

INSTITUTO TECNOLÓGICO DE COSTA RICA

**ÁREA ACADEMICA DE ADMINISTRACIÓN DE TECNOLOGÍAS DE
INFORMACIÓN**



**PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA
DE *SOFTWARE*, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS
APLICACIONES**

**PROYECTO DE GRADUACIÓN PARA OPTAR POR EL GRADO DE LICENCIATURA EN
ADMINISTRACIÓN DE TECNOLOGÍA DE INFORMACIÓN**

REALIZADO POR: WALTER CORDERO UREÑA

PROFESOR TUTOR: ANDRES SEGREDA JOHANNING

CARTAGO, 2016

DEDICATORIA/AGREDECIMIENTO

A mis padres: Walter Cordero Mora y Dulce María Ureña Mora, quienes son mi pilar y sin los cuales no hubiera sido posible llegar aquí; y a Krissia Cordero Fallas, que ha sido fundamental en los últimos años de mi vida para llegar a ser lo que soy.

Tabla de contenido

I INTRODUCCIÓN	11
I.1. Marco de referencia empresarial	13
I.2. Justificación del estudio	17
I.3. Alcances y limitaciones.....	21
II MARCO TEÓRICO	23
II.1. Application LifeCycle Management.....	24
II.2. Análisis de brechas	42
II.3. Estándares y políticas	45
II.4. Information Technology Infrastructure.....	48
II.5. Capability Maturity Model Integration	51
II.6. Pruebas de software.....	53
II.7. Manejo de fuentes de productos de software.....	56
II.8. Team Foundation Server	66
III MARCO METODOLÓGICO	68
III.1. Tipo de investigación.....	69
III.2. Sujetos y fuentes de Información.....	70
III.3. Técnicas de investigación	73
III.4. Procesamiento y análisis de los datos.....	78
IV ANALISIS DE LA SITUACIÓN	81
IV.1. Análisis de la situación deseada.....	82
IV.2. Análisis de la situación actual.....	83
V PROPUESTA A LA SITUACIÓN	109
V.1. Propuesta General	110
V.2. Plan de trabajo	119
V.3. Estándares, políticas, metodologías u otros.	129
VI CONCLUSIONES Y RECOMENDACIONES.....	185
VI.1. Conclusiones	186
VI.2. Recomendaciones	187
BIBLIOGRAFÍA.....	188
APÉNDICE	191
Apéndice 1.....	192
Apéndice 2.....	193

Apéndice 3.....	194
Apéndice 4.....	195
Apéndice 5.....	197
Apéndice 6.....	198
Apéndice 7.....	199
Apéndice 8.....	201
Apéndice 9.....	203
Apéndice 10.....	204
Apéndice 11.....	205
Apéndice 12.....	207
Apéndice 13.....	208
Apéndice 14.....	210
Apéndice 15.....	212
Apéndice 16.....	213
Apéndice 17.....	214
Apéndice 18.....	215
Apéndice 19.....	216
Apéndice 20.....	218
Apéndice 21.....	219
Apéndice 22.....	220
Apéndice 23.....	222
ANEXOS.....	224
Anexo 1.....	225
Anexo 2.....	226
GLOSARIO.....	227

Índice de ilustraciones

Ilustración 1 Aspectos fundamentales de ALM.....	25
Ilustración 2 Estrategia de branching Main Only.	59
Ilustración 3 Estrategia de branching Development Isolation.	60
Ilustración 4 Estrategia de branching Release Isolation.....	60
Ilustración 5 Estrategia de branching Development y Release Isolation.	61
Ilustración 6 Estrategia de branching Servicing y Release Isolation.	62
Ilustración 7 Estrategia de branching Servicing, hotfix y release.....	62
Ilustración 8 Estrategia de branching Code promotion.	63
Ilustración 9 Estrategia de branching Feature Isolation.....	65
Ilustración 10 Actores merge y branch.	138
Ilustración 11 Actores de manejo de código fuente.	151
Ilustración 12 CRM knowledge Base.	174
Ilustración 13 CRM Artículos publicados.....	175
Ilustración 14 CRM Categorías de documentos	176
Ilustración 15 CRM Plantilla de documentación	176
Ilustración 16 CRM Comentario en artículo.	177

Índice de tablas

Tabla 1 Principales productos.	14
Tabla 2 Áreas de ALM.	26
Tabla 3 Niveles de madurez.	28
Tabla 4 Resumen cuadro de análisis.	43
Tabla 5 Consideraciones de branch.	58
Tabla 6 Técnicas de investigación.	73
Tabla 7 Puntaje de niveles.	75
Tabla 8 Puntaje de respuestas.	75
Tabla 9 Análisis de datos por objetivo y técnica de investigación.	79
Tabla 10 Puntaje cuestionario arquitectura y diseño.	84
Tabla 11 Puntaje cuestionario ingeniería de requerimientos y experiencia de usuario.	86
Tabla 12 Puntaje cuestionario desarrollo.	89
Tabla 13 Puntaje cuestionario administración de la configuración de software.	92
Tabla 14 Puntaje cuestionario gobierno de TI.	95
Tabla 15 Puntaje cuestionario implementación y operaciones.	98
Tabla 16 Puntaje cuestionario aseguramiento de calidad.	101
Tabla 17 Puntaje cuestionario planificación y administración de proyectos.	104
Tabla 18 Tabla comparativa de niveles de ALM.	108
Tabla 19 Recomendaciones priorizadas.	120
Tabla 20 Plan de trabajo.	126
Tabla 21 Tipo campo.	135
Tabla 22 Objetos interfaz.	137
Tabla 23 Estrategia HotFix.	140
Tabla 24 Estrategia Casos Normales.	143
Tabla 25 Estrategia requerimientos.	146
Tabla 26 Branch administrativos.	149
Tabla 27 Metodología de cambios en BD.	155
Tabla 28 Datos de requerimientos.	159
Tabla 29 Tipos de requerimientos.	161
Tabla 30 Tipos de alcance.	162
Tabla 31 Métricas del negocio.	182
Tabla 32 Cuestionario de arquitectura y diseño.	192
Tabla 33 Cuestionario de ingeniería de requerimientos y experiencia de usuario.	193
Tabla 34 Cuestionario de desarrollo.	194
Tabla 35 Cuestionario de administración de la configuración del software.	195
Tabla 36 Cuestionario gobierno de TI.	197
Tabla 37 Cuestionario de implementación y operaciones.	198
Tabla 38 Cuestionario de aseguramiento de calidad.	199
Tabla 39 Cuestionario de planificación y administración de proyectos.	201
Tabla 40 Cuestionario de arquitectura y diseño con datos.	213
Tabla 41 Cuestionario de ingeniería de requerimientos y experiencia de usuario con datos.	214
Tabla 42 Cuestionario de desarrollo con datos.	215
Tabla 43 Cuestionario de planificación y administración de proyectos con datos.	216
Tabla 44 Cuestionario gobierno de TI con datos.	218

Tabla 45 Cuestionario de implementación y operaciones con datos.219
Tabla 46 Cuestionario de aseguramiento de calidad con datos.220
Tabla 47 Cuestionario de planificación y administración de proyectos con datos.222
Tabla 48 Glosario.....228

Índice de gráficos

Gráfico 1 Equipo de Trabajo.....	16
Gráfico 2 Resumen análisis.....	44
Gráfico 3 Etapas ALM.	83
Gráfico 4 Respuestas cuestionario arquitectura y diseño.....	84
Gráfico 5 Respuestas cuestionario ingeniería de requerimientos y experiencia de usuario.	86
Gráfico 6 Respuestas cuestionario desarrollo.....	89
Gráfico 7 Respuestas cuestionario administración de la configuración de software.	92
Gráfico 8 Respuestas cuestionario gobierno de TI.....	95
Gráfico 9 Respuestas cuestionario implementación y operaciones.	98
Gráfico 10 Respuestas cuestionario aseguramiento de calidad.	101
Gráfico 11 Respuestas cuestionario planificación y administración de proyectos.	104
Gráfico 12 Nivel de madurez ALM.....	107
Gráfico 13 Estrategia HotFix.....	139
Gráfico 14 Estrategia casos normales.	142
Gráfico 15 Estrategia requerimientos.	145
Gráfico 16 Metodología de cambios en BD.....	152
Gráfico 17 BD tropicalizaciones.....	153
Gráfico 18 BD estandarización.	154
Gráfico 19 Proceso de riesgos.	169
Gráfico 20 Gráfico radial.	226

RESUMEN

El siguiente trabajo se desarrolló en el centro de operaciones de una empresa dedicada a la creación y desarrollo de *software* para el sector financiero y gubernamental, con el objetivo de proponer una solución para la definición del proceso de entrega de *software* y guiar al centro de operaciones hacia la meta de alcanzar un estado avanzado en el ciclo de vida de las aplicaciones (ALM), según los niveles de madurez propuestos por Microsoft.

Para proponer una solución al centro de operaciones, se realizó un análisis de brechas, en el cual se compara el estado actual y el estado al que se desea llegar. Con el fin de cumplir con este objetivo se realizó una serie de cuestionarios, uno por proceso de ALM, con ellos se logró identificar el estado de cada uno de estos procesos y además conocer los puntos rezagados de la empresa en cuanto a lo que propone Microsoft.

Habiendo conocido las deficiencias del centro de operaciones y las iniciativas de Microsoft, se propuso un plan de trabajo en el cual se indicaron las diferentes acciones necesarias para alcanzar un estado de madurez avanzado en ALM, en cada una de sus etapas.

Con las acciones por realizar descritas se logró priorizar cada una de las actividades, según los criterios de la empresa, esto para mejorar las deficiencias de mayor importancia para ellos. De estas acciones se lograron identificar 9 que para la empresa son a las que se les debe poner mayor atención en la actualidad, por lo que, a estas 9 acciones, las cuales incluyen definición de estándares, políticas, metodologías, entre otras, se les desarrolló por el estudiante con el objeto de que una vez finalizado este trabajo, el centro de operaciones las pueda implementar.

Palabras Claves: ALM, ciclo de vida de las aplicaciones, estándares, políticas, metodologías, plantillas, Microsoft.

ABSTRACT

The following work was developed in the operations center of a company, dedicated to the creation and development of software for the financial and government sectors, with the aim of proposing a solution for the definition process of software delivery and guide the operations center to reach an advanced stage in the applications life cycle management (ALM), according to maturity levels proposed by Microsoft.

To propose a solution to the operations center, a gap analysis was created, in which the current state and the state they want to reach are compared. In order to meet this objective a series of questionnaires were applied, one for each ALM process, with which they were able to identify the status of each of these processes and also know the laggards points of the company in terms of what Microsoft states.

Knowing the deficiencies of the operations center and meeting the Microsoft initiatives, it is to propose a work plan which will indicate the different actions that need to be made to reach an advanced stage of maturity in ALM in each of its stages.

With the actions to be taken described above, it was able to prioritize each of the activities, according to the criteria of the company, to seek improvement on the shortcomings of most importance to them. Of these actions, 9 were identified, which for the company are the ones that they will have to pay more attention now a days, so these nine which include definition of standards, policies, methodologies, among others, were developed by the student, so that upon completion of this work, the operations center can implement.

Keywords: Applications life cycle management, ALM, standards, policies, methodologies, templates, Microsoft.

I INTRODUCCIÓN

El presente proyecto se realizó en el centro de operaciones de una empresa dedicada a la creación y desarrollo de *software*, para la cual se presentó una propuesta para la gestión del ciclo de vida de las aplicaciones.

Para realizar la propuesta al centro de operaciones se realizó un análisis de brechas con el propósito de conocer el estado actual y el estado al cual desean llegar, esto con base en los niveles de madurez propuestos por Microsoft, el cual tuvo como resultado que el centro de operaciones, a nivel general, tenía un estado “estandarizado” y que ninguna de las áreas de ALM (*Application Lifecycle Management*) tenía un estado “avanzado”. Por lo tanto, se identificaron diferentes mejoras posibles de realizar al proceso actual del ciclo de vida de las aplicaciones.

Las propuestas de mejora al ALM del centro de operaciones fueron plasmadas en un plan de trabajo, en el cual se detalló cada una de las acciones que la empresa puede realizar y con ellas lograr alcanzar el estado de “avanzado” en cada una de las áreas.

En el plan de trabajo, cada una de las actividades se categorizó por prioridad, tanto para la empresa como para la gestión del ciclo de vida de las aplicaciones, esto con el fin de que el centro de operaciones tenga identificadas las acciones a las cuales se les debe prestar mayor atención.

Para ayudar a la empresa a alcanzar el estado deseado en ALM se realizaron varias actividades con un estado de prioridad alta, detalladas en el plan de trabajo propuesto, las actividades que se realizaron fueron:

- Generación de políticas de código fuente.
- Definición de las estrategias de *merge* y *branch* de códigos fuentes para atención de casos.
- Política de manejo de código fuente para bases de datos y metodología de trabajo.

- Plantilla de requerimientos funcionales y técnicos.
- Definición del proceso de gestión de riesgos.
- Metodología para pruebas técnicas y funcionales.
- Generación de política para la creación y almacenamiento de conocimiento.
- Documentación y definición de políticas para el *deployment*.
- Generación de métricas del negocio.

Estas actividades realizadas, así como las demás que quedaron detalladas en el plan de trabajo, fueron basadas en lo que Microsoft propone, debido a que la empresa confía fielmente en lo que la multinacional plantea y además son *partner* de este.

I.1. Marco de referencia empresarial

La empresa donde se realizó el proyecto está dedicada a la creación y desarrollo de *software* para el sector financiero y gubernamental. En este último sector, son parte de la iniciativa Citinext, por sus productos de servicios en línea y recaudación. La empresa fue fundada en el año 2002.

Dentro de las principales herramientas para las áreas de negocio dicha empresa cuenta con las siguientes:

- Pagos electrónicos: Cuentan con aplicaciones para automatizar y fortalecer los pagos electrónicos por medio de operaciones interbancarias (SINPE), tarjetas de débito y crédito (procesadoras de pagos como ATH u otros), así como Hub de pagos para automatización de pagos de servicios públicos y privados orientados a mercados masivos.
- Aplicaciones de negocio: Han desarrollado aplicaciones para la operación de negocios (*CORE Systems*) de bancos, cooperativas y corredurías de seguros. Además, incluyen aplicaciones administrativas y financieras.
- Servicios electrónicos: Cuentan con aplicaciones que permiten a sus clientes brindar servicios electrónicos ágiles y seguros. Su cartera de productos incluye aplicaciones web transaccionales, como web *Banking*, servicios vía dispositivos celulares, entre otros. Así mismo, son líderes en la aplicación de la firma digital, para brindar servicios no tradicionales en línea desde la web.
- Consultoría: Brindan a sus clientes servicios de *outsourcing* y desarrollos a la medida. Cuentan con un equipo de profesionales de primer nivel, capacitados y orientados para trabajar con altos estándares de calidad y servicio.

Dentro de los principales productos de *software* comercializados con la marca de la empresa, es posible citar los siguientes indicadas en la Tabla 1:

Tabla 1 Principales productos.

Producto	Explicación
PS TRANSFER HUB	Sistema de pagos que permite la transferencia de fondos entre clientes de todas las entidades financieras inscritas en el sistema, así como retiros de efectivo y pagos de préstamos en cualquier entidad.
PS TRANSFER	Centro de gestión de pagos que permite a una entidad financiera conectarse en línea al sistema nacional de pagos de su país, realizar transferencias interbancarias, débitos electrónicos, negociación de divisas, y otros.
PS BANKER	Plataforma de aplicaciones para la operación de bancos y cooperativas de ahorro y crédito.
PS INSURER	<i>Software</i> diseñado para gestionar el negocio de corredurías y agencias de seguros desde una perspectiva diferenciadora de servicio al cliente.
PS WEB BANKING	Plataforma tecnológica optimizada para el establecimiento de portales web transaccionales en internet, con firma digital integrada.
PS FIRMA DIGITAL	<i>Software</i> diseñado para la utilización de mecanismos de autenticación oficiales en Costa Rica, usado en aplicaciones de internet, aplicaciones internas, manejo de documentos y otros.
PS SERVICES CARD	<i>Software</i> para el procesamiento y gestiones de tarjetas de crédito.
PS MÓVIL	Aplicaciones de banca celular para <i>smartphones</i> en plataforma iOS y Android.

Fuente: Elaboración propia, 2016.

Misión

La misión que tiene la organización donde se realizó el trabajo es:

Trabajamos para impactar la vida de nuestra gente, los resultados de nuestros clientes y el desarrollo integral de la sociedad.

Hacemos de la tecnología un espacio para potenciar nuestra creatividad y con ello, construir junto con nuestros clientes soluciones que fortalezcan sus ventajas competitivas (Empresa, 2016).

Visión

La empresa donde se realizó el trabajo tiene como visión:

“Ser una empresa modelo en la gestión del talento, reconocida por sus clientes como socio estratégico y apreciada por nuestra contribución a la sociedad” (Empresa, 2016).

Propuesta de valor

La organización donde se realizó el trabajo se basa en los siguientes valores:

- Buscar constantemente la innovación en todo lo que hacemos.
- Somos autodidactas nos encanta aprender.
- Vivir obsesionados por entender el negocio de nuestros clientes y el nuestro.
- Ser apasionados por cumplir nuestras metas.

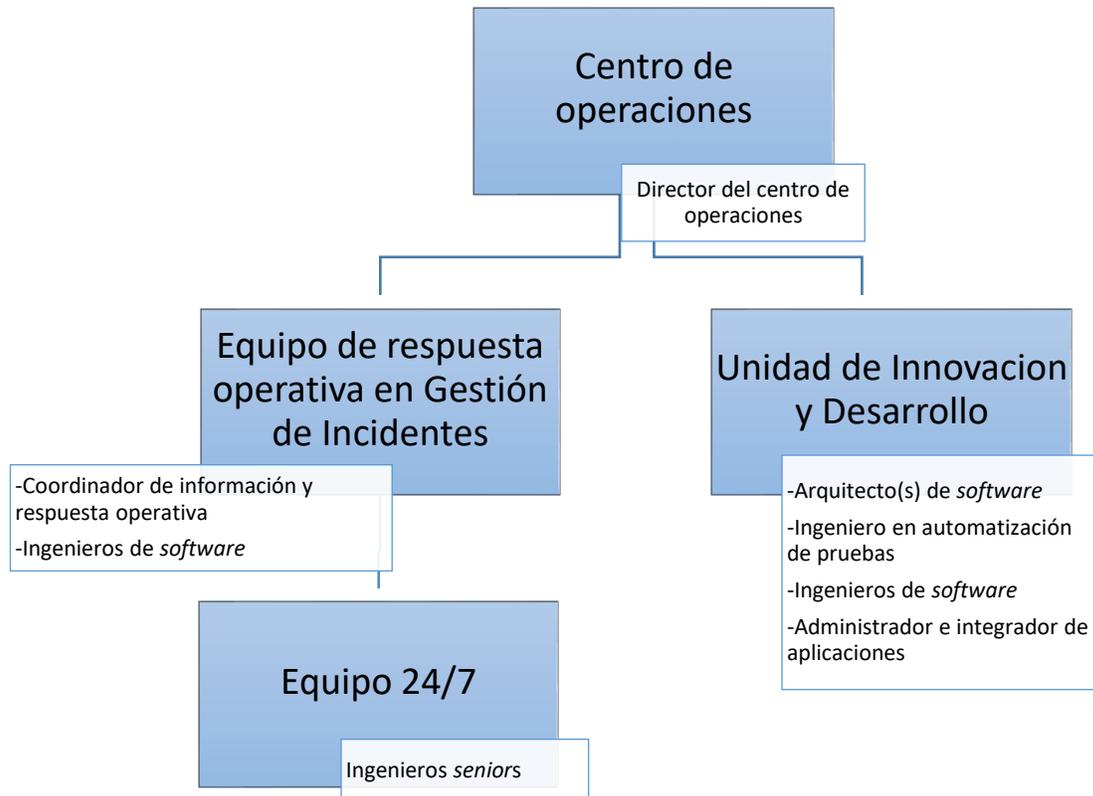
Equipo de trabajo

La empresa cuenta con diferentes equipos de trabajo, estos varían de acuerdo con los diferentes proyectos en los que se desempeña.

Con respecto al contexto de este proyecto, está desarrollado dentro del centro de operaciones de la empresa, anteriormente nombrado como área de soporte. Cada equipo de trabajo tiene responsabilidades diferentes.

En el Gráfico 1 se presenta el organigrama del centro de operaciones:

Gráfico 1 Equipo de Trabajo.



Fuente: Elaboración propia, 2016.

1.2. Justificación del estudio

Dentro del ciclo de vida de las aplicaciones, se debe tener definido lo que se debe realizar en cada una de las fases o etapas de dicho ciclo, esto para tener un mayor control y facilidad cuando surgen nuevos requerimientos, cambios o nuevos proyectos; dado que también al tener documentado todo el ciclo de vida de una aplicación, esta información va a servir de insumo para futuros proyectos desarrollados dentro de la empresa.

Actualmente, la empresa no tiene definido y/o estandarizado cada una de las fases del ciclo de vida de las aplicaciones que desarrollan, debido a esto ha identificado algunos problemas que han afectado la optimización de la labor realizada día tras día en la empresa, algunos de ellos son:

- Retrabajo.
- Falta de estandarización de mejoras.
- Falta de base de datos con conocimientos unificados.
- Manejo de fuentes de las aplicaciones provocan inconcistencias.
- Ausencia de pruebas automatizadas.
- Entre otros.

Por lo tanto, con este proyecto se pretende generar nuevas oportunidades y principalmente incrementar la efectividad del centro de operaciones.

A continuación, una lista de las oportunidades y beneficios que se esperaría obtener con la implementación de este proyecto:

- Con este proyecto la empresa podrá implementar alguna herramienta que gestione el ALM, ya que al tener claro cuáles son las entradas y salidas de los diferentes procesos. Además, se contará con estándares y políticas que ayudarán a la implementación de una herramienta.
- Con la solución propuesta se van a reducir los costos en cuanto al tiempo que demandan los procesos del ciclo de vida de las aplicaciones, ya que, gracias a la documentación generada por otros proyectos, se tendrá una

base de conocimientos definida, todos ellos van a servir para tomar mejores decisiones, así como para conocer el proceder en proyectos similares.

- Al tener estándares y políticas será fácil automatizar los procesos y las tareas.
- Se tendrá un mayor control de los códigos fuentes de las aplicaciones, dado que se implantarán estrategias que mejoren su gestión, esto evitará posibles conflictos en los *merge* de las aplicaciones o en los pases a producción.

Además, cabe mencionar que el estudiante a cargo de desarrollar el proyecto cuenta con los conocimientos necesarios para su adecuado desarrollo; ya que estos se adquirieron en muchos de los cursos aprobados a lo largo del plan de estudios de la carrera de Administración de Tecnología de Información, impartida en el Instituto Tecnológico de Costa Rica.

A lo largo del curso de las materias de dicha carrera, se analizaron los diferentes ciclos de vida, tanto de los proyectos que se desarrollaban de forma general como en los proyectos cuyos requerimientos eran específicamente de *software*.

Algunos de estos cursos, antes mencionados, que se relacionarán con la elaboración de este proyecto son los presentados en la siguiente lista:

- Administración de proyectos: En esta materia se estudian el ciclo de vida de los proyectos, así como los diferentes estándares o plantillas para su documentación.
- Ingeniería de requerimientos: Uno de los aspectos clave para el desarrollo de este trabajo es la elaboración de las plantillas de requerimientos funcionales y técnicos, los cuales fueron abordados en este curso.
- Diseño de *software*: En este curso se analizan conceptos de calidad en diseño de *software* y métodos, así como estrategias de diseño. Dicho conocimiento facilitará el diseño de los diferentes estándares y políticas que se mencionarán posteriormente.

- Producción, logística y calidad: En este curso se estudian aspectos para mejorar la productividad y la calidad de los productos de *software*, esto conformará el eje principal de este proyecto.
- Administración de servicios de tecnologías de información: En este curso se analiza ITIL, por lo que es de gran ayuda, ya que para la elaboración del proyecto muchas de las plantillas y políticas se van a basar en estas buenas prácticas.

Además, de estos conocimientos específicos, el estudiante ha aprendido tanto de habilidades técnicas, tal es el caso de la programación, como de habilidades blandas, lo cual es primordial para el desarrollo de este trabajo, dado que se deben tener fundamentos de programación para lograr generar los estándares y políticas, que van a utilizar los ingenieros de *software* de la empresa. Además, para lograr esto se va a tener que interactuar y comunicarse efectivamente con las diferentes partes del equipo de trabajo y personas involucradas directamente con el proyecto.

Seguidamente, se presenta la pregunta de investigación del proyecto, el objetivo general, objetivos específicos y meta del proyecto; con el fin de tener claro el alcance del proyecto

1.2.1. Pregunta de investigación

¿Cuál es el estado actual del centro de operaciones de la empresa en cada una de las etapas de ALM propuesto por Microsoft?

1.2.2. Objetivo general

Proponer una solución para definir el proceso de entrega de *software*, empleando estándares y políticas enfocados en la gestión del ciclo de vida de las aplicaciones.

1.2.3. Objetivos específicos

1. Realizar un análisis de brechas entre el estado actual del centro de operaciones de una empresa y el estado que se quiere alcanzar, basado en los niveles de madurez de ALM propuestos por Microsoft.
2. Proponer un plan de trabajo para lograr el estado deseado del Centro de Operaciones de una empresa, a partir de los resultados del análisis de brechas.
3. Emplear estándares, políticas, metodologías de los procesos asociados con la administración del ciclo de vida de las aplicaciones del centro de operaciones de una empresa.

1.2.4. Meta del proyecto

Poseer un mayor control en el centro de operaciones de la empresa cuando surjan nuevos proyectos, cambios en proyectos existentes o nuevos requerimientos, esto evitará el retrabajo, dado que estandarizará las mejoras de productos y tendrá una base de datos con conocimientos unificados.

1.3. Alcances y limitaciones

El proyecto se centrará en ALM, el cual es la gestión del ciclo de vida de las aplicaciones o, en otras palabras, desde la idea inicial de desarrollo de una aplicación, hasta el final de la vida de la misma.

Específicamente en este trabajo se realizarán las siguientes actividades:

- Realizar un análisis de brechas con el cual el centro de operaciones de la empresa logre conocer el estado actual y el estado deseado de cada una de las etapas de ALM al que desean llegar, basado en los niveles de madurez de ALM propuestos por Microsoft.
- Proponer un plan de trabajo con las actividades que se deben de realizar para cada una de las etapas de ALM, esto para lograr alcanzar el estado deseado por el centro de operaciones.
- Se desarrollarán las siguientes actividades (estándares, políticas, metodologías) para ayudar a la empresa a estar más cerca de alcanzar el estado deseado.
 - Generación de políticas de código fuente.
 - Definición de las estrategias de *merge* y *branch* de códigos fuentes para atención de casos.
 - Política de manejo de código fuente para bases de datos y metodología de trabajo.
 - Plantilla de requerimientos funcionales y técnicos.
 - Definición de proceso de gestión de riesgos.
 - Metodología para pruebas técnicas y funcionales.
 - Generación de política para la creación y almacenamiento de conocimiento.
 - Documentación y definición de políticas para el *deployment*.
 - Generación de métricas del negocio.

Quedan por fuera del alcance las implementaciones de los estándares, las políticas o las metodologías propuestas, además de desarrollar las actividades que no fueron citadas anteriormente, y que serán parte del plan de trabajo.

1.3.1. Limitaciones

Las limitaciones son todas las restricciones u obstáculos para la ejecución del trabajo, las limitaciones para este proyecto son:

- La empresa basa todos sus procesos en lo que propone Microsoft, por lo que para este trabajo las propuestas por realizar estarán basadas en lo que Microsoft defina.
- La empresa desea implementar la mayor cantidad de procesos en la plataforma Team Foundation Server, por lo que cada una de las propuestas debe estar basada en lo que se pueda realizar en esta plataforma.
- Por cuestiones de tiempo, solo se van a emplear 9 estándares / políticas / metodologías que mejoren el ALM del centro de operaciones de la empresa.

II MARCO TEÓRICO

En este apartado se presenta el sustento teórico al problema de investigación, los antecedentes y los términos referenciales.

II.1. Application LifeCycle Management

En la actualidad, las empresas, por lo general, buscan optimizar sus diferentes procesos y reducir los costos asociados a ellos. En muchos negocios surge la necesidad de gestionar proyectos, hacer seguimiento de estos, tener claras las actividades realizadas y los entregables o productos liberados; además, es importante probar que todo cumpla con lo requerido, pues ello permite obtener un resultado eficiente y de alta calidad. Así mismo, esto no termina en la entrega del producto final, sino que también se deben rescatar las lecciones aprendidas para seguir mejorando, monitoreando que el producto satisfaga las necesidades surgidas en el proceso. Igualmente es importante dar mantenimiento y ofrecer mejoras o actualizaciones de acuerdo con diferentes contextos planteados.

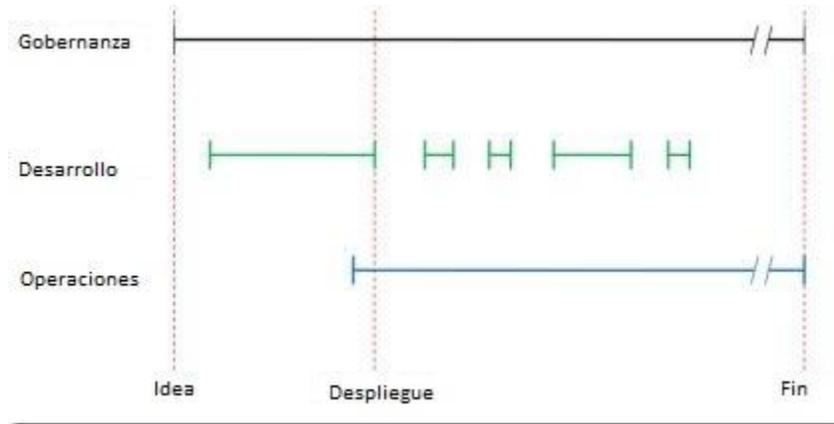
Lo citado anteriormente es totalmente aplicable al negocio del *software*, y es a partir de esto que nace la gestión del ciclo de vida de las aplicaciones (*Application LifeCycle Management* – ALM de ahora en adelante), según Chappell (2008), ALM incluye todo el tiempo durante el cual una organización está gastando dinero en este activo, desde la idea inicial, hasta el final de la vida de la aplicación; por lo que es un proceso continuo de la gestión de la vida de una aplicación a través de la gobernanza, desarrollo y mantenimiento.

Dentro de esta definición es posible encontrar 3 puntos clave íntimamente conectados, descritos por Chappell (2008) en “What is Application LifeCycle Management”. Los puntos clave son:

- **Gobernanza:** Asegurarse de que la aplicación siempre provea lo que el negocio necesite.
- **Desarrollo:** La parte fundamental de ALM. Es la construcción del producto por generar, que ocurre periódicamente en varias fases o iteraciones.
- **Operaciones:** El planeamiento y ejecución del despliegue y monitoreo de la aplicación.

En la Ilustración 1 se detalla los puntos clave mencionados anteriormente.

Ilustración 1 Aspectos fundamentales de ALM



Fuente: What is Application LifeCycle Management, David Chappell (2008)

Bajo estos puntos, surgen diferentes tipos de herramientas que durante el tiempo de implementación han ido evolucionando. Esto debido a que muchas herramientas se enfocaron mayoritariamente en el aspecto de desarrollo, fundamentalmente en el control de versiones.

Actualmente, los proveedores de soluciones ALM han alcanzado a entender estos 3 aspectos al ofrecer un conjunto de herramientas integradas que permiten muchas características contempladas por ALM; sin embargo, cabe decir que no todas cumplen en su totalidad con los 3 aspectos, esto debido a la complejidad y lo sofisticado que es el desarrollo de *software*.

II.1.1. Microsoft ALM

Microsoft ha sido una de las empresas que ha integrado ALM a sus herramientas, tal es el caso de Team Foundation Server (TFS), el cual engloba procesos, gestión de proyectos, compilaciones, monitoreo, control de versiones, entre otras. En resumen, TFS en conjunto con las herramientas de desarrollo Visual Studio, gestión Office, prototipado SketchFlow y gestión de documentos de proyecto en SharePoint, entre otras, llega a satisfacer lo que buscan las empresas con ALM.

II.1.1.1. Áreas de ALM

En la Tabla 2 se van a nombrar cada una de las fases donde interviene ALM y se realizará una pequeña descripción de ellas.

Tabla 2 Áreas de ALM.

Área	Descripción
Arquitectura y diseño	Área encargada de la estructuración a un alto nivel de las soluciones, así como del diseño lógico de los componentes.
Ingeniería de requerimientos y experiencia de usuario	Área encargada de la definición de requerimientos, así como de la comunicación con usuarios, encargada de que la experiencia del usuario sea agradable desde la toma de requerimientos.
Desarrollo	Área encargada de construir los componentes de <i>software</i> necesarios según la demanda de los requerimientos.
Administración de la configuración del <i>software</i>	Área encargada de la administración necesaria requerida por desarrollo, desde la administración de las fuentes hasta los despliegues y pases de ambientes.
Gobierno de TI	Área encargada de implementar normas, políticas o estándares según corresponda para que TI administre de manera eficaz a las necesidades de los clientes por medio de buenas prácticas.
Implementación y operaciones	Área encargada de la gestión de ambientes e instalación de los despliegues, gestión de riesgos en operaciones y contingencias necesarias.
Aseguramiento de la calidad	Área encargada de asegurar la calidad de los componentes que se construyen en la fábrica, así como la correcta integración de los componentes.

Área	Descripción
Planificación y administración de proyectos	Área encargada de administrar y dar seguimiento a los avances de los proyectos además de dictar la metodología y estrategia a seguir dentro de los ciclos de desarrollo.

Fuente: Elaboración propia, 2016.

Como se puede observar en la Tabla 2, el ALM está compuesto por 8 áreas, las cuales abarcan el ciclo de vida de las aplicaciones.

II.1.1.2. Niveles de Madurez de ALM

El ALM de Microsoft provee un marco para organizar pasos evolutivos en 4 niveles de madurez que establecen los cimientos para la mejora continua del proceso. Estos 4 niveles de madurez definen una escala para medir el proceso de *software* de una organización y para valorar su capacidad. Estos niveles ayudan también a la organización a establecer las prioridades de sus esfuerzos de mejora.

Un nivel de madurez consiste en una plataforma evolutiva bien definida, dirigida hacia el logro de un proceso de *software* maduro. Cada nivel de madurez constituye una capa en la cimentación para la mejora continua del proceso. Estos niveles incluyen una serie de objetivos del proceso, que cuando se satisfacen, estabilizan un componente importante del proceso de *software*.

En la Tabla 3, se detalla cada uno de los niveles de madurez propuestos por Microsoft (Visual Studio ALM Quick Reference Guide, 2015).

Tabla 3 Niveles de madurez.

Nivel	Descripción
Básico	<p>El equipo de desarrollo ha adoptado formas propias de las buenas prácticas.</p> <p>Las prácticas se llevan a cabo de manera informal ad hoc.</p> <p>Prácticas no son documentadas.</p> <p>Poca o ninguna comunicación entre los equipos.</p> <p>Algunos papeles claves no se realizan de manera consistente.</p>

Nivel	Descripción
Estandarizado	<p>El uso de las buenas prácticas está empezando a ser adoptados a través de algunos o todos los equipos de desarrollo.</p> <p>Las herramientas utilizadas son generalmente subutilizadas o no integradas con las diferentes áreas.</p> <p>Un uso relativamente informal de las herramientas sin uso de políticas definidas.</p> <p>Poca o informal documentación de las prácticas utilizadas.</p> <p>Falta de fortalecimiento de figuras claves y centralización de conocimiento.</p>
Avanzado	<p>El uso de herramientas de soporte es un fenómeno generalizado en todos los equipos.</p> <p>Las herramientas están totalmente integradas en el IDE (Integrated Development Environment).</p> <p>El uso de herramientas es formal con las políticas de uso documentadas y se hacen cumplir.</p> <p>La documentación se mantiene formalmente.</p> <p>Las mejores prácticas son adoptadas y documentadas.</p> <p>Inicios de automatización de algunos procesos.</p>
Dinámico	<p>Liderazgo en la industria.</p> <p>Práctica de desarrollos innovadores.</p>

Nivel	Descripción
	<p>La gestión de la cartera está integrada con herramientas y procesos.</p> <p>Capacidad de seguimiento de los requisitos y el uso de informes de análisis de impacto para las solicitudes de cambio.</p> <p>El uso de métricas de calidad de servicio de ayuda en el tiempo de entrega, coste de mantenimiento e identificación de errores.</p> <p>Automatización de procesos.</p>

Fuente: Elaboración propia, 2016.

Seguidamente, se expondrá cada uno de los puntos considerados como necesarios por parte de Microsoft para llegar al nivel de madurez deseado por cada área.

II.1.1.2.1 Arquitectura y diseño

Básico:

- Arquitectura documentada de forma incorrecta e ineficiente.
- Uso inconsistente o inexistente de las herramientas de modelado.
- No existe un proceso claro para transformar los requisitos del negocio en requisitos técnicos.
- Arquitectura de despliegue no considerada al inicio del proyecto.
Comprensión poco clara de la función de arquitecto.

Estandarizado:

- Rol de arquitectura entendido, claramente identificado y combinado con otros roles.
- Herramientas identificadas, fase de adopción temprana.
- Algunos hábitos se empiezan a formar, existe un poco de coherencia en los procesos.

- Se tiene documentación irregular, no de manera consistente, entre los equipos o proyectos.

Avanzado:

- Equipo de arquitectura dedicado.
- Al elegir las herramientas de arquitectura se tiene en cuenta el proceso de implementación.
- Herramientas integradas utilizadas en los diferentes equipos en los diferentes proyectos.
- Aprovechamiento de las prácticas y procedimientos.
- Aplicación de prácticas y procesos a través de diferentes equipos y proyectos.

Dinámico:

- Proceso de arquitectura formalizado y documentado. Inclusión constante de patrones y prácticas.
- Mecanismo claramente definido para compartir o forzar el uso de patrones y prácticas a través de proyectos y equipos.
- Contribuye a la comunidad de desarrollo interno y externo a través de la utilización de los artículos publicados, documentos técnicos, etc.

II.1.1.2.2 Ingeniería de requerimientos y experiencia de usuario (EU)

Básico:

- Equipo de desarrollo supone que sabe qué construir.
- Pocos o ningún requisito por escrito, ninguna herramienta específica.
- Rol de EU no definido, además el EU es hecho por programadores o quien esté disponible.
- Limitados o inexistentes comentarios por parte de los clientes, poca validación con los interesados.
- Diseño manejado por los procesos o datos.
- Documentación de usuario no definida, ni estandarizada, o definida bajo demanda.

Estandarizado:

- Calidad consistente, formatos de documentación y almacenamiento de requisitos.
- Seguimiento y hacer versiones de los requisitos.
- Rol de experiencia de usuario definido, no necesariamente a cargo de un experto de experiencia de usuario.
- Ciclo manual de revisión del cliente.
- Principios de diseño centrado en el usuario entendido, pero herramientas desconectadas.
- Existencia de alguna documentación consistente de usuario.

Avanzado:

- Seguimiento de tipos de requisitos, relaciones y trazabilidad de tareas.
- Uso de herramientas de trazabilidad por medio del uso de reportes de análisis de cobertura de requisitos y herramientas integradas para producir interfaz de usuario/requisito/matriz de tareas.
- Expertos en experiencia de usuario y especialistas en documentación de usuario involucrados al principio del proyecto.

- Herramienta de diseño centrado en el usuario (*storyboard*) integrado con requisitos y tareas.

Dinámico:

- Establecimiento de un comité de control de cambios del producto.
- Uso de reportes de análisis de impacto para las solicitudes de cambio.
- Indicadores del progreso de requisitos y solicitudes de cambios publicados.
- Expertos de experiencia de usuario incorporan los más recientes principios de IU (interfaz de usuario).
- Mejora continua de experiencia de usuario para uso posterior.
- Diseñadores entienden la intersección de fácil implementación contra interfaz de usuario genial.

II.1.1.2.3 Desarrollo

Básico:

- Los desarrolladores tienen pleno conocimiento de los componentes por utilizar antes de empezar a programar.
- Los desarrolladores utilizan la tutoría entre iguales.
- Desarrolladores siguen un estándar de codificación previamente acordado.

Estandarizado:

- Los desarrolladores utilizan un *framework* para desarrollo de *webservices*.
- Los desarrolladores tienen conocimiento suficiente de las últimas tecnologías Microsoft para aprovechar al máximo las inversiones en herramientas de desarrollo.
- Los desarrolladores utilizan el ciclo de vida de desarrollo seguro.

Avanzado:

- Los desarrolladores utilizan un *framework* que abstraer la capa de presentación de la lógica de negocios.
- Los desarrolladores utilizan un *framework* que asista la gestión de la identidad.
- Los desarrolladores utilizan un *framework* de máquina de estado.

Dinámico:

- Los desarrolladores tienen un plan de entrenamiento continuo en las últimas tecnologías de desarrollo.
- Los desarrolladores tienen un líder de desarrollo con áreas de conocimiento claramente asignadas.
- Los desarrolladores utilizan contratos de código y desarrollo guiados por pruebas (TDD).

II.1.1.2.4 Administración de la configuración del *software*

Básico:

- No se utiliza versiones de código fuente ni se hace uso de *check-ins* irregulares.
- Copias locales de código.
- Proceso de *build* manual no documentado o bajo demanda.
- No existe trazabilidad entre *build* y el contenido / el trabajo realizado y los requisitos.
- Poca comprensión de los conceptos de *branch/merge*.

Estandarizado:

- Uso de herramientas de gestión de código no integrados. Máquina dedicada para *builds*.
- Proceso de *build* informalmente documentado.
- Conceptos de *branch/merge* entendido por líderes de integración. *Check-Ins* realizados de manera regular o diariamente.

