2. | Change Request (Solicitud de cambio) | Cuando se requiere una modificación en partes de algún producto, se diligencia una de estas para solicitarla. | Implementación de software.  Cliente.  Gestión del proyecto. |
| 3. | Correction Register  (Registro de corrección) | Identifica las actividades establecidas para corregir una desviación o problema relacionado con la realización de un producto. | Gestión del proyecto. |
| 4. | Meeting Record (Acta de reunión) | Registro de los acuerdos con el equipo de trabajo del proyecto y con el cliente. | Gestión del proyecto. |
| 5. | Progress Status Record (Registro de estado de progreso) | Registro del avance en el proyecto con respecto al Plan del proyecto. | Gestión del proyecto. |
| 6. | Project Plan (Plan del Proyecto) | Presenta la planeación de todas las actividades del proyecto que se van a realizar con sus fechas. Esto asegura la finalización exitosa del proyecto. | Gestión del proyecto. |
| 7. | Project Repository (Repositorio del proyecto) | Almacenamiento de los productos y sus respectivas entregas del proyecto. | Gestión del proyecto. |
| 8. | Project Repository Backup (Copia de seguridad del repositorio del proyecto) | Repositorio de respaldo del repositorio principal, como medida de contingencia para poder realizar la recuperación de la información. | Gestión del proyecto. |
| **#** | **Nombre** | **Descripción** | **Proceso de donde proviene** |
| 9. | Software Configuration (Configuración del software) | Conjunto de productos de software identificados. Estos son:   * Especificación de requerimientos * Diseño de software * Registro de trazabilidad * Componentes de software * Software * Casos de prueba y procedimientos de prueba * Reporte de pruebas * Manual de operación * Manual de usuario * Manual de mantenimiento | Implementación de software. |
| 10. | Verification Results (Resultados de la verificación) | Documenta la ejecución de la verificación del plan del proyecto. | Gestión del proyecto.  Implementación de software. |
| 11. | Validation Results (Resultados de la validación) | Documento la ejecución de la validación. | Implementación de software. |

*Nota:* ISO/IEC. (2011). *Software engineering - Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*, p.33-40.

Y finalmente, la ISO/IEC 29110 define unos roles para el desarrollo del proyecto. Estos son: analista, cliente, diseñador, programador, gestor del proyecto, líder técnico y equipo de trabajo.

### Comparación entre metodologías de desarrollo de software

Se llevó a cabo la comparación entre metodologías de desarrollo de software, con el fin de determinar cuál de estas era la ideal para ser aplicada en este caso.

Existe un amplio número de metodologías que se pueden aplicar para el desarrollo de software, por lo cual, es muy difícil seleccionar una de ellas; no obstante, sería propicio determinar cuáles de estas metodologías pueden contribuir al mejoramiento de la productividad en el desarrollo y la calidad de un producto de software.

Con el fin de seleccionar qué metodología era la adecuada para aplicar en este proyecto, se hizo la comparación entre Scrum, XP e Iconix. En la tabla 7, se compararon estas metodologías en cuanto a las características que cada una posee.

Tabla 7. Comparación entre metodologías de desarrollo de software.

|  |  |  |
| --- | --- | --- |
| **Scrum** | **XP** | **Iconix** |
| * Scrum es una metodología ágil que brinda una gestión de proyecto y donde su enfoque es iterativo e incremental. * Scrum emplea en su proceso tres elementos: * Planificación del blacklog. * Seguimiento de Sprint * Revisión del Sprint. * El ciclo de vida de este marco de trabajo está compuesto de cuatro fases: planeación, puesta en escena, desarrollo y entrega. | * XP también es una metodología ligera que se basa en la simplicidad. * Esta metodología está fundamentada en pruebas y error. * Los requisitos pueden cambiar a lo largo del desarrollo. * Los grupos de trabajo por lo general pueden estar conformados de 2 -12 personas, pero lo más principal es de 2 personas. | * Iconix es una pesada-ligera donde su proceso es simplificado a comparación con otras metodologías y además es iterativo e incremental. * Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo. * Iconix contiene las siguientes bases: * Revisión del diseño. * Análisis y diseño preliminar. * Implementación y pruebas. |

*Nota:* Elaboración propia con información recopilada de: Alva Salcedo, A. B., & Reyes Laynes, J. B. (2019). *Desarrollo e implementación de un sistema de ventasa basado en la metodologia scrum y xp el proceso de ventas de servicio de la empresa emsoir*. Universidad Autónoma del Perú, Amavizca Valdez, L. O., Garcia Ruiz, A. C., Jiménez López, E., Duarte Guerrero, G. L., & Vázquez Brindis, J. C. (2014). *Aplicación de la metodología semi-ágil ICONIX para el desarrollo de software: Implementación y publicación de un sitio WEB para una empresa SPIN - OFF en el Sur de Sonora, México.* Universidad Tecnológica del Sur de Sonora. Lozano Argote, J., Bolivar Guzman, J., & Ramírez, J. J. (2019). *Desarrollo de los módulos de transferencia y comunicación de datos ambientales para la empresa de SANAMBIENTE de Cali*. Instituto Universitario Antonio Jose Camacho, Navarro Cadavid, A., Fernández Martínez, J. D., & Morales Vélez, J. (2013, septiembre 20). *Revisión de metodologías ágiles para el desarollo de software*. 11.

Luego de realizar la comparación, se seleccionó la metodología Iconix, debido a que las fases proporcionaban los elementos necesarios para la implementación del software propuesto por la empresa Sanambiente. La ISO/IEC 29110 contaba con una guía para la Implementación de software, pero los elementos que proporcionaba eran de difícil comprensión para ser aplicados y seguidos en este proyecto, por ende, se tomó la decisión de no emplearse.

### **Metodología Iconix.**

Metodología de desarrollo de software de proporción media, en la que el análisis y capacidad de su diseño se basa en UML (Unified Modeling Language) (Mnkandla & Dwolatzky, 2004, p. 1). Fue elaborada por Doug Rosenberg y Kendall Scott. Se dice que ICONIX se encuentra entre la complejidad de RUP (Rational Unified Processes) y la simplicidad de XP (eXtreme Programming), sin pasar por alto las tareas de análisis y de diseño que éste último no contempla (Amavizca Valdez et al., 2014, p. 3).

La esencia de ICONIX está en que un 80% de los casos pueden ser resueltos tan solo con un uso del 20% del UML, lo cual simplifica en gran medida el proceso, al dejar solo aquella documentación necesaria (ICONIX Brand Group, 2016, p. 1).

El proceso de ICONIX (ver figura 5) da facilidad en el paso de los casos de uso a la codificación de forma fiable, en el menor tiempo posible. Razón por la cual, la principal preocupación de ICONIX es el análisis y diseño por modelado de la producción de software. (Rosenberg et al., 2005, p. 41).

Figura 5. Procesos de ICONIX.



*Nota:* Rosenberg, D., Stephens, M., & Collins-Cope, M. (2005). *Agile Development with ICONIX Process*. United States of America: Apress, p.45.

* **Orígenes** (Rosenberg et al., 2005, p. 40):

El proceso de ICONIX se originó años antes de UML y el proceso unificado, como una agrupación que cuenta con las mejores técnicas de las metodologías originales que formaron UML: la Técnica de modelado de objetos (Object Modeling Technique (OMT)) de Jim Rumbaugh, el método Objectory de Ivar Jacobson y el método Booch de Grady Booch; debido a que las fortalezas y debilidades que estas metodologías poseían se complementaban entre sí.

* **Principales diagramas que maneja Iconix**

Como se mencionaba con anterioridad, ICONIX basa su análisis y diseño en UML, por lo tanto, hace uso de un conjunto de diagramas como apoyo para proveer un resultado óptimo. Estos diagramas son:

* + **Modelado de dominio:** es un glosario de proyecto o un diccionario de términos que se utilizan en el proyecto. Su principal función es asegurar que todos en el proyecto entiendan el problema que están enfrentando sin que haya equivocación (Rosenberg et al., 2005, p. 7).
  + **Diagrama de caso de uso:** describe la secuencia de acciones que un actor realiza dentro del sistema (Rosenberg et al., 2005, p. 42). Guía todo el proceso de la metodología ICONIX. Según (Bona, 2002, p. 68) es el centro conceptual del desarrollo.
  + **Diagrama de robustez:** es el encargado de identificar el primer conjunto de posibles objetos de primera estimación de cada caso de uso, todo logrado gracias al análisis del texto de estos. El diagrama se divide en tres estereotipos:
    - **Objetos de límite:** usados por los actores para comunicarse con el sistema.
    - **Objetos de entidad:** son objetos del modelado de dominio.
    - **Objetos de control:** son encargados de la integración entre los dos primeros objetos.
  + **Diagrama de secuencia:** diagrama que muestra todos los cursos posibles que los casos de uso pueden elegir y además muestra tanto los métodos pertenecientes a cada objeto como los mensajes que se producen al realizar una acción determinada al sistema.
* **Características de Iconix:**

Se destacan tres características fundamentales de ICONIX:

* + **Iterativo e incremental:** cuando se lleva a cabo el desarrollo del modelado de dominio y se precisan los casos de uso, se producen múltiples iteraciones. En el ciclo incremental se desarrolla el producto por partes para que se puedan integrar funcionalmente sin complicaciones y en cada iteración se procesa a su revisión y mejora (Fiestas Jacinto, 2015, p. 40) (Silva Ascuntar & García G., 2018, p. 34).
  + **Trazabilidad:** Cada paso culminado está ligado a un requerimiento. Su enfoque principal son las necesidades del usuario, no hay un momento en que durante el proceso se desvíe lejos de este (Fiestas Jacinto, 2015, p. 40).
  + **Aerodinámica del UML:** ofrece un uso flexible del UML, debido a que no exige la utilización de todos los diagramas que tiene ligados (Silva Ascuntar & García G., 2018, p. 34).

En (Rosenberg & Scott, 2001, p. 23) se presentan las fases principales de la metodología ICONIX. Estas fases cuentan con unos hitos específicos asociados, como se va a mostrar a continuación.

* **Análisis de requerimientos:**

Las actividades que se llevan a cabo en esta fase son (Rosenberg & Scott, 2001, p. 23) (Fiestas Jacinto, 2015, p. 46):

* Identificar en el “mundo real”, los objetos y todas las relaciones de agregación y generalización que tienen entre sí. En esta actividad se hace uso del **modelado de dominio**.
* Como sugerencia, crear un **prototipado** de la interfaz del sistema o unos diagramas de navegación con los que el cliente esté más familiarizado del sistema que propusieron.
* Realizar el análisis de los requerimientos e identificar los casos de uso y sus actores involucrados. En esta actividad se hace uso de los **diagramas de caso de uso**.
* Se deben organizar los casos de uso en grupos y para realizarlo se hace uso del **diagrama de paquete**.
* Realizar la asociación de los requerimientos funcionales que estén relacionados que se hallaron en los casos de uso y objetos de dominio.

**Hito:** Revisión de los requisitos.

* **Análisis y diseño preliminar:**

Las actividades que se llevan a cabo en esta fase son (Rosenberg & Scott, 2001, p. 24):

* Se deben escribir los **casos de uso**, como el flujo principal de las acciones, con la capacidad de contener el flujo alternativo y el flujo de excepción.
* Se realiza el **diagrama de robustez**. Ya que para cada caso de uso se debe realizar la localización de un conjunto de objetos y llevar a cabo la actualización del modelado de dominio.
* Culminar en su totalidad la modificación del **diagrama de clases.**

**Hito:** Revisión del diseño preliminar.

* **Diseño detallado:**

Las actividades que se llevan a cabo en esta fase son (Rosenberg & Scott, 2001, p. 24) (Bona, 2002, p. 64):

* Detallar los cursos posibles de cada caso de uso a través del **diagrama de secuencia,** e identificar los mensajes entre los diferentes objetos.
* Culminar en su totalidad el modelo estático, ultimando detalles del proyecto en el **diagrama de clase**.
* Comprobar junto con el equipo si el proyecto satisface los requerimientos definidos al inicio.

**Hito:** Revisión crítica del diseño.

* **Implementación:**

Las actividades que se llevan a cabo en esta fase son (Rosenberg & Scott, 2001, p. 25):

* Se escribe escribe/genera el **código** del software.
* Se realizan las pruebas unitarias y de integración al software creado.
* Se llevan a cabo las pruebas de aceptación del usuario del proyecto finalizado.

**Hito:** Entrega.

### Reconocimiento de las tecnologías utilizadas por los equipos de trabajo

En este subcapítulo se hará una breve descripción de las principales tecnologías que fueron utilizadas por parte de los equipos de trabajo (gestión, desarrollo y calidad) para llevar a cabo sus actividades durante el desarrollo del macro proyecto propuesto por la empresa Sanambiente.

#### 4.3.6.1 Angular Framework

El objetivo principal de este framework es facilitar la creación de aplicaciones cliente en HTML y JavaScript o en el lenguaje TypeScript que se agrupa en JavaScript. Una aplicación consta de plantillas de componentes HTML, clases de componentes que se encargan de administrar estas, y además de la lógica que se encuentra en los servicios y componentes (Bolívar Guzmán et al., 2020, p. 16).

#### 4.3.6.2 JavaScript

Es un lenguaje de programación interpretado, orientado a objetos, basado en prototipos y además se caracteriza por ser imperativo, dinámico y tipado (Bolívar Guzmán et al., 2020, p. 16).

#### 4.3.6.3 TypeScript

Lenguaje de programación de código abierto, orientado a objetos, ideal para ser usado en proyectos a gran escala (Bolívar Guzmán et al., 2020, p. 17).

#### 4.3.6.4 PostgreSQL

Es un gesto de base de datos relacional libre, de código abierto. Dirigido a la comunidad de desarrolladores. Está bajo licencia Berkeley Software Distribution (BSD) (Mina Posu & Romero Dorado, 2020, p. 33) (Bolívar Guzmán et al., 2020, p. 16).

#### 4.3.6.5 NodeJS

Es un entorno multiplataforma, que se encuentra en tiempo de ejecución, igualmente de código abierto. Útil para la creación de aplicaciones web altamente escalables (Bolívar Guzmán et al., 2020, p. 18).

#### 4.3.6.6 GitHub

Plataforma de desarrollo colaborativo de aplicativos, ideal para el alojamiento de proyectos mediante el sistema de control de versiones que esta proporciona Git.

#### 4.3.6.7 GitHub Desktop

Aplicación de escritorio, con interfaz gráfica que permite el fácil mantenimiento de versiones de un proyecto. Básicamente fue diseñado para brindar facilidad al realizar control de versiones sobre los proyectos mediante una interfaz gráfica amigable. Su principal diferencia con el software Git, es que este último se usa mediante líneas de comando por consola.

#### 4.3.6.8 Visual Studio Code

Editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Entre sus características incluye control integrado de Git, depuración, finalización inteligente de código y resaltado de errores por sintaxis (Mina Posu & Romero Dorado, 2020, p. 34).

#### 4.3.6.9 Jest

Plataforma de pruebas universal, la cual se adapta a cualquier biblioteca de JavaScript. Esta plataforma se puede ejecutar sobre proyectos que estén basado en Angular, Node, TypeScript, Vue, entre otros (Mina Posu & Romero Dorado, 2020, p. 34).

#### 4.3.6.10 Jmeter

Es un proyecto de Apache, que puede ser utilizado como herramienta para pruebas de carga, para analizar el rendimiento de una variedad de servicios. Además, es utilizada para realizar pruebas unitarias sobre conexiones de bases de datos y conexiones TCP genéricas (Mina Posu & Romero Dorado, 2020, p. 35).