Avanzado:

- Uso de un IDE integrado con las herramientas de gestión de código fuente.
- Rol dedicado de gestión de configuración.
- Proceso formal y documentado de *build*.
- Indicadores de *build* publicados regularmente. *Build* bajo demanda habilitado.
- Pruebas unitarias ejecutadas como pruebas de verificación de *builds*.

Dinámico:

- *Scripts* de construcción (*build*) reutilizables y centralizados.
- Integración de varios módulos de código, tanto internos como externos.
- Monitoreo y notificación de resultados de *build*. Rol *DevOp* dedicado para el mantenimiento del proceso de *build* y su infraestructura.

II.1.1.2.5 Gobierno de TI

Básico:

- Los proyectos se inician con justificación limitada.
- Proyectos financiados por las opiniones de importantes factores de influencia.
- Ninguna evaluación de ROI (*return on investment*) o retrospectiva. Ningún proceso de revisión del portafolio.
- Ningún programa de cumplimiento.
- Ninguna iniciativa de mejora de procesos en marcha.

Estandarizado:

- Certificación informal del programa de cumplimiento elegido.
- Certificación de cumplimiento aplicado e inconsistente monitoreo entre los equipos.
- Herramientas de procesos semi-manuales (listas de Excel, entre otras).
- Uso aleatorio de objetivos de la iniciativa.

Avanzado:

- Certificación formal del programa de cumplimiento elegido.
- Utilizando técnicas de gestión de cartera, pero la cartera y herramientas de gestión de proyectos no están necesariamente integrados.
- Recursos entre equipos gestionados y con asignación de tiempos. Integrado con el programa de certificación y cumplimiento.

Dinámico:

- Herramientas y procesos de gestión de la cartera completamente integrados.
- Participación en la creación y revisión de procesos de programas de cumplimiento de estándares de la industria.
- ROI y métricas de retrospectiva.

II.1.1.2.6 Implementación y operaciones

Básico:

- Poca o ninguna comunicación entre los equipos de operaciones y desarrollo. Ningún proceso de seguimiento mesa de ayuda/error, comunicación *ad hoc* vía correo.
- Problemas de implementación de infraestructura identificados y resueltos durante la implementación.
- Ambientes como desarrollo, pre-producción, pruebas, UAT (*User acceptance testing*), producción no documentados o segregados.
- Calendarización de *builds* no regulada o no planificada.

Estandarizado:

- Herramienta de mesa de ayuda aislada (entrenamiento, problemas de usuarios, infraestructura).
- Seguimiento de errores no integrado.
- Algunos procedimientos de monitoreo y despliegue y/o proceso de aprobación.
- Limitada automatización y validación de despliegue de *builds*. Rol de administrador de despliegue identificado.
- Infraestructura documentada, pero el responsable de ambientes segregados es incierto.

Avanzado:

- Mesa de ayuda integrada con gestión de errores.
- Instrumentación de monitoreo conectado a la infraestructura y las aplicaciones.
- Herramientas para implementar y validar la implementación exitosa de *builds* (pruebas de humo, entre otras.).
- Identificación clara del equipo multifuncional.
- Arquitectura de infraestructura documentada en herramientas integradas.

- Ambientes segregados, propietarios y procedimientos de promoción definidos.

Dinámico:

- Medición de la calidad de la mesa de ayuda en cuanto a tiempo de resolución, costos de mantenimiento e identificación de sistemas propensos a errores.
- Proceso de despliegue automatizado y trazable. Monitoreo continuo y proactivo.

II.1.1.2.7 Aseguramiento de calidad

Básico:

- Equipo de aseguramiento de calidad no dedicado.
- Pruebas funcionales no planificadas ejecutadas por el equipo de desarrollo. Acercamiento de depuración y no de pruebas.
- No existen indicadores de calidad.
- Ciclos largos de depuración, arreglo y despliegue. Alta tasa de defectos de regresión.

Estandarizado:

- Equipo de aseguramiento de calidad dedicado. Proceso de plan de pruebas definido.
- Herramientas de pruebas no integradas.
- Procedimientos y ambientes de pruebas informalmente documentados. Seguimiento de progreso rudimentario.

Avanzado:

- La cultura organizacional acepta las políticas de pruebas. La planificación de prueba comienza en la fase de requisitos.
- Realizar las pruebas es un proceso medible y cuantificable.
- Herramientas integradas generan métricas publicables.

Dinámico:

- Equipo de mejora del proceso de pruebas y herramientas en sitio.
- El liderazgo de la industria en la evaluación de herramientas y estrategias de pruebas.
- Prácticas para prevenir defectos.
- Pruebas basadas en muestreo estadístico, medición de la confianza, integridad y fiabilidad.

II.1.1.2.8 Planificación y administración de proyectos

Básico:

- Ningún plan de comunicación formal a los interesados del proyecto.
- Procesos informales o inexistentes para la estimación, planeación, gestión de riesgos y el alcance. Enfoque instintivo.
- Coordinación informal del equipo y asignación de tareas mediante correo electrónico, o de manera verbal.
- Las finanzas no son evaluadas de forma continua por el gerente del proyecto.
- La responsabilidad del gerente del proyecto no está definida claramente.

Estandarizado:

- Uso individual, no integrado, no estandarizado de herramientas de planeación de proyectos.
- El uso de la herramienta depende de las fortalezas de cada administrador de proyectos.
- Finanzas manualmente evaluadas por el administrador de proyectos.
- Responsabilidades del administrador de proyectos claramente asignadas.

Avanzado:

- Gestión integrada de errores, tareas y solicitudes de cambio.
- Uso de EPM para seguimiento financiero y de recursos a través de la integración de EPM (Enterprise Project Management) con TFS (Team Foundation Server).
- Recursos externos, las partes interesadas y los socios comparten la información del proyecto y tienen herramientas integradas para llevar a cabo su rol en el proyecto (*Sharepoint*, Portal del proyecto).
- Gerentes de proyecto dedicados.

Dinámico:

- Completa integración entre gestión de cartera y proyectos.
- Las métricas son utilizadas para impulsar los proyectos y ayudar en la estimación y reestimación.
- PMO (*project management office*) en su lugar.

II.2. Análisis de brechas

Según Ruiz (2012), el análisis de brechas es una herramienta de análisis para comparar el estado y el desempeño real de una organización, estado o situación en un momento dado, respecto a uno o más puntos de referencia seleccionados de orden local, regional, nacional y/o internacional.

De acuerdo con Ruiz (2012), se deben de realizar 4 pasos generales y, a manera de guía, cada paso busca responder una pregunta, las cuales son:

- 1) Decidir cuál es la situación actual que se desea analizar ("lo que es") y se quiere resolver. En este paso se responde a la pregunta: ¿Dónde estamos?
- 2) Delinear el objetivo o estado futuro deseado ("lo que debería ser"). Respondería la pregunta ¿En un año específico adónde deberíamos llegar?
- 3) Identificar la brecha entre el estado actual y el objetivo. Responde a la pregunta ¿Cuán lejos estamos de donde queremos estar?
- 4) Determinar los planes y las acciones requeridas para alcanzar el estado deseado. Responde a la pregunta de: ¿Cómo llegamos al año planteado?

Microsoft cuenta con una herramienta desarrollada por los expertos de ALM para ayudar a un gerente de desarrollo de aplicaciones en la recopilación de información sobre las prácticas y los procesos de desarrollo en el entorno de un cliente. Esta herramienta cuenta con varios cuestionarios implementados para identificar el estado actual de los procesos individuales de TI y decidir sobre un estado deseado para esos procesos. También se presenta orientación y varios resúmenes para ayudar con las recomendaciones para seguir adelante (Visual Studio ALM Quick Reference Guide, 2015).

A manera de ejemplo, se muestra, en la Tabla 4 y en el Gráfico 2, el resumen de un posible análisis de brechas, este fue realizado utilizando la herramienta ofrecida por Microsoft.

Tabla 4 Resumen cuadro de análisis.

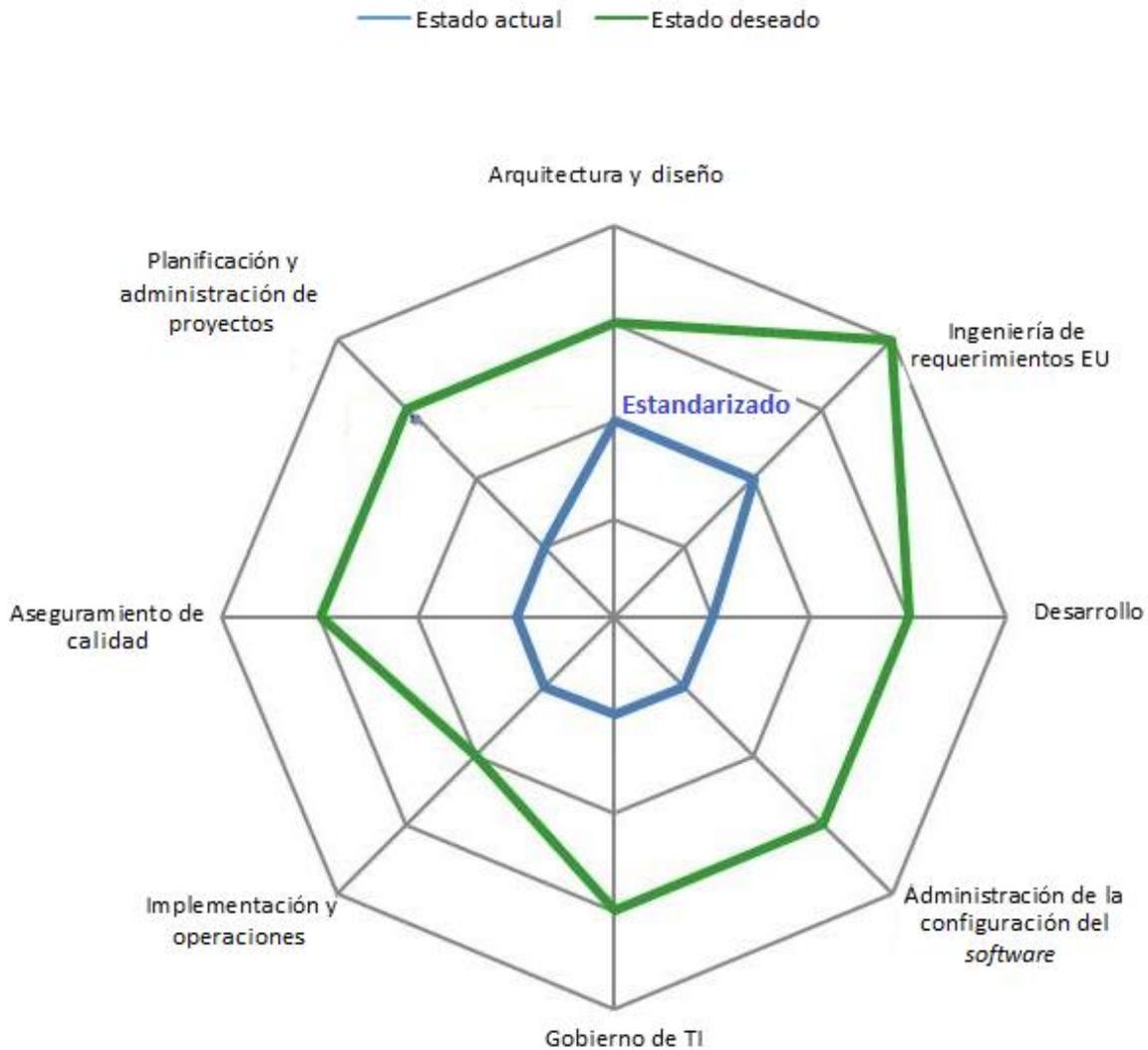
Áreas ALM	Estado Actual	Estado Deseado	Estado
Arquitectura y diseño	Estandarizado	Avanzado	!
Ingeniería de requerimientos y experiencia de usuario	Estandarizado	Dinámico	✘
Desarrollo	Básico	Avanzado	✘
Administración de la configuración	Básico	Avanzado	✘
Gobierno de TI	Básico	Avanzado	✘
Implementación y operaciones	Básico	Avanzado	✘
Aseguramiento de calidad	Básico	Estandarizado	!
Planificación y administración de proyectos	Básico	Avanzado	✘

Fuente: Microsoft (2016).

Para la Tabla 4 , el símbolo  significa precaución, en el cual se está a un nivel por debajo del estado deseado, y el  significa crítico, que se usa cuando se tiene más de un nivel por debajo del estado deseado, en cuanto a la comparación del estado actual y el estado al que se desea llegar.

Gráfico 2 Resumen análisis.

Nivel de madurez ALM



Fuente Microsoft (2016).

Como se puede observar con los ejemplos en la Tabla 4 y en el Gráfico 2, con esta herramienta (Visual Studio ALM Quick Reference Guide, 2015), es posible identificar el estado actual de cada uno de las áreas de ALM y compararlas con las directrices establecidas por Microsoft, de esta manera cumpliría con lo que sería un análisis de brechas. De igual manera se puede ver esta comparación de forma gráfica para un mejor y fácil entendimiento, con el fin de conocer cómo está la empresa actualmente comparada con el estado que se quiere lograr.

II.3. Estándares y políticas

Una de las herramientas de vital importancia cuando se habla de gestión de proyectos es un estándar y/o políticas para la gestión de estos. Según Kerzner (2009), el desarrollo de estándares representa un componente crítico para cualquier administrador de proyectos. Los estándares permiten: (a) construir un lenguaje común, (b) generar conocimiento y (c) organizar la información para su posterior análisis.

De acuerdo con el Institute of Electrical and Electronics Engineers (IEEE, 1963): “un estándar son publicaciones que establecen especificaciones y procedimientos destinados a garantizar la fiabilidad de los materiales, productos, métodos y/o servicios que la gente usa todos los días” (párr.2).

Los estándares se pueden clasificar en tres categorías, según Ruiz y Patiño (2014):

1) Estándares de facto (“de hecho” o “por convención”)

Los estándares de facto se subdividen en dos clases: Propietario y no propietario.

- Los estándares de propietario: “son aquellos originalmente inventados por una organización comercial como base para el funcionamiento de sus productos. Se llama de propietario porque son propiedad de la compañía que los inventó” (Ruiz y Patiño, 2014, párr.4).
- Los estándares no propietarios: “son aquellos originalmente desarrollados por grupos o comités que los han transferido al dominio público; también se llaman estándares abiertos porque abren las comunicaciones entre distintos sistemas” (Ruiz y Patiño, 2014, párr.4). En esta categoría califican la mayoría de estándares de *E-learning* existentes.

2) Estándares de jure (“por ley” o “por regulación”)

En relación con los estándares de jure, Ruiz y Patiño (2014), indican:

Los estándares de jure son aquellos que han sido legislados por un organismo oficialmente reconocido. Los estándares que no han sido aprobados por una organización reconocida, pero han sido adoptados como estándares por su amplio uso son estándares de facto. Los estándares de facto suelen ser establecidos a menudo por fabricantes que quieren definir la funcionalidad de un nuevo producto de tecnología (Ruiz y Patiño, 2014, párr.5).

3) Estándares de acuerdo

Los estándares de acuerdo “son aquellos que son definidos por convenio, alianza o pacto entre proveedores, usuarios, manufactureros entre otros” (Ruiz y Patiño, 2014, párr.6).

La importancia de crear estándares consiste en que permite a las organizaciones realizar actividades repetidas bajo un lenguaje común que ayude a los miembros del equipo a identificar los objetivos y las expectativas del proyecto de forma sencilla y coordinada (Ruiz y Patiño, 2014).

Según el Diccionario de la Real Academia Española, las políticas son:

“Orientaciones o directrices que rigen la actuación de una persona o entidad en un asunto o campo determinado” (2001, párr. 1). Por lo tanto, se puede notar que tanto un estándar como una política van a ser formas de guiar a las personas para que realicen una acción en específica. En el caso de este proyecto, se van a definir los pasos a seguir para que la ejecución de cada una de las áreas del ALM sea llevada a cabo de forma óptima, de manera que se obtenga el mayor beneficio para la empresa.

Para la ejecución de estos estándares y/o políticas muchas veces se deberán utilizar plantillas para que todas las personas lo hagan de una misma manera, por ello seguidamente se definirá este concepto.

II.3.1. Plantillas

De acuerdo con el PMBOK®, una plantilla es “un documento parcialmente completo en un formato predefinido, que proporciona una estructura definida para recopilar, organizar y presentar información y datos” (Project Management Institute, Inc, 2008, p.389).

Por lo tanto, las plantillas son utilizadas para facilitar y agilizar las labores de recopilación y de orden de la información. Además, es apropiado generar procedimientos e indicación acerca de cómo llenar las plantillas y con respecto a las reglas a seguir para documentar las plantillas de información.

El diseño y la selección de las plantillas de la metodología deben realizarse con sumo cuidado, ya que éstas deben estar enfocadas en guiar al usuario en la creación y el mantenimiento de los datos clave del proyecto de una manera formal y estructurada.

II.4. Information Technology Infrastructure

ITIL son las siglas de una metodología desarrollada a finales de los años ochenta por iniciativa del gobierno del Reino Unido, específicamente por la OGC u Oficina Gubernativa de Comercio Británica (Office of Government Commerce, 2007). Las siglas de ITIL significan Information Technology Infrastructure Library o Librería de Infraestructura de Tecnologías de Información.

ITIL se presenta como una guía y no un estándar, aun cuando está estrechamente alineado con el estándar ISO/IEC 20000. Al estar basado en “las mejores prácticas” de los principales proveedores de servicios, se le reconoce los siguientes beneficios (Office of Government Commerce, 2007):

- No es propietario: Al no basarse en ninguna tecnología o plataforma específicos, las prácticas en administración de servicios son aplicables a cualquier organización de TI.
- No es un precepto: Tiene aplicabilidad en todo tipo de organizaciones de servicio, grandes y pequeñas, públicas y privadas, con proveedores internos o externos.
- Basado en “mejores prácticas”: ITIL® representa las experiencias de aprendizaje y el liderazgo de pensamiento de los mejores proveedores de servicios de TI del mercado.
- Buenas prácticas: No todas las prácticas en ITIL® son consideradas las mejores, sino que ITIL® es un conjunto de prácticas habituales y buenas, lo cual, según los usuarios, es lo que le da sentido y lo hace útil.

El núcleo de ITIL está estructurado alrededor de un “ciclo de vida” de los servicios, este consiste en cinco fases: Estrategia del servicio, diseño del servicio, transición del servicio, operación del servicio y mejora continua del servicio. El ciclo de vida ordena las fases conforme un servicio avanza desde su concepción, pasando por la etapa de producción y hasta su retiro.

Este núcleo cuenta con cinco fases (para cada una existe un libro que la describe), según ITIL V3 (Office of Government Commerce, 2007):

- Estrategia del servicio (*service strategy*).
- Diseño del servicio (*service design*).
- Transición del servicio (*service transition*).
- Operación del servicio (*service operation*).
- Mejora continua del servicio (*continual service improvement*).

II.4.1. Objetivos de ITIL

- Promover la visión de TI como proveedor de servicios con el foco en el cliente, en lugar de la administración de tecnologías.
- Fomentar el alineamiento de la organización de TI con el negocio de la empresa, posicionándola como parte de la cadena de valor.
- Estandarizar los procesos de administración de TI y generar indicadores para Benchmarking.
- Promover el uso de conceptos comunes para los servicios de TI, para que estandaricen y mejoren la comunicación y el lenguaje entre las áreas de TI y para con el negocio.

ITIL como metodología propone el establecimiento de estándares que ayuden en el control, operación y administración de los recursos (ya sean propios o de los clientes). Igualmente, plantea hacer una revisión y reestructuración de los procesos existentes en caso de que estos lo necesiten (si el nivel de eficiencia es bajo o si hay una forma más eficiente de hacer las cosas), lo que lleva a una mejora continua.

Otra de sus propuestas es que cada actividad realizada se haga con la documentación pertinente, ya que esta puede ser de gran utilidad para otros miembros del área. Además, quedan asentados todos los movimientos realizados, lo cual permite que toda la gente esté al tanto de los cambios y no tome a nadie por sorpresa.

Finalmente, todas estas buenas prácticas que contiene ITIL van a servir de apoyo para este estudio, ya que algunas de sus fases tienen procesos que son acordes a los que Microsoft propone en su ALM.

II.5. Capability Maturity Model Integration

La mejora continua requiere de la evaluación de las capacidades de una organización contra una línea base, para luego determinar las brechas y constituir planes con el fin de subsanar las áreas con deficiencias. Los modelos de madurez proveen un instrumento objetivo para la identificación y validación del nivel de experiencia y disciplina que las organizaciones tienen con respecto a una metodología en particular (Becker, Knackstedt, y Pöppelbuß, 2009).

El modelo CMMI es uno de los adoptados para la medición de la madurez de las capacidades de desarrollo en una organización (Balani, 2008). Originalmente creado en el año 2000, por el SEI (Software Engineering Institute) de la Universidad de Carnegie Mellon, fue diseñado para diagnosticar y resolver problemas de desempeño en las empresas desarrolladoras de *software*. Actualmente, el CMMI se mercadea como una herramienta para medir el nivel de madurez de los procesos de cualquier empresa en cualquier industria (CMMI *Institute*, 2016).

Además, según la página de *Developer Network* de Microsoft (2016): La guía definitiva del modelo Capability Maturity Model Integration (CMMI) for Development está publicada por el Instituto de Ingeniería de Software como "*CMMI: Guidelines for Process Integration and Product Improvement*". Este libro describe específicamente CMMI para *Development* (CMMI-DEV) versión 1.3, que era uno de los modelos del conjunto actual del producto CMMI cuando se escribió este tema.

El modelo CMMI vio la luz en 1987 como Capability Maturity Model (CMM), un proyecto del Software Engineering Institute, que es un centro de investigación de la Universidad Carnegie-Mellon. Este centro lo fundó y lo financia el Departamento de Defensa de los Estados Unidos. En 1991, se publicó por primera vez el modelo CMM for software, que está basado en una lista de comprobación de los principales factores de éxito de los proyectos de desarrollo de *software* realizados a finales de los años setenta y principios de los años ochenta.

Aplicado principalmente a programas de defensa, el modelo CMM ha logrado una aceptación considerable y se ha sometido a varias revisiones e iteraciones. Su éxito condujo al desarrollo de modelos CMM para diversos ámbitos más allá del ámbito de *software*.

Lo más importante que debe saberse de CMMI-DEV es que se trata de un modelo. No se trata de un proceso ni de una prescripción que deba seguirse, sino de un conjunto de comportamientos organizativos de méritos demostrados en el marco del desarrollo de *software* y la ingeniería de sistemas.

Además de la información anterior, el sitio web <https://msdn.microsoft.com/es-es/library/ee461556.aspx> de Microsoft, 2016 indica que el modelo proporciona un lenguaje común y su propósito es evaluar la madurez de la organización, así como proporcionar orientación con respecto a cómo mejorar los procesos. Igualmente, delimita cinco niveles de madurez organizativos: Inicial, administrado, definido, administrado cuantitativamente y optimizado.

Con este modelo, al igual que con ITIL, se pueden generar mejoras para cada una de las áreas de ALM, por lo que es de gran importancia tener conocimiento de estas prácticas pues podría significar generar soluciones de gran utilidad para la organización.

II.6. Pruebas de *software*

Una de las propuestas brindadas por medio de este estudio consiste en las pruebas de *software* que se hacen actualmente en la empresa, para lo cual se debe tener un mayor conocimiento sobre este tema.

Según Abran et al (2004), hacer pruebas es una actividad que tiene el objetivo de evaluar y mejorar la calidad del producto, identificando defectos y problemas.

Además, según Abran et al (2004): “Las pruebas del *software* consisten en verificar el comportamiento de un programa dinámicamente a través de un grupo finito de casos de prueba, debidamente seleccionados del, típicamente, ámbito de ejecuciones infinito, en relación con el comportamiento esperado” (p.73). En la definición anterior las palabras subrayadas corresponden con aspectos esenciales en la identificación del “Área de conocimiento de las pruebas del *software*”. En particular:

Dinámicamente: Este término significa que hacer pruebas siempre supone ejecutar el programa con entrada de datos (valorados). Para precisar, es necesario afirmar que la entrada de valores no es siempre suficiente para definir una prueba, dado que un sistema complejo y no determinista podría tener diferentes comportamientos con las mismas entradas de datos, dependiendo del estado en el que se encuentre. En cualquier caso, se mantendrá el término de “entrada de datos”, asumiendo la convención de que el término incluye un estado del sistema específico, en los casos en que sea necesario (Abran, 2004, p.73).

Finito: Incluso en programas sencillos, teóricamente podría haber tantas pruebas que realizar, que hacer pruebas exhaustivas podría llevar meses o años. Esta es la razón por la que en la práctica el grupo completo de pruebas se podría considerar infinito. Hacer pruebas siempre supone un compromiso entre recursos y calendarios de trabajo limitados, por un lado, y necesidades inherentes de pruebas ilimitadas, por otro (Abran, 2004, p.73).

Seleccionados: La diferencia esencial entre las distintas técnicas de pruebas propuestas se encuentra en cómo se escoge el conjunto de pruebas. Los ingenieros informáticos deben ser conscientes de que criterios de selección distintos pueden producir grados de efectividad muy diferentes. La forma de identificar el criterio de selección de pruebas más apropiado para un conjunto de condiciones particulares es un problema complejo; en la práctica se usa la experiencia en el diseño de pruebas y técnicas de análisis de riesgo (Abran, 2004, p.73).

Esperado: Debería ser posible, aunque a veces no sea fácil, decidir si el resultado observado de la ejecución de un programa es aceptable o no, porque si no el esfuerzo de realizar las pruebas sería inútil. El comportamiento observado se puede comprobar con los resultados esperados por el usuario (normalmente conocido como pruebas de validación), con las especificaciones (pruebas de verificación), o, finalmente, con el comportamiento anticipado de requerimientos implícitos o expectativas razonables (Abran, 2004, p.73).

II.6.1. Tipos de pruebas

Existen diferentes tipos de pruebas, entre ellas algunas son más adecuadas para diferentes escenarios que otros. Según Abran et al (2004), existen los siguientes tipos de pruebas:

Pruebas de aceptación/calificación: Las pruebas de aceptación comparan el comportamiento del sistema con los requisitos del cliente, sea cual sea la forma en que estos se hayan expresado. El cliente realiza, o especifica, tareas típicas para comprobar que se satisfacen sus requisitos o que la organización los ha identificado para el mercado al que se destina el software. Esta actividad de pruebas puede incluir o no a los desarrolladores del sistema (Abran, 2004, p.77).

Pruebas de instalación: Normalmente, cuando las pruebas de aceptación han terminado, el software se puede comprobar una vez instalado en el entorno final. Las pruebas de instalación se pueden ver como pruebas del sistema realizadas en relación con los requisitos de la configuración de hardware. Los procedimientos para la instalación también se podrían verificar (Abran, 2004, p.77).

Pruebas alfa y beta: A veces, antes de poner el software en distribución, este se proporciona a un grupo representativo de usuarios potenciales para que puedan usarlo en pruebas en las instalaciones del desarrollador (pruebas alfa) o externamente (pruebas beta). Dichos usuarios notifican problemas con el producto. Normalmente, el uso de versiones alfa y beta sucede en entornos no controlados y no siempre se le hace referencia en los planes de pruebas (Abran, 2004, p.77).

Pruebas de conformidad/pruebas funcionales/pruebas de corrección: Las pruebas de conformidad tienen el objetivo de verificar si el comportamiento del software se corresponde con las especificaciones (Abran, 2004, p.77).

Materialización de la confiabilidad y evaluación: Las pruebas, al ayudar a identificar errores, son un medio para mejorar la confiabilidad. Por contraste, al generar casos de prueba aleatorios y al seguir el perfil de operaciones, se pueden derivar aproximaciones estadísticas de confiabilidad. Cuando se usan modelos que potencian la confiabilidad, ambos objetivos se pueden alcanzar al mismo tiempo (Abran, 2004, p.77).

Pruebas de regresión: Las pruebas de regresión son pruebas selectivas que se repiten en un componente para verificar que los cambios no han producido efectos indeseados. En la práctica, la idea es demostrar que cierto software que previamente pasó un conjunto de pruebas, aún las pasa. Beizer (1990) las define como cualquier repetición de pruebas que tiene como objetivo demostrar que el comportamiento del software no ha cambiado, excepto en aquellos aspectos en que se haya requerido así. Por supuesto se tiene que llegar a un compromiso entre realizar pruebas de regresión cada vez que se hace un cambio y los medios de que se dispone para realizar las pruebas (Abran, 2004, p.77).

Pruebas de rendimiento: Estas pruebas tienen el objetivo de verificar que el software alcanza los requerimientos de rendimiento especificados, particularmente los de capacidad y tiempo de respuesta. Un tipo particular de pruebas de rendimiento son las pruebas de volumen, en ellas las limitaciones internas del programa o sistema se ponen a prueba (Abran, 2004, p.77).

Pruebas de desgaste: Las pruebas de desgaste hacen funcionar el software a la máxima capacidad para la que fue diseñado y por encima de ella (Abran, 2004, p.77).

Pruebas de continuidad: Un grupo de pruebas se ejecuta en dos versiones diferentes de un producto software y los resultados se comparan (Abran, 2004, p.77).

Pruebas de recuperación: El objetivo de las pruebas de recuperación es verificar la capacidad del software para reiniciarse después de un desastre (Abran, 2004, p.77).

Pruebas de configuración: En los casos en los que el software se construye para dar servicio a distintos usuarios, las pruebas de configuración analizan el software en las diferentes configuraciones especificadas (Abran, 2004, p.78).

Pruebas de facilidad de uso: Este proceso evalúa lo fácil que le resulta usar y aprender a usar el software al usuario, incluyendo la documentación del usuario, la efectividad de las funciones del software para soportar las tareas de usuario y, finalmente, la habilidad de recuperarse de errores provocados por el usuario (Abran, 2004, p.78).

II.7. Manejo de fuentes de productos de *software*

Para este estudio, el manejo de fuentes de interés es el relacionado con las diferentes estrategias de ramificaciones y la forma de controlar las versiones del código fuentes.

Según Microsoft (2016) el ramificado permite el desarrollo en paralelo, esto ofrece a cada actividad de desarrollo un panorama de los archivos de código fuente necesarios, de las herramientas, de las dependencias externas y de los procesos automáticos. Tener ramificaciones aumenta la complejidad y el coste de las uniones. Sin embargo, hay varios escenarios en los que deberá considerarse mantener varias ramificaciones para conservar cierta velocidad de desarrollo. Ya sea para aislar o estabilizar temporalmente un cambio importante, desarrollar una funcionalidad o comenzar el desarrollo de una versión futura, se debería considerar lo siguiente:

- Hacer una ramificación desde un padre que tenga los últimos cambios (normalmente principal u otra ramificación de DEV) (Microsoft, 2016).
- Hacer una ramificación de todo lo necesario para desarrollar en paralelo, normalmente esto significa hacer una ramificación completa del padre (Microsoft, 2016).
- Hacer una unión desde la rama padre (FI) frecuentemente. Siempre FI (*forward integrate*) antes que RI (*reverse integrated*). Se debe hacer FI tantas veces como tenga sentido. Semanalmente es un buen punto de partida (Microsoft, 2016).
- Es preciso asegurarse de compilar y ejecutar las suficientes pruebas de verificación de *Build* (BVT, en inglés *Build Verification Tests*) para medir la estabilidad de la rama (Microsoft, 2016).
- Para estabilizar cambios, conservar los cambios, hacer un *forward integrate* (FI), compilar y pasar todos los BVT antes de hacer un *merge* (RI) con los cambios al padre (Microsoft, 2016).

- Un *merge* (RI) a principal es como una *feature release* en equipos de desarrollo. Una vez que la rama hace *merge* (RI), otros equipos usarán esa versión hasta la próxima unión (RI) (Microsoft, 2016).
- Cuando se hace una ramificación de *main* para *release* (versión actual), las correcciones que hacen que no se pueda liberar la versión se deben hacer en la rama *release*. La ramificación de nuevas características es solo para el trabajo de la próxima versión. En este ejemplo, cuando se hace el *branch* de principal para la próxima *release*, todas las ramificaciones de DEV pueden comenzar a trabajar para la próxima versión (Microsoft, 2016).

Además, según Microsoft (2016) es posible organizar las ramas (*branches*) en tres categorías: Principal (*main*), desarrollo (*development*) y liberación (*release*). Dependiendo del tipo de *branch*, es imperativo tener presente las siguientes consideraciones:

- Compilarlas regularmente (continuamente) ofrece una cadencia regular y permite identificar rápidamente problemas de calidad (Microsoft, 2016).
- Los movimientos de cambios producidos entre los *branch* padres e hijos deben ser entendidos y tenidos en cuenta por parte de todos los miembros del equipo (Microsoft, 2016).

En la Tabla 5, se muestra una lista de otras consideraciones por tener en cuenta de los tipos de *branch* mencionados antes.

Tabla 5 Consideraciones de branch.

Tipo de Branch	Consideraciones
Principal	<ul style="list-style-type: none"> • Los <i>branch</i> de principal y liberación están a un <i>merge</i> de distancia de principal. • El <i>branch</i> de la “verdad”, que debe ser “<i>buildable</i>”, cumple con unos mínimos de calidad y es una fuente de confianza para los equipos de aseguramiento de calidad. • Salvo en el caso de la estrategia <i>Main Only</i>, hay que evitar hacer cambios en principal. • A medida que aumenta el número de ramificaciones de desarrollo, el coste y la necesidad de <i>merge</i> (FI) tras un principal exitoso aumentan.
Desarrollo	<ul style="list-style-type: none"> • Todas las ramas son áreas auto contenidas (aisladas) que permiten a cada actividad de desarrollo avanzar, sin depender unas de otras. • Es preciso basar una ramificación DEV en un padre con un estado conocido y bueno, normalmente el principal. • Haz <i>merges</i> (FI) frecuentemente para reducir la complejidad del tipo “<i>big bang</i>”. • Se puede hacer uniones (FI) opcionalmente cada vez que la rama padre sea construida y pase los BVTs, pero esto suele conllevar una gran recarga de trabajo.
Liberación	<ul style="list-style-type: none"> • Las ramificaciones de liberación deben soportar el medio de liberación. • Los medios de liberación más comunes son <i>major release</i>, <i>hotfix</i> y <i>service pack</i>. • Como con todas las ramificaciones en general, ¡menos es mejor! • La relación padre/hijo entre <i>MAIN</i>→<i>SP</i>→<i>HOTFIX</i> permiten la unión de cambios en liberaciones futuras (por ejemplo: Los <i>hotfixes</i> se unen en la rama SP en su camino a <i>MAIN</i>), esto reduce el riesgo de “pulgas” de regresión en próximas liberaciones.

Fuente: Elaboración propia, 2016.

II.7.1 Estrategias de *branching*

II.7.1.1. Main Only

La estrategia Main Only puede estar basada en directorios o con el directorio principal promovido a una rama para soportar las características de control de versiones que ofrecen una visibilidad extra.

Según Microsoft (2016), esta estrategia es el recomendado punto de partida para cualquier equipo que no ha necesitado el aislamiento de código usando ramificaciones y cuando no hay problemas con las etiquetas mutables para marcar los hitos, en la única rama.

En la Ilustración 2, se detalla la estrategia Main Only.

Ilustración 2 Estrategia de branching Main Only.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.2. Development Isolation

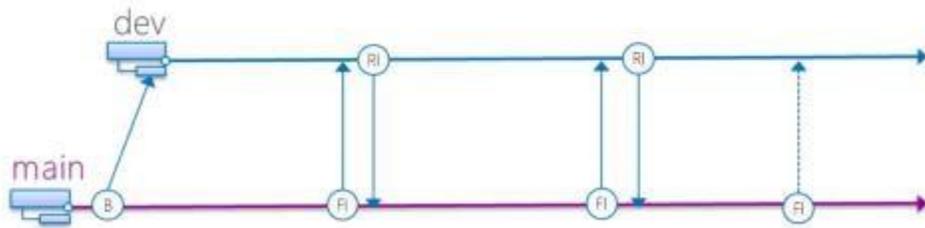
La estrategia de Development Isolation añade uno o más ramificaciones de desarrollo desde la principal, lo que permite el desarrollo concurrente para la próxima liberación, experimentos o resolución de *bugs* en ramificaciones aisladas.

Según Microsoft (2016) esta estrategia se usa cuando:

- Se necesite desarrollo aislado y concurrente, con el fin de proteger la rama estable principal.
- Se necesite solo una *major release*, soportada y entregada al cliente usando la ramificación principal.
- Se necesite un modelo de servicios para que sus clientes se actualicen a la próxima versión, por ejemplo: v1 → v2.

En la Ilustración 3, se detalla la estrategia Development Isolation.

Ilustración 3 Estrategia de branching Development Isolation.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.3. Release Isolation

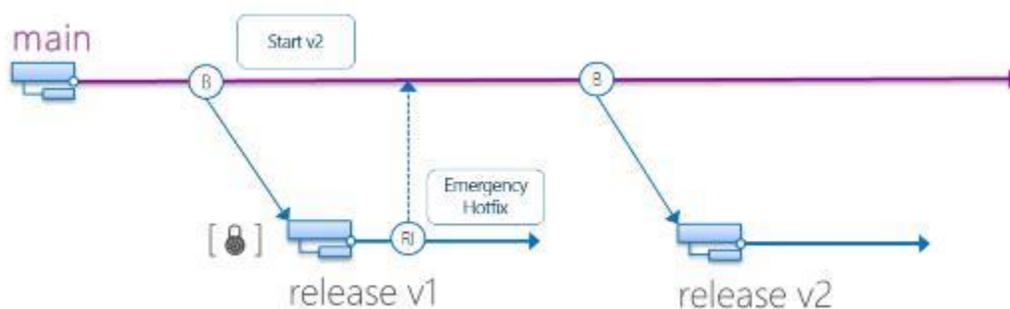
La estrategia Release Isolation incluye una o más ramificaciones de liberación desde principal, permitiendo así la gestión concurrente de liberaciones.

Según Microsoft (2016), esta estrategia se usa cuando:

- Se necesite aislamiento y liberaciones concurrentes, manteniendo la rama principal estable.
- Se necesite varias liberaciones en paralelo para soportar y entregar a los clientes la ramificación de liberación adecuada.
- Se necesite un modelo de servicios para que los clientes se actualicen a la próxima liberación, por ejemplo: v1 → v2.
- Se necesite un panorama del código fuente en tiempo de liberación.

En la Ilustración 4 se detalla la estrategia de Release Isolation.

Ilustración 4 Estrategia de branching Release Isolation.

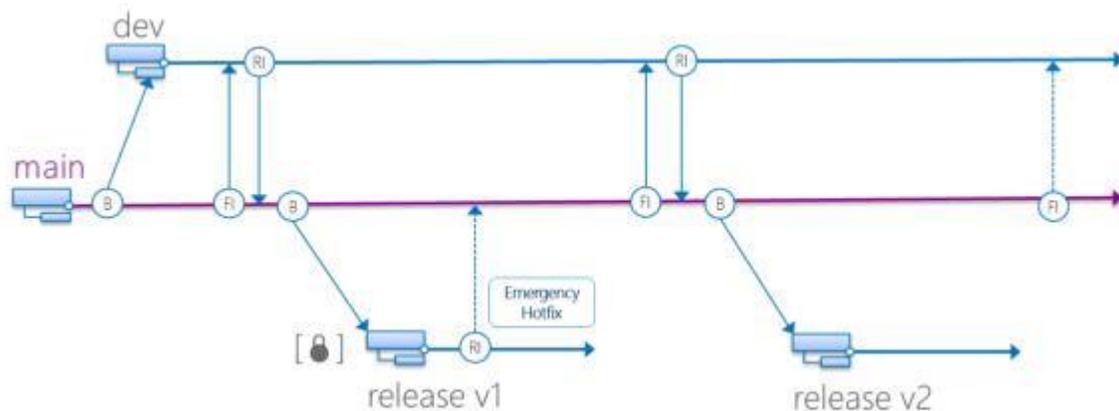


Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.4. Development y Release Isolation

La estrategia de Development y Release Isolation combina las estrategias de Development Isolation y Release Isolation, detalladas en la Ilustración 5 y, por lo tanto, se van a usar en los casos ya descritos.

Ilustración 5 Estrategia de branching Development y Release Isolation.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.5. Servicing y Release Isolation

La estrategia Servicing y Release Isolation añade ramificaciones de servicio, estas permiten la gestión concurrente de “pulgas” y *service packs*. Esta estrategia es una evolución de la estrategia de Development y Release Isolation.

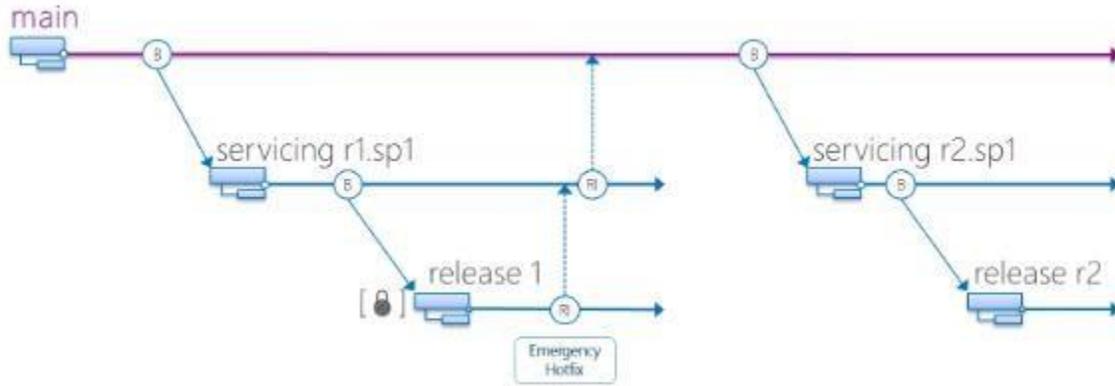
Según Microsoft (2016), esta estrategia se usa cuando:

- Se necesite liberaciones aisladas y concurrentes, manteniendo en un estado estable la rama principal.
- Se tengan varias *major releases*, permitiendo dar soporte y entregar a clientes la rama adecuada.
- Se necesite un modelo de servicios para que los clientes se actualicen a la siguiente *major release*, por ejemplo: $v1 \rightarrow v2$.
- Se necesite un modelo de servicios para que los clientes se actualicen a *service packs* adicionales, por ejemplo: $v1 \text{ SP1} \rightarrow v1 \text{ SP2}$.