## Marco legal

Todas las actividades propuestas y desarrolladas dentro del proyecto quedan inmersas dentro del acuerdo de cooperación interinstitucional realizado entre la empresa Sanambiente y la Institución Universitaria Antonio José Camacho, firmado en el mes de abril de 2019.

Conforme al (Estatuto de Investigaciones y Producción Intelectual, 2018, p. 8), un proyecto de investigación se puede definir como *“la unidad básica de trabajo que comprende un conjunto de actividades organizadas coherentemente en una propuesta de investigación…”* y para poder llevar a cabo estas actividades y lograr unos resultados concretos, se debe hacer uso de recursos tanto humanos como financieros, bibliográficos y de infraestructura.

Además, las normas internas de la Institución Universitaria Antonio José Camacho reglamentan sobre la Propiedad Intelectual, los derechos sobre las diversas formas de creación de productos derivados del talento y actividad investigativa en la Institución y *“…reconoce, respeta y se subordina a lo establecido en la Constitución Política de Colombia, así como en la ley 23 de 1982, sobre derechos de Autor, Decreto 1360 de 1989 sobre la inscripción del soporte lógico (software) en el Registro Nacional de derecho de autor…”*. Además, cuando un proyecto de investigación es aprobado por parte de la Institución, se debe suscribir el Acta de Propiedad Intelectual de este.

# Resultados

A partir del estudio realizado en el marco teórico, se determinó que para el desarrollo de este caso se ejecutaría el proceso de Gestión del proyecto definido en el estándar ISO/IEC 29110, junto con sus fases y artefactos correspondientes. La razón por la que se eligió seguir este estándar en particular, es porque proporcionaba los pasos y artefactos necesarios para llevar a cabo el seguimiento y control adecuado de todos los recursos que intervenían en el proyecto.

Cada uno de los artefactos se elaboró de acuerdo al contexto del proyecto de la empresa Sanambiente, eliminando algunos ítems que por defecto el estándar tenía definidos para estos y adicionando otros de los que carecían. Además, se optó por no incluir el artefacto de Copia de seguridad del repositorio que se elabora en la Fase 2, ya que no se vio necesario tener un repositorio adicional como respaldo del repositorio principal; sin embargo, se realizan copias de seguridad del repositorio en el servicio de la nube de Google Drive un día a la semana, como medida de contingencia ante cualquier pérdida de información.

Aclarado todo esto, se procede a detallar cómo fueron implementadas cada una de las fases y artefactos del proceso de Gestión en el proyecto de la empresa Sanambiente.

## Fase 1: Planificación del proyecto

En esta fase se realizó la apropiada documentación en detalle de los elementos necesarios de planificación para llevar a cabo la gestión del proyecto. Se produjeron los siguientes artefactos:

### **Crear el plan del proyecto.**

Plan que fue socializado y aprobado por el cliente (empresa Sanambiente) para ser ejecutado sobre el proyecto. A continuación, se presenta este:

**Plan del proyecto de Sanambiente**

**Descripción del producto:**

Se elaborará un sistema que va a permitir transferir, procesar y comunicar los datos ambientales de la empresa Sanambiente, este sistema involucra el desarrollo de una arquitectura robusta, bajo diseño centrado en el usuario y riguroso seguimiento a estándares de calidad internacionales.

**Propósito:**

Se elaborará un sistema que va a permitir transferir, procesar y comunicar los datos ambientales de la empresa Sanambiente, este sistema involucra el desarrollo de una arquitectura robusta, bajo diseño centrado en el usuario y riguroso seguimiento a estándares de calidad internacionales.

**Requisitos generales del cliente**

**Comunicación y traspaso de información.**

1. El sistema debe permitir el enlace y transferencia de datos de los datalogger mediante Internet.

2. El sistema debe permitir crear, modificar, borrar, visualizar comandos para ser utilizados por los protocolos de comunicación X y Y.

3. El sistema debe permitir elegir qué tipo de protocolo de comunicación usará una estación, pero en una organización puede haber múltiples estaciones con diferentes protocolos y a su vez, que usen diferentes comandos.

4. El sistema debe permitir establecer los parámetros necesarios para garantizar la comunicación vía FTP como cliente o servidor.

5. El sistema debe permitir crear plantillas para leer los archivos planos y cargar los datos al sistema.

6. El sistema debe permitir usar las plantillas de textos planos de FTP para otras estaciones.

7. El sistema debe permitir acceder al FTP de acuerdo a la base de tiempo de las estaciones, leer los archivos de texto y almacenar la información en la base de datos.

8. El sistema debe permitir establecer los parámetros necesarios para garantizar la comunicación vía Modbus como cliente o servidor.

9. El sistema debe permitir configurar los comandos de modbus para acceder a los dataloggers y obtener la información de estos.

10. El sistema debe permitir acceder a las estaciones mediante Modbus de acuerdo a la base de tiempo creado y almacenar la información en la base de datos.

11. El sistema debe permitir usar las plantillas comandos de Modbus para otras estaciones.

12. El sistema debe permitir subir textos planos de manera manual cuando una estación no tenga acceso a Internet.

**Procesamiento**

13. El sistema debe permitir recibir o capturar los datos de las estaciones y almacenarlos en la base de datos teniendo en cuenta que los datos pasan por dos formas de almacenar, la primera son los datos crudos, como llegan de las estaciones, la segunda son los datos editados, los cuales ya pasaron por el procesamiento de datos.

14. El sistema debe permitir crear reglas o criterios de validación de la siguiente manera: rangos máximos, mínimos, valores de pico, valores constantes, valores negativos, % de variación, pérdida de datos parciales o totales.

15. Permitir correcciones, estados, fórmulas lineales, fórmulas extendidas, corrección de alertas en los datos históricos por parámetro ambiental de forma manual por un rango de tiempo.

16. Permitir correcciones, estados, fórmulas lineales, fórmulas extendidas, corrección de alertas, de manera automática en los datos nuevos.

17. El sistema debe permitir que el usuario pueda crear plantillas de procesamiento de información para los parámetros ambientales, con lo anterior se podrá ej. SO2 en valor alto 200 ppb, valor bajo 30 ppb, valor constante, valor pico de 5%, etc. Que pueden ser aplicados a X estaciones del mismo cliente. (O de otros clientes) Sin necesidad de volverlos a crear.

18. El sistema debe permitir bloquear por un usuario de permisos altos como (Editor final) intervalos de tiempo donde los datos hayan sido validados tanto automática como manualmente, con el fin que no puedan ser modificados alterando la información oficial.

19. El sistema de permitir agregar banderas de estados de validación o recepción de datos para acompañar el dato almacenado. Algunos de los estados son "Datos Válidos, Inválidos, rango alto, rango bajo, Sin datos, Apagado, etc."

**Configuración**

20. El sistema debe permitir la creación, modificación, eliminación y visualización de organizaciones.

21. El sistema debe permitir la creación, modificación, eliminación y visualización de estaciones

22. El sistema debe permitir la creación, modificación, eliminación y visualización de parámetros por estación

23. El sistema debe permitir asignación de parámetros a estaciones.

24. El sistema debe permitir asignación de estaciones a organizaciones.

25. El sistema debe permitir asignación de estaciones a usuarios.

26. El sistema debe permitir que una estación tenga diferentes bases de tiempo.

27. El sistema debe permitir exportar datos a csv, excel, txt, pdf de manera manual o automática.

28. El sistema debe permitir agregar, modificar y visualizar la latitud, longitud y altura para mostrar las estaciones en un mapa.

**Seguridad**

29. El sistema debe permitir crear, modificar, eliminar y visualizar usuarios del sistema

30. El sistema debe permitir asignar o revocar permisos y roles del usuario por organización y estaciones.

31. El sistema debe permitir mantener los siguientes roles visualizador: solo puede ver datos de las estaciones, operador: puede modificar datos de las estaciones (parámetros, nombres, etc), editor preliminar: puede realizar validación y corrección de datos. Editor final puede realizar la validación y corrección de datos y deshacer ediciones del preliminar o cerrar datos para evitar modificación.

32. El sistema debe permitir que el administrador elija si desea que los datos se pueden publicar.

**Reportes**

33. Realizar el reporte de las estaciones (Reportes estadísticos tanto tabulares como gráficos [Incluidos rosas de vientos], de una estación o múltiples estaciones y parámetros sean uno o varios de una o varias estaciones).

34. El sistema debe permitir exportar datos a csv, excel, txt, pdf de manera manual o automática.

35. El sistema debe permitir generar reportes, de acuerdo a la base de tiempo, diarios, mensuales, anuales, grupo de estaciones/parámetros, reportes datos máximos, mínimos y desviación estándar, reporte percentil (porcentajes de datos válidos y promedios), reportes multiestaciones y multiparámetros, reportes de histogramas, rosa de los vientos y rosa contaminantes, requieran se visualicen en gráficas ya sean líneas o barras, en 2D o 3D, reportes tabulados de fecha / parámetro.

36. El sistema debe permitir generar reportes de manera automática y descargarlos o enviarlos por correo de acuerdo en csv, excel, txt, pdf, permitiendo a su vez crear una plantilla de los reportes con imágenes y descripciones, estos deben ser preparados de acuerdo a un intervalo de tiempo o una única vez, eligiendo a qué usuarios se deben enviar. Estos usuarios pueden ser o no propios del sistema.

37. El sistema debe permitir crear y visualizar índices de calidad de aire.

38. El mapa debe permitir mostrar las estaciones y su indicativo de color de acuerdo a la calidad del aire, en el caso que el parámetro no tenga colores, permitir crear una escala de color y numérica con el valor del parámetro.

39. El sistema debe permitir generar reportes sobre las estaciones que tengan mayores banderas de datos negativas para generar \*ordenes de revisión\* para visitar las estaciones.

**No funcionales**

40. El software debe ser web con la capacidad de ser accedido con móviles.

41. El sistema debe ser creado con herramientas libres.

42. Se debe configurar un servidor FTP.

43. El sistema debe ser responsive design

44. El sistema debe permitir S.A.A.S. u On Premise.

**Alcance**

El alcance del proyecto será el desarrollo de los requerimientos 1 y 2 del módulo de comunicación y transferencia de información. Dependiendo del tiempo que quede después de haberse solucionado estos requerimientos, se podría avanzar con la solución de otros.

**Objetivos**

**General**

* Desarrollar una herramienta que permita gestionar los datos ambientales de la empresa San Ambiente de Cali.

**Específicos**

* Establecer un marco de trabajo siguiendo los lineamientos dados por la ISO 29110 utilizando como caso de estudio la integración de procesos del proyecto.
* Desarrollar la interfaz de los módulos de comunicación y transferencia de datos ambientales de la empresa San Ambiente aplicando DCU
* Aplicar un modelo de calidad al módulo de comunicación de información del sistema de gestión de datos ambientales de la empresa San Ambiente de Cali.
* Desarrollar los módulos web y móvil de comunicación y transferencia de los datos ambientales de la empresa San Ambiente.
* Desarrollar un sistema geográfico que permita visualizar la información de estaciones y dispositivos de captura de datos de la empresa Sanambiente.

**Entregables**

Los entregables que se darán al cliente por parte del equipo de desarrollo son con base a los establecidos en las fases de la metodología seleccionada, ICONIX.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fases y entregables correspondientes** | | | |
| **Fase 1:**  **Análisis de requerimientos** | **Fase 2:**  **Análisis y diseño preliminar** | **Fase 3:**  **Diseño detallado** | **Fase 4:**  **Implementación** |
| Realizar el prototipado. | Diagrama de robustez. | Diagrama de secuencia. | Escribir/generar el código (Desarrollo de los módulos de comunicación y transferencia de datos). |
| Diagramas de caso de uso. | Diagrama de clases. | Diagrama de clases refinado. | Realizar pruebas unitarias y de integración. |
| --------------------- | --------------------- | Planes de prueba | Realizar pruebas de aceptación del usuario. |

**Tareas y su duración estimada**

**Nota:** En la fase 3, el refinamiento del diagrama de clases y los planes de prueba se realizarán de forma implícita en la fase 4, por lo cual no se mencionará como tarea independiente. Adicionalmente el diagrama de secuencia de la fase 3 se pasará a realizar en la fase de 2 en conjunto con el diagrama de clases.

Por lo tanto, la fase 3 se elimina por completo y quedan finalmente de la siguiente manera:

**Fase 1:** Análisis de requerimientos.

**Fase 2:** Análisis y diseño preliminar.

**Fase 3:** Implementación.

Las duraciones de las tareas serán estipuladas por caso o casos de uso.

Se realizaron agrupaciones de dos o tres casos de uso para ser entregados en la misma fecha debido a su poca complejidad, obteniendo así un ahorro de tiempo en días que se puede aplicar para otros casos de uso que sí requieren mayor tiempo para ser desarrollados y posteriormente ser entregados.

**Equipo de desarrollo (Requerimientos de comunicación y transferencia de información 1 y 2)**

**Casos de uso #03 y #07**

**Nombre CU03: Crear región**

**Nombre CU07: Crear ciudad**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz)  * Documentar. | 6 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 3 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 13 |

**Caso de uso #02 y #09**

**Nombre CU02: Crear mantenimiento (tabla parámetro)**

**Nombre CU09: Crear alerta (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | 6 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 3 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 13 |

**Casos de uso #05 y #06**

**Nombre CU05: Crear organización (tabla parámetro)**

**Nombre CU06: Crear categoría (tabla parámetro)**

**Nombre CU11: Validación de usuario**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | 6 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 3 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 13 |

**Caso de uso #04**

**Nombre CU04: Crear estación (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | 7 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 6 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 27 |

**Caso de uso #01**

**Nombre CU01: Seleccionar estación para conexión**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | 7 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 6 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 27 |

**Casos de uso #08 y #10**

**Nombre CU08: Crear rango (tabla parámetro)**

**Nombre CU10: Crear cuadro de tiempo (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | 6 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | 3 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | 13 |

**Entrega proporcionada por el equipo de desarrollo, para uso general en el proyecto**

|  |  |
| --- | --- |
| **Descripción** | **Duración (días)** |
| Modelo entidad - relación. |  |

**Equipo de calidad (Requerimientos de comunicación y transferencia de información 1 y 2)**

**Casos de uso #03 y #07**

**Nombre CU03: Crear región (tabla parámetro)**

**Nombre CU07: Crear ciudad (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 9 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 19 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 2 |

**Caso de uso #02 y #09**

**Nombre CU02: Crear mantenimiento (tabla parámetro)**

**Nombre CU09: Crear alerta (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 6 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 12 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 2 |

**Caso de uso #05 y #06**

**Nombre CU05: Crear organización (tabla parámetro)**

**Nombre CU06: Crear categoría (tabla parámetro)**

**Nombre CU11: Validación de usuario**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 6 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 12 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 2 |

**Caso de uso #04**

**Nombre CU04: Crear estación (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 7 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 12 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 2 |

**Caso de uso #01**

**Nombre CU01: Seleccionar estación para conexión**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 6 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 12 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 2 |

**Caso de uso #08 y #10**

**Nombre CU08: Crear rango (tabla parámetro)**

**Nombre CU10: Crear cuadro de tiempo (tabla parámetro)**

|  |  |  |
| --- | --- | --- |
| **#** | **Descripción** | **Duración (días)** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagramas de casos de uso. 2. Validación de diagramas de casos de uso. 3. Tipos de pruebas. | 6 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | 14 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de prueba. 2. Reporte de incidencias. | 6 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | 4 |

|  |  |
| --- | --- |
| **Descripción** | **Duración (días)** |
| Elaborar documento final | 8 |

**Recursos**

**-Recursos humanos:**

* Equipo de gerencia del proyecto.
* Equipo de desarrollo.
* Equipo de calidad.