- Se requieran panoramas del código en tiempo de liberación y servicio.

En la Ilustración 6 se detalla la estrategia Servicing y Release Isolation

Ilustración 6 Estrategia de branching Servicing y Release Isolation.

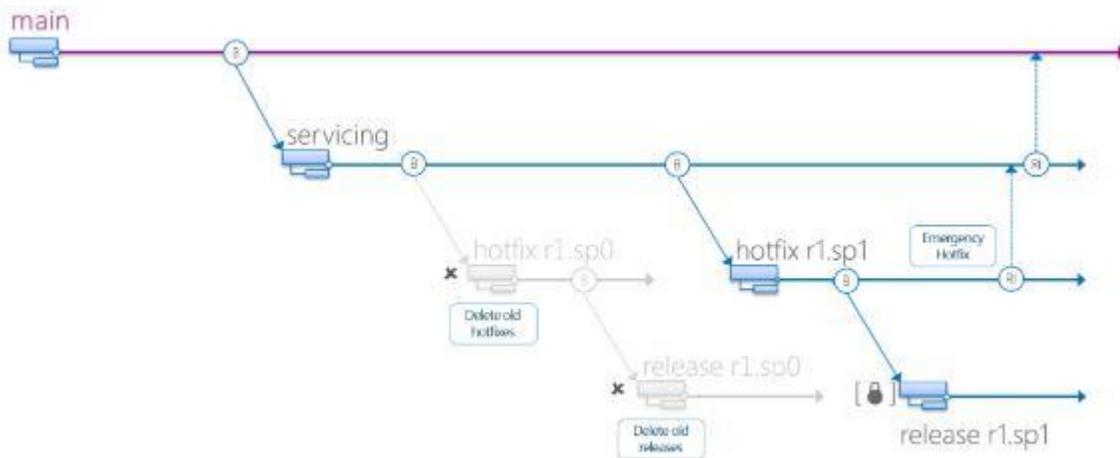


Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.6. Servicing, Hotfix, y Release Isolation

Es posible evolucionar la estrategia de Servicing y Release Isolation añadiendo ramificaciones adicionales para *hotfix*, de esta manera se consigue el soporte necesario para tener varios medios de liberación y escenarios de servicios, esta estrategia se detalla en la Ilustración 7.

Ilustración 7 Estrategia de branching Servicing, hotfix y release.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.7. Code Promotion

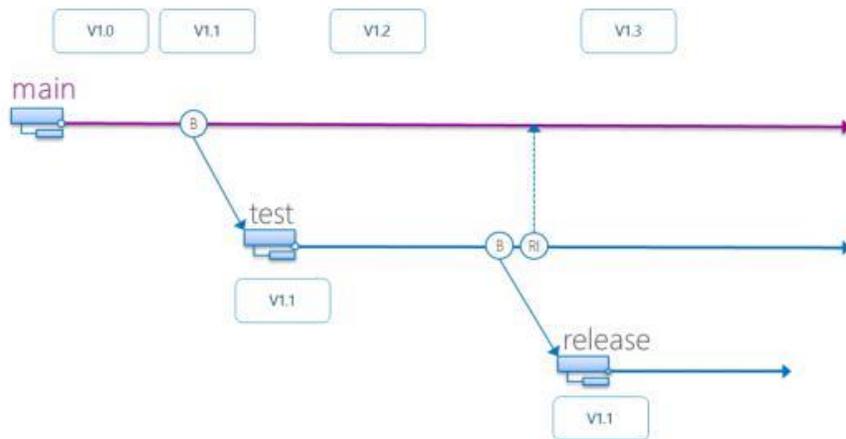
El plan Code Promotion promueve versiones del código con base en nuevos niveles a medida que se vuelve estable. Existen sistemas de control de versiones que implementan técnicas similares usando propiedades de archivo que muestran en qué nivel de promoción se está. A medida que el código de la rama principal se vuelve más estable, se promueve a la rama de pruebas (también conocido como rama de QA), allí se ejecutan todas las pruebas de regresión y pueden corregirse “pulgas” de manera adicional. Una vez que el código está listo para una liberación en la rama de pruebas, se une en la rama de producción, donde se realiza la certificación de calidad final y se prueba. El desarrollo concurrente puede realizarse en la rama principal, la cual promueve el concepto de “código no congelado”.

Según Microsoft (2016), esta estrategia se usa cuando:

- Se cuenta únicamente con una *major release*, soportada y entregada a los clientes desde la rama principal.
- Se tienen largos ciclos de prueba.
- Es preciso evitar código congelado y se desea tener aislada la *major release*.

La Ilustración 8 detalla esta estrategia.

Ilustración 8 Estrategia de branching Code promotion.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.7.1.8. Feature Isolation

La estrategia Feature Isolation añade una o más ramificaciones de funcionalidad desde la rama principal, esto permite el desarrollo concurrente de funcionalidades definidas claramente para la próxima liberación. La estrategia Feature Isolation es una evolución especial de la estrategia Development Isolation, esencialmente con dos o más ramas de desarrollo (funcionalidad).

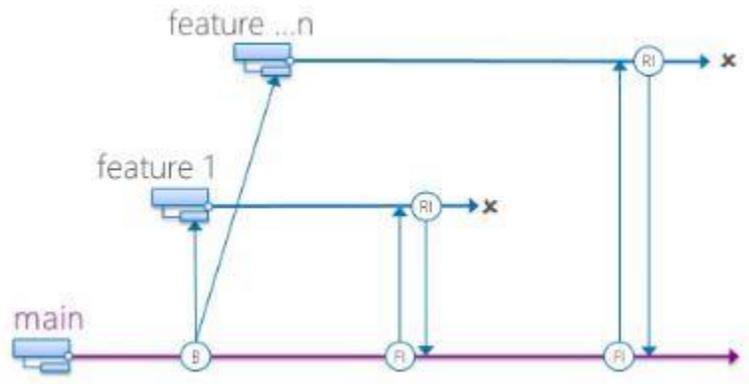
Según Microsoft (2016), con la estrategia de Feature Isolation es posible aislar cada funcionalidad en su propia rama, esto permite una gran flexibilidad en términos de cuándo mover una liberación a producción. Esta estrategia también ayuda a evitar escenarios en los que una ramificación necesite ser publicada con las funcionalidades terminadas en una rama, mientras que las funcionalidades que no estén terminadas se quedan en sus ramas. Unir funcionalidades a la rama principal, a medida que se vuelven estables para una liberación, borra y crea nuevas ramificaciones de funcionalidad a medida que son necesarias.

Esta estrategia se usa según Microsoft (2016) cuando:

- Se requiere asilamiento y desarrollo concurrente de funcionalidades bien definidas, con el fin de proteger a la rama principal.
- Se cuenta con equipos separados desarrollando varias funcionalidades para un mismo proyecto.
- Se esté creando un producto grande con muchas características que necesitan aislarse.
- Se desarrollan funcionalidades en paralelo que no deben estar en el mismo ciclo de liberación.
- Se necesite la habilidad de quitar funcionalidades de una liberación. Eliminar una funcionalidad puede ser costoso y cuando sea necesario hay que hacerlo con cuidado.

Con la Ilustración 9 se ejemplifica esta estrategia.

Ilustración 9 Estrategia de branching Feature Isolation.



Fuente: Microsoft (Use Team Foundation Version Control, 2016).

II.8. Team Foundation Server

Para este estudio se menciona el tema Team Foundation Server (TFS), por lo que es importante tener claro en que consiste.

Según Microsoft (2016), Team Foundation Service es un servicio que proporciona la herramienta de gestión de ciclo de vida de aplicaciones TFS bajo la modalidad de SaaS (*Software as a Service*). Esto quiere decir que, para obtener beneficios de las ventajas de usar esta herramienta, no será necesario pagar por diferentes licencias, ni montar una propia base de servidores o pasar por una serie de tediosos procesos de instalación. Bastará con darse de alta en una página web, y así se tendrán al alcance los beneficios de esta herramienta de gestión del ciclo de vida de las aplicaciones.

Características principales

Según Microsoft (2016), las principales características de TFS son:

- Gestión del equipo y su trabajo.
- Control de código fuente.
- Automatización de *builds*, pruebas y despliegues.

Otro punto por tener en cuenta es que TFS posee otros elementos, por ejemplo:

- Un control de versiones de código fuente, en formato centralizado o distribuido con GIT.
- Un servidor de integración continua (*build server*).
- Un registro de incidencias (*bug tracker*).
- Gestión de proyectos AGILE.
- Varios *dashboards* para gestionar *user stories* o *PBIs*, tareas, entre otros.
- Integración nativa con Microsoft Office.
- Gestión de pruebas y casos de pruebas automatizados.
- Integración con entornos no Microsoft como Eclipse, Cobol, etc.

Shelves

Uno de los términos que se utilizan en TFS es *shelves*. Para Microsoft (2016), los *shelvesets* consisten en un conjunto de cambios aplazados o pendientes, que se utilizan para cualquiera de los siguientes fines:

- Interrupción: Cuando se tienen cambios pendientes que no están listos para el *checkin*, pero hay que trabajar en una tarea diferente.
- Colaboración: Cuando se tienen cambios pendientes que no están listos para el *checkin* pero hay que compartirlos con otros miembros de equipo.
- Revisión código: Cuando se desea que otro miembro del equipo realice una revisión de su código.
- Compilaciones privadas: Antes de realizar un *checkin* de sus cambios, puede utilizar los *builds* automáticos para compilar y probar su código.
- Copia de seguridad: Cuando se tiene trabajo en curso que no se puede terminar ahora, por lo que desea una copia de seguridad que se almacena en el servidor y está a disposición de otros miembros de equipo que podrían necesitar de ella.
- Traspaso: Cuando se tiene trabajo en curso que desea entregar a otro miembro de equipo.

III MARCO METODOLÓGICO

El siguiente capítulo tiene como finalidad establecer el procedimiento a seguir para cumplir cada uno de los objetivos del proyecto. Así mismo, se va a definir el tipo de investigación a realizar, las fuentes de información que se consultarán, técnicas para obtener la información y la forma de procesamiento de los datos recopilados, con el fin de realizar un análisis de estos y proponer métodos, alternativas o soluciones que permitan a la organización resolver el problema en cuestión.

III.1. Tipo de investigación

La investigación aplicada, también conocida como investigación práctica, se realiza con fines prácticos, tanto para resolver un problema como para tomar decisiones, evaluar programas y, en general, para mejorar un producto o proceso (Venegas, 1986).

A su vez, es una investigación descriptiva porque busca especificar las propiedades, características y rasgos importantes del fenómeno deseable para analizar. Así mismo, recolecta datos sobre diversos aspectos permisivos con respecto a la descripción de lo investigado. Por tanto, es de fundamental importancia definir qué va a investigarse y de quiénes se obtendrá la información (Hernández, Fernández, y Baptista, 2006)

Mediante la aplicación de este tipo de metodología de investigación, y después de analizar los datos recopilados producto de la revisión documental, cuestionarios de madurez y diferentes sesiones, se determinará las brechas existentes en la organización en materia de los procesos actuales para determinar el ciclo de vida de las aplicaciones, así como el marco de referencia por utilizar y los requerimientos para la elaboración de la guía metodológica y/o plantillas necesarias para completar el alcance de este proyecto.

III.2. Sujetos y fuentes de Información

Las fuentes y sujetos de información serán todos aquellos puntos de referencia de los cuales se extraerán los datos de la investigación llevada a cabo.

III.2.1. Sujetos de información

Sabino (1978) se refiere a las personas físicas que de una u otra forma brindan información para desarrollar de la mejor manera esta indagación, para determinar las causas y posibles soluciones del problema planteado.

Dentro de los sujetos de información para el presente estudio, se utilizarán testimonios de colaboradores del centro de operaciones, específicamente aquellos relacionados, principalmente, con la formulación de la estrategia y a su rol protagónico en la gestión del ciclo de vida de las aplicaciones.

III.2.1.1. Población

Según Arias (2006), se entiende por población el conjunto finito o infinito de elementos con características comunes, para los cuales serán extensivas las conclusiones de la investigación. Esta queda limitada por el problema y por los objetivos del estudio. Es decir, se utilizará un conjunto de personas con características comunes que serán objeto de estudio.

Para este proyecto la población comprende a todos aquellos colaboradores de la empresa que pertenezcan al centro de operaciones de la misma y que tengan más de un año de trabajar para el departamento, con el fin de que tengan claro como es el funcionamiento actual del centro de operaciones.

El centro de operaciones cuenta con 31 colaboradores con más de un año laborando directamente con el departamento, esta información fue conocida gracias la empresa donde se realizará el proyecto. La población del proyecto serán estos 31 colaboradores.

III.2.1.2. Muestra

Según Morales (1994), la muestra es un "subconjunto representativo de un universo o población.

Para este proyecto el tipo de muestra que se utilizó fue la intencional, según lo define Arias (2006), el muestreo intencional u opinático, es aquel donde los elementos muestrales son escogidos con base a criterios o juicios preestablecidos por el investigador.

Por lo tanto, los sujetos de la muestra para este proyecto serán:

- Director del centro de operaciones: Es la jefatura máxima del departamento del centro de operaciones, encargado de asignar y controlar tareas, por lo que es esencial que se someta a estudio ya que conoce todo el entorno del centro de operaciones.
- Administrador e integrador de aplicaciones: Encargado de integrar las aplicaciones de la empresa en los ambientes del cliente, además es el encargado de controlar las fuentes de código fuente y de hacer estrategias y estándares para su documentación. Es importante esta persona para conocer todas las estrategias que el centro de operaciones tiene a nivel técnico.
- 10 ingenieros de software: Encargados de desarrollar los sistemas o requerimientos de los clientes, que al fin de cuentas son los que van a utilizar las propuestas de este proyecto.

III.2.2. Fuentes de información

Las fuentes de información son los lugares a partir de los cuales se obtienen los datos requeridos para la investigación. Estos datos son todos aquellos antecedentes o fundamentos necesarios para llegar a un conocimiento exacto del objeto de estudio. Dichos datos serán suficientes para sustentar y defender el trabajo de investigación desarrollado (Sabino, 1978).

Para la elaboración del presente trabajo se utilizarán las siguientes fuentes de información primaria y secundaria.

III.2.2.1. Fuentes primarias

Dentro de las fuentes primarias del presente trabajo de investigación se listan las siguientes:

- a. Libros impresos relacionados con la administración de proyectos e ingeniería de *software* y gestión del ciclo de vida de las aplicaciones de bibliotecas nacionales como la ubicada en el ITCR y la UCR.
- b. Regulaciones impresas sobre buenas prácticas del ciclo de vida de las aplicaciones.
- c. Artículos actualizados sobre metodologías y/ o plantillas que se ajusten al ciclo de vida de las aplicaciones.
- d. Materiales publicados por Microsoft y sus expertos en el tema de ALM (Visual Studio ALM Quick Reference Guide).
- e. Documentación, plantillas y técnicas utilizadas en lo interno de la organización.
- f. Guía de los Fundamentos para la Dirección de Proyectos. (Guía del PMBoK®, quinta edición, año 2013).
- g. Buenas practicas propuestas por ITIL V3.

III.2.2.2. Fuentes secundarias

Para las fuentes secundarias, también se toman en cuenta los criterios administrativos que circulan en torno al tema del ciclo de vida de las aplicaciones y que intervienen en la toma de decisiones de estos.

Para efectos de este estudio se consultan las siguientes fuentes secundarias:

- a. Información brindada por los sujetos de información, esta involucra experiencia en proyectos históricos en cuanto al ciclo de vida de las aplicaciones, plantillas y/o estándares para ALM y procedimientos realizados.
- b. Información de sitios en internet.
- c. Repositorios de trabajos finales de graduación de la biblioteca del ITCR.
- d. Página web de Microsoft y Team Foundation Server.

III.3. Técnicas de investigación

Se realizará un análisis por cada objetivo y se seleccionarán las herramientas consideradas como adecuadas para satisfacer el cumplimiento de dichos objetivos.

En algunos casos será necesario utilizar más de una herramienta o técnica de investigación. En la Tabla 6 se detalla las técnicas y/o herramientas utilizadas para cada uno de los objetivos del proyecto.

Tabla 6 Técnicas de investigación.

Objetivos	Herramientas o técnicas de investigación
Realizar un análisis de brechas entre el estado actual del centro de operaciones y el estado que se quiere alcanzar, basado en los niveles de madurez de ALM propuestos por Microsoft.	<ul style="list-style-type: none">• Cuestionarios de los procesos individuales de TI. Se pueden ver desde el Apéndice 1 al Apéndice 8 y se aplicará lo detallado en la sección III.3.1, los cuales se van a

Objetivos	Herramientas o técnicas de investigación
	<p>implementar mediante discusiones grupales según lo expuesto en la sección III.3.3.</p> <ul style="list-style-type: none"> • Gráfico radial o diagrama araña. Se puede ver su formato en el Anexo 2. • Cuadro de situación actual contra situación deseada, como se muestra en la Tabla 4. Esta tabla estará conformada por una fila por cada una de las áreas de ALM, y las columnas serán el estado actual y el estado deseado, además de una columna de estado, en la cual se coloca el icono correspondiente según lo detallado en la Tabla 4.
<p>Proponer un plan de trabajo para lograr el estado deseado del centro de operaciones, a partir de los resultados del análisis de brechas.</p>	<ul style="list-style-type: none"> • Juicio de expertos, los cuales se documentarán según lo explicado en la sección III.3.2. • Priorizar mejoras, según lo expuesto en la sección III.3.1. y según las variables detalladas en la sección
<p>Emplear estándares y políticas de los procesos asociados a la administración del ciclo de vida de las aplicaciones del centro de operaciones.</p>	<ul style="list-style-type: none"> • Revisión documental, según lo que se expresa en el título III.3.1. • Juicio de expertos, los cuales se documentarán según lo explicado en el apartado III.3.2.

Fuente: Elaboración propia, 2016.

III.3.1. Cuestionarios

Para los cuestionarios de la presente investigación (Apéndice 1 al Apéndice 8) es preciso retomar que existen diferentes niveles de madurez de ALM, los cuales se describen en la sección II.1.1.2, por lo tanto, cuando se hable de algún nivel de madurez, se estará haciendo referencia a estos. Además, como tabla de puntaje, cada nivel tendrá el valor asignado de acuerdo a la Tabla 7; es importante destacar que el puntaje se otorgó basándose en los 4 niveles de madurez de ALM y que cada uno de ellos ocupa una jerarquía mayor en este mismo, tomando como base el número diez y un máximo de cuarenta, para que al aplicar la fórmula (1) con estos números se obtenga un resultado en base diez.

Tabla 7 Puntaje de niveles.

Nivel	Puntaje
Básico	10
Estandarizado	20
Avanzado	30
Dinámico	40

Fuente: Elaboración propia, 2016.

Unido a lo anterior, cada una de las respuestas de los cuestionarios van a tener también su puntaje, en la Tabla 8 se detalla este.

Tabla 8 Puntaje de respuestas.

Nivel	Puntaje
Nunca	1
Algunas veces	2
Frecuentemente	3
Siempre	4

Fuente: Elaboración propia, 2016.

Adicionalmente, para obtener el puntaje total de cada cuestionario, se va aplicar la siguiente fórmula:

$$Puntaje\ total = Redondear \left(\frac{\sum Puntaje\ por\ respuesta}{Cantidad\ de\ preguntas} \right) * 10 \quad (1)$$

La implementación de la fórmula (1) consiste en obtener la suma total de los puntos obtenidos por respuesta y dividir esta cantidad entre el total de preguntas del cuestionario, este resultado se redondea a un decimal y el valor obtenido se multiplica por 10, para obtener un dígito con base 10, con el fin de que concuerde con los puntajes por nivel detallados en la Tabla 7.

Para la aplicación de los cuestionarios que se encuentran del Apéndice 1 al Apéndice 8, cabe resaltar la forma en que se llevarán a cabo los mismos, se desarrollarán una o varias discusiones grupales por cuestionario, la forma de proceder con las discusiones grupales se explicará en el apartado III.3.3.

III.3.2. Juicio de Expertos.

Para la realización de esta investigación se utilizará la técnica de juicios de expertos en la cual el estudiante propondrá sus ideas y someterá su validez a los expertos, para este proyecto esta técnica se someterá a consideración y análisis de 2 personas claves, la primera persona se trata del director del centro de operaciones el cual según la sección III.2.1 es la jefatura máxima del departamento del centro de operaciones, encargado de asignar y controlar tareas, además de poseer un gran conocimiento en el área del proyecto.

La segunda persona es el administrador e integrador de aplicaciones el cual según la caracterización descrita en la sección III.2.1 es el encargado de integrar las aplicaciones de la empresa en los ambientes del cliente, además es el encargado de controlar las fuentes de código fuente y de hacer estrategias y estándares para su documentación, unido a esto tiene gran conocimiento sobre el tema de ALM y posee certificaciones en el tema.

Estas dos personas serán las seleccionadas con el fin de comprobar criterios, ideas y propuestas y que estas sean ajustadas a lo que el centro de operaciones necesita. En el Apéndice 10, Apéndice 11 y Apéndice 12 , se encuentran la herramienta que se utilizarán para la validación por medio de juicio de experto.

III.3.3. Discusiones grupales.

Las discusiones grupales serán de utilidad de este proyecto para lograr aplicar los cuestionarios mencionados en el apartado III.3, ya que con esta técnica de investigación se buscará realizar entrevistas o cuestionarios colectivos para que los sujetos puedan expresarse de manera libre y espontánea sobre los diferentes temas y con esto lograr llegar a un consenso de respuesta en la cual la mayoría de sujetos estén de acuerdo.

Los sujetos que se necesitarán para aplicar esta técnica serán el director del centro de operaciones, el administrador e integrador de aplicaciones y 10 ingenieros de *software* (III.2.1).

La herramienta que se utilizará para la documentación de esta técnica serán los mismos cuestionarios, los cuales se encuentran desde el Apéndice 1 al Apéndice 8 y se aplicará lo detallado en la sección III.3.1.

III.3.4. Priorizar mejoras.

Para la priorización de las mejoras, el estudiante aplicará su conocimiento crítico para ordenarlas en cuanto a la importancia de cada una de las actividades dentro del ALM y para el centro de operaciones de la empresa. Las cuales se van a obtener al aplicar los cuestionarios y realizar el análisis de brechas, esta lista de mejoras junto con el juicio de expertos dará origen al plan de trabajo.

III.3.5. Revisión documental.

Esta técnica se implementará para recolectar información relevante de cada una de las actividades descritas en el plan de trabajo a desarrollar por el estudiante, como estándares, políticas y metodologías; y con esta información desarrollar las mismas de la manera más adecuada para la empresa.

La información que se recolectará comprenderá tanto fuentes bibliográficas como libros, revistas, entre otros, los cuales se describen en el apartado III.2.2. Además se realizará una revisión de la documentación existente en el centro de operaciones

y que influyen en los estándares, políticas y metodologías que se van a proponer por el estudiante.

III.3.6. Variables gráfico radial

Las variables con las que va a contar el gráfico radial serán las siguientes:

Las variables son cada una de las áreas de ALM expuestas en la sección II.1.1.1, las cuales son:

- Arquitectura y diseño.
- Ingeniería de requerimientos y experiencia de usuario.
- Desarrollo.
- Administración de la configuración del *software*.
- Gobierno de TI.
- Implementación y operaciones.
- Aseguramiento de calidad.
- Planificación y administración de proyectos.

Además, el gráfico estará diseñado en “tela de araña”, y será un octágono por cada uno de niveles de madurez, siendo el nivel más bajo el básico y será representado con octágono más pequeño y el nivel dinámico representado con el octágono más grande.

Con esto a todas las áreas se les identificará el estado actual en el que se encuentran según la información recolectada de los cuestionarios, este estado se va identificar con una línea color azul y será la línea que se encuentra más cerca del punto central del gráfico.

Unido a esta línea también se identificará el estado deseado para cada una de las áreas, para lo cual se identificarán por medio de una línea color verde y será la línea que estará en la parte más exterior del gráfico.

III.4. Procesamiento y análisis de los datos.

El procesamiento y análisis de los datos para el presente proyecto se realizará mediante el proceso detallado en los siguientes puntos:

- a. Recolección de los datos generados mediante la aplicación de cada una de las técnicas de investigación, para cada uno de los objetivos fijados anteriormente, descritos y detallados en el apartado III.3 de técnicas de investigación
- b. Se realizará un análisis de brechas de los datos recolectados mediante los cuestionarios de los procesos individuales de TI (Apéndice 1 al Apéndice 8), el cual se basó en los pasos descritos en la sección II.2. (Análisis de brechas). Este análisis dará origen a un diagrama araña y una tabla comparativa del estado actual contra el estado deseado.
- c. Los resultados de los análisis serán usados como entradas de información para la toma de decisiones y en el proceso de priorizar las mejoras a realizar para lo cual se basó en los criterios como importancia para la empresa y urgencia para la misma. Esta información también se utilizó para realizar el plan de acción.
- d. Se desarrollará el último objetivo mediante las técnicas descritas en el apartado III.3. (Técnicas de investigación) y con base en los resultados obtenidos.
- e. Conjuntamente con el punto anterior, se realizarán las propuestas y conclusiones de acuerdo con el alcance del proyecto.

En la Tabla 9 se presenta la manera en cómo se realizará el análisis de los datos, por cada uno de los objetivos del proyecto.

Tabla 9 Análisis de datos por objetivo y técnica de investigación.

Objetivos	Herramientas o técnicas de investigación	Análisis y procesamiento de datos
<p>Realizar un análisis de brechas entre el estado actual del centro de operaciones y el estado que se quiere alcanzar, basado en los niveles de madurez de ALM propuestos por Microsoft.</p>	<ul style="list-style-type: none"> • Cuestionarios de los procesos individuales de TI. • Gráfico radial (diagrama araña). • Cuadro de situación actual contra situación deseada. 	<p>Se aplicarán cuestionarios con preguntas cerradas (III.3.1), mediante discusiones grupales (III.3.3), los cuales se pueden ver desde el Apéndice 1 al Apéndice 8. Estos se utilizaron para determinar los siguientes puntos:</p> <ol style="list-style-type: none"> 1. Identificar el estado actual de cada uno de los procesos de ALM. 2. Identificar el estado al que se desea llegar. 3. Identificar posibles mejoras en cada uno de los procesos de ALM <p>Los resultados de los cuestionarios se analizarán mediante la comparativa del estado actual contra el estado deseado, usando un gráfico radial y un cuadro comparativo.</p>
<p>Proponer un plan de trabajo para lograr el estado deseado del centro de operaciones, a</p>	<ul style="list-style-type: none"> • Juicio de expertos. • Priorizar mejoras. 	<p>Se analizarán los resultados obtenidos y con estos se buscará identificar cada una de las mejoras posibles y priorizarlas (III.3.1) y junto al juicio de experto (III.3.2), se validarán cada una de las</p>

Objetivos	Herramientas o técnicas de investigación	Análisis y procesamiento de datos
partir de los resultados del análisis de brechas.		mejoras y prioridades y por lo tanto plasmar las actividades a desarrollar en el plan de trabajo.
Emplear estándares y políticas de los procesos asociados a la administración del ciclo de vida de las aplicaciones del centro de operaciones.	<ul style="list-style-type: none"> • Revisión documental. • Juicio de expertos. 	<p>Se realizará una revisión documental (III.3.1. Revisión documental.) sobre estándares y plantillas utilizados por empresas con el fin de identificar características que sirvan de fortalecimiento en la elaboración de los estándares. Además, se revisará la documentación existente en la empresa sobre cómo se ha trabajado en los últimos años, para ver los escenarios y/o flujos por los cuales ha pasado esta (Casos de clientes, excepciones de la empresa, casos críticos, pasos extra).</p> <p>Una vez que se recolecte la información se procederá a realizar cada una de las actividades seleccionadas o propuestas del plan de trabajo, las cuales serán sometidas al juicio de expertos (III.3.2) para validar su contenido y formato.</p>

Fuente: Elaboración propia, 2016.

IV ANALISIS DE LA SITUACIÓN

En este apartado se presentan los temas principales en torno a los que gira la propuesta. Entre estos, se incluyó el análisis de la situación deseada, el análisis de la situación actual, el plan de trabajo según las prioridades de la empresa y los estándares, políticas o metodologías propuestas.

IV.1. Análisis de la situación deseada

En esta sección se detalla información importante que permitió conocer el estado al que desea llegar el centro de operaciones, con respecto al ALM propuesto por Microsoft. Para esto, se implementó la técnica juicio de experto, la cual se encuentra en el Apéndice 15, con el director del centro de operaciones y con el administrador e integrador de aplicaciones.

Con la aplicación de esta técnica, y en conjunto con las personas mencionadas anteriormente y además teniendo en cuenta el análisis de la situación actual que se explica en la sección IV.2, se logró determinar que el estado al que se desea llegar es a un estado avanzado en cada una de las etapas del ALM. (Estas etapas están descritas en el capítulo I).

El estado avanzado al cual se desea llegar se determinó, ya que la empresa busca que todas las etapas estén en un mismo grado de madurez de ALM; esto con el fin de no dejar ninguna de estas etapas rezagadas y en un futuro seguir con el mejoramiento del ciclo de ALM, ya que el siguiente estado sería el dinámico y en este nivel es donde se automatizan las acciones de ALM.

IV.2. Análisis de la situación actual

En este apartado se detalla información relevante que permitió conocer la situación en que se encuentra el centro de operaciones en la actualidad con respecto al ALM propuesto por Microsoft. Para ello se desarrolló un análisis de brechas para responder al objetivo específico 1 de este proyecto.

Los sujetos de información son colaboradores de la empresa que gestionan y/o influyen en el ciclo de vida de las aplicaciones. Como se presentó en el capítulo dos (II) del marco metodológico, los sujetos de información son:

- Director del centro de operaciones.
- Administrador e integrador de aplicaciones.
- 10 ingenieros de *software*.

Para recolectar la información necesaria se realizó un cuestionario (III.3.1) por cada uno de los procesos individuales de TI o áreas de ALM, mediante la técnica de discusiones grupales (III.3.3), con el fin de aclarar este punto en el Gráfico 3 se muestran los procesos y/o etapas del ALM (estas etapas también están descritas en el capítulo I).



Fuente: Elaboración propia, 2016.

Seguidamente se detallan los principales resultados de la investigación para cada una de las etapas, así como el análisis de estos resultados y el estado actual en la cual se encuentra el proceso.

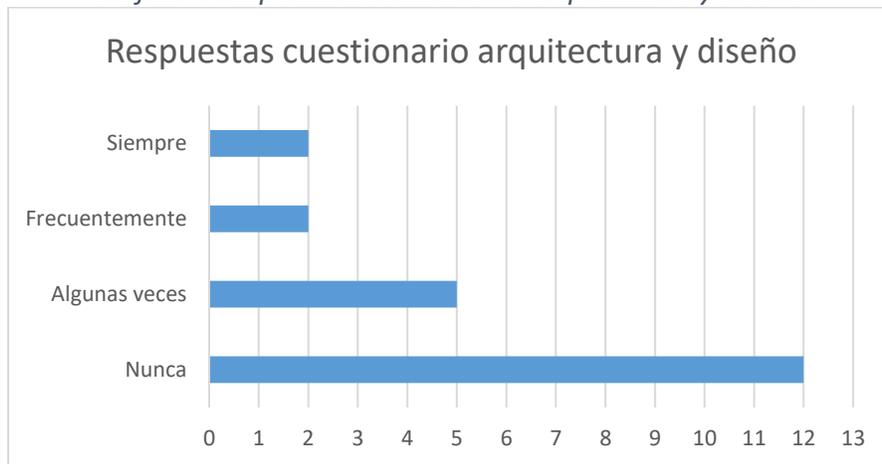
IV.2.1. Arquitectura y diseño

IV.2.1.1. Resultados

Con base en el cuestionario de arquitectura y diseño que se encuentra en el Apéndice 16, se logró obtener los siguientes resultados.

Para el total de 21 preguntas con las que cuenta el cuestionario, se obtuvieron 12 respuestas con el valor “nunca”, 5 con el valor “algunas veces”, 2 “frecuentemente” y 2 en “siempre”. El Gráfico 4 muestra las respuestas obtenidas de las preguntas:

Gráfico 4 Respuestas cuestionario arquitectura y diseño.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en la Tabla 8, en la Tabla 10, se detalla el puntaje obtenido del cuestionario:

Tabla 10 Puntaje cuestionario arquitectura y diseño.

Nivel	Cantidad	Puntaje
Nunca	12	12
Algunas veces	5	10
Frecuentemente	2	6
Siempre	2	8
Total		36

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1), se obtiene que el resultado del cuestionario es 20, el cual se detalla a continuación:

$$Puntaje\ total = Redondear\left(\frac{36}{21}\right) * 10 = 20$$

IV.2.1.2. Análisis de los resultados.

A manera de análisis, para la etapa de arquitectura y diseño, se obtiene que el rol de arquitectura está entendido y claramente definido, pero combinado con otras funciones. Además, se han identificado herramientas para trabajos de arquitectura, pero siguen en procesos de adopción, así mismo, se mantiene una documentación irregular e informal que no es de conocimiento de todos los miembros de los equipos.

Unido a lo detallado anteriormente, y gracias a los resultados obtenidos, se identifican algunos puntos importantes de esta etapa:

- Existen documentos informales no diagramados de la arquitectura del sistema en desarrollo.
- No se realizan modelados arquitectónicos al inicio de los requerimientos.
- No se realizan actualizaciones en los documentos cuando el sistema sufre cambios en su arquitectura.
- Actualmente existen dos *Sharepoint*, sin embargo, no se está sacando el provecho al máximo, pues el uso es mínimo.
- No se utilizan prototipos antes de los desarrollos en cuanto a diagramas ni diseños (por ejemplo: UML).
- No existen patrones de diseño definidos, tampoco se utilizan diagramas de secuencia.
- No existe documentación de la base de datos ni diccionario de datos.
- No hay forma de controlar el rendimiento de la base de datos en desarrollo.

IV.2.1.3. Estado actual

Por lo tanto, con base en los resultados obtenidos, y según la información de la Tabla 7, esta etapa se encuentra en un estado **estandarizado**.

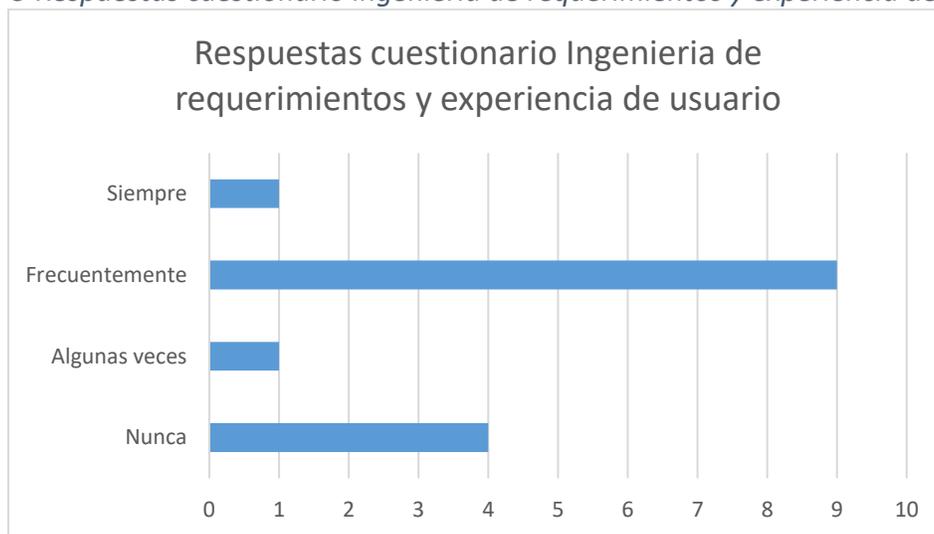
IV.2.2. Ingeniería de requerimientos y experiencia de usuario

IV.2.2.1. Resultados

De acuerdo con el cuestionario de ingeniería de requerimientos y experiencia de usuario que se encuentra en el Apéndice 17, se lograron obtener los siguientes resultados.

Para el total de 15 preguntas con las que cuenta el cuestionario, se obtuvieron 4 respuestas con el valor “nunca”, 1 con el valor “algunas veces”, 9 “frecuentemente” y 1 en “siempre”. El Gráfico 5 muestra las respuestas obtenidas a partir de las preguntas:

Gráfico 5 Respuestas cuestionario ingeniería de requerimientos y experiencia de usuario.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en Tabla 8, en la Tabla 11, se detalla el puntaje obtenido a partir del cuestionario:

Tabla 11 Puntaje cuestionario ingeniería de requerimientos y experiencia de usuario.

Nivel	Cantidad	Puntaje
Nunca	4	4
Algunas veces	2	2
Frecuentemente	9	27
Siempre	1	4
Total		37

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1), se obtiene que el resultado del cuestionario es 20, el cual se detalla de la siguiente manera:

$$Puntaje\ total = Redondear\left(\frac{37}{21}\right) * 10 = 20$$

IV.2.2.2. Análisis de los resultados

A manera de análisis, para la etapa de ingeniería de requerimientos y experiencia de usuario se identificó que existe un nivel de calidad constante en el formato de los requerimientos documentados. Igualmente, hay diferentes versiones de los requerimientos y se da cierto seguimiento (control de cambios). Así mismo, hay acceso a los requerimientos por parte de cualquier interesado en el proyecto y existe un rol de usuario experto definido, aunque no necesariamente es experto en temas de requerimientos. También existen algunas herramientas no integradas para el desarrollo de prototipos de interfaz de usuario.

Unido a lo detallado anteriormente, y gracias a los resultados obtenidos, se identifican algunos puntos importantes de esta etapa:

- Actualmente no se utilizan diagramas de casos de uso ni se documentan posibles escenarios según el uso de los requerimientos.
- En la definición de requerimientos no se definen constantemente criterios de aceptación para el proceso de pruebas.
- No hay trazabilidad desde los requerimientos, pruebas, tareas y secciones de código fuente, hasta hace un tiempo se ha empezado a utilizar el TFS para el seguimiento de tareas, pero no es una fortaleza aún, pues hay deficiencias en el conocimiento de la herramienta.
- El uso de prototipos de interfaz de usuario no se aplica en todos los casos, muchas veces no hay claridad de lo que el usuario necesita hasta que se les entrega el producto, lo que trae como consecuencia la identificación de requerimientos no atendidos hasta el proceso de pruebas de aceptación.
- No hay visibilidad de la totalidad de requerimientos pendientes en cuanto a una pila de requerimientos.

- No existen reportes ni *dashboards* de seguimientos o de avance, con sus respectivas tareas.
- Existe poca retroalimentación para el usuario sobre el trabajo realizado por desarrollo y estatus actual.
- El proceso de toma de requerimientos o afinación de estos aún no está estandarizado. Al día de hoy se evalúan formas de llevar a cabo reuniones o contar con personal, por lo que es inestable la forma de realizarlo.
- El rol de dueño de producto debe fortalecerse, pues los potenciales candidatos para ejecutar dichas funciones son multifuncionales y es difícil tomar el tiempo para establecer directrices de dueño de producto que establezcan comunicación entre desarrollo y el negocio.
- En la empresa hay poca libertad para temas de configuración del Team Foundation Server en cuanto a administración de elementos de trabajo, como requerimientos o los mismos *bugs*. De igual manera, en la empresa no se tiene definida una metodología para el uso de estos, pues se utiliza como catálogo de requerimientos.

IV.2.2.3. Estado actual

Con base en los resultados obtenidos, y según la Tabla 7, esta etapa se encuentra en un estado **estandarizado**.

IV.2.3. Desarrollo

IV.2.3.1. Resultados

Según el cuestionario de la etapa de desarrollo que se encuentra en el Apéndice 18, se lograron obtener los siguientes resultados.

Para el total de 20 preguntas con las que cuenta el cuestionario se obtuvieron 13 respuestas con el valor “nunca”, 3 con el valor “algunas veces”, 2 “frecuentemente” y 2 en “siempre”. El Gráfico 6, muestra las respuestas obtenidas de las preguntas:

Gráfico 6 Respuestas cuestionario desarrollo.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en Tabla 8, en la Tabla 12, se detalla el puntaje obtenido del cuestionario:

Tabla 12 Puntaje cuestionario desarrollo.

Nivel	Cantidad	Puntaje
Nunca	13	13
Algunas veces	3	6
Frecuentemente	2	6
Siempre	2	8
Total		33

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1) se obtiene que el resultado del cuestionario es 20, el cual se detalla seguidamente:

$$Puntaje\ total = Redondear\left(\frac{33}{20}\right) * 10 = 20$$

IV.2.3.2. Análisis de los resultados

A manera de análisis, para esta etapa se tiene que el equipo de desarrollo trabaja de la mano con Team Foundation Server como marco para la gestión de código. Además, existe un paradigma orientado a objetos en la construcción de componentes para el sistema, pero existen bastantes deficiencias en la parte de administración de código fuente, también se notó que el equipo tiene jerarquías altamente identificadas (arquitectos, líderes técnicos, desarrolladores seniors, analistas de calidad).

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa.