**-Estándares:**

* ISO/IEC 29110.
* ISO/IEC 25010.

**-Equipos:**

* Los equipos básicos de cómputo que serán utilizados para llevar a cabo todo el desarrollo del proyecto, poseen las siguientes configuraciones:

SO: Windows 10

Procesador: i5 7200U

RAM: 8GB

Disco duro: 1TB

Cabe mencionar que estos equipos son equipos de cómputo portátiles.

**-Herramientas:**

* Todas las herramientas que se utilizarán para llevar a cabo el desarrollo del proyecto son de uso libre, ejemplo: Visual Studio Code.

**Composición del equipo de trabajo:**

El equipo de trabajo se encuentra conformado de la siguiente manera:

* Equipo de gerencia del proyecto: conformado por 2 personas.
* Equipo de desarrollo: conformado por 3 personas.
* Equipo de calidad: conformado por 2 personas.

**Calendario de las tareas del proyecto**

Las fechas de las tareas serán estipuladas por caso o casos de uso.

Se realizaron agrupaciones de dos o tres casos de uso para ser entregados en la misma fecha debido a su poca complejidad, obteniendo así un ahorro de tiempo en días que se puede aplicar para otros casos de uso que sí requieren mayor tiempo para ser desarrollados y posteriormente entregados.

**Equipo de desarrollo (Requerimientos de comunicación y transferencia de información 1 y 2)**

**Casos de uso #03 y #07**

**Nombre CU03: Crear región**

**Nombre CU07: Crear ciudad**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Plantilla Bootstrap. 2. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  16/06/2019  **Actividad 1.2:**  16/06/2019 | **Actividad 1.1:**  22/06/2019  **Actividad 1.2: 6**  22/06/2019 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  22/06/2019  **Actividad 2.2:**  22/06/2019 | **Actividad 2.1:**  25/06/2019  **Actividad 2.2: 3**  25/06/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  25/06/2019  **Actividad 3.2:**  29/06/2019  **Actividad 3.3:**  02/07/2019  **Actividad 3.4:**  05/07/2019 | **Actividad 3.1: 4**  29/06/2019  **Actividad 3.2: 3**  02/07/2019  **Actividad 3.3: 3**  05/07/2019  **Actividad 3.4: 3**  08/07/2019 |

**Caso de uso #02 y #09**

**Nombre CU02: Crear mantenimiento (tabla parámetro)**

**Nombre CU09: Crear alerta (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  08/07/2019 | **Actividad 1.1: 6**  14/07/2019 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  14/07/2019  **Actividad 2.2:**  14/07/2019 | **Actividad 2.1:**  17/07/2019  **Actividad 2.2: 3**  17/07/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  17/07/2019  **Actividad 3.2:**  21/07/2019  **Actividad 3.3:**  24/07/2019  **Actividad 3.4:**  27/07/2019 | **Actividad 3.1: 4**  21/07/2019  **Actividad 3.2: 3**  24/07/2019  **Actividad 3.3: 3**  27/07/2019  **Actividad 3.4: 3**  30/07/2019 |

**Casos de uso #05, #06 y #11**

**Nombre CU05: Crear organización (tabla parámetro)**

**Nombre CU06: Crear categoría (tabla parámetro)**

**Nombre CU11: Validación de usuario**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  30/07/2019 | **Actividad 1.1: 6**  05/08/20190 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  05/08/2019  **Actividad 2.2:**  05/08/2019 | **Actividad 2.1:**  08/08/2019  **Actividad 2.2: 3**  08/08/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  08/08/2019  **Actividad 3.2:**  12/08/2019  **Actividad 3.3:**  15/08/2019  **Actividad 3.4:**  18/08/2019 | **Actividad 3.1: 4**  12/08/2019  **Actividad 3.2: 3**  15/08/2019  **Actividad 3.3: 3**  18/08/2019  **Actividad 3.4: 3**  21/08/2019 |

**Caso de uso #04**

**Nombre CU04: Crear estación (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  21/08/2019 | **Actividad 1.1: 7**  28/08/2019 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  28/08/2019  **Actividad 2.2:**  01/09/2019 | **Actividad 2.1: 3**  01/09/2019  **Actividad 2.2: 3**  04/09/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  04/09/2019  **Actividad 3.2:**  19/09/2019  **Actividad 3.3:**  22/09/2019  **Actividad 3.4:**  27/09/2019 | **Actividad 3.1: 15**  19/09/2019  **Actividad 3.2: 3**  22/09/2019  **Actividad 3.3: 5**  27/09/2019  **Actividad 3.4: 4**  01/10/2019 |

**Caso de uso #01**

**Nombre CU01: Seleccionar estación para conexión**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  01/10/2019 | **Actividad 1.1: 7**  08/10/2019 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  08/10/2019  **Actividad 2.2:**  11/10/2019 | **Actividad 2.1: 3**  11/10/2019  **Actividad 2.2: 3**  14/10/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  14/10/2019  **Actividad 3.2:**  29/10/2019  **Actividad 3.3:**  01/11/2019  **Actividad 3.4:**  06/11/2019 | **Actividad 3.1: 15**  29/10/2019  **Actividad 3.2: 3**  01/11/2019  **Actividad 3.3: 5**  06/11/2019  **Actividad 3.4: 4**  10/11/2019 |

**Casos de uso #08 y #10**

**Nombre CU08: Crear rango (tabla parámetro)**

**Nombre CU10: Crear cuadro de tiempo (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Análisis y requerimientos**  **Obj:** Revisión de los requerimientos.  **Actividades:**   1. Realizar el prototipado (diseño de la interfaz).  * Documentar. | **Actividad 1.1:**  10/11/2019 | **Actividad 1.1: 6**  16/11/2019 |
| 2 | **Fase 2: Análisis y diseño preliminar**  **Obj:** Revisión del diseño preliminar.  **Actividades:**   1. Diagrama de clases. 2. Diagrama de secuencia.  * Documentar. | **Actividad 2.1:**  16/11/2019  **Actividad 2.2:**  16/11/2019 | **Actividad 2.1:**  19/11/2019  **Actividad 2.2: 3**  19/11/2019 |
| 3 | **Fase 3: Implementación**  **Obj:** Codificación y creación del software.  **Actividades:**   1. Escribir/generar código. 2. Diccionario de funciones o módulos. 3. Refinamiento de acuerdo a las pruebas unitarias y de integración. 4. Realizar pruebas de aceptación del usuario.  * Documentar. | **Actividad 3.1:**  19/11/2019  **Actividad 3.2:**  23/11/2019  **Actividad 3.3:**  26/11/2019  **Actividad 3.4:**  29/11/2019 | **Actividad 3.1: 4**  23/11/2019  **Actividad 3.2: 3**  26/11/2019  **Actividad 3.3: 3**  29/11/2019  **Actividad 3.4: 3**  02/12/2019 |

**Entregas proporcionadas por el equipo de desarrollo, para uso general en el proyecto**

|  |  |  |
| --- | --- | --- |
| **Descripción** | **Fecha inicio** | **Fecha fin** |
| Modelo entidad - relación. |  |  |