- Actualmente existen estándares para los cuales no se tiene control si se aplican, sino hasta el momento que se encuentran con situaciones puntuales en las que se evidencian.
- No se tienen políticas para controlar el código fuente.
- Falta de seguimiento a las normas indicadas para la escritura de código.
- Algunos casos no hay correcta documentación de código fuente.
- Falta de revisiones formales de código o controles cruzados, en la estimación no se toman en cuenta estas revisiones.
- No se utilizan herramientas de cobertura de código o análisis unitario de las pruebas.
- No existen métricas ni reportes que evidencien las buenas o malas prácticas en escritura de código.
- No hay un equipo de base de datos integrado con el equipo de desarrollo, los mismos desarrolladores de forma independiente dan seguimiento a sus bases de datos de desarrollo.

- La estrategia de *branching* y *merging* provoca varios conflictos, además de la forma de subir el código como un único entregable ocasiona problemas con los *merges*.
- Actualmente no se utilizan *builds* para asegurar código correcto.
- No existen métricas ni reportes para evidenciar avances o retrasos con el equipo de desarrollo.
- Se evidencia una falta de metodología que permita entregar valor continuo al negocio de una forma ágil.
- En cuanto al desarrollo hay una subutilización de la herramienta Team Foundation Server, ya que básicamente se utiliza como base de datos del código fuente.
- Hay una deficiencia en cuanto a los privilegios con los que cuenta el equipo de desarrollo para la utilización correcta del TFS.
- No hay forma de controlar las versiones o proyectos de las bases de datos, lo único que se versiona son los *scripts*, pero no hay manera de generar scripts incrementales que permitan actualizaciones completas de las bases de datos.

IV.2.3.3. Estado actual

Ahora bien, de acuerdo con los resultados obtenidos y a la Tabla 7 esta etapa se encuentra en un estado **estandarizado**.

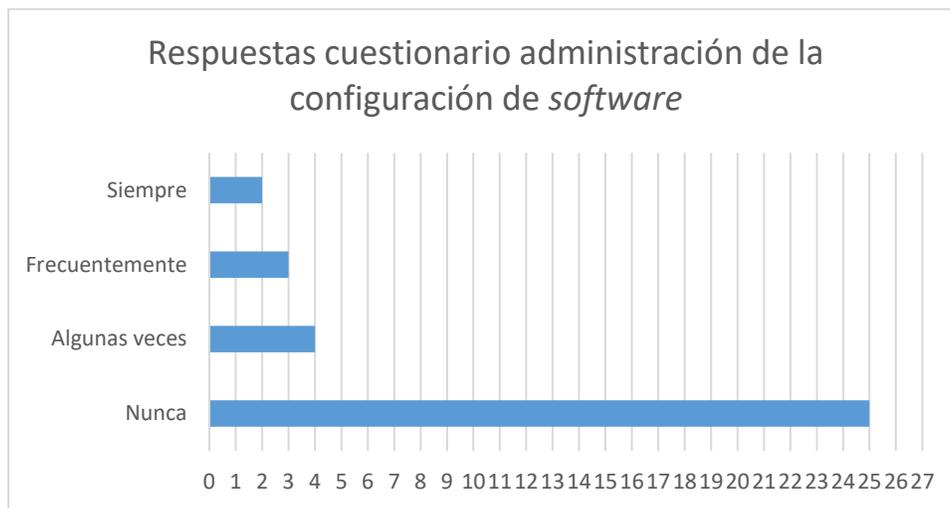
IV.2.4. Administración de la configuración del *software*

IV.2.4.1. Resultados

Con base en cuestionario de la etapa de administración de la configuración del *software* que se encuentra en el Apéndice 19, se logró obtener los siguientes resultados:

Para el total de 34 preguntas con las que cuenta el cuestionario, se obtuvieron 25 respuestas con el valor “nunca”, 4 con el valor “algunas veces”, 3 “frecuentemente” y 2 en “siempre”. El Gráfico 7 muestra las respuestas obtenidas de las preguntas:

Gráfico 7 Respuestas cuestionario administración de la configuración de software.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en Tabla 8, en la Tabla 13, se detalla el puntaje obtenido del cuestionario:

Tabla 13 Puntaje cuestionario administración de la configuración de software.

Nivel	Cantidad	Puntaje
Nunca	25	25
Algunas veces	4	8
Frecuentemente	3	9
Siempre	2	8
Total		50

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1), se obtiene que el resultado del cuestionario es 10, el cual se obtuvo de la siguiente forma:

$$Puntaje\ total = Redondear\left(\frac{50}{34}\right) * 10 = 10$$

IV.2.4.2. Análisis de los resultados

A manera de análisis, para la etapa de administración de la configuración del *software*, se descubrió que existe una administración básica del control de código fuente, en ciertos casos se utilizan respaldos locales del código fuente, así mismo, existen procesos de configuración no documentados. Además, existe poca trazabilidad sobre el trabajo realizado en términos de requerimientos y configuraciones en diferentes ambientes y hay mucha irregularidad con el término *check-in*. También se identificó que las funcionalidades fundamentales de la herramienta Team Foundation Server no se han habilitado.

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa.

- La estrategia de Branch y Merge da muchos conflictos debido a la forma de trabajar y la metodología implementada en desarrollo.
- Falta libertad en la parte de administración de la configuración para manipular funcionalidades básicas que ofrece el TFS. Dicha situación podría bien tratarse de desconocimiento y falta de capacitación.
- El rol de administración de la configuración no trabaja de forma integrada con los equipos de desarrollo, por lo que hay mucha información omitida en los procesos de configuración.
- No se utilizan etiquetas de código para proveer versiones específicas en los pases y documentar las versiones implementadas.
- Los desarrollos en paralelo se convierten en cascadas aisladas, lo que ocasiona que no se tenga conocimiento del código desarrollado por otros equipos, por tanto, a la hora de realizar *merge* entre diferentes *branches* hay pérdida de código o configuraciones.

- Los *scripts* de migración de las bases de datos se realizan de manera manual y cuando se necesita actualizar la base de datos se desarrolla el *script*. Al día de hoy no se ha investigado si existe alguna herramienta para manejar proyectos de bases de datos de Oracle que permitan ejecutar *scripts* incrementales y manejar los proyectos de estos.
- No hay un proceso de carga de datos de prueba ni encriptación de estos, se utilizan datos de producción, inclusive datos sensibles.
- No hay documentación de la estrategia de Branch y Merge que respalde la forma de trabajar de la empresa y que no represente “cuellos de botella” en las liberaciones.
- No hay políticas de control de código fuentes que permitan ingresar información para una mayor trazabilidad.
- Existe documentación como manuales y otros documentos de uso de la aplicación; sin embargo, no hay guías de instalación y no hay certeza que la documentación del *SharePoint* esté bien actualizada.
- Hay una deficiencia en cuanto a los planes para realizar algún *rollback* o planes de instalación para evitar errores durante el proceso.
- No se han realizado *builds* automáticos para asegurar la correcta construcción de los componentes de *software*, por lo tanto tampoco hay aprobaciones respaldadas por algún historial de compilaciones.
- No hay reportes, métricas o *dashboards* para validar el estado de compilaciones, componentes y requerimientos de configuración.

IV.2.4.3. Estado actual

Por lo tanto, con base en los resultados obtenidos, y según la Tabla 7, esta etapa se encuentra en un estado **básico**.

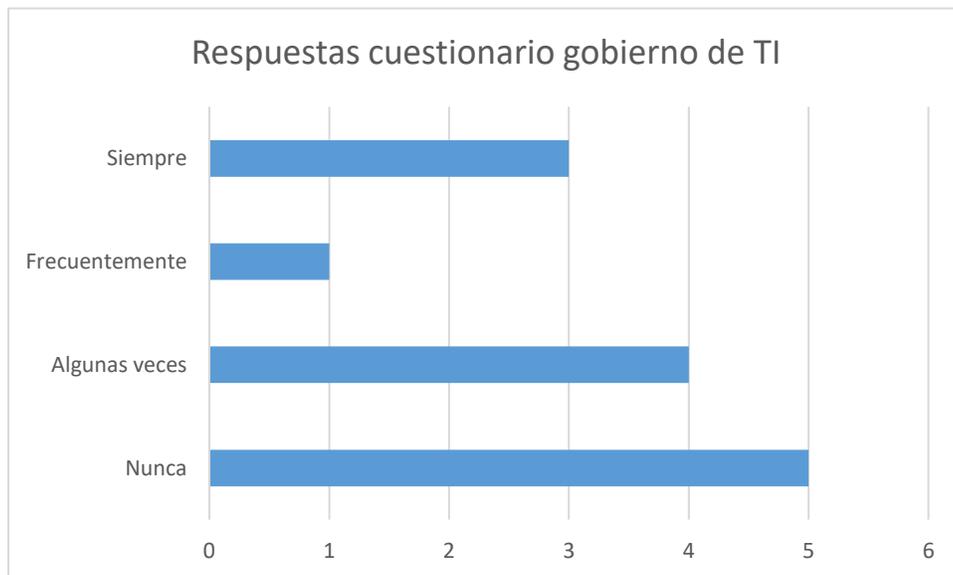
IV.2.5. Gobierno de TI

IV.2.5.1. Resultados

De acuerdo con el cuestionario de gobierno de TI, que se encuentra en el Apéndice 20, se obtuvieron los siguientes resultados.

Para el total de 13 preguntas con las que cuenta el cuestionario, se obtuvieron 5 respuestas con el valor “nunca”, 4 con el valor “algunas veces”, 1 “frecuentemente” y 3 en “siempre”. El Gráfico 8 muestra las respuestas obtenidas de las preguntas:

Gráfico 8 Respuestas cuestionario gobierno de TI.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en Tabla 8, en la Tabla 14, se detalla el puntaje obtenido del cuestionario:

Tabla 14 Puntaje cuestionario gobierno de TI.

Nivel	Cantidad	Puntaje
Nunca	5	5
Algunas veces	4	8
Frecuentemente	1	3
Siempre	3	12
Total		28

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1) se obtiene que el resultado del cuestionario es 20, el cual se detalla a continuación:

$$Puntaje\ total = Redondear\left(\frac{28}{13}\right) * 10 = 20$$

IV.2.5.2. Análisis de los resultados

A manera de análisis, para la etapa de gobierno de TI se descubrió que ha alcanzado una mejoría en los procesos e implementación de buenas prácticas. Así mismo, se han implementado prácticas que ayudan al fortalecimiento de un marco de trabajo, por lo que se podría decir que hay una certificación informal adoptada por la empresa para el cumplimiento de los objetivos. También hay herramientas semimanuales que fortalecen la ejecución de procesos fundamentales.

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa.

- La metodología utilizada por la empresa no está estrictamente apoyada por ningún marco de trabajo (ITIL, COBIT, entre otros).
- Actualmente los requerimientos se aclaran y tienen varios filtros antes de ser pasados a desarrollo, lo cual ha fortalecido el compromiso del usuario a la hora de definirlos.
- No hay métricas o indicadores claves de rendimiento que evidencien retornos de inversión o avances positivos en cualquier campo con respecto a los servicios brindados por la fábrica.
- No hay cuadros de mando para monitorear progreso. Existen herramientas manuales, pero no han sido creadas bajo la necesidad y no hay documentación ni plantillas oficiales.
- No hay herramientas para la integración con el portafolio de proyectos y los procesos realizados en ellos.
- No hay una gestión de riesgos de la empresa que permita una mejoría continua por medio de una implementación de mitigación de riesgos o matriz de seguimiento.

IV.2.5.3. Estado actual

Por lo tanto, con base en los resultados obtenidos, y según la Tabla 7, esta etapa se encuentra en un estado **estandarizado**.

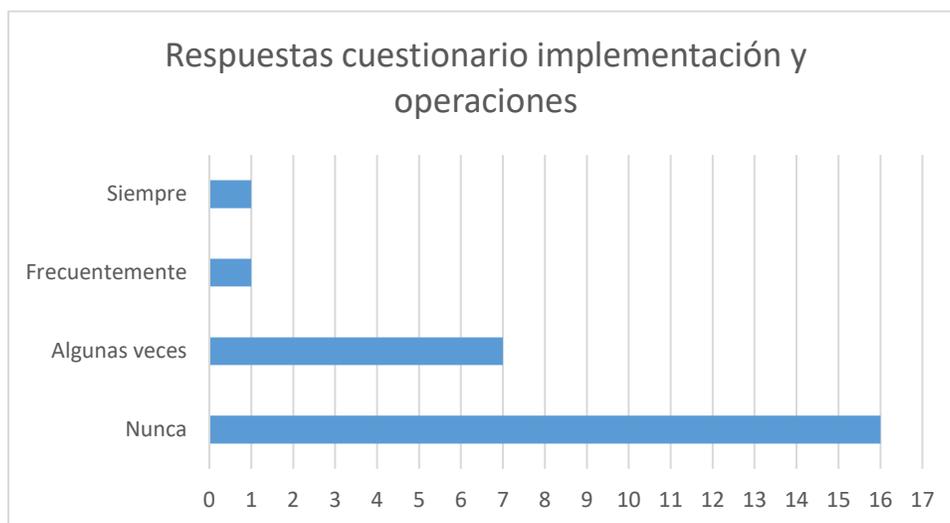
IV.2.6. Implementación y operaciones

IV.2.6.1. Resultados

Con base en el cuestionario de la etapa de implementación y operaciones, que se encuentra en el Apéndice 21, se obtuvieron los siguientes resultados.

Para el total de 25 preguntas con las que cuenta el cuestionario, se obtuvieron 16 respuestas con el valor “nunca”, 7 con el valor “algunas veces”, 1 “frecuentemente” y 1 en “siempre”. El Gráfico 9 muestra las respuestas obtenidas a partir de las preguntas:

Gráfico 9 Respuestas cuestionario implementación y operaciones.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en la Tabla 8, en la Tabla 15, se detalla el puntaje obtenido del cuestionario:

Tabla 15 Puntaje cuestionario implementación y operaciones.

Nivel	Cantidad	Puntaje
Nunca	16	16
Algunas veces	7	14
Frecuentemente	1	3
Siempre	1	4
Total		37

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1) se obtiene que el resultado del cuestionario es 10, obtenido de la siguiente manera:

$$Puntaje\ total = Redondear\left(\frac{37}{25}\right) * 10 = 10$$

IV.2.6.2. Análisis de los resultados

A manera de análisis, para la etapa de implementación y operaciones se tiene que hay poca o nula comunicación entre los equipos de desarrollo y operaciones. Adicionalmente, en el centro de operaciones no hay un procedimiento formal de seguimiento de errores o alguna mesa de ayuda para la gestión de incidentes. Por su parte, los problemas en los despliegues correspondientes a infraestructura o configuración son resueltos en tiempos de despliegue. También se identificó que la configuración y segregación de ambientes para diferentes fines no están documentadas con las configuraciones básicas requeridas.

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa:

- Existe poca retroalimentación del resultado de los despliegues en los diferentes ambientes.
- La administración de la plataforma TFS, a pesar de que hay cooperación, trabaja de forma aislada.
- No hay monitoreo para ambientes que no sean producción para diagnosticar problemas desde antes que se publiquen objetos en producción.
- No existen guías o base de conocimientos para la solución de incidentes en los diferentes pases.
- El diseño e implementación de la infraestructura no está documentado, tanto para producción como para desarrollo u otros ambientes.
- En la empresa se omiten los procesos de pases de conocimientos a los equipos de desarrollo o equipos de QA, estos se van comprendiendo en el transcurso de los desarrollos.

- La administración de requerimientos o *bugs*, a partir de una mejora en la parte de operaciones, es ambigua y se resuelve dependiendo de su criticidad, es decir, no hay un estándar definido.
- No hay un control de versiones o políticas de retención de *builds* para el caso de pruebas o implementaciones en la producción.

IV.2.6.3. Estado actual

Por lo tanto, con base en los resultados obtenidos y según la Tabla 7 esta etapa se encuentra en un estado **básico**.

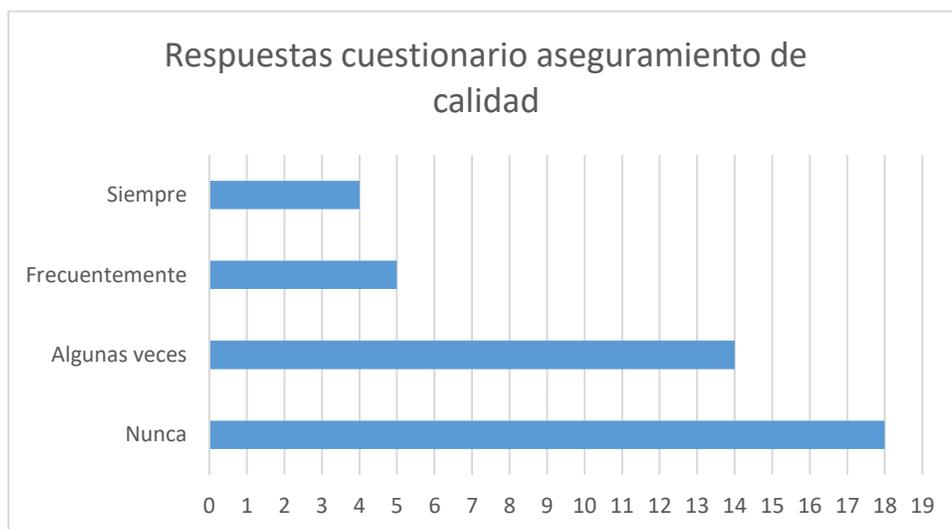
IV.2.7. Aseguramiento de calidad

IV.2.7.1. Resultados

Según el cuestionario de aseguramiento de calidad, que se encuentra en el Apéndice 22, se obtuvieron los siguientes resultados.

Para el total de 41 preguntas con las que cuenta el cuestionario, se obtuvieron 18 respuestas con el valor “nunca”, 14 con el valor “algunas veces”, 5 “frecuentemente” y 4 en “siempre”. El Gráfico 10 muestra las respuestas obtenidas de las preguntas:

Gráfico 10 Respuestas cuestionario aseguramiento de calidad.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en la Tabla 8, en la Tabla 16 se detalla el puntaje obtenido del cuestionario:

Tabla 16 Puntaje cuestionario aseguramiento de calidad.

Nivel	Cantidad	Puntaje
Nunca	18	18
Algunas veces	14	28
Frecuentemente	5	15
Siempre	4	16
Total		77

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1) se obtiene que el resultado del cuestionario es 20, el cual se detalla enseguida:

$$Puntaje\ total = Redondear\left(\frac{77}{41}\right) * 10 = 20$$

IV.2.7.2. Análisis de los resultados

A manera de análisis, para la etapa de aseguramiento de la calidad se determinó que la empresa cuenta con un grupo especializado en temas de aseguramiento de calidad. Así mismo, existe un proceso definido para la creación de planes de prueba. Igualmente, cuentan con herramientas de pruebas, pero estas no están integradas con los procesos o iteraciones del equipo de desarrollo. Además, existen métodos de prueba y ambientes que no están formalmente documentados.

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa:

- Actualmente no se cuentan con laboratorios virtuales para la ejecución de pruebas manuales o automáticas.
- El tema de pruebas automáticas o técnicas, como de rendimiento y estrés, no se ha adaptado de una manera correcta, debido a problemas con la arquitectura de aplicación y falta de conocimiento.
- Actualmente hay herramientas para la automatización de pruebas; sin embargo, estas no se utilizan.
- La trazabilidad de las pruebas con respecto a requerimientos y bloques de código no es tan precisa, debido a que en general la metodología no se adapta a todas las áreas y trabajan como “islas” que comparten información, pero no se integran formalmente.
- La estrategia de pruebas se define solo según el tiempo y división de trabajo, no se realiza con respecto al tipo de pruebas o criterios de aceptación.
- La priorización de pruebas no se realiza por medio de alguna herramienta en la cual se lleve el control, sino según la demanda o la premura de desarrollo por liberar producto.

- No existen reportes o *dashboards* que evidencien la calidad del producto en *testing* o cantidad de *bugs* dentro del proyecto.
- Los planes de prueba no se integran con los *builds* o versión de *builds*, para asegurar ciertas versiones.
- El equipo de calidad no asegura las instalaciones ni reciben guías o diagramas de flujo agregados en los requerimientos.
- El equipo de calidad no está en igualdad de conocimientos técnicos de calidad, en ocasiones se omiten tipos de pruebas.
- No existen informes formales para el seguimiento del proceso de calidad, algunas veces se crean reportes, pero no se cumple ningún estándar.
- En ocasiones, no todos los usuarios brindan criterios de aceptación, inclusive muchos usuarios no tienen claro lo que son, lo que provoca que las pruebas de aceptación que se realicen, y no se validen correctamente en términos de pruebas de negocio.
- Ni el equipo de QA ni el dueño de producto realizan acompañamiento al usuario para la ejecución de pruebas de aceptación.
- Actualmente, no se están realizando pruebas de regresión formales para asegurar que los cambios realizados en desarrollo no afecten objetos desarrollados que estaban en buen estado.
- Existe una debilidad fuerte en temas de automatización de pruebas, para la recuperación o inversión de tiempo en funcionalidades nuevas.
- Hace falta presencia del equipo de QA en los procesos, pues la percepción de desarrollo es que encuentren errores, pero esto no sucede debido a que ellos interactúan en solo una parte del proceso y no en toda la gestión.

IV.2.1.3. Estado actual

Por lo tanto, con base en los resultados obtenidos, y según la Tabla 7, esta etapa se encuentra en un estado **estandarizado**.

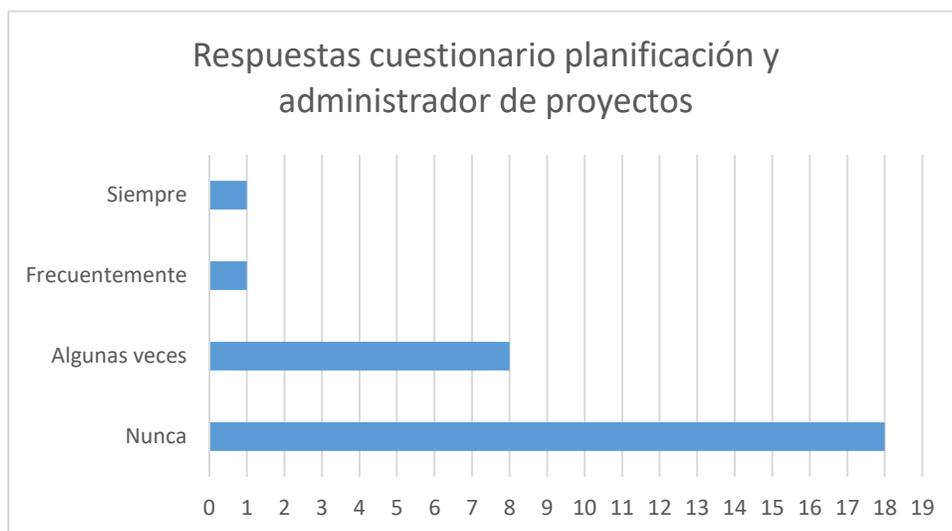
IV.2.8. Planificación y administración de proyectos

IV.2.8.1. Resultados

Según el cuestionario de planificación y administración de proyectos, que se encuentra en el Apéndice 23, se obtuvieron los siguientes resultados.

Para el total de 28 preguntas con las que cuenta el cuestionario, se obtuvieron 18 respuestas con el valor “nunca”, 8 con el valor “algunas veces”, 1 “frecuentemente” y 1 en “siempre”. El Gráfico 11 muestra las respuestas obtenidas de las preguntas:

Gráfico 11 Respuestas cuestionario planificación y administración de proyectos.



Fuente: Elaboración propia, 2016.

Con base en las respuestas obtenidas y conforme a lo descrito en la Tabla 8, en la Tabla 17 se detalla el puntaje obtenido del cuestionario:

Tabla 17 Puntaje cuestionario planificación y administración de proyectos.

Nivel	Cantidad	Puntaje
Nunca	18	18
Algunas veces	8	16
Frecuentemente	1	3
Siempre	1	4
Total		41

Fuente: Elaboración propia, 2016.

Al aplicar la fórmula (1) se obtiene que el resultado del cuestionario es 10, el cual se obtuvo de la siguiente manera:

$$\text{Puntaje total} = \text{Redondear}\left(\frac{41}{28}\right) * 10 = 10$$

IV.2.8.2. Análisis de los resultados

A manera de análisis, para la etapa de planificación y administración de proyectos se obtuvo que en el centro de operaciones de la empresa no hay un plan formal de comunicación con los interesados del proyecto. Por su parte, los procesos para estimación, planificación, gestión de riesgos son inexistentes o informales, de igual manera la asignación de tareas y la coordinación de estas no está totalmente estandarizada por medio de la herramienta de TFS y se notó que muchas veces es informal o verbal. Además, se identificó que por el tipo de negocio el tema de costos no se evalúa de acuerdo con el rendimiento de los equipos de trabajo.

Unido a lo detallado anteriormente y gracias a los resultados obtenidos se identifican algunos puntos importantes de esta etapa:

- La metodología actual es muy ambigua, aún no se ha madurado, así mismo hay problema con la agilidad y la entrega de valor continuo.
- El análisis de las habilidades de los desarrolladores no está bien definido, pues hay varios miembros polifuncionales, y otros unifuncionales.
- Como la metodología es bajo demanda y necesidad, la comunicación entre usuarios y el proyecto como tal está muy marcada; igualmente, no hay un rol de retroalimentación al usuario.
- No existe calidad continua en la administración del proyecto, ya que las fases están segmentadas y son excluyentes.
- Existen plantillas para realizar los pases; sin embargo, estos documentos no siempre se almacenan o tienen alguna política de retención.
- En el centro de operaciones de la empresa no todos los miembros tienen identificados los factores de éxito que ayuden a alcanzar el objetivo del proyecto.

- El equipo, en general, no tiene visión de los entregables por presentar, ni se da seguimiento por medio de herramientas a estos entregables.
- El equipo no tiene claro los roles de los equipos, también hay ambigüedades en cuanto a las líneas de trabajo de la empresa.
- No existen métricas, reportes ni *dashboards* para las diferentes jerarquías que evidencien el estado actual del proyecto.
- No se documentan todas las decisiones del proyecto, así mismo, existen herramientas que no se utilizan o se subutilizan.
- No se gestionan los riesgos y, en algunas ocasiones, no son de conocimiento del equipo.

IV.2.1.3. Estado actual

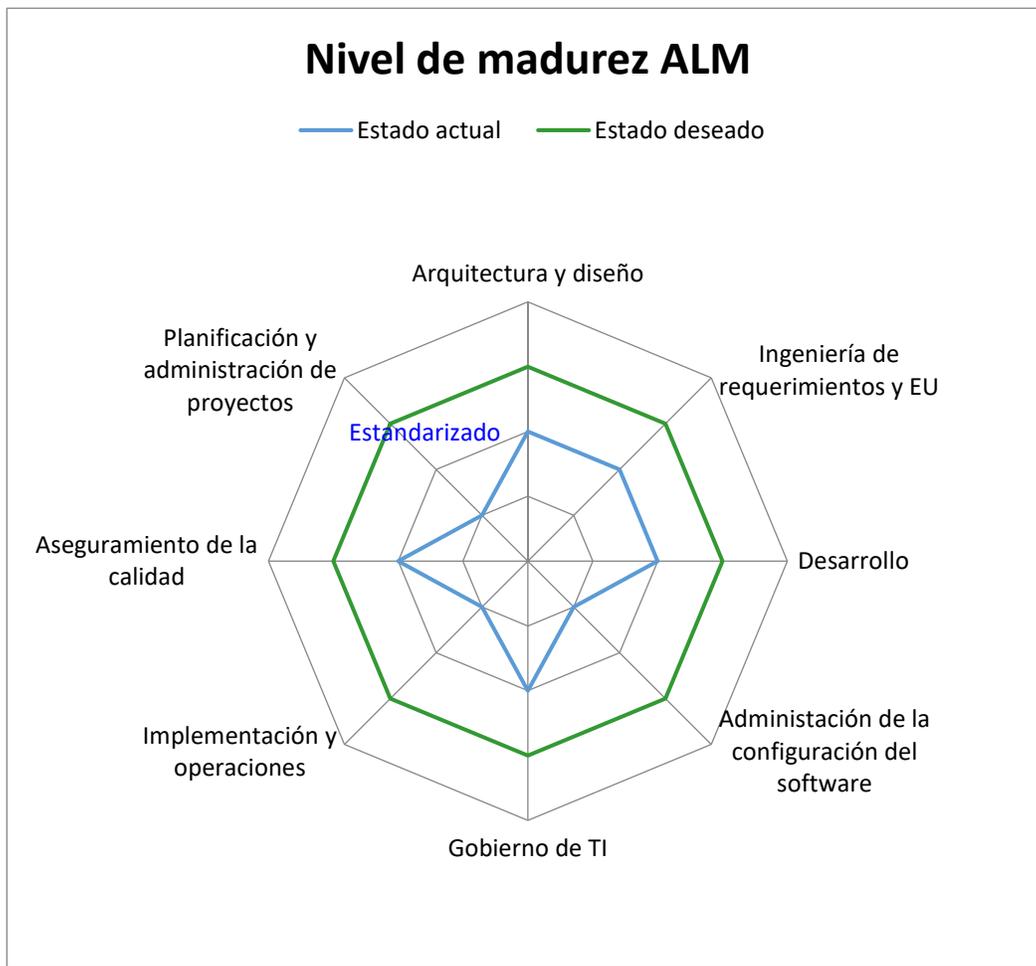
Por lo tanto, con base en los resultados obtenidos, y de acuerdo con la Tabla 7, esta etapa se encuentra en un estado **básico**.

IV.2.9. Resumen

A modo de resumen, a partir del análisis realizado al centro de operaciones de la empresa, se identifica que existe un proceso claro, este ha desembocado con un estilo de trabajo estandarizado; sin embargo, varios puntos de la evaluación evidencian que hay procesos que aún hace falta depurar y mejorar. Una de las principales fortalezas es que ya existen herramientas para interactuar con las personas y los procesos; sin embargo, estas herramientas no se están aprovechando al máximo.

Se presenta en el Gráfico 12 el estado actual de cada una de las áreas estudiadas, en comparación con el estado deseado, según los niveles de ALM:

Gráfico 12 Nivel de madurez ALM.



Fuente: Elaboración propia, 2016.

Como se puede observar en el Gráfico 12, para cada área se han adoptado buenas prácticas, puesto que predomina el nivel estandarizado, aun así se identifica poca integración, y todavía no se han culturizado ni documentado políticas, estándares y controles que permitan el seguimiento correcto de la empresa.

En resumen, se muestra en la Tabla 18 la situación actual por áreas y el estado deseado por alcanzar:

Tabla 18 Tabla comparativa de niveles de ALM.

Áreas ALM	Estado Actual	Estado Deseado	Estado
Arquitectura y diseño	Estandarizado	Avanzado	!
Ingeniería de requerimientos y experiencia de usuario	Estandarizado	Avanzado	!
Desarrollo	Estandarizado	Avanzado	!
Administración de la configuración	Básico	Avanzado	✘
Gobierno de TI	Estandarizado	Avanzado	!
Implementación y operaciones	Básico	Avanzado	✘
Aseguramiento de calidad	Estandarizado	Avanzado	!
Planificación y administración de proyectos	Básico	Avanzado	✘

Fuente: Elaboración propia, 2016.

Para la Tabla 18, el símbolo ! significa precaución, en el cual se está a un nivel por debajo del estado deseado, y el ✘ significa crítico, que se usa cuando se tiene más de un nivel por debajo del estado deseado, en cuanto a la comparación del estado actual y el estado al que se desea llegar.

V PROPUESTA A LA SITUACIÓN

En este apartado se detallan las propuestas que se recomiendan a la empresa para mejorar su situación actual.

V.1. Propuesta General

Este apartado se muestra la situación propuesta a nivel general, seguidamente se indica su distribución en cada área, de manera que se pueda identificar puntualmente y priorizar los puntos. Cabe destacar que a partir de esta propuesta general se genera el plan de acción para mejorar estas áreas.

V.1.1. Generalidades

Los siguientes puntos son consideraciones generales que la empresa debería de conocer o realizar:

- Es importante realizar una actualización de la herramienta TFS, de manera que se puedan aprovechar ciertas características nuevas que ofrece la herramienta, pues estas aplican a la mejoría del funcionamiento del centro de operaciones de la empresa. Además, se deben implementar ciertas configuraciones para el control y ejecución del trabajo diario.
- A nivel de la empresa, tras evaluar las habilidades de ciertas personas, surge la necesidad de crear un comité técnico que evalúe nuevas tecnologías. Además, es preciso que vayan creando un plan de actualización de la aplicación a versiones recientes de Visual Studio para aprovechar la integración de herramientas nuevas. Igualmente, se deben evaluar nuevas tendencias de patrones de desarrollo (*MVC*, *Entity Framework*, entre otras) y estar a la vanguardia de lo que ofrece el mercado.
- Es de suma importancia implementar retrospectivas de parte de los usuarios hacia la empresa, esto con el propósito de percibir, constantemente, oportunidades de mejora y fortalecer buenas prácticas.

V.1.2. Arquitectura y diseño

Los siguientes puntos son consideraciones de la etapa de arquitectura y diseño que la empresa debería de conocer o realizar:

- Definir un equipo de arquitectura dedicado.
- Iniciar proceso de documentación de la arquitectura y diseño del sistema.
- Implementar herramientas para diseñar y documentar la arquitectura, de manera que el conocimiento sea compartido y no se centralice en personas específicas.
- Utilizar el *SharePoint* como base de conocimiento de arquitectura para versionar documentos e información al alcance.
- Definir patrones de diseño, los cuales incorporen diagramas de secuencia para un mayor entendimiento de la comunicación de los proyectos lógicos de la solución.
- Realizar la documentación de la base de datos e incluir diccionario de datos.
- Incorporar herramientas que permitan controlar el rendimiento de los servidores en ambientes de desarrollo.
- Implementar equipo de arquitectura que sea facilitador y coordinador de desarrollo, de manera que se centralicen dudas y se estandaricen los estilos de desarrollo.

V.1.3. Ingeniería de requerimientos y experiencia de usuario

Los siguientes puntos son consideraciones de la etapa de ingeniería de requerimientos y experiencia de usuario que la empresa debería de conocer o realizar:

- Utilizar TFS para centralizar la documentación de requerimientos y casos de uso, así como hacer uso de un estándar de definición y documentación, de manera que exista trazabilidad de los requerimientos desde su respectiva documentación.
- En la definición de requerimientos, definir en conjunto con el usuario criterios de aceptación claros que permitan focalizar los desarrollos.
- Dar trazabilidad desde los requerimientos, a través de elementos de trabajo como casos de pruebas, tareas y conjuntos de cambios de código.

- Implementar el uso de *storyboarding* para la toma y presentación de requerimientos, de manera que se pueda graficar y obtener mayor claridad del fin de un requerimiento.
- Crear una plantilla de requerimientos funcionales y técnicos, esto con el propósito de estandarizar todos los requerimientos de la empresa.
- Confeccionar reportes y *dashboards* que permitan dar seguimiento a los respectivos requerimientos, además de compartir este tipo de información con el equipo para concientizar los avances.
- Fortalecer la figura del dueño de producto, el cual pueda colaborar en la retroalimentación al usuario sobre el trabajo realizado por desarrollo.
- Proveer a la fábrica mayor libertad en cuanto a temas de configuración y utilización del Team Foundation Server.

V.1.4. Desarrollo

Los siguientes puntos son consideraciones de la etapa de desarrollo que la empresa debería de conocer o realizar:

- Implementar el uso de revisiones de código para asegurar estándares, esta sería una revisión únicamente de estándar, no de entendimiento de código.
- Implementar, dentro de TFS, políticas para controlar el código fuente por medio de estándares de comentarios, así como utilizar elementos de trabajo y revisores de código.
- Inicializar un proceso de generación de pruebas unitarias para asegurar la correcta cobertura de código y correcta ejecución de este.
- Generar reportes y métricas a nivel de TFS para evidenciar avances o retrasos en términos de desarrollo versus elementos de trabajo asignados.
- Generar políticas de código fuente.
- Centralizar la administración y desarrollo de base de datos en ambiente de construcción de componentes, además de analizar la posibilidad de

utilizar proyectos de bases de datos, esto con el fin de versionar por esquema los objetos en un proyecto adicional de la solución.

- Reestructurar la estrategia de *branching* y *merging*, con el objeto de permitir desarrollos en paralelo y evitar realizar un único *checkin* del código de un requerimiento.
- Implementar compilaciones automáticas que se disparen luego de cada *checkin*, lo cual asegurará el correcto funcionamiento y construcción de los componentes desde el desarrollo y antes de una etapa de *testing*.

V.1.5. Administración de la configuración del *software*

Los siguientes puntos son consideraciones de la etapa de administración de la configuración del *software* que la empresa debería de conocer o realizar:

- Fortalecer el rol de administración de la configuración y establecer una metodología y políticas para la gestión de *branch* y *merge*, de manera que se implemente una forma para integrar los productos constantemente y evitar conflictos de versiones o requerimientos no aprobados mezclados con versiones estables, además de documentar las estrategias utilizadas.
- Establecer perfiles de administración de *source* control a los facilitadores del área de desarrollo, de manera que puedan realizar configuraciones básicas en su respectivo ambiente.
- Integrar la administración de la configuración con el área de desarrollo, de modo que se agilicen temas de configuración y traspaso de conocimiento.
- Para los respectivos pases, implementar el etiquetado de versiones con el fin de identificar, de una manera más ordenada, los despliegues en ambientes.
- Buscar e implementar la manera de automatizar los *scripts* de actualización de bases de datos para agilizar los pases y versionamiento de las fuentes de bases de datos.
- En ambientes de prueba, encriptar la información sensible para evitar utilizar datos de producción que puedan sufrir de alguna fuga de información.

- Implementar políticas para la administración y control de códigos fuentes que permita ingresar información para una mayor trazabilidad.
- Estandarizar las guías de instalación y proveerlas a los diferentes equipos, incluido el equipo de QA, además de tenerla actualizada y versionada.
- Documentar los procesos de *rollback* y hacerlos de conocimiento común para el equipo de desarrollo, esto por medio de la utilización de herramientas tales como *SharePoint*.
- Realizar la configuración de *builds* automáticos y documentarlos, además de configurar políticas de retención para realizar integraciones con las versiones de prueba del equipo de QA.
- Implementar reportes, métricas y *dashboards* correspondientes a temas de configuración y que sean de conocimiento del equipo por medio del portal de TFS.

V.1.6. Gobierno de TI

Los siguientes puntos son consideraciones de la etapa de gobierno de TI que la empresa debería de conocer o realizar:

- Definir métricas que evidencien el funcionamiento del centro de operaciones de la empresa, de manera que responda de manera positiva el trabajo realizado por todos los equipos de trabajo.
- Implementar herramientas que integren el portafolio de proyectos y los procesos realizados en ellos.
- Iniciar un proceso de gestión de riesgos de la empresa que permita una mejoría continua, por medio de la cual se lleve el pulso de factores críticos que puedan afectar la continuidad de la empresa.

V.1.7. Implementación y operaciones

Los siguientes puntos son consideraciones de la etapa de implementación y operaciones que la empresa debería de conocer o realizar:

- Incluir un proceso para medir la retroalimentación de los despliegues que se realizan en los diferentes ambientes, incluyendo producción.

- Integrar el rol de administración de la configuración con la empresa, de manera que trabajen menos aislados y se solventen problemas triviales de una manera ágil.
- En los ambientes de desarrollo y pruebas, implementar sistemas de monitoreo de rendimiento, o bien ejecutar pruebas de rendimiento que permitan disparar alertas desde ambientes previos a producción.
- Documentar y compartir, por medio de una base de conocimiento, las guías de instalación; o bien definir protocolos para atender incidentes por medio de la identificación de patrones con los que ocurren actualmente.
- Desarrollar sesiones de pase de conocimiento desde operaciones hasta la producción, incluyendo a los equipos de QA, para la atención de posibles incidentes y entendimiento de los contextos en que estos ocurren.
- Documentar y definir políticas para el *deployment*.
- Definir el protocolo para la administración de requerimientos o *bugs* a partir de una mejora en la parte de operaciones, de manera que se resuelvan de manera rápida y no afecte los desarrollos actuales.

V.1.8. Aseguramiento de calidad

Los siguientes puntos son consideraciones de la etapa de aseguramiento de calidad que la empresa debería de conocer o realizar:

- Evaluar y, de ser factible, habilitar las características del TFS para la creación y virtualización de laboratorios de pruebas.
- Fortalecer e incentivar la cultura de automatización de pruebas, tanto funcionales como de rendimiento y carga, de manera que se libere tiempo en la ejecución de pruebas manuales y se identifiquen problemas de rendimiento de forma temprana.
- Acoplar al equipo de pruebas la correcta utilización del TFS, para que se brinde trazabilidad a los elementos de trabajo.
- Utilizar criterios de aceptación como insumo principal de la creación de planes de pruebas y fortalecer la creación de estrategias en ciclos cortos, y no hasta el final de los ciclos largos de desarrollo.

- Realizar una priorización de pruebas por medio del Test Manager, de manera que se ordenen las pruebas y se ejecuten con un orden lógico.
- Crear reportes, métricas y *dashboards* para mejorar el control de los ciclos de pruebas, y que exista una retroalimentación en línea al equipo de desarrollo y demás interesados.
- Establecer umbrales de aceptación de producto para medir efectividad del equipo de QA y al mismo tiempo incentivar competencias sanas y mejoramiento continuo.
- Integrar la configuración de *builds* con la utilización de planes de prueba, para asegurar las versiones probadas.
- Proveer al equipo de QA las guías necesarias para que prueben los procesos de instalación e identifiquen irregularidades antes de los despliegues oficiales, además de proveer los flujos necesarios para asegurar procesos de sistema.
- Igualar conocimientos técnicos al equipo de QA, reducir las brechas de dependencias hacia personas y, de igual manera, implementar nuevos tipos de pruebas que actualmente se omiten.
- Implementar plantillas de informes de testing para finalizar ciclos de pruebas y fortalecer el seguimiento de parte de QA y el equipo de desarrollo, esto con el fin de que exista compromiso para la calidad continua.
- Proveer acompañamiento a los usuarios expertos en cuanto a la creación de criterios de aceptación, esto no implica realizar el trabajo de los usuarios, más bien, significa servir como guía al usuario para que los requerimientos sean robustos y entendibles inclusive para los mismos miembros de equipo de desarrollo.
- Durante la ejecución de pruebas de aceptación, realizar acompañamientos, ya sea de parte de algún miembro del equipo de QA o preferiblemente del dueño de producto para aclarar dudas y evitar re-trabajos producto de malos entendidos u otras situaciones típicas de este proceso.

- De no existir automatización, se deben implementar estrategias de muestreo de pruebas de aceptación para asegurar que los cambios o correcciones realizadas no afecten el producto que ya era funcional.
- Fortalecer la cultura en todo el equipo acerca del concepto “miembro de equipo” y asegurador de calidad. De igual manera incentivar la idea de que el equipo busca detectar la mejora continua y no los errores.
- Definir una metodología base a seguir para pruebas funcionales y técnicas.

V.1.9. Planificación y administración de proyectos

Los siguientes puntos son consideraciones de la etapa de planificación y administración de proyectos que la empresa debería de conocer o realizar:

- Implementar estrategias de planificación y administración para la entrega continua de valor al negocio, así como asegurar el entendimiento de todo el equipo para evitar ambigüedades en el proceso.
- Dentro de la planificación, buscar respetar las habilidades y destrezas de los miembros de equipo, en la cual deben existir roles facilitadores.
- Incluir con cierta frecuencia a los usuarios en las planeaciones (para evaluar disponibilidad y cuidado), con el fin de que exista mayor transparencia e integración en las fases de desarrollo y se evite que los usuarios pierdan el pulso de los requerimientos.
- Implementar un modelo de calidad continua y no segmentar fases, de manera que el equipo de QA se involucre en todo el proceso y fases.
- Hacer de conocimiento común los valores, los cuales son factores de éxito en la fábrica, para que se trabaje bajo una focalización de ideas, compromiso, apertura a cambios, coraje para afrontar los retos, iniciativa y auto-organización.
- Dar seguimiento a los entregables, tanto de desarrollo como requerimientos, tareas, pruebas y entregables a los usuarios.
- Creación de reportes y métricas según las jerarquías en la empresa que evidencien los avances o atrasos en la gestión del proyecto en cada área específica (desarrollo, QA, operación, entre otros).

- Creación de plantilla para *dashboards* gerenciales para el seguimiento de los proyectos.
- Creación de plantillas de minutas y controles de cambios a nivel de desarrollo, por medio de los elementos de trabajo que ofrece el TFS.
- Gestionar riesgos o *issues* que pueden ser bloqueantes o no en la fábrica para dar seguimiento y delegarlos.

V.2. Plan de trabajo

En esta sección se desarrollará el plan que se utilizará para alcanzar el estado deseado por parte del centro de operaciones de la empresa. Este plan de trabajo va estar dividido en dos secciones, en la primera se van a detallar las actividades categorizadas por prioridad, las cuales se proponen a la empresa para alcanzar el estado deseado, y en la segunda estará detallado el plan de trabajo del estudiante, con el fin de colaborar con la empresa para lograr el estado deseado.

Para la elaboración de este apartado (V.2. Plan de trabajo), se establecieron cada una de las actividades propuestas para alcanzar un estado avanzado de ALM, y se determinó la prioridad de cada una de ellas, una vez desarrollado lo anterior se realizó una sesión con expertos según lo explicado en el apartado III.3.2. Juicio de Expertos, lo que da como resultado el Apéndice 13, esto con el propósito de definir las prioridades de la empresa, según los puntos de mejora citados en la sección V.1. Propuesta General, por lo que seguidamente se mostrará el resultado de priorizar las actividades.

La implementación y desarrollo de las demás actividades del plan de trabajo quedan abiertas a que la empresa las desarrolle con sus recursos, mediante este plan se busca que la empresa tenga identificadas cada una de las actividades necesarias para alcanzar el estado deseado, pero depende de la misma la ejecución, dado que está es una propuesta por parte del estudiante.

Cabe resaltar que se definió que la empresa desea llegar todas las etapas de ALM a un estado avanzado, según lo expuesto en el apartado IV.1 (Análisis de la situación deseada), como primer paso, una vez finalizada este paso la empresa buscara alcanzar el estado dinámico con sus propios recursos.

V.2.1. Plan de trabajo general

En este plan de trabajo se detallan todas las acciones propuestas como necesarias para lograr alcanzar el estado avanzado en cada una de las etapas de ALM, estas acciones fueron categorizadas por prioridad con base en lo que la empresa, expertos y Microsoft consideren más importante. Por medio de la Tabla 19 se especifica el plan de trabajo general.

Tabla 19 Recomendaciones priorizadas.

Área	Plan de acción / Recomendación	Prioridad
General	Creación de Comité de Innovación.	Media
	Retrospectivas al finalizar ciclos de desarrollo por parte del usuario, para evaluar los puntos fuertes y los puntos de mejora.	Alta
	Definir un esquema de privilegios que va tener la fábrica para temas de configuración y aprovechamiento de la plataforma.	Alta
Arquitectura y diseño	Creación de equipo de arquitectos encargado de documentar paulatinamente la arquitectura por medio de diagramas de secuencia generados por Visual Studio y documentar la comunicación de clases, incluyendo las bases de datos, además de la creación de sitios en <i>SharePoint</i> exclusivos para esto.	Baja
	Integrar el equipo de QA y arquitectos para poner a prueba la arquitectura o nuevos componentes por medio de pruebas de estrés y rendimiento.	Media
Ingeniería de requerimientos	Capacitar para utilizar el TFS como medio de administración de requerimientos, además de utilizar herramientas como <i>Storyboarding</i> como apoyo en la gestión de los requisitos.	Alta
	Capacitar a los involucrados en la empresa con respecto a toma de requerimientos y realizar acompañamientos al usuario para la definición de criterios de aceptación del producto.	Alta
	Definir una plantilla de requerimientos funcionales y técnicos.	Alta

Área	Plan de acción / Recomendación	Prioridad
	Realizar e implementar en reportes y <i>dashboards</i> generadores de métricas.	Alta
Desarrollo	Generación de políticas de Código Fuente.	Alta
	Capacitar e implementar revisiones de código, además de la creación de políticas que adapten la gestión de requerimientos y tareas con los desarrollos y la creación de compilaciones automáticas que permitan ser un filtro adicional de la correcta ejecución de los componentes desarrollados.	Media
	Analizar e implementar pruebas unitarias de código que pruebe código, esto para asegurar el correcto funcionamiento de los objetos de <i>software</i> creados.	Media
	Capacitar al equipo en la creación de reportes y <i>dashboards</i> que evidencien el avance, la tendencia o el historial del equipo de desarrollo.	Alta
	Implementar proyectos de bases de datos que permitan versionar el proyecto de base datos y no solo los <i>scripts</i> , para generar <i>scripts</i> incrementales que se apliquen en otros ambientes al momento de realizar la actualización.	Media
	Capacitar sobre la nueva estrategia de Branch y Merge, además de su respectiva implementación, la cual debe estar acorde con la metodología y evitar problemas o conflictos.	Alta
Administración de la configuración	Implementar técnicas de integración de código que permitan integrar diariamente el código, además del etiquetado correspondiente, así como su complementación con las compilaciones para asegurar pequeñas integraciones diarias y no solo una integración de grandes dimensiones que provoque conflictos difíciles de resolver y que demanden mucho tiempo.	Alta
	Realizar un análisis de los datos sensibles y realizar una encriptación para ambientes de desarrollo y pruebas.	Alta

Área	Plan de acción / Recomendación	Prioridad
	Capacitar sobre la implementación de políticas de código fuente que permitan dar trazabilidad a las tareas y revisores de código, además de aprender sobre la utilización de conjuntos de cambios aplazados y gestión de los <i>workspaces</i> para agilizar procesos de desarrollo.	Alta
	Definir plantillas de guías de instalación e incluirlas en <i>SharePoint</i> para estar en constante actualización y que sean de conocimiento del equipo, esto con el fin de desarrollar despliegues en ambientes o cambios en la arquitectura de las soluciones.	Baja
	Realizar una documentación de procesos de configuración, tanto de ambientes como de soluciones, así como planes de <i>rollback</i> .	Baja
Gobierno de TI	Implementar buenas prácticas de algún marco de trabajo que se alinee con la planeación estratégica de la empresa, de modo que facilite el cumplimiento de los objetivos de TI y se mejoren los procesos.	Baja
	A partir de la información registrada en TFS, establecer métricas o indicadores que evalúen o permitan mejorar el servicio brindado por la fábrica, esto para establecer metas a corto y mediano plazo.	Media
	Realizar una integración del Project Server con el TFS con el fin de integrar el portafolio de proyectos con la ejecución de estos, lo cual al mismo tiempo genere métricas o indicadores clave.	Media
	Definir un proceso de gestión de riesgos a nivel del centro de operaciones para centrar esfuerzos en la mitigación de estos motivando el mejoramiento continuo.	Media
Implementación y operaciones	Brindar capacitación para generar tareas a nivel de TFS de implementación, de manera que en estas se documente y lleve historial de los posibles cambios, así como la idea de que dichas tareas sean hijas de un requerimiento, así se puede tener conocimiento acerca de si los requerimientos ya han sido implementados. Es importante, además, definir la información que debe contener dicha tarea.	Baja

Área	Plan de acción / Recomendación	Prioridad
	Implementación de sistemas de monitoreo de ambientes, integrados con la plataforma de TFS por medio de SCVMM, los cuales generen información útil para el área de operaciones y den retroalimentación al desarrollo para mejoramiento de rendimiento o situaciones específicas de las soluciones creadas.	Baja
	Documentar y definir políticas para el <i>deployment</i> .	Alta
	Capacitar y definir el proceso de gestión de incidentes reportados en producción, el cual se integre con la estrategia de Branch y Merge para la sincronización a tiempo de la resolución de estos. Además de implementar las vías correctas de resolución por medio de la priorización de los conflictos o incidentes.	Media
Aseguramiento de calidad	Capacitar al equipo de aseguramiento de calidad en el uso correcto de la herramienta Test Manager, así como desarrollar buenas prácticas para la ejecución de pruebas, definir estrategias de pruebas de regresión y promover la estandarización de la definición y ejecución de casos de prueba.	Alta
	Capacitar al equipo de <i>testing</i> en la automatización de casos de prueba a partir de pruebas manuales, así como a partir de pruebas de rendimiento y carga, además de incentivar el uso de pruebas unitarias de código.	Alta
	Definir los criterios para indicar la prioridad de casos de prueba. De igual manera, fortalecer el cumplimiento de documentación de los criterios de aceptación.	Alta
	Crear reportes de aseguramiento de calidad sobre avances en ejecución de casos de prueba, al igual que seguimiento de <i>bugs</i> y estado de aplicación en pruebas. Igualmente, se deberá tomar en cuenta la tendencia del equipo en cuanto a generación de inconsistencias.	Alta
	Implementar umbrales de aceptación de producto para medir la efectividad de <i>testing</i> , además de promover una cultura de cero errores a partir de políticas de aceptación de producto definidas por <i>testing</i> .	Media
	Configurar los planes de prueba con los <i>builds</i> automáticos, para tener certeza de la versión que se está probando.	Media

Área	Plan de acción / Recomendación	Prioridad
	Definir metodología para pruebas técnicas y funcionales.	Alta
Planificación y administración de proyectos	Definir proceso de gestión de riesgos.	Alta
	Generar políticas para la creación y almacenamiento de conocimiento.	Alta
	Incluir paulatinamente a ciertos usuarios en los procesos de planeación o retrospectiva, de manera que trabajen cerca de la planeación y conozcan las necesidades en las que él mismo podría ayudar a mover obstáculos.	Media
	Capacitar y enseñar los valores claves en el éxito de las metodologías ágiles y hacerlas comunes en el día tras día de la empresa (focalización de ideas, compromiso, apertura a cambios, coraje para afrontar los retos, iniciativa, auto-organización).	Baja
	Creación de plantilla para <i>dashboards</i> gerenciales con el fin de dar seguimiento de los proyectos.	Alta
	Generar métricas del negocio para poder medir y saber cómo se están haciendo las cosas en el centro de operaciones.	Alta

Fuente: Elaboración propia, 2016.

Con base en la Tabla 19, y juicios de expertos, se definieron las actividades por realizar en este proyecto, con el fin de ayudar a conseguir el nivel deseado de ALM. Las actividades que se definieron para realizar y que son de prioridad alta para la empresa son:

- Generación de políticas de código fuente.
- Definición de estrategia de Merge y Branch de códigos fuentes para atención de casos.
- Creación de Política de Manejo de Código Fuente para Bases de Datos y Metodología de Trabajo.
- Elaboración de plantilla de requerimientos funcionales y técnicos.
- Definición del proceso de gestión de riesgos.
- Implementación de metodología para pruebas técnicas y funcionales.
- Generación de política para la creación y almacenamiento de conocimiento.
- Documentación y definición de políticas para el *deployment*.
- Generación de métricas del negocio.

V.2.2. Plan de trabajo del estudiante

En la Tabla 20 se detalla el plan de trabajo del estudiante con las actividades que se definieron realizar.

Tabla 20 Plan de trabajo.

Acción	Tiempo	¿Qué se va hacer?	¿Cómo se va hacer?	¿Con quién se va hacer?
Generación de políticas de código fuente.	6 días	Se va a desarrollar una política general, con los aspectos fundamentales e instrucciones claras de cómo generar código fuente.	Se harán discusiones grupales para citar puntos importantes, además de que se tomará en cuenta la opinión de los expertos, unido a esto se va hacer una revisión documental para conocer los aspectos más importantes a tomar en cuenta a la hora de generar una política de código fuente.	<ul style="list-style-type: none"> • Ingenieros seniors. • Director del centro de operaciones. • Administrador e integrador de aplicaciones.
Definir estrategia de Merge y Branch de códigos fuentes para atención de casos.	6 días	Se va a definir una estrategia para los <i>merge</i> y <i>branch</i> de códigos fuentes, en la que se van a incluir los <i>branch</i> que van a existir y la funcionalidad de cada uno de ellos.	Se harán revisiones documentales para conocer las mejores estrategias, además de validar lo que Microsoft propone para su herramienta de TFS. Y con este conocimiento se va a proponer la estrategia que la empresa debería implementar.	<ul style="list-style-type: none"> • Administrador e integrador de aplicaciones.
Definir estrategia de manejo de código fuente para bases de datos y metodología de trabajo.	5 días	Se va definir una estrategia o política que expone la metodología de desarrollo para objetos o cambios con base de datos.	Se harán revisiones documentales para conocer las mejores estrategias, además de validar lo que Microsoft propone para su herramienta de TFS. Y con este	<ul style="list-style-type: none"> • Administrador e integrador de aplicaciones.

Acción	Tiempo	¿Qué se va hacer?	¿Cómo se va hacer?	¿Con quién se va hacer?
			conocimiento se va a proponer la estrategia que la empresa debería de implementar.	
Plantilla de requerimientos funcionales y técnicos.	5 días	Se van a definir las secciones principales con las que debe contar cualquier toma de requerimientos, tanto para requerimientos funcionales como para los técnicos.	Para esta tarea se harán revisiones documentales, tanto de libros como de empresas que ya tienen plantillas. También, se investigará lo que tiene TFS como definición de requerimientos y así se adecuarán las plantillas a los datos que se solicitan en esta herramienta.	<ul style="list-style-type: none"> • Director del centro de operaciones. • Administrador e integrador de aplicaciones.
Definir proceso de gestión de riesgos.	5 días	Se va a realizar un proceso general acerca de cómo dar seguimiento a los riesgos por los que puede pasar la empresa en un proyecto. Y con esto lograr mitigarlos.	Se harán discusiones grupales con los ingenieros y expertos para tomar en cuenta los puntos importantes con base en su experiencia.	<ul style="list-style-type: none"> • Ingenieros de <i>software</i>. • Director del centro de operaciones.
Metodología para pruebas técnicas y funcionales.	6 días	Se va a definir una metodología para la ejecución de las pruebas técnicas y funcionales para los principales productos de la empresa, de manera que se pueda garantizar la entrega de <i>software</i> de alta calidad.	Se harán discusiones con los ingenieros que se han encargado de la calidad de los productos y con el director del centro de operaciones, esto para definir consideraciones generales que se deben de considerar en la metodología propuesta.	<ul style="list-style-type: none"> • Ingenieros de <i>software</i> (Control de Calidad). • Director del centro de operaciones.
Generar política para la creación y almacenamiento de conocimiento.	4 días	Se va a establecer una política para generar una base de datos de conocimiento que permita	Se harán discusiones grupales para citar puntos importantes, además de que se tomara en cuenta la opinión de los expertos,	<ul style="list-style-type: none"> • Ingenieros seniors.

Acción	Tiempo	¿Qué se va hacer?	¿Cómo se va hacer?	¿Con quién se va hacer?
		compartir el mismo entre todos los ingenieros de la empresa.	unido a esto se va hacer una revisión documental para conocer los aspectos más importantes por tomar en cuenta a la hora de la creación y almacenamiento de conocimiento.	<ul style="list-style-type: none"> • Director del centro de operaciones. • Administrador e integrador de aplicaciones.
Documentar y definir políticas para el <i>deployment</i> .	6 días	Se van a establecer las políticas y las pautas que se deben seguir en el <i>deployment</i> de los aplicativos, de manera que se minimicen al máximo los incidentes que se puedan generar debido a malas prácticas en las publicaciones y/o configuraciones de los componentes de un producto.	Para esta tarea se hará revisiones documentales y se tomará en cuenta el juicio de expertos y de los ingenieros de la empresa. También se investigará lo que tiene TFS para los <i>deployment</i> y así adecuar las políticas o pautas a lo que la herramienta requiere o concede.	<ul style="list-style-type: none"> • Ingenieros seniors. • Director del centro de operaciones. • Administrador e integrador de aplicaciones.
Generar métricas del negocio.	4 días	Se van a desarrollar métricas para medir el desempeño del negocio en cuanto a las diferentes etapas de ALM.	Se van a definir las métricas con base en los conocimientos adquiridos y una vez que están definidas se van a exponer al director del centro de operaciones para validar cualquier consideración adicional que se deba tener.	<ul style="list-style-type: none"> • Director del centro de operaciones.

Fuente: Elaboración propia, 2016.

V.3. Estándares, políticas, metodologías u otros.

En esta sección se desarrollarán los estándares, políticas, metodologías u otras actividades de diferentes procesos, que se realizaron para acercar al centro de operaciones al estado deseado en ALM. Los puntos que se realizaron son los nombrados en la sección del V.2.2. Plan de trabajo del estudiante, por lo que esta sección estará dividida en cada una de las propuestas.

Hay que tomar en cuenta que, para esta sección, en algunos casos, no se crearon plantillas (Word) para los procesos, dado que la información, por lo general, se va a registrar en el TFS, por lo que no es necesario tener una plantilla esquematizada, más bien lo que se necesita es conocer los datos necesarios que se deben recolectar o el proceso requerido para cada una de las acciones.

Para el desarrollo de esta sección se realizaron revisiones documentales según lo expuesto en el apartado III.3.1 (Revisión documental) y unido al conocimiento del estudiante se desarrollaron cada una de las actividades, estas actividades fueron sometidas a validez por parte de los expertos (III.3.2. Juicio de Expertos), el cual se presenta en el Apéndice 14, teniendo en cuenta las opiniones del juicio de expertos se aplicaron las mejoras y se tuvo como resultado lo presentado a continuación.

V.3.1. Política de código fuente

Este apartado tiene como objetivo principal establecer los estándares necesarios en el proceso de desarrollo de sistemas en la empresa, de manera que exista una política que permita facilitar el entendimiento del código fuente por cualquier ingeniero del centro de operaciones, y a su vez, que permita darle un mejor mantenimiento a este.

Las directrices propuestas en esta sección aplican para todos los nuevos desarrollos sin excepción alguna.

Uno de los aspectos importantes por tener en cuenta es que, con respecto a la empresa, la mayoría de sus proyectos son realizados en .NET por lo que muchas de los estándares propuestos estarán dirigidos a este lenguaje *framework*.

V.3.1.1. Estándares de programación para .NET

V.3.1.1.1 Estándares generales

Los estándares generales propuestos para la programación en .NET Framework son los siguientes:

- 1) La versión de .NET Framework por utilizarse debe ser 4.0 o superior.
- 2) Todos los nuevos desarrollos deben realizarse sobre la herramienta Microsoft Visual Studio 2012 o superior.
- 3) El lenguaje de programación por utilizar debe ser C#.
- 4) El acceso a las bases de datos debe realizarse solamente a través de la tecnología *Entity Framework* de Microsoft.

V.3.1.1.2. Estándares de nomenclatura

Esta sección contiene los estándares propuestos para la nomenclatura en .NET Framework.

V.3.1.1.2.1 Nombres de soluciones y proyectos

Los nombres de las soluciones deberán llevar el nombre del producto. Si la extensión del nombre del producto es mayor a los 20 caracteres, se deberá utilizar sus siglas o, en su defecto, un acrónimo que identifique el producto.

En cuanto a los nombres de los proyectos, estos quedan a criterio del desarrollador, pero debe ser un nombre significativo y único que identifique al proyecto.

V.3.1.1.2.2 Nombres de clases

Los nombres de las clases deben tener el siguiente formato:

“CL_” + <Nombre descriptivo>

El nombre descriptivo de la clase debe hacer referencia al nombre de la clase, y la primera letra debe ser en mayúscula.

V.3.1.1.2.3 Documentación interna de clases

Los estándares para la documentación interna de las clases son las siguientes:

- 1) Todas las clases deben iniciar con una sección (región llamada “Descripción”) que contenga los siguientes valores:
 - Nombre del proyecto
 - Nombre de la clase
 - Descripción breve de la clase
 - Nombre del ingeniero que la crea
 - Fecha de creación de la clase

Ejemplo:

```
#region <Descripción>
////////////////////////////////////
// Proyecto: CGP
// Clase: CL_Transaccion
// Descripción: Clase que contiene la información de las
// transacciones
// Ingeniero: Walter Cordero Ureña
// Fecha: 27/10/2016
////////////////////////////////////
#endregion
```

2) Todos los métodos que contenga la clase deben ser documentados utilizando el estándar XML de .NET Framework y debe contener, como mínimo, los siguientes *tags*:

- Summary: Descripción breve del método.
- Param: Descripción de cada uno de los parámetros que reciba.
- Returns: Descripción del valor de retorno.

V.3.1.1.2.4 Estructura de clases

Todas las clases desarrolladas deben tener la siguiente estructura, manteniendo el orden establecido:

- 1) Descripción de la clase (ver la sección V.3.1.1.2.3).
- 2) Importación de *namespaces*.
- 3) Definición de *namespace*.
- 4) Definición de la clase.
- 5) Propiedades (debe existir una región con este nombre).
- 6) Variables privadas (debe existir una región con este nombre).
- 7) Constructores (debe existir una región con este nombre).
- 8) Métodos (debe existir una región con este nombre).

V.3.1.1.2.5 Nombres de variables y objetos

Los nombres de las variables privadas deben tener el siguiente formato:

“_” +< Nombre descriptivo>

El nombre descriptivo de la variable debe hacer referencia a la función específica que se realizará con esa variable, y la primera letra debe ser en mayúscula.

V.3.1.1.2.6 Nombres de métodos y propiedades

Los nombres de los métodos y de las propiedades de una clase deben iniciar con mayúscula y el resto debe ser en minúscula, pues de esta manera sigue el estándar de C#.

V.3.1.1.2.7 Nombres de parámetros

Los nombres de los parámetros deben tener el siguiente formato:

"p" + <Nombre descriptivo>

El nombre descriptivo del parámetro debe hacer referencia a la función específica que se realizará con este, y la primera letra debe ser en mayúscula.

V.3.1.2. Estándares de programación para base de datos

V.3.1.2.1. Estándares generales

Los estándares generales de programación para base de datos son los siguientes:

- 1) Todos los nuevos desarrollos deben realizarse sobre la herramienta Microsoft SQL Server Management Studio 2012 o superior.
- 2) Debe existir un proyecto de base de datos en Microsoft Visual Studio 2012 o superior. Dicho proyecto debe estar debidamente versionado en *Team Foundation Services* (ver sección V.3.3.)

V.3.1.2.2. Estándares de nomenclatura

Las normas generales que se deben utilizar para definir los nombres de los objetos incluidos en la base de datos de un sistema son las siguientes:

- 1) Los nombres de tablas y vistas deben tener el siguiente formato:

<Tipo de objeto> + <Nombre descriptivo>

Donde:

- Tipo de objeto: Este valor puede tener solamente 2 valores, T para tablas y V para vistas.
- Nombre descriptivo: Nombre que identifica la naturaleza del objeto, debe iniciar siempre con mayúscula y las demás letras en minúscula.

- 2) Los nombres de los campos deben seguir el siguiente formato:

<Tipo de campo> + “_” + <Nombre descriptivo>

Donde:

- Tipo de campo: Corresponde a un código de caracteres que indica el uso del campo de acuerdo a la Tabla 21:

Tabla 21 Tipo campo.

Código	Tipo de campo
Id	Identificador
Can	Cantidad
Cod	Código
Dir	Dirección
Dsc	Descripción
Fec	Fecha
Img	Imagen
Mon	Monto
Nom	Nombre
Num	Número
Txt	Texto

Fuente: Elaboración propia, 2016.

- Nombre descriptivo: Nombre que identifica la naturaleza del campo, debe iniciar siempre con mayúscula y las demás letras deberán escribirse en minúscula.

V.3.1.2.3. Estándares para TRANSACT-SQL

Los estándares de programación para las sentencias de TRANSACT-SQL son los siguientes:

- 1) Todas las palabras reservadas se deben escribir en mayúscula.
- 2) Los nombres de las tablas se rigen por los estándares de nomenclatura establecidos (ver la sección V.3.1.2.2).
- 3) Los nombres de procedimientos almacenados deben seguir el siguiente formato:

“pr_” + <objeto que afecta> + <función que cumple>

En caso de que el procedimiento almacenado afecte varios objetos, puede omitirse el nombre del objeto que afecta. Por lo que el formato sería el siguiente:

“pr_” + <función que cumple>

4) Todos los procedimientos almacenados deben tener un encabezado que contenga los siguientes valores:

- Nombre del procedimiento almacenado
- Función que realiza
- Lista de parámetros y su descripción
- Lista de valores de retorno y su descripción
- Fecha de creación

Ejemplo:

```
-----  
-- Nombre: pr_GenerarDevoluciones  
-- Función: Procedimiento que genera las devoluciones de un ciclo  
-- Recibe: @FecCiclo DATETIME – Fecha del ciclo  
-- Retorna: Lista transacciones  
-- Fecha: 04/11/2016  
-----
```

5) Los parámetros de los procedimientos almacenados deben declararse uno por línea acompañados de su respectivo tipo. El nombre debe tener el siguiente formato:

“p_” + <nombre descriptivo>

El nombre descriptivo del parámetro debe hacer referencia a la función específica que se realizará con este, y la primera letra debe ser en mayúscula.

6) Para cualquier sentencia SQL, la cláusula inicial funcionará como encabezado y cualquier otra cláusula de la sentencia tendrá una alineación de un “tab” con respecto a la cláusula inicial.

V.3.1.3. Estándar de nomenclatura de objetos de la interfaz

Los nombres de los objetos deben iniciar con las letras que identifiquen el objeto al que hacen referencia, seguido por un nombre significativo de acuerdo con el tipo de dato mostrado, el cual debe iniciar siempre con mayúscula y las demás letras deberán estar escritas en minúscula.

En la Tabla 22, se detallan los objetos de mayor importancia:

Tabla 22 Objetos interfaz.

Objeto	Nombre
Cuadro de texto (<i>TextBox</i>)	txt + Nombre campo
Botones	btn + Nombre de la acción que realiza el botón
<i>Radio Buttons</i>	rb + Nombre del campo
<i>Check Boxes</i>	chk + Nombre del campo
<i>DropDownList</i>	drp + Nombre del campo
<i>Label</i>	lbl + Nombre de la etiqueta
<i>Link Button</i>	lnk + Nombre del link
<i>DataGrid</i>	dg + Nombre de los datos que muestre el gris
<i>Image</i>	img + Nombre de la imagen
<i>Calendar</i>	cld + Nombre significativo de la fecha a la que se refiere
<i>ListBox</i>	lst + Nombre significativo de la lista
<i>RequiredFieldValidator</i>	rfl + Nombre significativo del campo a validar

Fuente: Elaboración propia, 2016.

V.3.2. Estrategia de Merge y Branch de códigos fuentes para atención de casos

A continuación, se presenta una guía rápida en la que se expone la metodología de desarrollo para potencializar el uso del explorador de control de código fuente que ofrece el Team Foundation Server, además de las políticas y metodologías por implementar, con el fin de reestructurar y ordenar el manejo de las fuentes con los diferentes clientes, en pro de una respuesta efectiva a las necesidades.

Se ha identificado que, a nivel de centro de operaciones, la metodología de desarrollo de *software* se divide en tres grandes grupos (*Hotfix*, Casos y Requerimientos), según la necesidad del cliente, estas se describen posteriormente unidas a la forma de trabajar en cada área, de manera que se puedan ordenar estas áreas bajo una estandarización de los procesos.

Tomando en cuenta los grupos de tipos de casos, se describen las metodologías de desarrollo para la administración de código fuente.

Importante: Para entender los gráficos metodológicos se indica en la Ilustración 10 los diferentes actores.

Ilustración 10 Actores merge y branch.



Cliente



Coordinador de Información y
Respuesta Operativa



Desarrolladores



Integrador de
Aplicaciones

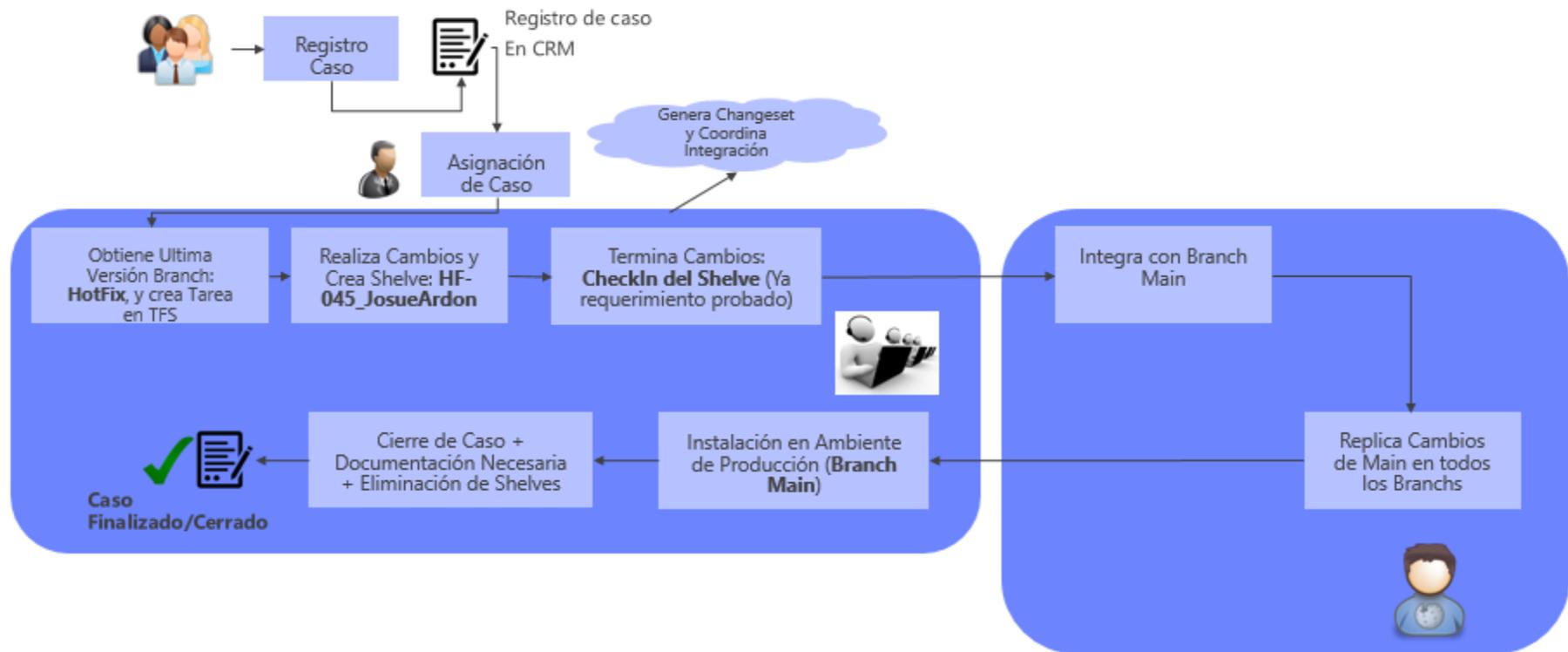
Fuente: Elaboración propia, 2016.

V.3.2.1. Tipo de Caso: HotFix

Se utiliza para atención de casos de carácter urgente y críticos, los cuales provocan bloqueos en la ejecución del negocio del cliente, dichos casos deben ser atendidos inmediatamente y no pueden durar más de 24 horas sin atenderse, una característica es que el pase a ambientes de producción debe ser casi inmediato.

En el Gráfico 13 se muestra la propuesta de la estrategia para el grupo *HotFix*.

Gráfico 13 Estrategia HotFix.



Fuente: Elaboración propia, 2016.

En la Tabla 23 se va a presentar una explicación con respecto a los actores de la metodología propuesta para el grupo de *Hotfix*.

Tabla 23 Estrategia HotFix.

Actor	Descripción
Cliente	Inicialmente, el proceso comienza cuando el cliente detecta alguna inconsistencia del producto en funcionamiento, posteriormente reporta el caso a través de la plataforma del CRM.
Coordinador de información y respuesta operativa	Una vez se detecta el caso, este actor procede a evaluar la criticidad del caso, en cuanto evalúa y se da cuenta de que corresponde a un caso de tipo <i>Hotfix</i> , realiza la asignación del caso, y de manera paralela en el proceso lleva a cabo su seguimiento.
Desarrolladores	<p>Una vez que el caso es asignado al equipo de desarrollo se disparan varias actividades:</p> <ul style="list-style-type: none"> - Inicialmente, el desarrollador debe obtener la última versión, del <i>branch</i> de <i>Hotfix</i>, y crear una tarea en TFS, en la cual se documente el trabajo por realizar. - Seguidamente, se trabaja en los cambios necesarios para la atención del incidente, esto llevando a cabo la modificación de los objetos necesarios. - Se crea el <i>shelve</i>, con el siguiente formato: HF-[CódigoCaso]-[usuarioempresa]. Ejemplo: HF-CAS045-corderouw. - Se realizan las pruebas necesarias para asegurar que el cambio se implementó correctamente. - Una vez asegurado el cambio, se procede a realizar un <i>checkin</i> del <i>shelve</i>.

Actor	Descripción
	<ul style="list-style-type: none"> - Se debe coordinar con el integrador de aplicaciones la integración con los otros <i>branch</i>, especialmente producción. - Una vez integrado con el <i>Branch</i> de Producción, se deben obtener las fuentes para realizar la actualización en el ambiente del cliente. - Cuando ya la actualización se haya terminado exitosamente se debe realizar el cierre del caso, generar la documentación necesaria y, por último, eliminar el <i>shelve</i> creado, pues el cambio ya ha sido replicado en todos los <i>branches</i>.
Integrador de aplicaciones	Una vez que se realiza un <i>checkin</i> en el <i>branch</i> de <i>hotfix</i> , se coordina con el integrador de aplicaciones, el trabajo de este es integrar los cambios realizados con el <i>branch</i> de producción y replicar los cambios en todos los otros <i>branches</i> para mantener el código actualizado.

Fuente: Elaboración propia, 2016.

En resumen, los pasos a seguir para los casos de tipo *Hotfix* son los siguientes:

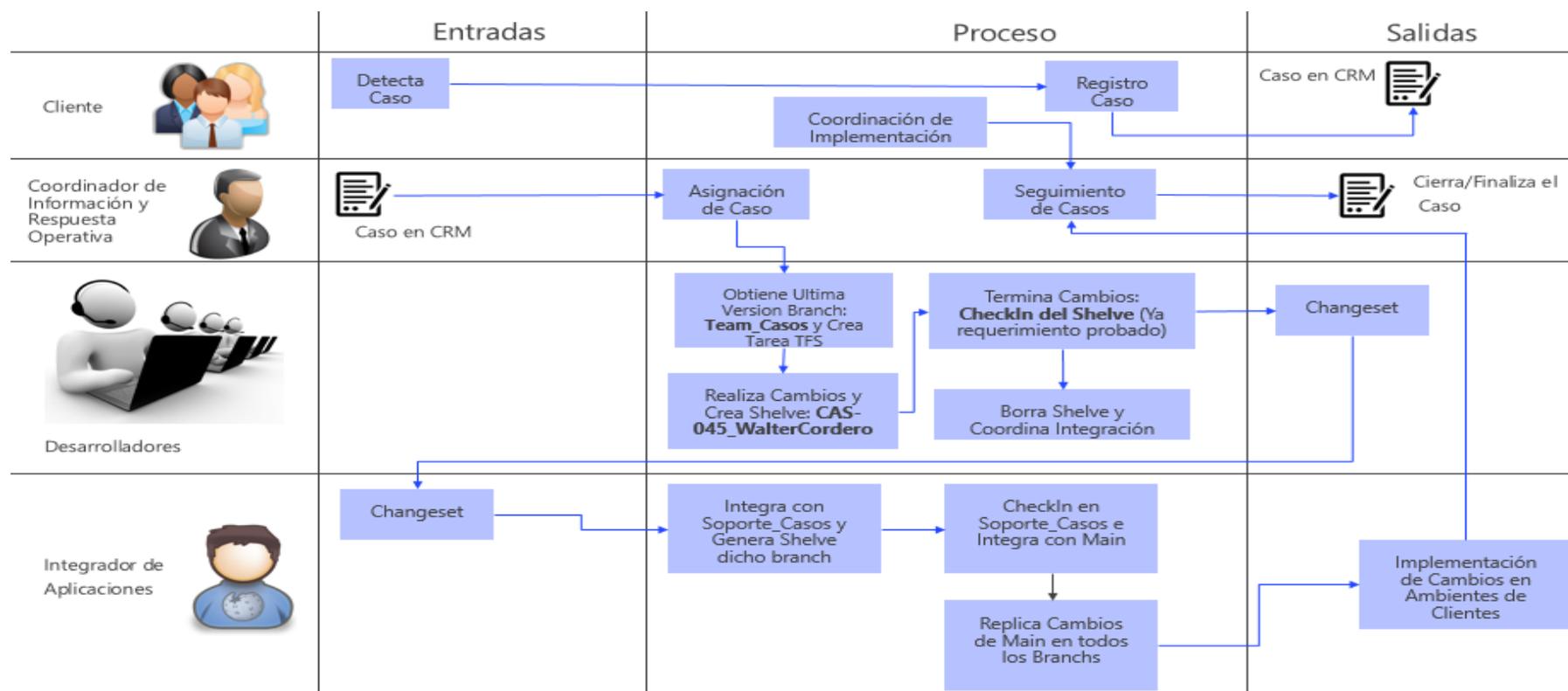
- Obtener la última versión del *branch* de *HotFix* y crea la tarea en TFS.
- Crear *Shelve*, realizar cambios, y generar *re-shelve* de los cambios (HF-[CódigoCaso]-[usuarioempresa]). Ejemplo: HF-CAS045-corderouw).
- Cuando termina los cambios: Obtiene la última versión del *branch*, compila localmente y seguidamente realiza *Checkin* del *Shelve*, esto siempre tomando en cuenta las políticas necesarias (Comentario de Cambios, asociar tarea creada en TFS).
- Una vez que el integrador de aplicaciones integra con el *branch* de producción, se obtienen los cambios, es decir, la última versión de producción. Igualmente se generan los publicados y se actualiza el ambiente de producción con los cambios realizados.
- Se cierra la tarea de TFS y se genera la documentación necesaria para dar por terminado el caso.

V.3.2.2. Tipo de caso: Casos normales

Corresponde a la atención de casos que el cliente normalmente ingresa en la plataforma del CRM y estos son los casos que pueden durar un día o más tiempo; sin embargo, no afectan la operativa del negocio del cliente, se caracterizan porque los pases a ambientes de producción pueden ser coordinados o aprobados por el cliente.

En el Gráfico 14, se muestra la propuesta de la estrategia para estos tipos de casos.

Gráfico 14 Estrategia casos normales.



Fuente: Elaboración propia, 2016.

A continuación, se va a presentar en la Tabla 24, una explicación con respecto a los actores de la metodología propuesta estos casos.

Tabla 24 Estrategia Casos Normales.

Actor	Descripción
Cliente	Inicialmente, el proceso comienza cuando el cliente detecta alguna inconsistencia del producto en funcionamiento, posteriormente este reporta el caso a través de la plataforma del CRM.
Coordinador de información y respuesta operativa	Una vez se detecta el caso, este actor procede a evaluar la criticidad del caso. En cuanto evalúa y se da cuenta de que corresponde a un caso normal, realiza la asignación del caso y, de manera paralela en el proceso, lleva acabo su seguimiento.
Desarrolladores	<p>Una vez que el caso es asignado al equipo de desarrollo, se disparan varias actividades:</p> <ul style="list-style-type: none"> • Inicialmente, el desarrollador debe obtener la última versión del <i>branch</i> de Team_Casos, y crear una tarea en TFS, donde se documente el trabajo por realizar. • Seguidamente, se trabaja en los cambios necesarios para la atención del incidente, en este punto se realizan las modificaciones necesarias. • Se crea el <i>shelve</i>, con el siguiente formato: HF-[CódigoCaso]-[usuarioempresa]. Ejemplo: DEVC-CAS045-corderouw. • Se realizan las pruebas necesarias para asegurarse de que el cambio se implementó correctamente. • Una vez que se asegure el cambio, se procede a realizar un <i>checkin</i> del <i>shelve</i> y seguidamente se borra el <i>shelve</i> recién creado. • Se debe coordinar con el integrador de aplicaciones la integración con los otros <i>branch</i>.