**Equipo de calidad (Requerimientos de comunicación y transferencia de información 1 y 2)**

**Casos de uso #03 y #07**

**Nombre CU03: Crear región (tabla parámetro)**

**Nombre CU07: Crear ciudad (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. | **Actividad 1.1**  18/06/2019  **Actividad 2:**  20/06/2019  **Actividad 1.3:**  22/06/2019 | **Actividad 1.1:2**  20/06/2019  **Actividad 1.2:2**  22/06/2019  **Actividad 1.3:5**  27/06/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  27/06/2019  **Actividad 2.2:**  01/07/2019  **Actividad 2.3:**  03/07/2019  **Actividad 2.4:** 06/07/2019  **Actividad 2.5:**  09/07/2019  **Actividad 2.6:**  12/07/2019 | **Actividad 2.1:4**  01/07/2019  **Actividad 2.2:3** 03/07/2019  **Actividad 2.3:3**  06/07/2019  **Actividad 2.4:3**  09/07/2019  **Actividad 2.5:3**  12/07/2019  **Actividad 2.6:3**  15/07/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  15/07/2019  **Actividad 3.2:**  17/07/2019  **Actividad 3.3:**  19/07/2019 | **Actividad 3.1: 2**  17/07/2019  **Actividad 3.2:2**  19/07/2019  **Actividad 3.3:2**  21/07/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  21/07/2019 | **Actividad 4.1: 2**  23/07/2019 |

**Caso de uso #02 y #09**

**Nombre CU02: Crear mantenimiento (tabla parámetro)**

**Nombre CU09: Crear alerta (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. | **Actividad 1.2:**  23/07/2019  **Actividad 1.2:**  25/07/2019  **Actividad 1.3:**  27/07/2019 | **Actividad 1.2:2**  25/07/2019  **Actividad 1.2:2**  27/07/2019  **Actividad 1.3:2**  29/07/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  29/07/2019  **Actividad 2.2:**  31/07/2019  **Actividad 2.3:**  02/08/2019  **Actividad 2.4:** 04/08/2019  **Actividad 2.5:**  06/08/2019  **Actividad 2.6:**  08/08/2019 | **Actividad 2.1:2**  31/07/2019  **Actividad 2.2:2** 02/08/2019  **Actividad 2.3:2**  04/08/2019  **Actividad 2.4:2**  06/08/2019  **Actividad 2.5:2**  08/08/2019  **Actividad 2.6:2**  10/08/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  10/08/2019  **Actividad 3.2:**  12/08/2019  **Actividad 3.3:**  14/08/2019 | **Actividad 3.1:2**  12/08/2019  **Actividad 3.2:2**  14/08/2019  **Actividad 3.3:2**  16/08/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  16/08/2019 | **Actividad 4.1: 2**  18/08/2019 |

**Caso de uso #05 y #06**

**Nombre CU05: Crear organización (tabla parámetro)**

**Nombre CU06: Crear categoría (tabla parámetro)**

**Nombre CU11: Validación de usuario**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. | **Actividad 1.1:**  18/08/2019  **Actividad 1.2:**  20/08/2019  **Actividad 1.3:**  22/08/2019 | **Actividad 1.1:2**  20/08/2019  **Actividad 1.2:2**  22/08/2019  **Actividad 1.3:2**  24/08/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  24/08/2019  **Actividad 2.2:**  26/08/2019  **Actividad 2.3:**  28/08/2019  **Actividad 2.4:** 30/08/2019  **Actividad 2.5:**  02/09/2019  **Actividad 2.6:**  04/09/2019 | **Actividad 2.1:2**  26/08/2019  **Actividad 2.2:2** 28/08/2019  **Actividad 2.3:2**  30/08/2019  **Actividad 2.4:2**  02/09/2019  **Actividad 2.5:2**  04/09/2019  **Actividad 2.6:2**  06/09/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  06/09/2019  **Actividad 3.2:**  08/09/2019  **Actividad 3.3:**  10/09/2019 | **Actividad 3.1: 2**  08/09/2019  **Actividad 3.2:2**  10/09/2019  **Actividad 3.3:2**  12/09/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  12/09/2019 | **Actividad 4.1: 2**  14/09/2019 |

**Caso de uso #04**

**Nombre CU04: Crear estación (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. 4. Validación de diagrama de casos de uso. 5. Tipos de pruebas. | **Actividad 1.1:**  14/09/2019  **Actividad 1.2:**  16/09/2019  **Actividad 1.3:**  18/09/2019 | **Actividad 1.1:2**  16/09/2019  **Actividad 1.2:2**  18/09/2019  **Actividad 1.3:2**  20/09/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  20/09/2019  **Actividad 2.2:**  22/09/2019  **Actividad 2.3:**  24/10/2019  **Actividad 2.4:** 26/09/2019  **Actividad 2.5:**  28/09/2019  **Actividad 2.6:**  30/09/2019 | **Actividad 2.1:2**  22/09/2019  **Actividad 2.2:2** 24/09/2019  **Actividad 2.3:2**  26/09/2019  **Actividad 2.4:2**  28/09/2019  **Actividad 2.5:2**  30/09/2019  **Actividad 2.6:2**  02/10/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  02/10/2019  **Actividad 2:**  04/10/2019  **Actividad 3.3:**  06/10/2019 | **Actividad 3.1: 2**  04/10/2019  **Actividad 3.2:2**  06/10/2019  **Actividad 3.3:2**  08/10/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  08/10/2019 | **Actividad 4.1: 2**  10/10/2019 |

**Caso de uso #01**

**Nombre CU01: Seleccionar estación para conexión**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. | **Actividad 1.1:**  10/10/2019  **Actividad 1.2:**  12/10/2019  **Actividad 1.3:**  14/10/2019 | **Actividad 1.1:2**  12/10/2019  **Actividad 1.2:2**  14/10/2019  **Actividad 1.3:2**  16/10/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  16/10/2019  **Actividad 2.2:**  19/10/2019  **Actividad 2.3:**  21/10/2019  **Actividad 2.4:** 23/10/2019  **Actividad 2.5:**  25/10/2019  **Actividad 2.6:**  27/10/2019 | **Actividad 2.1:3**  19/10/2019  **Actividad 2.2:2** 21/10/2019  **Actividad 2.3:2**  23/10/2019  **Actividad 2.4:2**  25/10/2019  **Actividad 2.5:2**  27/10/2019  **Actividad 2.6:2**  29/10/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  29/10/2019  **Actividad 3.2:**  31/10/2019  **Actividad 3.3:**  02/11/2019 | **Actividad 3.1: 2**  31/10/2019  **Actividad 3.2:2**  02/11/2019  **Actividad 3.3:2**  04/11/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  04/11/2019 | **Actividad 4.1: 2**  06/11/2019 |

**Caso de uso #08 y #10**

**Nombre CU08: Crear rango (tabla parámetro)**

**Nombre CU10: Crear cuadro de tiempo (tabla parámetro)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descripción** | **Fecha inicio** | **Fecha fin** |
| 1 | **Fase 1: Planeación**  **Obj:** Se define el alcance del proyecto a desarrollar al igual que los roles del equipo de trabajo.  **Actividades:**   1. Diagrama de caso de uso. 2. Validación de diagrama de casos de uso. 3. Tipos de pruebas. | **Actividad 1.1:**  06/11/2019  **Actividad 1.2:**  08/11/2019  **Actividad 1.3:**  10/11/2019 | **Actividad 1.1:2**  08/11/2019  **Actividad 1.2:2**  10/11/2019  **Actividad 1.3:2**  12/11/2019 |
| 2 | **Fase 2: Preparación de pruebas según las características de la ISO/IEC 25010 (adecuación funcional, eficiencia de desempeño, usabilidad, seguridad y mantenibilidad)**  **Obj:** Se busca elaborar las pruebas a las cuales se va a someter el software.  **Actividades:**   1. Modelado de dominio. 2. Plan de pruebas. 3. Especificación del diseño de pruebas. 4. Especificación de casos de prueba. 5. Procedimiento de pruebas. 6. Reporte de trasmisión de pruebas. | **Actividad 2.1:**  12/11/2019  **Actividad 2.2:**  15/11/2019  **Actividad 2.3:**  17/11/2019  **Actividad 2.4:** 19/11/2019  **Actividad 2.5:**  21/11/2019  **Actividad 2.6:**  23/11/2019 | **Actividad 2.1:3**  15/11/2019  **Actividad 2.2:2** 17/11/2019  **Actividad 2.3:2**  19/11/2019  **Actividad 2.4:2**  21/11/2019  **Actividad 2.5:2**  23/11/2019  **Actividad 2.6:2**  25/11/2019 |
| 3 | **Fase 3: Ejecución de pruebas**  **Obj**: Se busca con los datos de la fase 2 ejecutar las pruebas.  **Actividades:**   1. Logs de pruebas. 2. Realizar pruebas unitarias y de integración. 3. Reporte de incidencias. | **Actividad 3.1:**  25/11/2019  **Actividad 3.2:**  27/11/2019  **Actividad 3.3:**  29/11/2019 | **Actividad 3.1: 2**  27/11/2019  **Actividad 3.2:2**  29/11/2019  **Actividad 3.3:2**  01/12/2019 |
| 4 | **Fase 4: Finalización de pruebas**  **Obj:** Documentar los resultados  **Actividades**   1. Reporte final de pruebas. | **Actividad 4.1:**  01/12/2019 | **Actividad 4.1: 4**  05/12/2019 |