Actor	Descripción
	<ul style="list-style-type: none"> • Una vez que se integra con el <i>Branch</i> de producción, se deben obtener las fuentes para realizar la actualización en el ambiente del cliente. • Cuando ya la actualización se haya terminado exitosamente, se debe realizar el cierre del caso, así como generar la documentación necesaria.
Integrador de aplicaciones	Una vez que se realiza un <i>checkin</i> en el <i>branch</i> de Team_Casos, se coordina con el integrador de aplicaciones, el trabajo de este es integrar los cambios realizados con el <i>branch</i> de Soporte_Casos y una vez que el cliente lo indique se integra con el <i>branch</i> de producción. Por último, se replican los cambios en todos los <i>branches</i> .

Fuente: Elaboración propia, 2016.

En resumen, los pasos que se deben seguir para este tipo de casos son los siguientes:

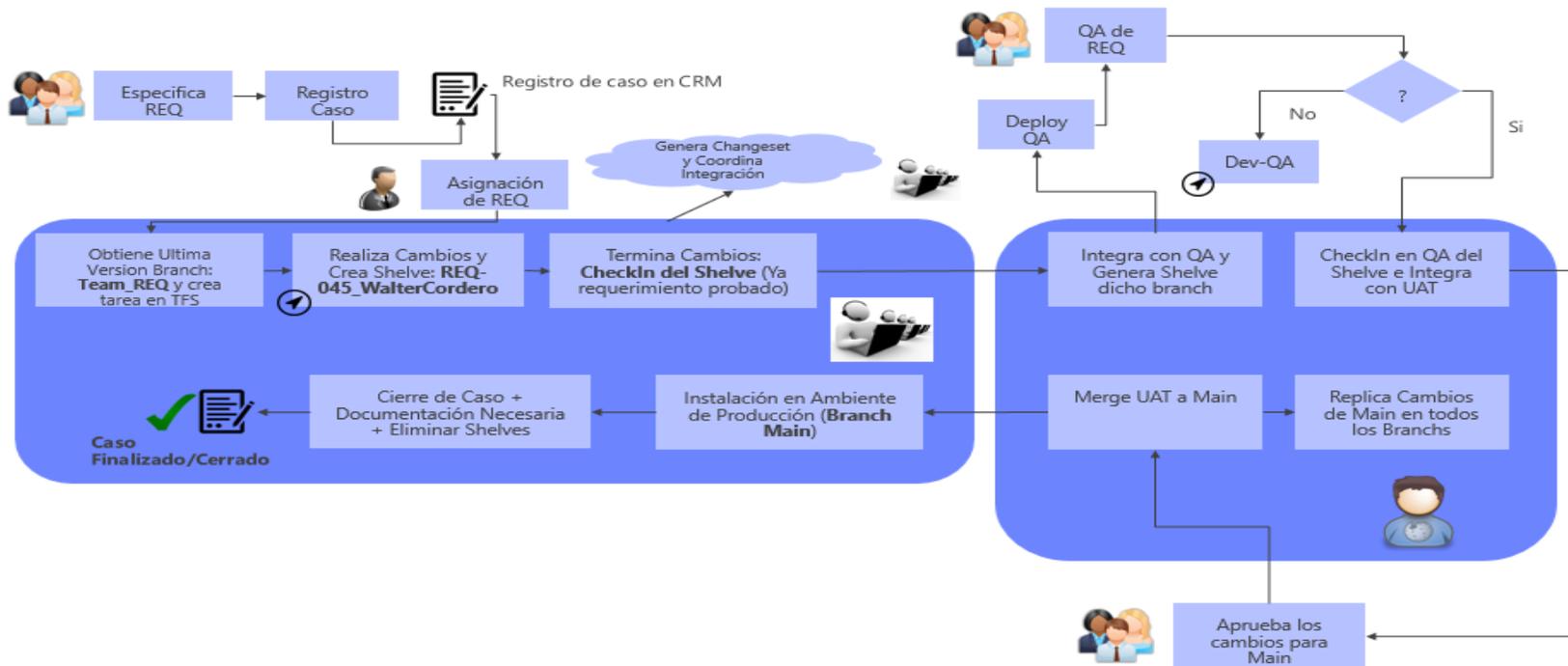
- Obtener la última versión del *branch* de Team_Casos y crea la tarea en TFS.
- Crear *shelve*, realizar cambios, y generar *re-shelve* de los cambios (DEV-[CódigoCaso]-[usuarioempresa]. Ejemplo: DEV-CAS045-corderouw).
- Cuando termina los cambios: Obtiene la última versión del *branch*, compila localmente y seguidamente realiza *checkin* del *shelve*, siempre tomando en cuenta las políticas necesarias (comentario de cambios y asociar tarea creada en TFS).
- Una vez que el integrador de aplicaciones integra el *branch* de producción, se obtienen los cambios, es decir, la última versión de producción, y se generan los publicados, además se actualiza el ambiente de producción con los cambios realizados al coordinar los cambios con el cliente.
- Se cierra la tarea de TFS, se elimina el *shelve* y se genera la documentación necesaria para dar por terminado el caso.

V.3.2.3. Tipo de caso: Requerimientos

Los requerimientos son usualmente versiones mejoradas del sistema base, es decir, nuevas funcionalidades específicas por cada cliente, los cuales registran los casos; pero estos requerimientos tienen una estimación de parte del área de desarrollo y pueden durar varios días, inclusive meses. Este grupo se caracteriza porque estos requerimientos o casos deben pasar por una etapa de aseguramiento de calidad, tanto de desarrollo como por parte del cliente, y eventualmente el pase a producción de estos ítems depende de la aprobación del cliente y no es tan crítica como las anteriores.

En el Gráfico 15 se muestra la propuesta de la estrategia para los requerimientos.

Gráfico 15 Estrategia requerimientos.



Fuente: Elaboración propia, 2016.

Se va a presentar en la Tabla 25, una explicación de acuerdo con los actores de la metodología propuesta para el grupo de requerimientos.

Tabla 25 Estrategia requerimientos.

Actor	Descripción
<p>Cliente</p>	<p>Inicialmente, el proceso comienza cuando el cliente detecta alguna oportunidad de mejora del producto en funcionamiento, de manera que se convierte en un nuevo requerimiento, este reporta el caso a través de la plataforma del CRM.</p> <p>Cuando se realiza la instalación en QA, el cliente debe realizar las pruebas necesarias y aprobar los cambios para coordinar el pase a producción de los desarrollos realizados.</p> <p>Adicionalmente, el cliente debe coordinar y aprobar el pase a producción de los requerimientos desarrollados.</p>
<p>Coordinador de información y respuesta operativa</p>	<p>Una vez se detecta el caso, este actor procede a evaluar su criticidad. En cuanto evalúa y se da cuenta de que corresponde a un requerimiento, realiza la asignación del caso y, de manera paralela en el proceso, lleva a cabo su seguimiento.</p>
<p>Desarrolladores</p>	<p>Una vez que el caso es asignado al equipo de desarrollo se disparan varias actividades:</p> <ul style="list-style-type: none"> • Inicialmente, el desarrollador debe obtener la última versión del <i>branch</i> de Team_REQ y crear una tarea en TFS, por medio de la cual se documente el trabajo por realizar. En este punto es importante definir el <i>branch</i> en el que se va a trabajar, ya que eventualmente para estos casos puede existir un <i>branch</i> por requerimiento en desarrollo (<i>feature branching</i>).

Actor	Descripción
	<ul style="list-style-type: none"> • Seguidamente, se trabaja en los cambios necesarios para la atención del incidente, de manera que se realicen las modificaciones necesarias. • Se crea el <i>shelve</i> con el siguiente formato: DEVR-[CódigoCaso]-[usuarioempresa]. Ejemplo: DEVR-CAS045-corderouw. • Se realizan las pruebas necesarias para asegurarse de que el cambio se implementó correctamente. • Una vez que se asegure el cambio, se procede a realizar un <i>checkin</i> del <i>shelve</i>, y el <i>shelve</i> debe mantenerse registrado hasta que el requerimiento pase a producción. • Se debe coordinar con el integrador de aplicaciones la integración con el <i>branch</i> de QA. • Una vez que se integra con el <i>branch</i> de QA, se deben obtener las fuentes a partir del <i>shelve</i> de QA, esto es necesario para realizar la actualización en el ambiente de QA en el cual el cliente realizará las pruebas. • En el caso de que salgan inconsistencias en la fase de pruebas, las correcciones de estas deben realizarse sobre el <i>shelve</i> del <i>branch</i> de desarrollo de requerimientos (no en el <i>branch</i> de QA) y se repite el proceso desde el primer punto. • Una vez que el integrador de aplicaciones realice el <i>merge</i> con el <i>branch</i> de producción, se deben generar los publicados y actualizar el ambiente de producción del cliente • Cuando ya la actualización se haya terminado exitosamente, se debe realizar el cierre del caso, generar la documentación necesaria y eliminar el <i>shelve</i>.
Integrador de aplicaciones	En la metodología de requerimientos, el integrador de aplicaciones realiza varias actividades:

Actor	Descripción
	<ul style="list-style-type: none"> • Una vez que se realiza un <i>checkin</i> en el <i>branch</i> de Team_REQ, se debe de integrar los cambios realizados con el <i>branch</i> de QA. • En dicho <i>branch</i> se genera un <i>shelve</i> de los cambios con el cual se realiza el <i>deploy</i> en conjunto con el equipo de desarrollo para que el cliente realice las pruebas necesarias. • Cuando el cliente aprueba las pruebas, se realiza el <i>checkin</i> de los cambios en el <i>branch</i> de QA y se integra con el <i>branch</i> de UAT, para esperar la aprobación y pasar a producción. • Una vez que el cliente aprueba el pase a producción, se realiza una integración del <i>branch</i> de UAT con el <i>branch</i> de producción y, seguidamente, se replican los cambios en todos los <i>branches</i>.

Fuente: Elaboración propia, 2016.

En resumen, los pasos que se deben seguir para los requerimientos son los presentados en los siguientes puntos:

- Obtener la última versión del *branch* de Team_REQ y crea la tarea en TFS.
- Crear *shelve*, realizar cambios, y generar *re-shelve* de los cambios (DEV-[CódigoCaso]-[usuarioempresa]. Ejemplo: DEVR-CAS045-corderouw).
- Cuando termina los cambios: Obtiene la última versión del *branch*, compila localmente y seguidamente realiza *checkin* del *shelve*, siempre tomando en cuenta las políticas necesarias (comentario de cambios y asociar tarea creada en TFS).
- Una vez que el integrador de aplicaciones integra con el *branch* de QA, se obtiene el *shelve* creado y se genera el instalador para actualizar el ambiente de pruebas del cliente, con el fin de que se realicen las pruebas necesarias.
- Una vez que el integrador de aplicaciones se integra con el *branch* de producción, se obtiene la última versión de dicho *branch* y se genera el instalador para actualizar el ambiente de producción.
- Se cierra la tarea de TFS, se elimina el *shelve* en el *branch* de desarrollo y se genera la documentación necesaria para dar por terminado el caso.

V.3.2.4. *Branch* administrativos o de punto de control

En la Tabla 26 se detallan los *branch* administrativos propuestos como puntos de control y por seguridad, además de las indicaciones que se deberán seguir para un correcto uso.

Tabla 26 *Branch* administrativos.

Branch	Notas
Main	<ul style="list-style-type: none">• Solamente desde este <i>branch</i> se pueden generar instaladores o publicados para realizar pases a producción.• El único que puede hacer <i>checkin</i> en este <i>branch</i> es el integrador de aplicaciones del centro de operaciones.• Cuando se realiza un <i>checkin</i> en este <i>branch</i>, el integrador de aplicaciones actualizará los otros <i>branches</i>.• Los desarrolladores pueden bajar códigos de esta sección para generar publicados, pero no pueden realizar <i>checkin</i>.
Soporte_casos	<ul style="list-style-type: none">• Solamente el integrador de aplicaciones puede hacer <i>checkin</i> sobre este <i>branch</i>.• En este <i>branch</i> se tienen las fuentes de los casos ya solucionados a la espera de aprobación del cliente para ser publicados en producción.
QA	<ul style="list-style-type: none">• Solamente el integrador de aplicaciones puede hacer <i>checkin</i> sobre este <i>branch</i>.• El formato para manejo de <i>shelves</i> en este <i>branch</i> es: QA-[CódigoCaso]-[usuarioempresa]. Ejemplo: QA-CAS045-corderouw.• En este <i>branch</i> se encuentran las fuentes de los requerimientos separados por <i>shelves</i>.• Las publicaciones para pruebas de usuarios se realizan desde los <i>shelves</i> por requerimiento.

Branch	Notas
	<ul style="list-style-type: none"> • En este <i>branch</i> ningún desarrollador puede editar código, en caso de incidentes, estos se atienden desde el <i>branch</i> de requerimientos con su respectivo <i>shelve</i>.
UAT	<ul style="list-style-type: none"> • Solamente el integrador de aplicaciones puede hacer <i>checkin</i> sobre este <i>branch</i>. • En este <i>branch</i> se encuentran las fuentes de los requerimientos ya probados por el usuario y pendientes de pasar a producción. • Ningún desarrollador puede editar código en este <i>branch</i>.

Fuente: Elaboración propia, 2016.

V.3.3. Estrategia de manejo de código fuente para bases de datos y metodología de trabajo

A continuación, se presenta una guía donde se expone la metodología de desarrollo para objetos o cambios en las bases de datos para potencializar el uso del *Source Control Explorer* que ofrece el Team Foundation Server, además de las políticas y metodologías por implementar, todo esto con el fin de reestructurar, ordenar y estandarizar el manejo de las fuentes con los diferentes clientes, en pro de una respuesta efectiva a las necesidades.

Se ha identificado que, a nivel de centro de operaciones, los cambios en objetos de base de datos se dan bajo una necesidad de cambio del negocio o bien bajo la construcción y actualización de componentes, por lo que hay tropicalizaciones de estructuras de base de datos en diferentes clientes, por esta razón se propone dividir la atención de incidentes de base de datos en dos grupos principales: Estandarización y tropicalización.

- Estandarización: Conjunto de estructuras que comparten todos los clientes. Es la base principal del modelo de base de datos de los productos, cuando se va a realizar una instalación del producto se toman estas fuentes.
- Tropicalizaciones: Corresponde a *scripts* o estructuras adicionales a nivel de los modelos de base de datos, que se han tenido que ajustar según el negocio del cliente, estas difícilmente se repiten en otros clientes, pues son muy propias de las necesidades o iniciativas de cada uno.

Tomando en cuenta los grupos, se describe la metodología de desarrollo para la administración de código fuente de bases de datos.

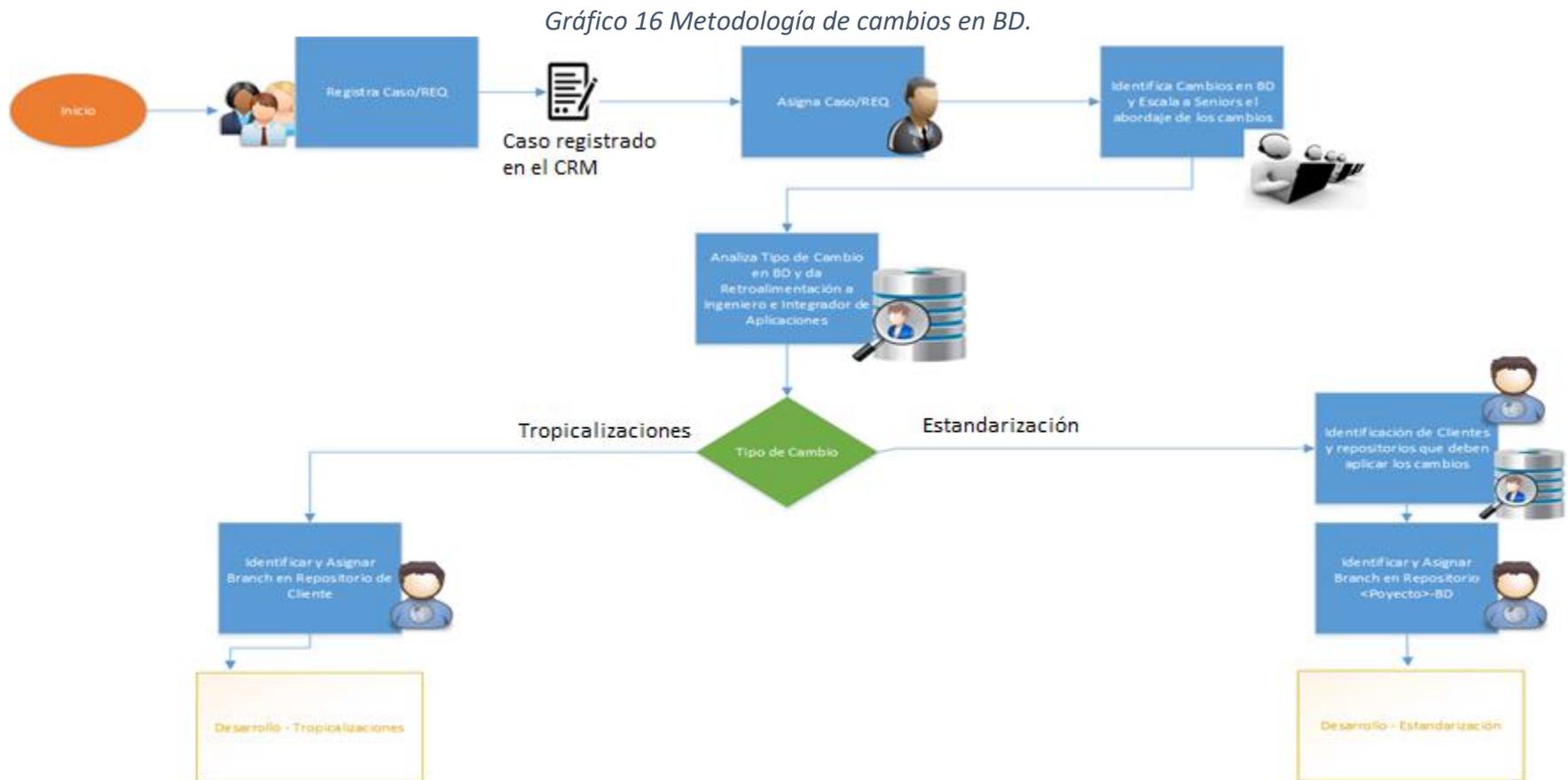
Importante: Para entender los gráficos metodológicos, se indican en la Ilustración 11 los diferentes tipos de actores:



Fuente: Elaboración propia, 2016.

Se mostrarán 3 gráficos en los cuales se detalla la metodología propuesta para el manejo de cambios a nivel de base datos, la cual dependerá del tipo de cambio que sea.

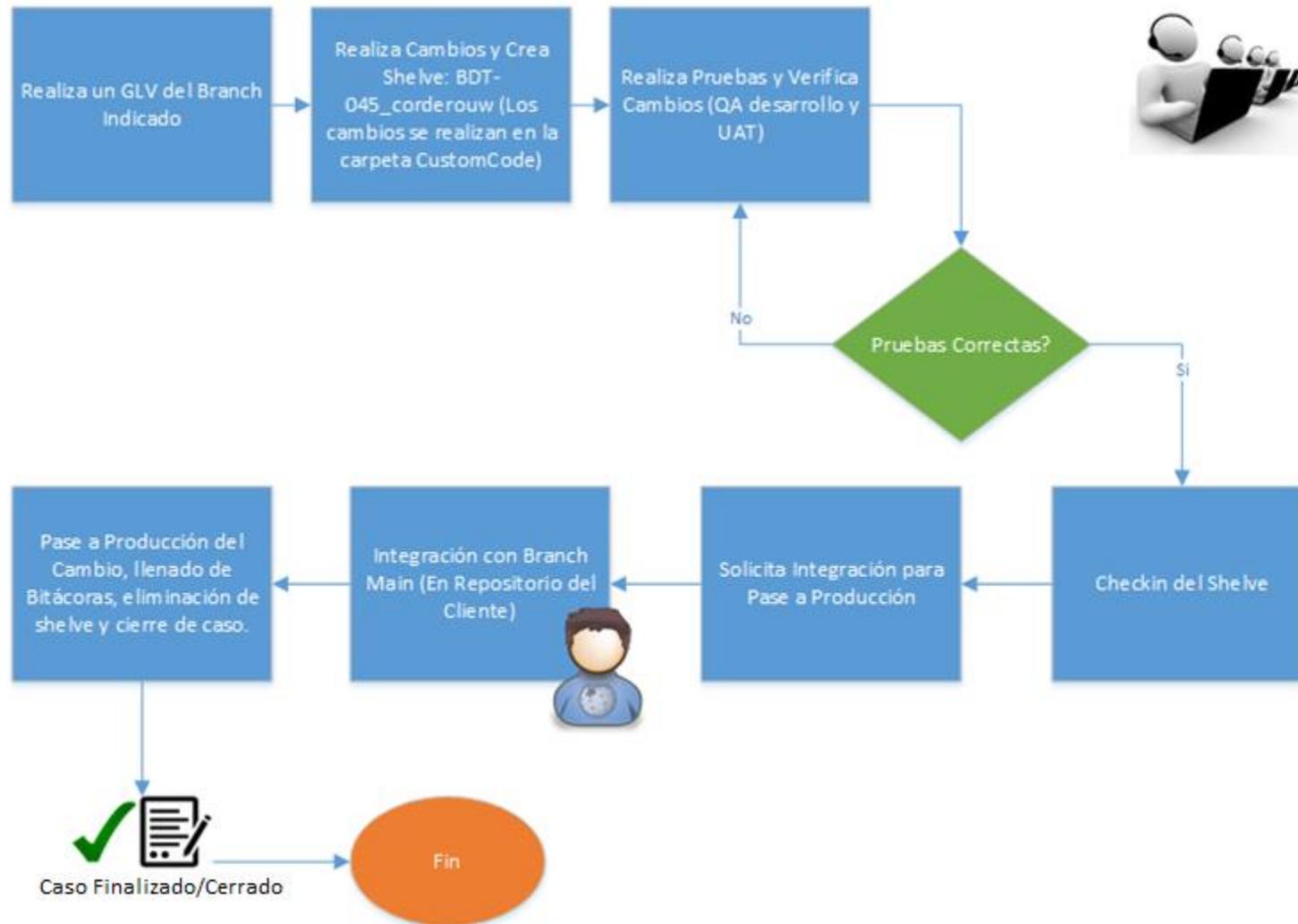
En el Gráfico 16 se va a mostrar el proceso normal de un cambio hasta la etapa en la que se identifica qué tipo de cambio es, según lo descrito anteriormente.



Fuente: Elaboración propia, 2016.

En el Gráfico 17 se desarrolla la parte de los cambios de tipo tropicalizaciones.

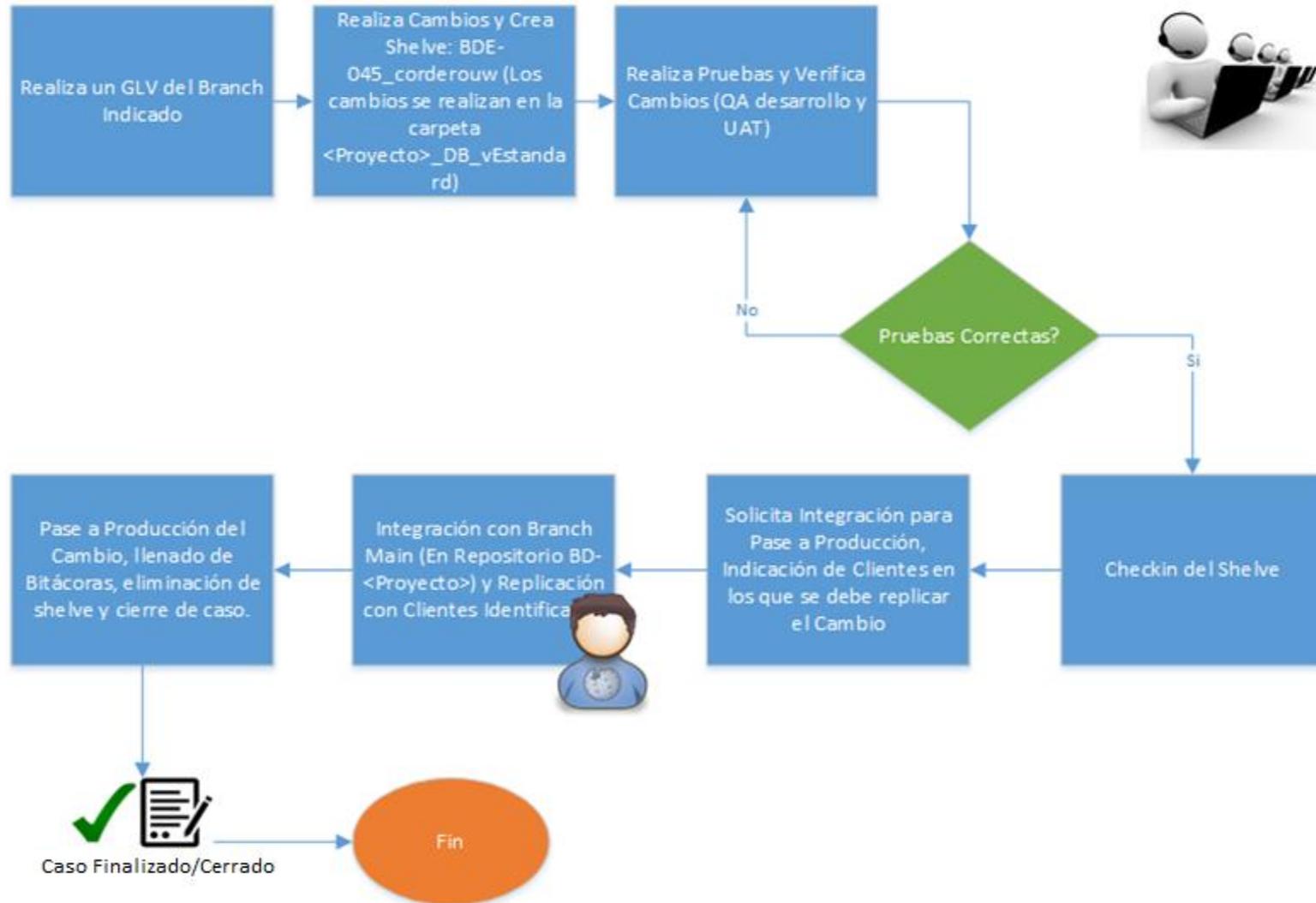
Gráfico 17 BD tropicalizaciones.



Fuente: Elaboración propia, 2016.

En el Gráfico 18 se desarrolla la parte de los cambios de tipo estandarización.

Gráfico 18 BD estandarización.



Fuente: Elaboración propia, 2016.

Se va a presentar, en la Tabla 27, una explicación según los actores de la metodología propuesta para los cambios en las bases de datos.

Tabla 27 Metodología de cambios en BD.

Actor	Descripción
Cliente	Inicialmente el proceso comienza cuando el cliente detecta alguna inconsistencia o requerimiento del producto en funcionamiento, el mismo reporta el caso a través de la plataforma del CRM o los medios brindados.
Coordinador de Información y Respuesta Operativa	Una vez se detecta el caso, este actor procede a evaluar la criticidad del caso (Política de Fuentes del proyecto), en cuanto evalúa y se da cuenta que tipo de caso corresponde, realiza la asignación del caso, y de manera paralela en el proceso, realiza el seguimiento del mismo.
Desarrolladores	<p>Cuando el caso es asignado al equipo de desarrollo varias actividades:</p> <ul style="list-style-type: none"> • Primeramente, el ingeniero realiza un análisis del caso, es en este momento cuando identifica cambios en estructuras de base de datos, seguidamente eleva el caso a los Seniors con el diagnostico previamente realizado para evaluar si corresponde a una tropicalización o a un cambio a nivel de estandarización. • En el caso que corresponda en que se identifica el tipo de cambio se realizan las siguientes actividades. <p>Tropicalización:</p> <ul style="list-style-type: none"> • El desarrollador realiza un <i>Get Latest Version</i> (GLV), sobre el <i>branch</i> indicado por el integrador de aplicaciones, para obtener la última versión de los cambios de dicho <i>branch</i>. • Realiza los cambios necesarios en dicho <i>branch</i> dentro de la carpeta <i>CustomCode</i> y guarda los cambios en el <i>shelve</i> con el

Actor	Descripción
	<p>siguiente formato: BDT-[codCaso]-[usuario], ejemplo: BDT-XB34 896-corderouw.</p> <ul style="list-style-type: none"> • Realiza las pruebas necesarias para asegurarse que los cambios pueden ir a producción incluyendo las pruebas técnicas de desarrollador y las de aceptación de usuario. • Hasta que las pruebas sean exitosas se realiza un <i>checkin</i> del <i>shelve</i>. • Una vez que se realiza el <i>checkin</i> se solicita la integración de las fuentes para realizar un pase a producción. • Cuando las fuentes ya se encuentren integradas, el usuario debe tomar del <i>branch Main</i>, de la sección de BD-<Poyecto> los cambios para realizar el pase a producción y seguidamente eliminar el <i>shelve</i> de desarrollo, y llenar las bitácoras de publicación para finalmente cerrar el caso. <p>Estandarización:</p> <ul style="list-style-type: none"> • El desarrollador realiza un <i>Get Latest Version</i> (GLV), sobre el <i>branch</i> indicado por el integrador de aplicaciones, para obtener la última versión de los cambios de dicho <i>branch</i>. • Realiza los cambios necesarios en dicho <i>branch</i> dentro de la carpeta <i>vEstandar</i> y guarda los cambios en el <i>shelve</i> con el siguiente formato: BDE-[codCaso]-[usuario], ejemplo: BDE-XB34 896-corderouw. • Realiza las pruebas necesarias para asegurarse que los cambios pueden ir a producción incluyendo las pruebas técnicas de desarrollador y las de aceptación de usuario. • Hasta que las pruebas sean exitosas se realiza un <i>checkin</i> del <i>shelve</i>. • Una vez que se realiza el <i>checkin</i> se solicita la integración de las fuentes para realizar un pase a producción.

Actor	Descripción
	<ul style="list-style-type: none"> • Cuando las fuentes ya se encuentren integradas, el usuario debe tomar del <i>branch Main</i>, de la sección de BD-<Proyecto> los cambios para realizar el pase a producción y seguidamente eliminar el <i>shelve</i> de desarrollo, y llenar las bitácoras de publicación para finalmente cerrar el caso. • Finalmente se evalúa los otros clientes a los que debe realizar la publicación de los cambios.
Seniors	<ul style="list-style-type: none"> • Una vez que el encargado del caso, le eleva el caso al Senior, este debe realizar un análisis para identificar si el cambio correspondiente al caso, aplica solamente como código tropicalizado a la necesidad del cliente o bien corresponde a un cambio en las estructuras de bases de datos que afecta a todos los clientes. • A partir del diagnóstico indicado por el Senior el desarrollador debe realizar las indicaciones brindadas por el Senior, adicionalmente si el Senior identifica que es un cambio a nivel de estandarización debe proveer al desarrollador la lista de clientes a la cual se deben replicar los cambios desarrollados a partir del caso.
Integrador de Aplicaciones	<ul style="list-style-type: none"> • Según el tipo de cambio indicado por el Senior, el integrador de aplicaciones debe indicar y proveer la dirección exacta del <i>branch</i> en el cual deben realizarse los cambios, con el fin de que, una vez terminado realizar la estrategia de integración correspondiente al tipo de cambio a realizar. • En conjunto con el Senior debe identificar en el caso de estandarizaciones los clientes a los cuales se deben replicar los cambios una vez concluidos. • Debe de realizar las integraciones previamente identificadas para pasar los cambios en las bases de datos a producción.

Fuente: Elaboración propia, 2016.

En resumen, los pasos propuestos para los cambios en las bases de datos son:

Desarrollador:

- Identificar cambios en las bases de datos, a partir de la revisión del caso.
- Solicitar *branch* de trabajo para atención de caso.
- Realizar un GLV del *branch* indicado, crear el *shelve* para guardar los cambios, desarrollar los cambios, y realizar las pruebas necesarias.
- Cuando las pruebas están listas, realizar *checkin* del código de base de datos, solicitar integración de fuentes, realizar los pases a producción necesarios en los clientes indicados, llenar las bitácoras, eliminar el *shelve* creado, y cierre de caso.

Seniors:

- Identificar el tipo de cambio a realizar (tropicalizaciones o estandarización).
- En el caso de estandarizaciones, se debe identificar los clientes a los que se debe realizar la actualización de estructuras de bases de datos.
- Seguimiento y acompañamiento necesario para asegurar que los cambios realizados tengan el alcance necesario.

Integrador de aplicaciones:

- Analizar tipo de cambio en conjunto con los *seniors*.
- Proveer los *branches* para desarrollo de casos.
- Integraciones de los cambios y validación del proceso y la política establecida.

V.3.4. Plantilla de requerimientos funcionales y técnicos

A continuación, se presenta una guía rápida donde se expone la plantilla y/o datos necesarios para la toma de requerimientos, las cuales deben contener la información de mayor importancia para la empresa dependiendo de los tipos de requerimientos, y que esta información recolectada también pueda ser utilizada en el Team Foundation Server, esto con el fin de reestructurar y ordenar los requerimientos del centro de operaciones, en pro de un mayor control y generación de documentos de valor.

Tomando en cuenta las clases de requerimientos, en la Tabla 28 se describen los datos necesarios que se deben de documentar para la especificación de requerimientos.

Tabla 28 Datos de requerimientos.

Dato	Explicación
Nombre del proyecto	Nombre que se le dará al proyecto o que ya tiene asignado.
Código del proyecto	Código que se le dará al proyecto o que ya tiene asignado.
Código del requerimiento	Identifica el requerimiento como único dentro del sistema, para futura trazabilidad en relación con las actividades desarrolladas, <ul style="list-style-type: none">• Sugerencia: [CONSECUTIVO]- [Área Impactada]- [Descripción Global]• Ejemplo: CAS-04568934-Gestión Riesgos BAC – Creación de Reportes.
Tipo de requerimiento	Define el requerimiento como funcional o técnico. Basado en la Tabla 29.
Tipo de alcance	Define el alcance del requerimiento alto o bajo. Basado en la Tabla 30.
Requerimientos asociados	Lista de todos aquellos requerimientos que participan en su implementación y funcionamiento.
Descripción	Especificación del requerimiento y su importancia dentro del sistema para el cliente.

Dato	Explicación
Criterios de aceptación	Argumento con el cual se definirá después de la implementación si el requerimiento es aceptado o no.
Modulo	Especifican los modelos a los cuales se encuentra asociado.
Valor de área	Se indica a cuál área impactará el requerimiento: <ul style="list-style-type: none"> • Negocio. • Arquitectura. Separando la parte del cliente y la parte técnica.
Prioridad	Describe qué tan importante es el requerimiento para el cliente, el arquitecto, el analista de requerimientos y el gerente de proyecto (Valores de 1 a 4).
Versión	Define la gestión de cambio del requerimiento.
Estado	Indica el estado en que se encuentra el requerimiento. <ul style="list-style-type: none"> • Nuevo • Activo • Cerrado • Eliminado • Resuelto
Riesgo	Describe qué tan riesgoso puede llegar a ser el requerimiento sino se realiza. <ul style="list-style-type: none"> • 1- Alto. • 2- Medio. • 3- Bajo.
Datos	Muestra los datos de salida si el requerimiento está asociado directamente con el cliente o de entrada si el requerimiento está asociado con funcionalidades del sistema.
Comentarios	Información de importancia relacionada al requerimiento.

Fuente: Elaboración propia, 2016.

Tipos de requerimientos

En la Tabla 29 se detallan los dos tipos (macros) de requerimientos que la empresa utiliza.

Tabla 29 Tipos de requerimientos.

Dato	Explicación
Requerimientos funcionales	<p>Estos tipos de requerimientos son los relacionados con la descripción del comportamiento fundamental de los componentes del <i>software</i>. Las funciones son especificadas en términos de entradas, procesos y salidas. Una vista dinámica podría considerar aspectos como el control, el tiempo de las funciones (de comienzo a fin) y su comportamiento en situaciones excepcionales.</p> <p>Por lo tanto, un requerimiento funcional describe lo que el sistema debe hacer, es decir, especifica acciones que el sistema debe ser capaz de realizar, sin considerar restricciones físicas. Los requerimientos funcionales especifican el comportamiento del sistema.</p>
Requerimientos técnicos	<p>Para la empresa estos tipos de requerimientos son los que recibe un informático para que sean desarrollados, son especificaciones de más bajo nivel, por lo que se necesita conocimiento en informática. Por lo general, estos requerimientos están más completos y consistentes, ya que son redactados por funcionalidad, por ejemplo:</p> <ul style="list-style-type: none">• Crear un servicio Windows que realice consultas a la tabla SQL con el nombre ABC de acuerdo con una cantidad de minutos configurada en el “web.config” del proyecto.

Fuente: Elaboración propia, 2016.

Tipos de alcance

En la Tabla 30 se detallan los dos tipos de alcance que la empresa utiliza.

Tabla 30 Tipos de alcance.

Dato	Explicación
Requerimientos de alto alcance	Esta clase de requerimientos por lo general tienen un gran alcance y son utilizados cuando el cliente solicita requerimientos de un proyecto que se va a iniciar, el desarrollo de estos requerimientos tiene una duración elevada, ya que son requerimientos que contienen diferentes funcionalidades.
Requerimientos de bajo alcance	Son requerimientos “fáciles” de realizar por lo general tienen una duración máxima de 2-3 días.

Fuente: Elaboración propia, 2016.

V.3.5. Definir proceso de gestión de riesgos

Seguidamente, se presenta una guía rápida donde se expone el proceso que se debe realizar para dar seguimientos a los riesgos de las aplicaciones o de los proyectos de la empresa, la cual va a contar con las etapas que se deben de seguir y realizar en cada una de estas secciones. Además, este proceso debe ser coherente con lo que en TFS se puede efectuar en cuando a la administración de riesgos, esto con el fin de poder mitigar de mejor manera los riesgos que se dan en los proyectos o en las aplicaciones y poder evitarlos, si es posible.

Se ha identificado que, a nivel del centro de operaciones, no se maneja un proceso formal de gestión de riesgos, por lo que para el desarrollo de este proceso se van a tomar como referencia las acciones que actualmente se hacen para la administración de riesgos.

Además, se recopilaron diversas fuentes de información para definir el proceso de gestión de riesgos, que sea de mayor utilidad y beneficio para la empresa. Así mismo, se tomará en cuenta lo que propone TFS para la administración de riesgos, ya que es en este ambiente en el cual en un futuro se van a registrar de forma digital.

Propuesta de proceso de gestión de riesgos

A continuación, se define la propuesta para el seguimiento o administración de los riesgos.

V.3.5.1. Identificar los riesgos

Es el proceso en el cual se determinan los riesgos que pueden afectar el proyecto o la aplicación y se documentan sus características.

Entre las personas que participan en la identificación de riesgos se pueden incluir: El director del proyecto, los miembros del equipo del proyecto, el equipo de gestión de riesgos (si está asignado), clientes, expertos en la materia externos al equipo del proyecto, usuarios finales, otros directores del proyecto, interesados y expertos en gestión de riesgos.

Identificar los riesgos es un proceso iterativo debido a que se pueden descubrir nuevos riesgos o pueden evolucionar conforme el proyecto avanza a lo largo de su ciclo de vida. La frecuencia de iteración y quiénes participan en cada ciclo varía de una situación a otra.

Entradas

En los siguientes puntos, se detallan las entradas de utilidad para la identificación de los riesgos.

- Estimaciones de costos de las actividades.
- Estimaciones de la duración de la actividad.
- Línea base del alcance
- Registro de interesados
- Plan de gestión de costos
- Plan de gestión del cronograma
- Plan de gestión de calidad
- Documentos del proyecto
- Factores ambientales de la empresa
- Activos de los procesos de la organización

Salidas

Por lo general, las salidas principales del proceso identificar los riesgos figuran en el registro de riesgos.

- Registro de riesgos

La preparación del registro de riesgos comienza en el proceso identificar los riesgos con la siguiente información, y luego queda a disposición para otros procesos de dirección de proyectos y de gestión de los riesgos del proyecto.

- Lista de riesgos identificados: Cada uno de los riesgos debe contar con las siguientes secciones:
 - Título
 - Área a la que pertenece el riesgo.
 - Iteración del riesgo.
 - Asignado a: Una persona que es responsable de la respuesta al riesgo.
 - Descripción
 - Estado

- Lista de respuestas potenciales.

V.3.5.2. Análisis cualitativo de los riesgos

Realizar el análisis cualitativo de riesgos es el proceso que consiste en priorizar los riesgos para realizar otros análisis o acciones posteriores, evaluando y combinando la probabilidad de ocurrencia y el impacto de dichos riesgos.

Entradas

Las entradas que son de utilidad para el análisis cualitativo de los riesgos son:

- Registro de riesgos.
- Enunciado del alcance del proyecto.
- Activos de los procesos de la organización.

Salidas

El registro de riesgos se actualiza con la información procedente del proceso realizar el análisis cualitativo de riesgos. Las actualizaciones al registro de riesgos provenientes del proceso Realizar el Análisis Cualitativo de Riesgos incluyen:

- Probabilidad estimada de que ocurra el riesgo.
- La prioridad del riesgo (valores: 1, 2 o 3).
- Disparadores: Para cada riesgo, identificar los desencadenantes o síntomas, esto con el fin de ayudar a proporcionar evidencia temprana de que el riesgo está a punto de ocurrir.
- Actualizar el estado.

V.3.5.3. Análisis cuantitativo de los riesgos

Realizar el análisis cuantitativo de riesgos es el proceso que consiste en analizar numéricamente el efecto de los riesgos identificados sobre los objetivos generales del proyecto.

El proceso de realizar el análisis cuantitativo de riesgos se aplica a los riesgos priorizados mediante el proceso de análisis cualitativo de riesgos, al tener un posible impacto significativo sobre las demandas concurrentes del proyecto.

Por lo general, el proceso del análisis cuantitativo de riesgos se realiza después del proceso del análisis cualitativo de riesgos. En algunos casos, es posible que el proceso del análisis cuantitativo de riesgos no sea necesario para desarrollar una respuesta efectiva a los riesgos.

Entradas

Las entradas que pueden ser de utilidad para el análisis cuantitativo de los riesgos, se describen en los siguientes puntos:

- Registro de riesgos.
- Plan de gestión de costos.
- Plan de gestión del cronograma.
- Activos de los procesos de la organización.

Salidas

El registro de riesgos se actualiza para incluir un informe cuantitativo de riesgos que detalla los enfoques cuantitativos, las salidas y las recomendaciones. Las actualizaciones incluyen los siguientes componentes principales:

- Análisis probabilístico del proyecto: Se realizan estimaciones de los resultados potenciales del cronograma y costos del proyecto, enumerando las fechas de conclusión y los costos posibles con sus niveles de confianza asociados.
- Severidad o impacto que puede provocar el riesgo (valores: Bajo, medio o alto).

V.3.5.4. Planificar las respuestas a los riesgos

Planificar la respuesta a los riesgos es el proceso por el cual se desarrollan opciones y acciones para mejorar las oportunidades, así como reducir las amenazas a los objetivos del proyecto.

Las respuestas a los riesgos planificadas deben adaptarse a la importancia del riesgo, ser rentables con relación al desafío por cumplir, realistas dentro del contexto del proyecto, acordadas por todas las partes involucradas y deben estar a cargo de una persona responsable. También deben ser oportunas.

Entradas

A continuación, se detallan diversas entradas que pueden ser de utilidad para planificar las respuestas a los riesgos.

- Registro de riesgos.

Salidas

- Actualizaciones al registro de riesgos: En el marco del proceso, planificar la respuesta a los riesgos, se seleccionan y se acuerdan las respuestas apropiadas, y se incluyen en el registro de riesgos.
 - Plan de mitigación: Se llena normalmente para riesgos críticos y altos, contiene instrucciones sobre cómo reducir la probabilidad o las consecuencias de un evento de riesgo a un umbral aceptable, esto reduce el valor monetario esperado de un evento de riesgo y la probabilidad de ocurrencia, o ambos.
 - Las estrategias de respuesta acordadas.
 - Plan de contingencia: Contiene la lista de acciones necesarias para reducir la probabilidad o las consecuencias del impacto en los objetivos del proyecto. Estos planes describen el curso de acción a tomar en caso de que se produzca el desencadenante del riesgo o el síntoma.
 - Describir las acciones por seguir cuando ocurre el evento de riesgo (determina qué riesgos deben tener un plan de contingencia).
- Acuerdos contractuales relacionados con los riesgos: Acuerdos nuevos que se definieron con el cliente, fueron provocados por los riesgos realizados o mitigados.

- Actualizaciones a los documentos del proyecto

V.3.5.5. Realizar el seguimiento y control de los riesgos

Es el proceso por el cual se implementan planes de respuesta a los riesgos, se rastrean los riesgos identificados, se monitorean los riesgos residuales, se identifican nuevos riesgos y se evalúa la efectividad del proceso contra los riesgos a través del proyecto.

El proceso de seguimiento y control de los riesgos aplica técnicas, tales como el análisis de variación y de tendencias, estos requieren del uso de información del desempeño generada durante la ejecución del proyecto. Otras finalidades del proceso de seguimiento y control de los riesgos son determinar si:

- Los supuestos del proyecto siguen siendo válidos.
- Los análisis muestran que un riesgo evaluado ha cambiado o puede descartarse.
- Se respetan las políticas y los procedimientos de gestión de riesgos.
- Las reservas para contingencias de costo o cronograma deben modificarse para alinearlas con la evaluación actual de los riesgos.

Entradas

- Registro de riesgos.

Salidas

- Los resultados de las reevaluaciones, auditorías y revisiones periódicas de los riesgos. Estos resultados pueden incluir la identificación de nuevos riesgos, actualizaciones a la probabilidad, al impacto, a la prioridad, a los planes de respuesta, a la propiedad y a otros elementos del registro de riesgos. Los resultados también pueden incluir el cierre de riesgos que ya no se aplican y la desafectación de las reservas correspondientes.
- Los resultados reales de los riesgos del proyecto y de las respuestas a los riesgos.
- Documentar lecciones aprendidas y guardar los registros en la base de conocimientos.

Como resumen del proceso, se muestra el Gráfico 19 que lo detalla.

Gráfico 19 Proceso de riesgos.



Fuente: Basado en PMBOK (2008).

V.3.6. Metodología para pruebas técnicas y funcionales.

En este apartado se va a definir una metodología para la ejecución de las pruebas técnicas y funcionales para los principales productos de la empresa, de manera que se pueda garantizar la entrega de *software* de alta calidad.

Para lograr lo anterior, se pretende establecer un estándar para la documentación de las pruebas funcionales y técnicas que se aplicarán a las diferentes soluciones según corresponda.

V.3.6.1. Consideraciones generales

Algunas de las consideraciones generales para la metodología a seguir en el caso de las pruebas funcionales y técnicas de los productos son las siguientes:

1. Las directrices establecidas en esta metodología aplican en la atención de todos los casos cuya solución implique alguna de las siguientes acciones:
 - a. Creación o modificación del código fuente de componentes en la capa aplicativa de un producto (ya sea *frontend* o *backend*).
 - b. Creación o modificación de objetos de base de datos de un producto.
 - c. Modificación en los archivos de configuración de componentes en la capa aplicativa de un producto (ya sea *frontend* o *backend*).
 - d. Modificación en el ambiente donde se encuentran instalados los componentes de un producto (actualización de sistema operativo, motor de base de datos, versión de *framework*, entre otros).
2. Se debe asignar un ingeniero senior, el cual debe apoyar a los ingenieros encargados de la solución del caso.
3. El ingeniero encargado de la atención del caso debe proporcionar la documentación de las pruebas técnicas y funcionales como parte de la entrega de la solución del caso al responsable que corresponda (ingeniero senior a cargo o director del centro de operaciones).
4. El responsable del caso debe verificar que los resultados de las pruebas ejecutadas por el ingeniero se encuentren debidamente documentadas y que estas sean exitosas, antes de realizar la entrega final de la solución al cliente.

5. En caso de que el cliente maneje otro formato para la documentación de los escenarios de prueba, el ingeniero encargado de la ejecución de las pruebas debe producir dicho documento también, como parte de los entregables al responsable que corresponda.
6. El documento de pruebas funcionales y técnicas debe ser adjuntado al caso correspondiente del CRM de la empresa en forma de nota, de manera que sea de fácil acceso para todos los involucrados.

La metodología que se va utilizar va estar dividida en 2 secciones las cuales son:

- Definición de casos de prueba: Esta sección va consistir en definir el proceso que se debe seguir para definir los casos de prueba.
- Documentación de pruebas y resultados: Para esta sección se va a detallar cómo se deben documentar las pruebas y/o resultados de estas.

V.3.6.2. Definición de casos de prueba

La definición de los casos de prueba para los incidentes o requerimientos deberán realizarse según los siguientes lineamientos:

1. El ingeniero encargado del caso debe realizar la definición de los casos de prueba basándose en el documento de análisis y diseño de la solución para el requerimiento.
2. Los casos de prueba definidos por el ingeniero encargado del caso deben contemplar todos los posibles escenarios que se pueden ejecutar en el aplicativo en el cual se utilicen todos los componentes del producto que fueron creados o modificados como parte de la solución.
3. Todos los casos de prueba deben ser ejecutados sin excepción alguna. En caso de fuerza mayor, el ingeniero debe solicitar el visto bueno del jefe inmediato para omitir uno o varios de los escenarios de prueba definidos previamente.
4. En el caso de que no exista interfaz gráfica que permita la ejecución de uno o varios de los casos de prueba definidos por el ingeniero encargado del caso, es responsabilidad de él diseñar una metodología que le permita ejecutar dichos escenarios o, en su defecto, solicitar apoyo al ingeniero en automatización de pruebas con alguna herramienta o solución para realizar la ejecución de las pruebas.

V.3.6.3. Documentación de pruebas y resultados

Para la documentación de los casos de prueba y los resultados obtenidos de su ejecución, se cuenta con una plantilla, la cual se puede encontrar en el apéndice 9 del presente documento.

Todos los escenarios de prueba deben ser documentados desde el momento en el que se definen, de manera que, el ingeniero encargado de realizar las pruebas pueda seguir el guion de los escenarios paso a paso y documentar los resultados en la sección respectiva para cada caso de prueba (ver plantilla en el Apéndice 9), y así, ir completando el documento que deberá presentar al encargado.

V.3.7. Política para la creación y almacenamiento de conocimiento.

En esta sección propondrá una política para generar una base de datos de conocimiento que permita compartir los datos entre todos los ingenieros.

Adicionalmente, dichas políticas indicarán la manera en que se almacenará toda documentación generada y la forma de acceder a dichos documentos.

La política va estar dividida en 3 (partes) las cuales son:

- Creación de documentos.
- Almacenamiento del documento
- Consulta de documentos y retroalimentación

Seguidamente, se van a explicar cada una de estas partes.

V.3.7.1. Creación de documentos

Las pautas que se deben de seguir para la creación de documentación son:

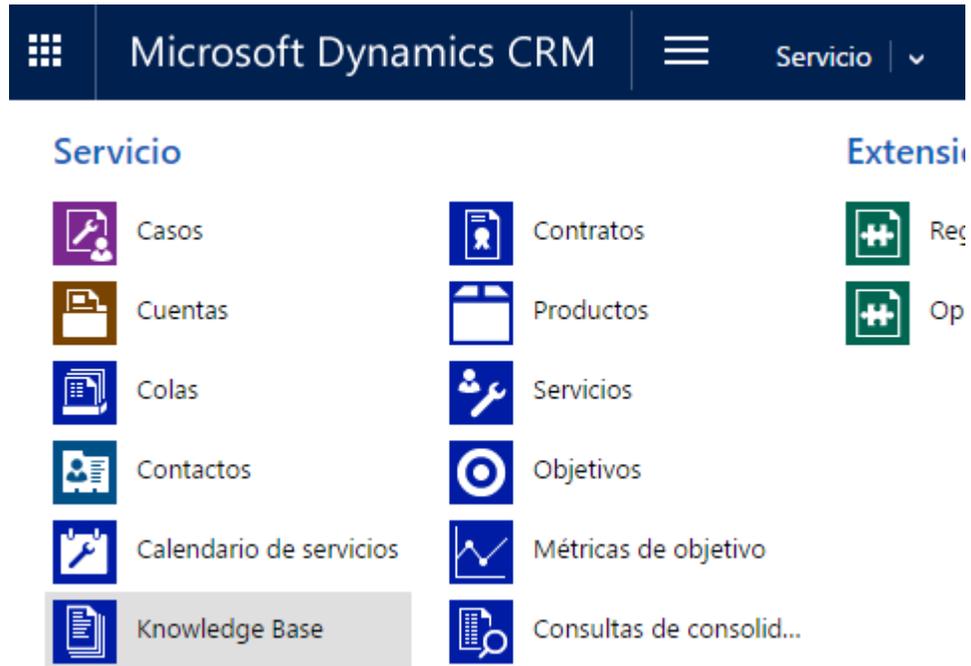
- 1) Se nombrará un colaborador responsable, el cual se encargará de dar el mantenimiento adecuado a la base de datos de conocimiento. Adicionalmente, debe velar por que todos los meses ingrese contenido en ella.
- 2) El colaborador responsable debe asignar mensualmente a un ingeniero, como mínimo, para que produzca algún documento que contenga información de las tecnologías utilizadas, mejores prácticas o cualquier otro dato que considere relevante y que sea de provecho para todos los ingenieros del centro de operaciones. El ingeniero debe producir la documentación correspondiente y remitirá al coordinador responsable de la base de datos de conocimiento.
- 3) En caso de que algún ingeniero considere conveniente documentar el diagnóstico y la solución de un caso (debido a que se puede presentar en dos o más clientes) y almacenarlo en la base de datos de conocimiento, puede hacerlo sin ninguna restricción.
- 4) El formato del contenido de la documentación generada queda a criterio del ingeniero encargado de producirla.
- 5) Todos los documentos generados por los ingenieros encargados deben ser en formato PDF.

V.3.7.2. Almacenamiento del documento

Las consideraciones que se deben de seguir para el almacenamiento del conocimiento, son:

- 1) La inclusión de los artículos/documentos a la base de datos de conocimiento debe realizarse por medio del coordinador responsable de esta.
- 2) Toda la documentación generada será almacenada en la sección de base de datos de conocimiento del CRM de Microsoft, de manera que pueda ser accedida por cualquiera de los ingenieros a través de dicha herramienta, en la sección Servicio -> *Knowledge Base*. En la Ilustración 12 se muestra esta funcionalidad.

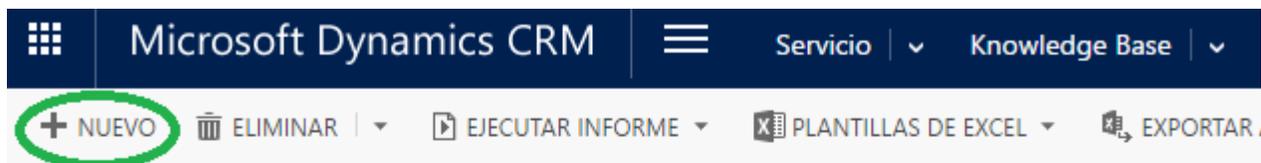
Ilustración 12 CRM knowledge Base.



Fuente: Elaboración propia, 2016.

- 3) Para almacenar un documento se debe de entrar en la sección *Knowledge Base*, la cual mostrará una vista de todos los artículos publicados, para agregar un documento nuevo hay que hacer clic en el botón nuevo en la parte superior izquierda, tal como se muestra en la Ilustración 13.

Ilustración 13 CRM Artículos publicados.



➔ Artículos publicados ▼

✓	Número ↑	Asunto	Idioma	Título
	AKB-01004-Z5J7	Información	Español(3082)	Clientes - Información estratégica

Fuente: Elaboración propia, 2016.

- 4) Después se deberá escoger la categoría a la que va a pertenecer el documento, las cuales son:
- Artículo estándar de *Knowledge Base*: Utilice esta categoría para proporcionar información general acerca de un asunto
 - Pregunta y respuesta: Utilice esta categoría para crear una pregunta y su respuesta para un problema o situación específicos.
 - Procedimiento: Utilice esta categoría para crear un procedimiento que defina los pasos para resolver un problema específico.
 - Solución a un problema: Utilice esta categoría para proporcionar una solución a un problema específico.

En la Ilustración 14 se puede ver la manera de como seleccionar las distintas categorías.

Ilustración 14 CRM Categorías de documentos

Idioma: Español

Plantillas internas

- Artículo estándar de Knowledge Base
- Pregunta y respuesta
- Procedimiento
- Solución a un problema

Título: Artículo estándar de Knowledge Base

Autor: SYSTEM

Descripción: Utilice esta plantilla para proporcionar información general acerca de un asunto.

Fuente: Elaboración propia, 2016.

- Una vez que se selecciona la categoría, esta se dirigirá a una plantilla donde se deberá asignarle un título al artículo, un pequeño resumen acerca de qué trata, comentarios adicionales, y una vez que se guarda es posible adjuntar el archivo PDF que ya antes se había generado, lo descrito se puede observar en la Ilustración 15.

Ilustración 15 CRM Plantilla de documentación

Adjuntar archivo | Agregar nota

Artículo : Information

Artículo: AKB-01007-N5F2

General

Título*: Procesos ALM | Asunto*: Asunto predeterminado

Palabras clave: ALM, aplicaciones, ciclo de vida | Idioma: Español

Procesos ALM
AKB-01007-N5F2

Resumen
Este es el resumen del documento

Comentarios adicionales
Acá se detallaran comentarios adicionales acerca del documento

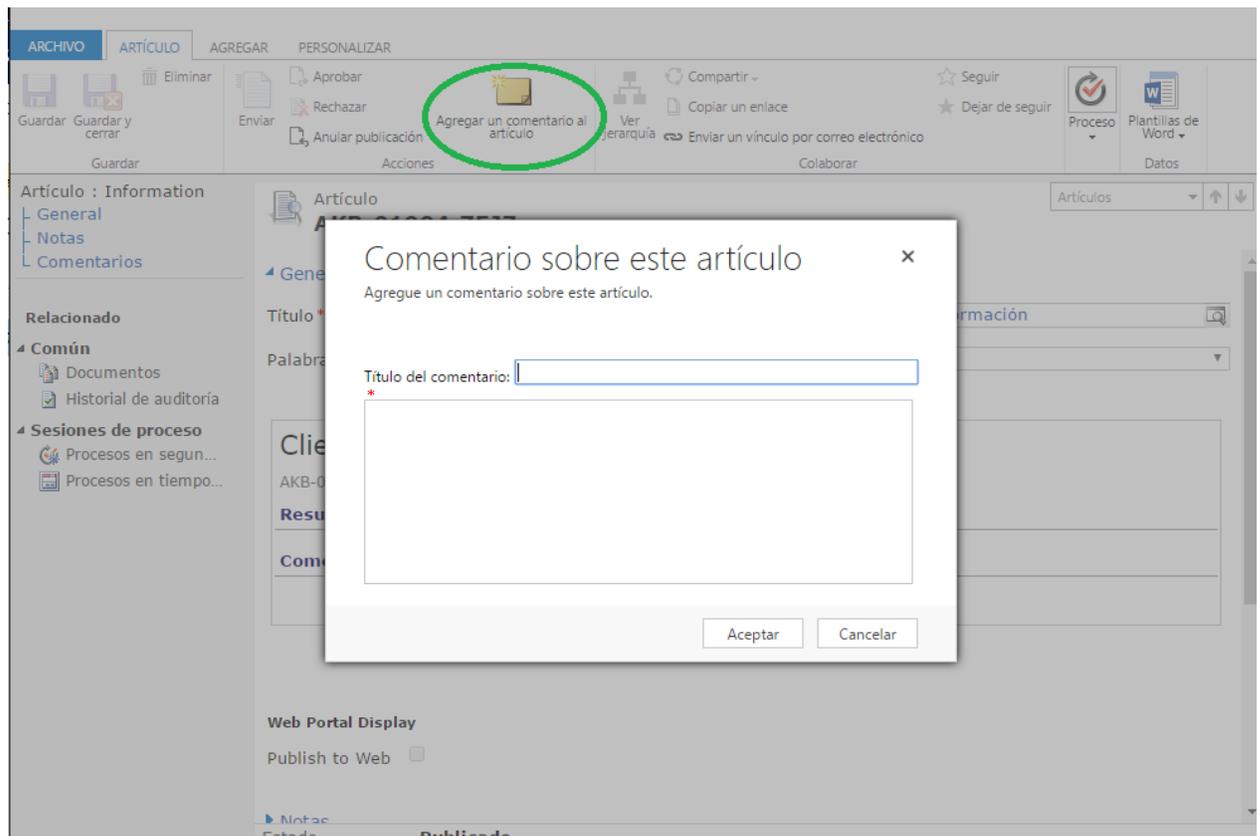
Fuente: Elaboración propia, 2016.

- Estos artículos quedarán en estado borrador hasta que el colaborador responsable de la base de conocimiento lo apruebe y, por lo tanto, quede publicado y visible para los demás colaboradores.

V.3.7.3. Consulta de documentos y retroalimentación

- 1) Para acceder a los artículos publicados se debe de ingresar a la plataforma CRM de la empresa, en la sección Servicio → *Knowledge Base* → Artículos publicados, en esta vista se van a encontrar todos los artículos que han sido aprobados y que pueden ser consultados.
- 2) Si en alguna situación se considera que se puede agregar más información algún documento publicado, solo se debe dar clic en el artículo y agregar un comentario a este en el que se proponga lo que se quiere agregar. El encargado de aprobar esta información complementaria será el colaborador responsable de la base de conocimiento. En la Ilustración 16 se muestra como agregar un comentario.

Ilustración 16 CRM Comentario en artículo.



Fuente: Elaboración propia, 2016.

V.3.8. Políticas para el *deployment*.

Este apartado tiene como objetivo principal establecer las políticas y las pautas que se deben seguir en el *deployment* de los aplicativos, de manera que se minimicen al máximo los incidentes posibles debido a malas prácticas en las publicaciones y/o configuraciones de los componentes de un producto.

Las políticas propuestas en esta sección permitirán tener un mayor conocimiento sobre la versión de los componentes que se publicarán, mediante un control riguroso en la generación de los entregables (compilados de los aplicativos, archivos de configuración y scripts de base de datos que se deben ejecutar), además de la documentación del proceso.

V.3.8.1. Consideraciones generales

Las consideraciones generales respecto a las políticas establecidas respecto al *deployment* de las aplicaciones son las siguientes:

1. Las directrices mencionadas en este apartado aplican para el *deployment* de aplicaciones el cual implique alguna de las siguientes acciones:
 - a. Instalación de nuevos componentes de un producto. Dichos componentes incluyen, pero no se limitan a: Servicios Windows, servicios web, sitios web, objetos de base de datos y aplicaciones de escritorio.
 - b. Modificación de componentes existentes de un producto, sin importar la naturaleza del cambio (nueva versión de compilados, archivos de configuración, entre otros).
 - c. Cambios en el ambiente donde se encuentran instalados los componentes de un producto como, por ejemplo, cambios en la versión del framework, actualización de sistema operativo, actualización en la versión del motor de base de datos, entre otros.
2. Las políticas establecidas en este documento aplican para los siguientes ambientes:
 - a. Ambientes de pruebas.
 - b. Ambientes de QA (Quality assurance).
 - c. Ambientes de UAT (User acceptance testing).
 - d. Ambientes de pre-producción.
 - e. Ambientes de producción.

3. Cualquier documentación generada por el ingeniero encargado de hacer el deployment debe ser adjuntada al caso correspondiente del CRM en forma de nota, de manera que sea de fácil acceso para todos los involucrados.
4. Se debe asignar un ingeniero responsable del deployment, el cual se encargará de realizar la publicación de los componentes.
5. Todas las pautas que se encuentran en este documento deben ser cumplidas por los ingenieros sin excepción alguna.

V.3.8.2. Generación de entregables

En esta sección se detallan las políticas propuestas para la generación de los entregables en los procesos de *deployment*.

Compilados de los aplicativos y archivos de configuración

Las directrices establecidas para la generación de los compilados que se publicarán son:

1. El ingeniero responsable del *deployment* debe indicar al integrador de aplicaciones el *branch* del repositorio correspondiente que contiene la solución y los componentes que se publicarán. Dicha solicitud se debe realizar a través de un correo electrónico.
2. El integrador de aplicaciones realizará los *merge* correspondientes para actualizar las ramas respectivas del repositorio necesarias. Además, generará los compilados de cada uno de los componentes y se los entregará al ingeniero encargado del *deployment*.
3. El ingeniero responsable del *deployment* debe realizar un documento con el paso a paso detallado de la publicación de todos los componentes necesarios, esto debido a que la publicación podría ser realizada por otro ingeniero, o en su defecto, por el personal del cliente. Adicionalmente, el ingeniero responsable debe realizar otro documento con el paso a paso detallado que permita realizar un *rollback* en caso de que el *deployment* no sea exitoso por algún motivo.
4. El formato de dichos documentos queda a criterio del ingeniero responsable del *deployment*, y debe contener como mínimo la siguiente información:
 - a. Ambiente en el cual se publicarán los cambios.
 - b. Equipos donde se realizarán las publicaciones.
 - c. Componentes o archivos que deben ser respaldados antes de realizar las modificaciones.

- d. Ruta física de las carpetas donde se encuentran instalados los componentes que sufrirán cambios (en caso de ser una modificación).
 - e. Paso a paso detallado de los cambios a realizar.
 - f. Posibles pruebas que el cliente puede ejecutar con el fin de validar que el *deployment/rollback* haya sido exitoso.
5. En el caso de que los archivos de configuración de los componentes sufran algún cambio, queda a criterio el ingeniero responsable realizar alguna de las siguientes acciones:
- a. Solicitar al cliente los archivos de configuración de los componentes que serán modificados, realizar los cambios respectivos e incluirlos como parte de los archivos que se publicarán.
 - b. No incluir los archivos de configuración como parte de lo que se publicará e indicar en el documento con el paso a paso de la publicación, los cambios que deben realizarse sobre los archivos de configuración ya existentes.
6. Una vez realizada la publicación, el ingeniero debe solicitar al cliente el visto bueno de que la publicación de los cambios fue realizada con éxito e informarlo al integrador de Aplicaciones.
7. En el caso de que el *deployment* se haya realizado de manera exitosa, el integrador de aplicaciones se encargará de hacer los *merge* correspondientes a las ramas respectivas del repositorio. Adicionalmente, debe informar a todos los ingenieros del centro de operaciones, a través de un correo electrónico, que el repositorio de dicho cliente fue actualizado, de manera que puedan obtener y trabajar sobre la última versión del aplicativo.

V.3.8.3. Scripts de base de datos

Las directrices que se proponen para la generación de script de base de datos son:

1. El ingeniero encargado del *deployment*, será quien genere los *scripts* de base de datos que se ejecutarán. Adicionalmente, debe realizar un documento con el paso a paso detallado de la ejecución de todos los scripts necesarios, esto debido a que dicha tarea podría ser realizada por otro ingeniero, o en su defecto, por el personal del cliente. Además, debe realizar otro documento con el paso a paso detallado que permita realizar un *rollback* en caso de que el *deployment* no sea exitoso por algún motivo.
2. El formato de dichos documentos queda a criterio del ingeniero responsable del *deployment*, y debe contener, como mínimo, la siguiente información:
 - a. Ambiente en el cual se ejecutarán los scripts.
 - b. Instancia y puerto donde se encuentra la o las bases de datos sobre las cuales se ejecutarán los scripts.
 - c. Objetos de bases de datos que deben ser respaldados.
 - d. Paso a paso detallado de la ejecución de los scripts.
 - e. Posibles pruebas que el cliente puede ejecutar con el fin de validar que el *deployment/rollback* haya sido exitoso.
3. El ingeniero responsable del *deployment* debe indicar al integrador de aplicaciones el *branch* del repositorio correspondiente que contiene la solución de base de datos con los scripts aplicados. Dicha solicitud se debe realizar a través de un correo electrónico.
4. El integrador de aplicaciones realizará los *merge* correspondientes para actualizar las ramas respectivas del repositorio necesarias.
5. Una vez realizada la publicación, el ingeniero debe solicitar al cliente el visto bueno de que la publicación de los cambios fue realizada con éxito e informarlo al integrador de aplicaciones.
6. En el caso de que el *deployment* se haya realizado de manera exitosa, el integrador de aplicaciones se encargará de hacer los *merge* correspondientes a las ramas respectivas del repositorio. Adicionalmente, debe informar a todos los ingenieros del centro de operaciones, a través de un correo electrónico, que el repositorio de dicho cliente fue actualizado, de manera que puedan obtener y trabajar sobre la última versión de las bases de datos.

V.3.9. Métricas del negocio.

A continuación, se presentan las métricas que se proponen utilizar a lo largo del ciclo de vida de las aplicaciones en el centro de operaciones. No se puede mejorar aquello que no se conoce y no se puede llegar realmente a conocer aquello que no se puede medir. Es indispensable que la organización defina una serie de métricas que permitan determinar si se han alcanzado los objetivos propuestos, así como la calidad y rendimiento de los procesos y tareas involucradas.

Se ha identificado que, a nivel del centro de operaciones, no existen definiciones formales de las métricas para verificar el cumplimiento del ciclo de vida de las aplicaciones, por lo que con este apartado se busca formalizar las métricas que usará la empresa.

Para el desarrollo de las métricas se va a utilizar la documentación no formal con la que cuenta la empresa para la verificación de procesos, además de recopilar diversas fuentes de información y definir métricas nuevas que sean de utilidad para el centro de operaciones y que puedan acoplarse con base en la herramienta de TFS, las fuentes consultadas para la definición de las métricas fueron ITIL V3 (Office of Government Commerce, 2007), y Microsoft (Visual Studio ALM Quick Reference Guide, 2015).

En la Tabla 31, se definen las métricas que se pueden aplicar en el centro de operaciones.

Tabla 31 Métricas del negocio.

Métrica	Descripción
Tiempo promedio de nuevas características	El tiempo que necesitamos para entregar una nueva característica.
Tiempo promedio de correcciones de errores	El tiempo que se tarda corrigiendo errores en alguna aplicación.
Tiempo promedio para completar la tarea	El tiempo necesario para dar por finalizada las tareas.
Promedio de desviación de estimaciones	Suma de la desviación del tiempo (por ejemplo, en días) contra el calendario previsto de todos los proyectos activos.
Porcentaje de solicitudes de servicio resueltas	Porcentaje de solicitudes de servicio resueltas dentro de un período de tiempo acordado / aceptable.

Métrica	Descripción
Costo de entrega de servicios	Costo de entrega de servicios como se define en el Acuerdo de Nivel de Servicio (SLA) basado en un período establecido como mes o trimestre.
Porcentaje de interrupción (indisponibilidad)	Porcentaje de interrupción (indisponibilidad) debido a la implementación de cambios planificados, en relación con las horas de servicio.
Tiempo promedio de resolución de incidentes	Tiempo promedio (por ejemplo, en horas) entre la ocurrencia de un incidente y su resolución.
Tiempo promedio de indisponibilidad de algún servicio.	El tiempo medio entre el fallo (MTBF) y el tiempo medio de reparación (MTTR)
Número de quejas	Número de quejas recibidas dentro del período de medición.
Porcentaje incidentes resueltos con la primera llamada.	Porcentaje de problemas de clientes que fueron resueltos por la primera llamada telefónica.
Tiempo promedio desde solicitud de funcionalidad hasta entrega	Tiempo de ciclo desde la solicitud de funcionalidad hasta la entrega.
Promedio de satisfacción de clientes por servicios	Percepción de los usuarios respecto a la atención prestada mediante encuestas de satisfacción.
Tiempo promedio de resolución de incidentes por categoría	Muestra el tiempo medio de resolución y la categoría de incidentes cerrados en un periodo.
Número de incidentes repetidos	Cantidad de incidentes que se han detectados repetidos en cuanto a funcionalidad.
Cantidad de clientes nuevos adquiridos	Cantidad de clientes nuevos adquiridos en un periodo.
Cantidad de clientes perdidos	Cantidad de clientes perdidos en un periodo.
Cumplimiento de niveles de servicio	Cantidad de servicios/ SLA's que cumplen con los niveles de servicio acordados.

Métrica	Descripción
Cantidad de defectos identificados durante las pruebas.	Cantidad de defectos identificados con los mecanismos de pruebas.
Cantidad de cambios mayores en los proyectos	Cantidad de cambios mayores evaluados por el director de proyectos.
Tiempo para autorización para cambios	Tiempo promedio desde que solicita el cambio por parte del cliente hasta que se aprueba este.
Retrasos del proyecto	Fechas de finalización de proyecto, real vs. planificadas
Promedio de riesgos detectado por proyecto	Cantidad promedio de riesgos detectados en los proyectos.
Promedio de tiempo tardado en realizar pruebas	Tiempo promedio que se tardó en realizar pruebas en un periodo de tiempo.
Cantidad de fallos en pruebas	Cantidad de fallos presentados en las ejecuciones de pruebas.
Horas de esfuerzo por proyecto, iteración o sprint	Horas hombre que se han dedicado al proyecto, iteración o sprint.
Total de tareas por actividad en el proyecto	Permite identificar la distribución de las tareas según las actividades desarrolladas.
Inconsistencias en el desarrollo de software por cada miembro del equipo	Muestra la cantidad de inconsistencias en las actividades de desarrollo de software que tiene cada miembro de equipo según la gravedad de estas.

Fuente: Elaboración propia, 2016.

VI CONCLUSIONES Y RECOMENDACIONES

Este apartado muestra las conclusiones y recomendaciones que se obtienen del proyecto.

VI.1. Conclusiones

- Basado en los niveles de madurez de ALM propuestos por Microsoft, el estado en el que se encontraba el centro de operaciones antes de la realización de este proyecto, en los procesos de arquitectura y diseño, ingeniería de requerimientos y experiencia de usuario, desarrollo, gobierno de TI y aseguramiento de calidad, era estandarizado.
- Los procesos de administración de la configuración del *software*, implementación y operaciones, así como planificación y administración de proyectos, antes de realizar este trabajo se encontraban en un estado básico, según los niveles de madurez de ALM propuestos por Microsoft.
- Existen 41 actividades que se tienen que realizar para lograr alcanzar un estado avanzado de ALM, de estas 41 son fundamentales 22 para cumplir con este objetivo.
- Las acciones de mayor prioridad para la empresa, con el fin de alcanzar un estado avanzado en ALM, son los expuestos en la Tabla 20.
- Los estándares, políticas y/o metodologías desarrolladas están basadas en lo que Microsoft define y lo que la empresa necesita, por lo que son de fuentes confiables y de utilidad para la empresa.
- Con la definición de los estándares, políticas y/o metodologías se tiene establecido cómo se deben de realizar cada una de las tareas involucradas, ya que estas son reglas, guías, características o principios que se van a tener que tomar en cuenta para el proceso de entrega de *software* del centro de operaciones.
- El centro de operaciones tiene controles o documentación para los procesos de ALM, pero no definidos formalmente, esto según el capítulo IV.
- Actualmente los equipos de calidad del centro de operaciones no reciben guías o diagramas de flujo agregados en los requerimientos.
- El centro de operaciones no tiene patrones de diseño definidos y esto causa problemas en rendimiento.
- Los colaboradores del centro de operaciones no tienen reportes ni *dashboards* de seguimientos o de avance, de sus tareas o las del equipo a cargo, impidiendo tener un seguimiento y control más eficaz.

- El centro de operaciones no tiene un proceso de carga de datos de prueba ni encriptación de estos, se utilizan datos de producción, inclusive datos sensibles, lo que puede provocar modificación de los mismos sin el consentimiento del cliente
- Tener definido en que se va a basar un proyecto ayuda como guía para saber dónde y de que fuentes buscar, para este proyecto fue Microsoft, sin embargo, por este mismo aspecto se están dejando de lado información o practicas útiles que se realizan en otras entidades o investigaciones.

VI.2. Recomendaciones

- El director del centro de operaciones deberá emitir una directriz para que los estándares, metodologías o políticas de este documento sean de aplicación obligatoria para todos los colaboradores.
- Se recomienda asignar a una persona para que mejore constantemente el ciclo de vida de las aplicaciones y le dé seguimiento a los procesos que se van mejorando, esto con el fin de conocer los procesos que se van quedando rezagados.
- Se debe de capacitar a los colaboradores de la empresa para que implementen los estándares, metodologías o políticas de este documento.
- Se recomienda investigar las funcionalidades de TFS para que unido a ella se mejoren los procesos de ALM.
- Se recomiendan asignar funcionalidades a los colaboradores según los actores descritos en los estándares, metodologías o políticas de este documento.
- Se recomiendan seguir el plan de trabajo con las acciones priorizadas de este documento para continuar con la mejora del ciclo de vida de las aplicaciones.
- Se recomienda que el conocimiento que posee el integrador de aplicaciones sea compartido a otros colaboradores, ya que se presenta dependencia a esta persona.
- Se recomienda realizar una actualización de la herramienta TFS, de manera que se puedan aprovechar ciertas características nuevas que ofrece la herramienta, pues estas aplican a la mejoría del funcionamiento del centro de operaciones de la empresa. Además, se deben implementar las configuraciones para el control y ejecución del trabajo diario.
- Se recomienda implementar retrospectivas de parte de los usuarios hacia la empresa, esto con el propósito de percibir, constantemente, oportunidades de mejora y fortalecer buenas prácticas.
- Se recomienda seguir otros estándares o metodologías diferentes a las ofrecidas por Microsoft, con el fin de realmente aplicar las mejores prácticas y no centrarse específicamente en lo que dice Microsoft.

BIBLIOGRAFÍA

- Abran, A., Moore, J., Bourque, P., Dupuis, R., & Tripp, L. L. (2004). Guía al cuerpo de conocimiento de la ingeniería del software. California: IEEE Computer Society.
- Arias, F. (2006). El proyecto de investigación: Introducción a la metodología científica. Caracas - Venezuela: EPISTEME S.A.
- Balani, B. P. (2008). Governing Agile Projects Using CMMI. *Journal of the Quality Assurance*, 22, 17-19.
- Becker, J., Knackstedt, R., & Pöppelbuß, J. (2009). Developing Maturity Models for IT Management - A Procedure Model and its Application. *Business & Information Systems Engineering*, 1(3), 213-222.
- Beizer, B. (1990). *Software Testing Techniques*. New York: International Thomson Computer Press.
- Chappell, D. (2008). *What is Application Lifecycle Management*. CHAPPELL & ASSOCIATES.
- CMMI Institute. (Agosto de 2016). What is CMMI. Obtenido de <http://cmmiinstitute.com/what-is-cmmi>
- Hernández Sampieri, R., Fernández Callado, C., & Baptista Lucio, P. (2006). *Metodología de la investigación*. México D.F: McGRAW-HILL.
- Institute of Electrical and Electronics Engineers. (1 de Enero de 1963). IEEE. Obtenido de IEEE: <http://trystandards.org/mod/page/view.php?id=3#was>
- Kerzner, H. (2009). *Project Management: A system approach to planning, scheduling, and controlling*. New York: John Wiley & Sons.
- Microsoft. (15 de Septiembre de 2015). Visual Studio ALM Quick Reference Guide. Obtenido de Visual Studio ALM Quick Reference Guide: <https://vsarquickguide.codeplex.com/>
- Microsoft. (Agosto de 2016). Información general de CMMI. Obtenido de <https://msdn.microsoft.com/es-es/library/ee461556.aspx>
- Microsoft. (Agosto de 2016). Team Foundation Service, tu herramienta ALM en la nube. Obtenido de <https://msdn.microsoft.com/es-es/library/dn168158.aspx>
- Microsoft. (Agosto de 2016). Use Team Foundation Version Control. Obtenido de <https://www.visualstudio.com/en-us/docs/tfvc/overview>
- Morales, V. (1994). *Planeamiento y análisis de investigaciones* (8 ed). Caracas: El Dorado.
- Office of Government Commerce. (2007). *ITIL® v3 Service Transition*. United Kingdom: The Stationary Office.

- Office of Government Commerce. (2007). The Official Introduction to the ITIL® Service Lifecycle. United Kingdom: The Stationary Office.
- Project Management Institute. (2008). Guía de los fundamentos para la dirección de proyectos (Guía del PMBOK)-Cuarta Edición. Newtown Square, Pennsylvania: Project Management Institute.
- Real Academia Española. (2016). Diccionario de la lengua española. Obtenido de Diccionario de la lengua española: <http://dle.rae.es/?id=Ta2HMYR>
- Ruiz Ballén, X. (2012). GUÍA ANÁLISIS DE BRECHAS . Bogotá: Universidad Nacional de Colombia.
- Ruiz, M. A., & Patiño, L. D. (20 de Octubre de 2014). Fundamentos redes de comunicación. Obtenido de Fundamentos redes de comunicación: <http://fundamentosderedes.jimdo.com/2-protocolos-arquitectura-de-redes-y-modelo-osi-iso/tipos-de-estándares-y-organismos-de-estandarización/>
- Sabino, C. (1978). El Proceso de la Investigación. Argentina: El Cid Editor.
- Venegas Jiménez, P. (1986). Algunos elementos de investigación. San José: UNED.
-

Apéndice 1.

Cuestionario de arquitectura y diseño

En la Tabla 32 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de arquitectura y diseño.

Tabla 32 Cuestionario de arquitectura y diseño.

#	Pregunta	Respuesta	Puntaje
1	¿Está utilizando herramientas para documentar y compartir modelos de arquitectura?		
2	¿Usted realiza un modelado arquitectónico inicial en el comienzo mismo de su proyecto?		
3	¿Revisa y actualiza periódicamente los documentos de arquitectura?		
4	¿Toman grandes decisiones arquitectónicas a seguir en un proceso definido?		
5	¿La arquitectura se alinea con todas las políticas de seguridad, confidencialidad y privacidad de la organización?		
6	¿La arquitectura proporciona la capacidad para el intercambio de servicios de software comunes y reutilización de los componentes?		
7	¿Tiene una dirección que apunta a la gestión de arquitectura, cuando el software se pone en operación? (Por ejemplo: a través de SNMP)		
8	¿Dirigen escalabilidad, adaptabilidad y la interacción con el cliente durante la fase de arquitectura y diseño?		
9	¿Todos los miembros del equipo tienen acceso a los diagramas de diseño?		
10	¿Revisa periódicamente el diseño de sus proyectos durante el ciclo de vida?		
11	¿Son los diagramas actualizados durante todo el ciclo de vida del proyecto?		
12	¿Estos diagramas se almacenan y controlan versión?		
13	¿Se realiza ingeniería de avance / retroceso de entre el código y los diagramas?		
14	¿Utiliza prototipos antes de su desarrollo?		
15	¿Utilizamos patrones y prácticas bien definidas durante la fase de diseño.?		
16	Si usa de UML, ¿se crean diagramas de secuencia?		
17	¿Utiliza formalmente metodologías de modelado?		
18	¿Es su base de datos documentada usando herramientas de integración?		
19	¿Se sigue un esquema de base de datos consistente con nombramientos de objetos y convenciones?		
20	¿Tiene una efectiva reutilización de código y demostrable sobre la capa de base de datos?		
21	¿Puede controlar el rendimiento de la base de datos de desarrollo?		
		Total de Puntaje	
			Nivel

Fuente: Basado en Microsoft (2016).