|  |  |  |
| --- | --- | --- |
| **Descripción** | **Fecha inicio** | **Fecha fin** |
| Elaborar documento final | 05/12/2019 | 13/12/2019 |

**Esfuerzo y costo estimado**

**- Esfuerzo**

Para este proyecto se requiere un esfuerzo del 100%, que está repartido entre los distintos equipos de trabajo que lo conforman, de la siguiente manera:

|  |  |
| --- | --- |
| **Equipo de trabajo** | **Esfuerzo (%)** |
| Equipo de gerencia del proyecto | 25 |
| Equipo de desarrollo | 30 |
| Equipo de calidad | 45 |

**- Costo estimado**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Equipos** | **Dedicación (horas/semana)** | **Semanas (desde junio hasta diciembre)** | **Valor hora ($)** | **Total** |
| Equipo de gerencia del proyecto | 12 | 28 | 15000 | 5040000 |
| Equipo de desarrollo | 16 | 28 | 15000 | 6720000 |
| Equipo de calidad | 20 | 28 | 15000 | 8400000 |
| **Total** |  |  |  | **20160000** |

**Estrategia de control de versiones**

Para la creación del repositorio del proyecto se hará uso de la plataforma de desarrollo colaborativo GitHub y se utilizará para el control de versiones del proyecto la aplicación de escritorio de GitHub: GitHub Desktop, Interfaz Gráfica de Usuario (GUI) diseñada para facilitar el uso de Git. Adicional, se almacenarán semanalmente copias de respaldo de las versiones del proyecto en el servicio de la nube de Google Drive. De esta manera si se desea volver a un estado anterior del proyecto, se tiene tanto la herramienta y su sistema de control de versiones como las copias de seguridad almacenadas en Google Drive para hacerlo.

El mecanismo de almacenamiento que se utilizará para el proyecto será el gestor de base de datos relacional PostgreSQL.

**Instrucciones de entrega**

* Se acuerda la entrega completa de documentación y productos por caso o grupo de casos de uso (definidos anteriormente) y siempre y cuando todas las tareas con las que estos cuentan estén culminadas en su totalidad.
* Se van a realizar pruebas de las entregas con el gerente del proyecto y con el usuario.
* Se hará un acta de aceptación para las entregas que se le van a proporcionar al cliente.

### **Crear repositorio del proyecto**

Para la creación del repositorio del proyecto se hará uso de la plataforma de desarrollo colaborativo GitHub y se utilizará para el control de versiones del proyecto, la aplicación de escritorio de GitHub: GitHub Desktop. A continuación, se muestra una captura de imagen de referencia de la aplicación de escritorio de GitHub (ver figura 6):

Figura 6. Captura de imagen de referencia de la aplicación de escritorio de GitHub: GitHub Desktop.



*Nota:* Elaboración propia.

Cada equipo de trabajo tenía bajo su responsabilidad, la entrega de unas tareas definidas. Para la entrega ordenada de estas tareas por parte de los equipos de trabajo, se hizo la creación de ramas (branches) específicas para cada uno con el objetivo de que, al momento de subirlas, se lograra tener un seguimiento y control de las mismas. A continuación, se muestra cómo es visto el repositorio desde la aplicación web de GitHub (ver figura 7):

Figura 7. Repositorio del proyecto, visto desde la aplicación web de GitHub.



*Nota:* Elaboración propia.

El uso del repositorio como estrategia para la entrega ordenada de las actividades fue de crucial importancia, puesto que facilitó el seguimiento continuo del trabajo realizado por cada equipo, a través de la revisión de fecha, hora y descripción que cada cambio proporcionaba en el momento en que era producido.

### **Registrar los resultados de la verificación**

Antes de que se ejecutara el Plan del proyecto (Fase 2), se diligenciaba este artefacto, con el fin de verificar que los elementos del Plan eran viables o no completamente, y se llevaban a cabo (si las había) las correcciones correspondientes hasta que el documento fuera aprobado por todo el equipo, y de esta manera fuera ejecutado sin ningún problema o contratiempo. A continuación, se muestra un ejemplo de una verificación realizada al Plan del proyecto (ver figura 8):

Figura 8. Ejemplo de una plantilla de resultados de la verificación del plan de proyecto.

****

*Nota:* Elaboración propia con base en lo propuesto por la ISO/IEC 29110.

## Fase 2: Ejecución del plan del proyecto

En esta fase se llevó a cabo la ejecución del plan sobre el proyecto de la empresa Sanambiente, y el uso del Repositorio del proyecto definido en la fase 1.

### **Diligenciar un acta de reunión**

Plantilla que fue diligenciada los días sábados entre el mes de agosto e inicios del mes de diciembre en cada reunión lograda con el equipo de trabajo de forma presencial o virtual. A continuación, un ejemplo (ver figura 9):

Figura 9. Ejemplo de un acta de reunión lograda con el equipo de trabajo.



*Nota:* Elaboración propia con base en lo propuesto por la ISO/IEC 29110.

### **Diligenciar el registro de estado de progreso**

Para la realización de esta plantilla se tomó como base el calendario de tareas definido en el Plan del proyecto. A continuación, un ejemplo de un registro de estado que se realizó sobre los casos de uso 03 y 07 pertenecientes al equipo de desarrollo (ver figura 10):

Figura 10. Ejemplo de un registro de estado de progreso sobre los casos de uso 04 y 08 pertenecientes al equipo de desarrollo.



*Nota:* Elaboración propia con base en lo propuesto por la ISO/IEC 29110.

### **Diligenciar una solicitud de cambio**

Plantilla que fue diligenciada por cada equipo de trabajo (incluido el equipo de Gestión) cuando era necesario.

Cuando una solicitud era iniciada, el equipo de gestión era informado para realizar su seguimiento; y si eran notificados de su solución, se procedía a corroborar su veracidad y llevar a cabo el cambio respectivo del estado de iniciada o evaluada al estado de resuelta en la plantilla y se notificaba a los equipos implicados acerca de esta actualización. A continuación, se muestra un ejemplo de una solicitud de cambio en estado iniciada (ver figura 11):

Figura 11. Ejemplo de una solicitud de cambio iniciada.



*Nota:* Elaboración propia con base en lo propuesto por la ISO/IEC 29110.

## Fase 3: Evaluación y control del proyecto

En esta fase se llevó a cabo la evaluación de desempeño y progreso del proyecto, con respecto a lo documentado y pactado en el Plan del proyecto, dando uso para la comparación el Registro de estado de progreso. Adicional, se siguieron realizando solicitudes de cambio para abordar desviaciones importantes que surgieron y se realizó seguimiento de estas hasta que fueron resueltas. Se produjeron los siguientes artefactos:

### **Diligenciar el registro de corrección**

Plantilla que fue encargada para ser diligenciada por cada equipo de trabajo.

Cuando una corrección era iniciada, el equipo de gestión era informado para realizar su seguimiento; y si eran notificados de su realización, se procedía a corroborar su veracidad y llevar a cabo el cambio respectivo del estado de pendiente al estado de realizada en la plantilla y se notificaba a los equipos implicados acerca de esta actualización. A continuación, se muestra un ejemplo de un registro de corrección en estado pendiente (ver figura 12):

Figura 12. Ejemplo de un registro de corrección en estado pendiente.



*Nota:* Elaboración propia con base en lo propuesto por la ISO/IEC 29110.

## Fase 4: Cierre del proyecto

En esta fase se llevó a cabo la entrega de documentación y productos al cliente, tal y como se detalló en las Instrucciones de entrega del Plan del proyecto. Se produjeron los siguientes artefactos:

### **Diligenciar un registro de aceptación.**

Esta plantilla se diligenció por cada equipo (calidad y desarrollo), cuando se hizo la entrega completa de la documentación y productos al cliente a cargo de cada uno. A continuación, se muestra un ejemplo de un registro de aceptación diligenciado y firmado por el cliente receptor (ver figura 13):

# Conclusiones

# Referencias

Amavizca Valdez, L. O., García Ruíz, A. C., Jiménez López, E., Duarte Guerrero, G. L., & Vásquez Brindis, J. C. (2014). *Aplicación de la metodología semi-ágil ICONIX para el desarrollo de software: Implementación y publicación de un sitio WEB para una empresa SPIN -OFF en el Sur de Sonora, México.* [Científico].

Bolívar Guzmán, J., Lozano Argote, J., & Ramírez, J. J. (2020). *Desarrollo de los módulos de transferencia y comunicación de datos ambientales para la empresa de Sanambiente de Cali*. Institución Universitaria Antonio José Camacho.

Bona, C. (2002). *AVALIAÇÃO DE PROCESSOS DE SOFTWARE: UM ESTUDO DE CASO EM XP E ICONIX*. Universidade Federal de Santa Catarina.

Calvo Manzano, J., & García, I. (2008). Hacia la gestión cuantitativa en la gestión de proyectos en el ámbito de las pymes. *Revista Español de Innovación, Calidad e Ingeniería del Software*, *4*(2), 7-19.

Cataldi, Z. (2000). *Una metodología para el diseño, desarrollo y evaluación de software educativo*. Universidad Nacional de la Plata.

Estrada Reyes, J. N. (2015). Análisis de la gestión de proyectos a nivel mundial. *Palermo Business Review*, *12*, 38.

Fiestas Jacinto, J. E. (2015). *La implementación de un sistema de inteligencia de negocios que permita mejorar la toma de decisiones respecto a las remuneraciones de la empresa pesquera Carlos Eduardo S.R.L.-2014*. Universidad Nacional de Trujillo.

González Toste, D., Valdés González, A. A., Romero Gómez, Y., & García Pérez, Y. (2017). *El uso de las TIC como apoyo al proceso de formación máster en «Ciencias de la educación»*. *2, núm. 4*(octubre-diciembre (2014)), 1-11.

Hugues, P., Fisher, P., & Mc Daniel, J. (2010). *System development life cycle models and methodologies*. Canadian Society for International Health Certificate Course in Health Information Systems.

ICONIX Brand Group. (2016). *Manual introductorio de ICONIX*.

IEEE. (1991). *A Compilation of IEEE Standard Computer Glossaries”, IEEE Standard Computer Dictionary (IEEE-STD-610)*.

ISO/IEC. (2011). *Software engineering—Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile*.

Jacobson, I., Booch, G., & Rumbaugh, J. (2000). *El proceso unificado de desarrollo de software* (S. Sánchez, M. Á. Sicilia, C. Canal, & F. J. Durán, Trads.). Addison Wesley.

Laporte, C., O´connor, R., & García, L. (2016). *THE IMPLEMENTATION OF ISO/IEC 29110 SOFTWARE ENGINEERING STANDARDS AND GUIDES IN VERY SMALL ENTITES*. École de technologie supérieure, Montréal, Canada, School of Computing, Dublin City University, Dublin, Ireland, Universidad Peruana de Ciencias Aplicadas, Lima, Peru.

LEGARIA. (2018). *IMPLEMENTACIÓN DE PROCESO ORGANIZACIONAL DE GESTIÓN DE PROYECTOS EN DEVELOPIT*.

Lugo García, J. A., & Marín Sánchez, J. (2016). *Control de proyectos de software: Actualidad y retos para la industria cubana*. *24*(1). http://dx.doi.org/10.4067/S0718-33052016000100010

Madruñero Padilla, E. R. (2018). *Implementación del estándar ISO/IEC 29110 en el proceso de desarrollo de software de la dirección de desarrollo tecnológico e informático de la Universidad Técnica del Norte*. Universidad Técnica del Norte.

Matla Cruz, E. O. (2014). *Desarrollo de software guiado por la norma ISO/IEC 29110 y Scrum: SIDEP v.2.0*. Universidad Nacional Autónoma de México.

Mina Posu, F. A., & Romero Dorado, J. D. (2020). *Aplicación de un modelo de calidad al módulo de comunicación y traspaso de información del sistema de gestión de datos ambientales de la empresa Sanambiente*. Institución Universitaria Antonio José Camacho.

Mnkandla, E., & Dwolatzky, B. (2004). *A Survey of Agile Methodologies* [Científico].

PORRAS. (2019). *“METODOLOGÍA ÁGIL ICONIX EN LA CALIDAD DEL PRODUCTO SOFTWARE, LIMA, 2017”*. Universidad Nacional Federico villareal.

Pressman, R. S. (2002). *Ingeniería del Software. Un enfoque práctico* (R. Martín Ojeda, V. Yagüe Galaup, I. Morales Jareño, & S. Sánchez Alonso, Trads.; Quinta Edición). Concepción Fernández Madrid.

Puerta Marín, H. A., & Espinosa Bedoya, A. (2015). UNA ADAPTACIÓN DE PMBOK AL CICLO DE VIDA DE DESARROLLO DE PROYECTOS SOFTWARE EN PEQUEÑAS EMPRESAS. *VI Congreso Iberoamericano de Ingeniería de Proyectos - CIIP*.

Ramos, C., & Mendoza, L. (2014). *Implementación del estándar ISO/IEC 29110-4-1 para pequeñas organizaciones de desarrollo de software*. Universidad Peruana de Ciencias Aplicadas.

Roberts, A., & Wallace, W. (2002). *Gestión de Proyectos* (1.a ed., p. 54). Edinburg Business School, Heriot-Watt University.

Rosenberg, D., & Scott, K. (2001). *Applying Use Case Driven Object Modeling with UML: An Annotated e-Commerce Example* (First Edition). Addison Wesley.

Rosenberg, D., Stephens, M., & Collins-Cope, M. (2005). *Agile Development with ICONIX Process*. Apress.

Scalone, F. (2006). *Estudio comparativo de los modelos y estándares de calidad del software*. Universidad Tecnológica Nacional Facultad Regional Buenos Aires.

Silva Ascuntar, S., & García G., Y. C. (2018). *Documentación de la metodología ICONIX a través del desarrollo del caso «Oriéntate Cali»*. Institución Universitaria Antonio José Camacho.

UNE. (1995). *Gestión de la calidad y aseguramiento de la calidad. Vocabulario. (ISO 8402)*.

Estatuto de Investigaciones y Producción Intelectual, Pub. L. No. Acuerdo No. 14, 11 (2018).

Vásquez González, P. (2007). *Metodologías de Gestión de proyectos, alcance, impacto y tendencias*. Universidad de Chile.