Apéndice 2

Cuestionario de ingeniería de requerimientos y experiencia de usuario

En la Tabla 33 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de ingeniería de requerimientos y experiencia de usuario.

Tabla 33 Cuestionario de ingeniería de requerimientos y experiencia de usuario.

#	Pregunta	Respuesta	Puntaje
1	¿Su equipo comprende la visión del cliente para el proyecto (s) que está haciendo para ellos?		
2	¿Su equipo tiene una comprensión de los tipos de usuarios que van a utilizar la aplicación (por ejemplo, las Personas)?		
3	¿Su equipo entiende UML por lo menos casos de uso y lo utiliza apropiadamente?		
4	¿Se crea pruebas de aceptación de los requisitos del cliente?		
5	¿Todas las personas en su equipo comprenden cómo documentar los requisitos y lo hace de la misma manera?		
6	¿Los requerimientos que cambian son actualizados?		
7	¿Hay trazabilidad de los requisitos a las pruebas?		
8	¿Su equipo investiga sobre los tipos de usuarios que tendrá para que pueda crear personajes y clasificarlos?		
9	¿Tiene una manera de crear prototipos de interfaces de usuario?		
10	¿Se crítica la accesibilidad realizada para asegurar la solución puede ser utilizado por usuarios con discapacidad?		
11	¿Se utiliza guiones gráficos? (Stoyboarding)		
12	¿Se llevan a cabo evaluaciones regulares de diseño de interfaz de usuario?		
13	¿Su equipo tiene las capacidades para crear buenas interfaces de usuario (UX) habilidades?		
14	¿Es la experiencia de usuario y / o el diseño de interfaz de usuario exitoso en la comunicación dentro de su equipo y con el cliente?		
15	¿Es cualquier documentación, como material de formación, guías de usuario, ayuda en línea, considerado como un entregable del núcleo si el cliente lo quiere?		
		Total de Puntaje	Nivel

Fuente: Basado en Microsoft, (2016).

Apéndice 3

Cuestionario de desarrollo

En la Tabla 34 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de desarrollo.

Tabla 34 Cuestionario de desarrollo.

#	Pregunta	Respuesta	Puntaje
1	¿Existen normas en vigor para escribir código seguro?		
2	¿Existen normas para la escritura de código que garantiza comentarios, nombres de variables / funciones están libres de malas palabras, declaraciones políticas, religiosas?		
3	¿Hay una práctica de codificación en general bien definido para los namespaces, funciones y nombres de variables?		
4	¿Está bien documentado el código?		
5	¿Hay análisis de código estático eficaz?		
6	¿Hay algún tipo de herramienta de aplicación de estilo de código que estén utilizando?		
7	¿Existen normas para la cobertura de código de pruebas?		
8	¿Hay patrones y prácticas establecidas para la reutilización de código?		
9	¿Hay algunas bibliotecas y repositorios compartidos para la reutilización de código?		
10	¿Los frameworks están disponibles o personalizados y se utilizan para resolver las tareas comunes de codificación?		
11	¿Son los code snippets (fragmentos de código) personalizados, creados y compartidos en todos los miembros del equipo y los equipos?		
12	¿Se llevan acabo revisiones de código efectivas?		
13	¿Se utilizan revisiones de pares?		
14	¿Hay algun artefacto producidos durante las revisiones de código para su uso como una métrica para la mejora?		
15	¿Hay un proceso de registro bien definido y exhaustiva que incluye controles de calidad?		
16	¿Su equipo hace uso de métricas de calidad para mejorar la presentación de informes de código y procesos?		
17	¿Son medidas los fallos de unidades y dirigidos?		
18	¿Está el equipo de DBA o base de datos bien integrado con el resto del equipo del proyecto?		
19	¿Hay revisiones de código para el código de base de datos?		
20	¿Tiene normas formales para el código de base de datos?		
		Total de Puntaje	Nivel

Fuente: Basado en Microsoft, (2016).

Apéndice 4

Cuestionario de administración de la configuración del *software*

En la Tabla 35 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de administración de la configuración del *software*.

Tabla 35 Cuestionario de administración de la configuración del software.

#	Pregunta	Respuesta	Puntaje
1	¿Hay un control de versiones y una estrategia apropiada de ramificación para el equipo y el producto / servicio que se está desarrollando?		
2	¿Su source control y el sistema de ALM tienen un nivel apropiado de acuerdo al nivel de servicio (SLA) para asegurar un impacto mínimo para los equipos de desarrollo en el caso de cualquier punto único de fallo?		
3	¿El sistema de control de código fuente permite la actividad de desarrollo en diferentes sitios geográficos?		
4	Su sistema de control de código fuente permite branch, merge, etiquetado?		
5	¿La estructura del repositorio de código y permisos permiten el desarrollo paralelo?		
6	¿Tiene algún mecanismo de control de versiones de las bases de datos?		
7	¿Su estructura de control de las fuentes de base de datos coincide con la ramificación y fusión de los planes de su código de aplicación?		
8	¿Los scripts de migración de bases de datos son creadas a partir de un proceso de construcción automatizado?		
9	¿Es la creación de datos de prueba repetible con la automatización?		
10	¿Los datos de prueba sensibles ya sea fabricados en prueba o depurados de producción?		
11	¿Todo el código se gestiona a través de un sistema de control de código que proporciona una funcionalidad adecuada tales como el rendimiento, la disponibilidad y características?		
12	¿La estructura del repositorio de código sigue las mejores practicas de documentación o estándar de la compañía?		
13	¿Hay una política de etiquetado consistente?		
14	¿Está el control de código debidamente asegurado?		
15	¿Son las políticas de control de fuentes bien documentadas?		
16	¿Toda la propiedad intelectual del proyecto (código fuente, documentación, etc.) está bajo el control de código fuente efectiva, segura?		
17	¿Sus publicaciones incluyen planes de despliegue y de rollback documentados?		
18	¿Se introdujeron paquetes de implementación temprana en testing de forma similar a cómo la aplicación se está desplegando a la producción?		
19	¿Los despliegues en producción estan a cargo de una persona o de un grupo diferente al los que desarrollaron la aplicación?		

#	Pregunta	Respuesta	Puntaje
20	¿Son sus liberaciones a producción instalados por un script automatizado para el despliegado de forma automática o a través de una herramienta de implementación?		
21	¿Existe un proceso formal para la promoción de las aprobaciones de construcción?		
22	Es producción el punto de partida para implementaciones automáticas de la aplicación y la base de datos?		
23	¿Tiene una persona responsable de la gestión de las compilaciones para asegurarse de que se mantienen hasta la fecha en que la aplicación evoluciona?		
24	¿Para las implementaciones automatizadas en la prueba, no se utiliza el estado de producción actual como la línea de base y después de implementar los cambios en el desarrollo?		
25	¿Hay una biblioteca de builds con éxito?		
26	¿Hay un horario regular de builds?		
27	Se automatiza el proceso de build?		
28	¿Los builds producen binarios que tienen un esquema de versiones significativa?		
29	¿Son sitios de entrega estructurados / organizarse eficazmente para apoyar despliegues en múltiples entornos, tales como desarrollo / pruebas / producción?		
30	¿Existen criterios formales de check-in que rigen los cambios de código fuente?		
31	¿Hay auditoría efectiva de personas que realizan cambios a control de código fuente?		
32	¿Hay auditoría efectiva de por qué se realizan cambios en control de código fuente?		
33	¿Hay seguimiento eficaz de los builds al control de código fuente?		
34	El cumplimiento de SOX se aborda en el sistema de control de código fuente?		
		Total de Puntaje	Nivel

Fuente: Basado en Microsoft, (2016).

Apéndice 5

Cuestionario de gobierno de TI

En la Tabla 36 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de gobierno de TI.

Tabla 36 Cuestionario gobierno de TI.

#	Pregunta	Respuesta	Puntaje
1	¿Está su empresa siguiendo de un marco de gobierno de TI?		
2	¿Tiene una idea clara del caso de negocio antes de iniciar el desarrollo?		
3	¿Hay un conjunto de criterios para decidir sobre la financiación de proyectos?		
4	Se mide el ROI en el desarrollo de software?		
5	¿Hay un programa de métricas en su lugar?		
6	¿Los datos son utilizados por PMO para decidir sobre la priorización de proyectos / aplicación?		
7	¿Se definen los indicadores clave de rendimiento?		
8	¿Hay cuadros de mando en el lugar donde los interesados pueden monitorear el progreso?		
9	¿Realiza reuniones retrospectivos para ajustar la dirección general del proyecto?		
10	¿Se han integrado herramientas de gestión del portafolio y procesos?		
11	¿Se ha integrado la gestión de riesgos en el proceso de gestión del portafolio?		
12	¿Las herramientas que se utilizan para apoyar el marco elegido, incluyen, el riesgo y la oportunidad de negocio umbrales presupuestarios financieros entre otras métricas para evaluar el estado de un proyecto?		
13	¿Se realiza un seguimiento requisitos de gobierno?		
		Total de Puntaje	Nivel

Fuente: Basado en Microsoft (2016).

Apéndice 6

Cuestionario de implementación y operaciones

En la Tabla 37 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de implementación y operaciones.

Tabla 37 Cuestionario de implementación y operaciones.

#	Pregunta	Respuesta	Puntaje
1	¿Hay una arquitectura global en el lugar de la infraestructura?		
2	¿Es diseñado el proceso de la infraestructura en el lugar?		
3	¿Hay alguna retroalimentación del equipo de operaciones para el equipo de desarrollo?		
4	¿IT gestiona la infraestructura de desarrollo?		
5	¿Son las configuraciones de entorno implementado correctamente cada vez?		
6	¿Hay ambientes separados para el desarrollo y producción?		
7	¿El diseño de aplicaciones debe tener en cuenta el modelo de implementación que la empresa está utilizando?		
8	¿Existe la instrumentación efectiva de la aplicación utilizando tecnologías tales como registros de eventos, contadores de rendimiento, y WMI?		
9	¿Están las configuraciones de aplicaciones bajo control de versiones?		
10	¿Hay guías de operación y guías de solución de problemas bajo el control de versiones?		
11	¿Los requisitos son presentados y aprobados antes de hacer cualquier cambio?		
12	¿Son los nuevos clientes y servidores instalados de forma automática?		
13	¿Se mantienen los documentos para el diseño e implementación de infraestructura?		
14	¿Es la arquitectura general de Infraestructura bien entendido por los desarrolladores y testers?		
15	¿Está la infraestructura para las pruebas automatizadas?		
16	¿Tiene las funciones y la gestión de derechos en el lugar de la gestión del entorno de desarrollo?		
17	¿Ha realizado la planificación de capacidad para su entorno de desarrollo?		
18	¿Hay una solución de vigilancia que tenga todos los componentes (red, hardware, sistema operativo, aplicaciones y datos) de operaciones?		
19	¿Hay un proceso de recuperación de desastres en su lugar?		
20	¿Hay una gestión eficaz del entorno de desarrollo?		
21	¿Pueden volver a crear fácilmente entornos de producción?		
22	¿Existe un proceso de escalamiento eficaz de apoyo a la ingeniería?		
23	¿Están cambios de base de datos bajo control de código fuente?		
24	¿Normalmente los cambios de base de datos son validados contra el esquema antes de la implementación?		
25	¿Se han automatizado los builds de su esquema de base de datos?		
		Total de Puntaje Nivel	

Fuente: Basado en Microsoft (2016).

Apéndice 7

Cuestionario de aseguramiento de calidad

En la Tabla 38 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de a aseguramiento de calidad.

Tabla 38 Cuestionario de aseguramiento de calidad.

#	Pregunta	Respuesta	Puntaje
1	¿Hay un líder de prueba dedicada en su lugar?		
2	¿El líder de prueba participa en las fases de planificación y estimación?		
3	¿Están disponibles en el equipo de prueba todas las habilidades de prueba que se requieren?		
4	¿Hay suficientes instalaciones de laboratorio para entornos de prueba disponible para la prueba?		
5	¿Existen herramientas adecuadas disponibles para realizar pruebas manuales (herramientas de depuración, las herramientas de pruebas de carga, etc.)?		
6	¿Existen herramientas adecuadas disponibles para realizar pruebas automatizadas?		
7	¿Existe un sistema de gestión de pruebas en el lugar para realizar un seguimiento de elementos de trabajo, defectos y solicitudes de cambio?		
8	¿Hay alguna manera de realizar un seguimiento de las pruebas de nuevo a los requisitos?		
9	¿Hay un plan de prueba o estrategia en su lugar antes del comienzo de la prueba?		
10	¿Es el equipo de prueba representado en la fase de revisión del diseño?		
11	¿Son los casos de prueba y diseños creados de acuerdo con el diseño?		
12	¿Se han identificado las áreas de mayor riesgo y las pruebas priorizado en consecuencia?		
13	¿Los planes de prueba incluyen la configuración y actualización de escenarios de prueba?		
14	¿Los planes de pruebas consideran la simulación de diferentes entornos de usuario?		
15	¿Los planes de prueba consideren las pruebas de integración con otros sistemas y productos de terceros?		
16	¿Los planes de prueba consideran laboratorios o eventos de pruebas de usabilidad?		
17	¿Se rastreó la ejecución de pruebas en contra del plan de pruebas?		
18	¿Se sigue el plan de pruebas?		
19	¿Hay reportes de procesos correctos en el lugar?		
20	¿Son los criterios de salida de prueba bien definidos y evaluados?		
21	¿Son los criterios para el usuario final o cliente-aceptación bien definidos y evaluados?		
22	¿Se ha asignado suficiente tiempo para la estabilización antes de cada lanzamiento de la solución o producto?		

#	Pregunta	Respuesta	Puntaje
23	¿Tienen en cuenta los casos de pruebas los siguientes requisitos no funcionales : rendimiento, escalabilidad, seguridad, accesibilidad, la regresión, la localización, y la carga / estrés		
24	¿Se ha considerado en su caso de prueba la cobertura de código?		
25	¿Hay herramientas para medir las pruebas pasadas que se deben ejecutar de nuevo cuando una nueva versión de la aplicación se construye?		
26	¿Se utiliza Pruebas de aceptación del usuario (UAT)?		
27	¿Se utiliza pruebas exploratorias?		
28	¿Se utilizan pruebas de regresión formales?		
29	¿Se realizan pruebas de interfaz de usuario automatizados?		
30	¿Se utilizan pruebas de secuencia o pruebas de orden?		
31	¿Se construyen pruebas de validación (prueba de humo)?		
32	¿Se automatizan las pruebas de integración		
33	¿Cualquier herramienta de generación de datos?		
34	¿Alguna herramienta de pruebas de estrés?		
35	¿Cualquier herramienta de análisis de rendimiento?		
36	¿Se han automatizado las pruebas de sus bases de datos?		
37	¿Se prueba distintos elementos del esquema de la base de datos (procedimientos almacenados, funciones, etc.) antes de la implementación?		
38	¿Tiene un conjunto formal de las pruebas que se ejecutan después de cada cambio de esquema?		
39	¿Se ejecuta pruebas de base de datos con los datos de producción?		
40	¿Hay datos de prueba adecuados para asegurar las pruebas de aplicación son válidas?		
41	¿Se ha establecido un conjunto de datos repetibles para la prueba?		
		Total de Puntaje	
		Nivel	

Fuente: Basado en Microsoft (2016).

Apéndice 8

Cuestionario de planificación y administración de proyectos

En la Tabla 39 se muestra el cuestionario que se aplicó para conocer el estado actual del centro de operaciones en el área de planificación y administración de proyectos.

Tabla 39 Cuestionario de planificación y administración de proyectos.

#	Pregunta	Respuesta	Puntaje
1	¿El equipo del proyecto ha sido educado o entrenado en metodologías de desarrollo estándar de la empresa y debe ser capaz de realizar en niveles compatibles con otros equipos?		
2	¿Ha habido algún análisis de recursos realizado en base a los requerimientos del proyecto? Por ejemplo, ¿hay un recurso asignado ETL si hay importaciones / Movimiento datos necesarios?		
3	¿Los miembros del equipo saben dónde encontrar la documentación del proyecto, tales como los requisitos y planes de proyecto y tienen el acceso adecuado a estos documentos? ¿Hay alguna cuenta a los miembros del equipo para revisar y entender estos documentos?		
4	¿Su proyecto sigue una metodología formal para la gestión de proyectos tales como CMM, ágil,?		
5	¿Se revisó el plan del proyecto con frecuencia y se ajusta a las necesidades del cliente?		
6	¿Son los niveles individuales de habilidades considerados cuando se determinan las estimaciones de recursos y otras responsabilidades de estos individuos (Por ejemplo: reuniones, apoyo, otros proyectos)		
7	¿Es la visión y el alcance del proyecto definidos, documentados, y adoptaron como una visión compartida por todo el equipo?		
8	¿Son actividades de control de calidad (seguridad, rendimiento, facilidad de uso, la capacidad de prueba) y Gestión de la Entrega (implantación en la web y base de datos, migraciones) previstas para llevar a cabo simultáneamente con las actividades de desarrollo?		
9	¿Tiene plantillas de documentos con instrucciones para llenar, y ejemplos de documentos para todos los artefactos utilizados en su metodología de gestión de proyectos almacenados en un repositorio único?		
10	¿Tiene el equipo a identificado y acordado los factores de éxito internos que ayuden a alcanzar el objetivo del proyecto?		
11	¿Se realiza un seguimiento del estado del proyecto en contra de la programación del proyecto y refleja si el proyecto se encuentra detrás o en la fecha prevista?		
12	¿La asignación de recursos se gestionan con una forma de calcular la capacidad en comparación con la cantidad de trabajo que se les asigna?		
13	¿Están todos los entregables definidos y entendidos por todos en el equipo y son estos entregables revisados y actualizados con regularidad?		
14	¿¿Están los roles de los miembros del proyecto y las líneas de trabajo claramente definidas y documentadas por el equipo?		

#	Pregunta	Respuesta	Puntaje
15	¿Los individuos trabajan en varios proyectos al mismo tiempo, a menudo cambiando entre ellos durante una semana determinada?		
16	Si procede: ¿Los socios tienen la visibilidad que necesitan para apoyar eficazmente el proyecto?		
17	¿Los proyectos requieren aprobación formal en las etapas de ajuste con el fin de avanzar o liberar presupuesto?		
18	¿Son todas las decisiones del proyecto documentados y fácilmente disponible para todo el equipo?		
19	¿Tiene la firma del Cliente formalmente en el proceso de control de cambio?		
20	¿Está el reporte de estado del proyecto en gran medida calculada automáticamente por indicadores clave de rendimiento predeterminados y estándares que se han acordado métricas en función de la organización?		
21	¿Los riesgos son conocidos y gestionan de forma activa?		
22	¿Se utilizan planes de acción correctiva con disparadores formales?		
23	¿Dependencias de terceros y el cliente son bien administrados?		
24	¿Se sabe todos los grupos de interés internos cuando un proyecto se inicia?		
25	¿Se conocen los grupos de interés externos cuando se inicia un proyecto?		
26	¿Han sido todos los entregables aceptado formalmente? ¿Las salidas se ajustan a los criterios de aceptación definidos?		
27	¿Son los datos de la iteración y la pila de producto preparadas para cerrar una historia o tarea abierta?		
28	¿La documentación del proyecto ya está archivada en un lugar permanente?		
		Total de Puntaje	
		Nivel	

Fuente: Basado en Microsoft (2016).

1. Información general de las pruebas

Número de caso CRM:	<i>Número de caso al que corresponde la prueba</i>
Responsable confección:	<i>Ingeniero encargado de la solución del caso</i>
Versión:	<i>Versión de la prueba</i>
Fecha confección:	<i>Fecha de confección de la prueba</i>
Producto:	<i>Producto afectado en la solución</i>
Componentes afectados:	<i>Componentes del producto afectados en la solución</i>

2. Información detallada de las pruebas**2.1 Caso de prueba X****2.1.1 Descripción de la prueba**

Propósito:	<i>Objetivo de la prueba</i>
Dependencias:	<i>Dependencias de la prueba</i>
Ambiente de prueba:	<i>Ambiente en el que se ejecuta la prueba (sistema operativo, explorador, versión de framework, entre otros)</i>
Precondiciones:	<i>Precondiciones para la ejecución de la prueba</i>
Datos entrada:	<i>Datos de entrada para la ejecución de la prueba</i>
Resultados esperados:	<i>Datos de salida esperados de la ejecución de la prueba</i>
Resultados obtenidos:	<i>Datos de salida obtenidos de la ejecución de la prueba</i>

2.1.2 Pantallas de evidencia

Capturas de pantalla del paso a paso de la ejecución de la prueba.

2.1.3 Observaciones generales

Fecha de ejecución:	<i>Fecha de ejecución de la prueba</i>
Ejecutado por:	<i>Ingeniero responsable de ejecutar la prueba</i>
Revisado por:	<i>Ingeniero responsable de revisar los resultados de la prueba</i>
Estado de la prueba:	<i>Estado final de la prueba (Prueba satisfactoria, Prueba no satisfactoria, Prueba satisfactoria con observaciones)</i>
Detalle observaciones:	<i>Detalle de las observaciones de la prueba</i>

Copie y pegue esta plantilla según la cantidad de pruebas.

Apéndice 10

Verificación de juicio de expertos para el plan de trabajo

1. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Plan de trabajo
Fecha:	<i>Fecha en la que se realiza el juicio de expertos</i>
Expertos:	<i>Personas expertas con los cuales se validarán los temas</i>

2. Validaciones

Ítem	Validado		Recomendaciones
	Si	No	
Recomendaciones descritas correctamente:			
Prioridades entendibles (alta, media, baja):			
Clasificación de las recomendaciones acorde a lo que la empresa espera:			
Las actividades sugeridas son aplicables en la empresa:			
Las recomendaciones son acordes a lo que se espera en ALM			
Las actividades a realizar en son a las que se le debe dar prioridad			
Los tiempos de las actividades a realizar son los idóneos:			
Las personas establecidas para la ejecución de las actividades son las correctas			

Apéndice 11

Verificación de juicio de expertos para estándares y políticas.

1. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Estándares y políticas de ALM
Fecha:	
Expertos:	

2. Validaciones

Ítem	Validado		Recomendaciones
	Si	No	
La política de código de fuente está de acuerdo a los requerimientos de la empresa			
Las estrategias de merge y branch en el código fuente para la atención de casos fue desarrollada de manera correcta para el centro de operaciones			
La estrategia de manejo de código fuente para bases de datos y metodología de trabajo fue desarrollada acorde con las especificaciones de la empresa			

<p>La plantilla de requerimientos funcionales y técnicos se adecua al trabajo diario del centro de operaciones</p>			
<p>La definición del proceso de gestión de riesgos está realizada de la manera óptima</p>			
<p>La metodología planteada para las pruebas técnicas y funcionales tendrá los resultados esperados por el centro de operaciones</p>			
<p>La política para la creación y almacenamiento de conocimiento es de fácil uso</p>			
<p>Las políticas para el deployment son aplicables a la empresa</p>			
<p>Con las métricas del negocio propuestas se podrá determinar si se están cumpliendo con los objetivos</p>			

Apéndice 12

Verificación de juicio de expertos del estado deseado.

1. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Estado deseado
Fecha:	29/08/2016
Expertos:	Michael Gonzales: director del centro de operaciones. Josué Argón: administrador e integrador de aplicaciones.

2. Validaciones

Ítem	Comentarios
¿Cuál es la visión de la empresa proyectada a futuro en cuanto ALM?	
¿Cuáles consideran que son los pasos a seguir para lograr el estado deseado?	
¿Cada una de las etapas de ALM (Arquitectura y diseño, Desarrollo, Gobierno de TI, entre otras) tienen la misma prioridad para la empresa?	
Observaciones:	

Apéndice 13

Verificación de juicio de expertos del plan de trabajo con las respuestas.

1. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Plan de trabajo
Fecha:	05/09/2016
Expertos:	Michael Gonzales: director del centro de operaciones. Josué Argón: administrador e integrador de aplicaciones.

2. Validaciones

Ítem	Validado		Recomendaciones
	Si	No	
Recomendaciones descritas correctamente	X		
Prioridades entendibles (alta, media, baja)	X		
Clasificación de las recomendaciones acorde a lo que la empresa espera:	X		1- Realizar un análisis de los datos sensibles y realizar una encriptación para ambientes de desarrollo y pruebas →Media→Alta 2- Realizar una documentación de procesos de configuración, tanto de ambientes como de soluciones, así como planes de <i>rollback</i> →Media →Baja 3- Implementar proyectos de bases de datos que permitan versionar el proyecto de base datos y no solo los <i>scripts</i> →Alta→Baja 4- Creación de equipo de arquitectos encargado de documentar paulatinamente la arquitectura →Media→Baja
Las actividades sugeridas son aplicables en la empresa	X		
Las recomendaciones son acordes a lo que se espera en ALM	X		

Las actividades a realizar en son a las que se le debe dar prioridad	X		<p>Quitar → Creación de plantilla para dashboards gerenciales con el fin de dar seguimiento de los proyectos</p> <p>Poner → Generar métricas del negocio para poder medir y saber cómo se están haciendo las cosas en el centro de operaciones.</p>
Los tiempos de las actividades a realizar son los idóneos:	X		
Las personas establecidas para la ejecución de las actividades son las correctas	X		<p>En las actividades de:</p> <ul style="list-style-type: none"> • Documentar y definir políticas para el deployment • Generar política para la creación y almacenamiento de conocimiento. • Generación de políticas de código fuente. <p>Definir que solo se harán con ingenieros seniors</p>

Apéndice 14

Verificación de juicio de expertos para políticas y estándares con las respuestas.

1. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Estándares y políticas de ALM
Fecha:	13-09-2017 a 08-11-2017
Expertos:	Michael Gonzales: director del centro de operaciones. Josué Argón: administrador e integrador de aplicaciones.

2. Validaciones

Ítem	Validado		Recomendaciones
	Si	No	
La política de código de fuente está de acuerdo a los requerimientos de la empresa	X		Agregar: <ul style="list-style-type: none"> El acceso a las bases de datos debe realizarse solamente a través de la tecnología Entity Framework de Microsoft. En caso de que el procedimiento almacenado afecte varios objetos, puede omitirse el nombre del objeto que afecta.
Las estrategias de merge y branch en el código fuente para la atención de casos fue desarrollada de manera correcta para el centro de operaciones	X		<ul style="list-style-type: none"> Eliminar los shelves una vez que los cambios son puestos en producción.

			<ul style="list-style-type: none"> • Quitar la función al coordinador de crear tarea en TFS y dársela al desarrollador a cargo. • Coordinar con el integrador de aplicaciones la integración con los otros branch. • Finalizar o dar por terminadas las tareas en el TFS.
La estrategia de manejo de código fuente para bases de datos y metodología de trabajo fue desarrollada acorde con las especificaciones de la empresa	X		
La plantilla de requerimientos funcionales y técnicos se adecua al trabajo diario del centro de operaciones	X		<ul style="list-style-type: none"> • Los tipos de requerimientos son funcionales y técnicos, incluirlos en la plantilla. • Dividir los requerimientos en alto y bajo alcance.
La definición del proceso de gestión de riesgos está realizada de la manera óptima	X		
La metodología planteada para las pruebas técnicas y funcionales tendrá los resultados esperados por el centro de operaciones	X		<ul style="list-style-type: none"> • Agregar a la plantilla de pruebas el estado de la prueba y versión de la prueba.
La política para la creación y almacenamiento de conocimiento es de fácil uso	X		<ul style="list-style-type: none"> • Agregar más imágenes para recursos que no conocen muy bien el CRM.
Las políticas para el deployment son aplicables a la empresa	X		<ul style="list-style-type: none"> • El documento que debe realizar el ingeniero para hacer rollback debe contener como mínimo: <ul style="list-style-type: none"> ○ Ambiente ○ Equipos ○ Componentes ○ Rutas
Con las métricas del negocio propuestas se podrá determinar si se están cumpliendo con los objetivos	X		<ul style="list-style-type: none"> • Agregar una métrica para mostrar la cantidad de inconsistencias en las actividades de desarrollo de software

Apéndice 15

Verificación de juicio de expertos del estado deseado con respuestas.

3. Información general del juicio de expertos

Nombre del proyecto:	PROPUESTA DE UNA SOLUCIÓN PARA LA DEFINICIÓN DEL PROCESO DE ENTREGA DE SOFTWARE, BASADA EN LA GESTIÓN DEL CICLO DE VIDA DE LAS APLICACIONES
Tema:	Estado deseado
Fecha:	29/08/2016
Expertos:	Michael Gonzales: director del centro de operaciones. Josué Argón: administrador e integrador de aplicaciones.

4. Validaciones

Ítem	Comentarios
¿Cuál es la visión de la empresa proyectada a futuro en cuanto ALM?	Se desea tener automatizado todas las etapas de ALM, por lo tanto, la empresa espera tener un estado dinámico.
¿Cuáles consideran que son los pasos a seguir para lograr el estado deseado?	Primero se necesita estandarizar todas las etapas de ALM a un mismo nivel y a partir de este estado se mejorarían todas las etapas a dinámico.
¿Cada una de las etapas de ALM (Arquitectura y diseño, Desarrollo, Gobierno de TI, entre otras) tienen la misma prioridad para la empresa?	Si, la empresa desea mejorar cada una de las etapas y no dejar rezagadas ninguna de ellas. Lo que se desea es tener todas las etapas a un mismo nivel.
Observaciones:	
Se determina que lo más efectivo, en este momento para la empresa es primero nivelar cada una de las etapas a un estado avanzado; por lo que este proyecto se centrará en esta primera fase en mejorar las etapas a un nivel avanzado. Una vez alcanzado este estado la empresa buscará llegar a un estado dinámico pero este aspecto queda fuera del alcance de este proyecto.	

Apéndice 16

Cuestionario de arquitectura y diseño con datos

En la Tabla 40 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 32.

Tabla 40 Cuestionario de arquitectura y diseño con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Está utilizando herramientas para documentar y compartir modelos de arquitectura?	Nunca	1
2	¿Usted realiza un modelado arquitectónico inicial en el comienzo mismo de su proyecto?	Nunca	1
3	¿Revisa y actualiza periódicamente los documentos de arquitectura?	Nunca	1
4	Toman grandes decisiones arquitectónicas a seguir en un proceso definido?	Algunas Veces	2
5	¿La arquitectura se alinea con todas las políticas de seguridad, confidencialidad y privacidad de la organización?	Algunas Veces	2
6	¿La arquitectura proporciona la capacidad para el intercambio de servicios de software comunes y reutilización de los componentes?	Siempre	4
7	¿Tiene una dirección que apunta a la gestión de arquitectura, cuando el software se pone en operación? (Por ejemplo: a través de SNMP)	Algunas Veces	2
8	¿Dirigen escalabilidad, adaptabilidad y la interacción con el cliente durante la fase de arquitectura y diseño?	Algunas Veces	2
9	¿Todos los miembros del equipo tienen acceso a los diagramas de diseño?	Siempre	4
10	¿Revisa periódicamente el diseño de sus proyectos durante el ciclo de vida?	Nunca	1
11	¿Son los diagramas actualizados durante todo el ciclo de vida del proyecto?	Nunca	1
12	¿Estos diagramas se almacenan y controlan versión?	Nunca	1
13	¿Se realiza ingeniería de avance / retroceso de entre el código y los diagramas?	Nunca	1
14	¿Utiliza prototipos antes de su desarrollo?	Algunas Veces	2
15	¿Utilizamos patrones y prácticas bien definidas durante la fase de diseño.?	Nunca	1
16	Si usa de UML, ¿se crean diagramas de secuencia?	Nunca	1
17	¿Utiliza formalmente metodologías de modelado?	Nunca	1
18	¿Es su base de datos documentada usando herramientas de integración?	Nunca	1
19	¿Se sigue un esquema de base de datos consistente con nombramientos de objetos y convenciones?	Frecuentemente	3
20	¿Tiene una efectiva reutilización de código y demostrable sobre la capa de base de datos?	Frecuentemente	3
21	¿Puede controlar el rendimiento de la base de datos de desarrollo?	Nunca	1
		Total de Puntaje	20
		Nivel Estandarizado	

Fuente: Basado en Microsoft (2016).

Apéndice 17

Cuestionario de ingeniería de requerimientos y experiencia de usuario con datos

En la Tabla 41 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 33.

Tabla 41 Cuestionario de ingeniería de requerimientos y experiencia de usuario con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Su equipo comprender la visión del cliente para el proyecto (s) que está haciendo para ellos?	Frecuentemente	3
2	¿Su equipo tiene una comprensión de los tipos de usuarios que van a utilizar la aplicación (por ejemplo, las Personas)?	Frecuentemente	3
3	¿Su equipo entiende UML por lo menos casos de uso y lo utiliza apropiadamente?	Nunca	1
4	¿Se crea pruebas de aceptación de los requisitos del cliente?	Frecuentemente	3
5	¿Todas las personas en su equipo comprenden cómo documentar los requisitos y lo hace de la misma manera?	Frecuentemente	3
6	¿Los requerimientos que cambian son actualizados?	Siempre	4
7	¿Hay trazabilidad de los requisitos a las pruebas?	Frecuentemente	3
8	¿Su equipo investiga sobre los tipos de usuarios que tendrá para que pueda crear personajes y clasificarlos?	Frecuentemente	3
9	¿Tiene una manera de crear prototipos de interfaces de usuario?	Nunca	1
10	¿Se crítica la accesibilidad realizada para asegurar la solución puede ser utilizado por usuarios con discapacidad?	Nunca	1
11	¿Se utiliza guiones gráficos? (Stoyboarding)	Nunca	1
12	¿Se llevan a cabo evaluaciones regulares de diseño de interaz de usuario?	Algunas Veces	2
13	¿Su equipo tiene las capacidades para crear buenas interfaces de usuario (UX) habilidades?	Frecuentemente	3
14	¿Es la experiencia de usuario y / o el diseño de interfaz de usuario exitoso en la comunicación dentro de su equipo y con el cliente?	Frecuentemente	3
15	¿Es cualquier documentación, como material de formación, guías de usuario, ayuda en línea, considerado como un entregable del núcleo si el cliente lo quiere?	Frecuentemente	3
		Total de Puntaje	20
		Nivel Estandarizado	

Fuente: Basado en Microsoft (2016).

Apéndice 18

Cuestionario de desarrollo con datos

En la Tabla 42 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 34.

Tabla 42 Cuestionario de desarrollo con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Existen normas en vigor para escribir código seguro?	Nunca	1
2	¿Existen normas para la escritura de código que garantiza comentarios, nombres de variables / funciones están libres de malas palabras, declaraciones políticas, religiosas?	Nunca	1
3	¿Hay una práctica de codificación en general bien definido para los namespaces, funciones y nombres de variables?	Algunas Veces	2
4	¿Está bien documentado el código?	Algunas Veces	2
5	¿Hay análisis de código estático eficaz?	Nunca	1
6	¿Hay algún tipo de herramienta de aplicación de estilo de código que estén utilizando?	Frecuentemente	3
7	¿Existen normas para la cobertura de código de pruebas?	Nunca	1
8	¿Hay patrones y prácticas establecidas para la reutilización de código?	Siempre	4
9	¿Hay algunas bibliotecas y repositorios compartidos para la reutilización de código?	Siempre	4
10	¿Los frameworks están disponibles o personalizados y se utilizan para resolver las tareas comunes de codificación?	Frecuentemente	3
11	¿Son los code snippets (fragmentos de código) personalizados, creados y compartidos en todos los miembros del equipo y los equipos?	Algunas Veces	2
12	¿Se llevan acabo revisiones de código efectivas?	Nunca	1
13	¿Se utilizan revisiones de pares?	Nunca	1
14	¿Hay algun artefacto producidos durante las revisiones de código para su uso como una métrica para la mejora?	Nunca	1
15	¿Hay un proceso de registro bien definido y exhaustiva que incluye controles de calidad?	Nunca	1
16	¿Su equipo hace uso de métricas de calidad para mejorar la presentación de informes de código y procesos?	Nunca	1
17	¿Son medidas llos fallos de unidades y dirigidos?	Nunca	1
18	¿Está el equipo de DBA o base de datos bien integrado con el resto del equipo del proyecto?	Nunca	1
19	¿Hay revisiones de código para el código de base de datos?	Nunca	1
20	¿Tiene normas formales para el código de base de datos?	Nunca	1
		Total de Puntaje	20
		Nivel Estandarizado	

Fuente: Basado en Microsoft (2016).

Apéndice 19

Cuestionario de administración de la configuración del *software*

En la Tabla 43 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 35.

Tabla 43 Cuestionario de planificación y administración de proyectos con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Hay un control de versiones y una estrategia apropiada de ramificación para el equipo y el producto / servicio que se está desarrollando?	Algunas Veces	2
2	¿Su source control y el sistema de ALM tienen un nivel apropiado de acuerdo al nivel de servicio (SLA) para asegurar un impacto mínimo para los equipos de desarrollo en el caso de cualquier punto único de fallo?	Algunas Veces	2
3	¿El sistema de control de código fuente permite la actividad de desarrollo en diferentes sitios geográficos?	Nunca	1
4	Su sistema de control de código fuente permite branch, merge, etiquetado?	Frecuentemente	3
5	¿La estructura del repositorio de código y permisos permiten el desarrollo paralelo?	Nunca	1
6	¿Tiene algún mecanismo de control de versiones de las bases de datos?	Nunca	1
7	¿Su estructura de control de las fuentes de base de datos coincide con la ramificación y fusión de los planes de su código de aplicación?	Algunas Veces	2
8	¿Los scripts de migración de bases de datos son creadas a partir de un proceso de construcción automatizado?	Nunca	1
9	¿Es la creación de datos de prueba repetible con la automatización?	Nunca	1
10	¿Los datos de prueba sensibles ya sea fabricados en prueba o depurados de producción?	Nunca	1
11	¿Todo el código se gestiona a través de un sistema de control de código que proporciona una funcionalidad adecuada tales como el rendimiento, la disponibilidad y características?	Siempre	4
12	¿La estructura del repositorio de código sigue las mejores practicas de documentación o estándar de la compañía?	Nunca	1
13	¿Hay una política de etiquetado consistente?	Nunca	1
14	¿Está el control de código debidamente asegurado?	Siempre	4
15	¿Son las políticas de control de fuentes bien documentadas?	Nunca	1
16	¿Toda la propiedad intelectual del proyecto (código fuente, documentación, etc.) está bajo el control de código fuente efectiva, segura?	Frecuentemente	3
17	¿Sus publicaciones incluyen planes de despliegue y de rollback documentados?	Nunca	1
18	¿Se introdujeron paquetes de implementación temprana en testing de forma similar a cómo la aplicación se está desplegando a la producción?	Algunas Veces	2
19	¿Los despliegues en producción estan a cargo de una persona o de un grupo diferente al los que desarrollaron la aplicación?	Frecuentemente	3

#	Pregunta	Respuesta	Puntaje
20	¿Son sus liberaciones a producción instalados por un script automatizado para el despliegado de forma automática o a través de una herramienta de implementación?	Nunca	1
21	¿Existe un proceso formal para la promoción de las aprobaciones de construcción?	Nunca	1
22	Es producción el punto de partida para implementaciones automáticas de la aplicación y la base de datos?	Nunca	1
23	¿Tiene una persona responsable de la gestión de las compilaciones para asegurarse de que se mantienen hasta la fecha en que la aplicación evoluciona?	Nunca	1
24	¿Para las implementaciones automatizadas en la prueba, no se utiliza el estado de producción actual como la línea de base y después de implementar los cambios en el desarrollo?	Nunca	1
25	¿Hay una biblioteca de builds con éxito?	Nunca	1
26	¿Hay un horario regular de builds?	Nunca	1
27	Se automatiza el proceso de build?	Nunca	1
28	¿Los builds producen binarios que tienen un esquema de versiones significativa?	Nunca	1
29	¿Son sitios de entrega estructurados / organizarse eficazmente para apoyar despliegues en múltiples entornos, tales como desarrollo / pruebas / producción?	Nunca	1
30	¿Existen criterios formales de check-in que rigen los cambios de código fuente?	Nunca	1
31	¿Hay auditoría efectiva de personas que realizan cambios a control de código fuente?	Nunca	1
32	¿Hay auditoría efectiva de por qué se realizan cambios en control de código fuente?	Nunca	1
33	¿Hay seguimiento eficaz de los builds al control de código fuente?	Nunca	1
34	El cumplimiento de SOX se aborda en el sistema de control de código fuente?	Nunca	1
		Total de Puntaje	10
		Nivel	Básico

Fuente: Basado en Microsoft (2016).

Apéndice 20

Cuestionario de gobierno de TI con datos

En la Tabla 44 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 36.

Tabla 44 Cuestionario gobierno de TI con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Está su empresa siguiendo de un marco de gobierno de TI?	Algunas Veces	2
2	¿Tiene una idea clara del caso de negocio antes de iniciar el desarrollo?	Siempre	4
3	¿Hay un conjunto de criterios para decidir sobre la financiación de proyectos?	Siempre	4
4	Se mide el ROI en el desarrollo de software?	Frecuentemente	3
5	¿Hay un programa de métricas en su lugar?	Nunca	1
6	¿Los datos son utilizados por PMO para decidir sobre la priorización de proyectos / aplicación?	Siempre	4
7	¿Se definen los indicadores clave de rendimiento?	Algunas Veces	2
8	¿Hay cuadros de mando en el lugar donde los interesados pueden monitorear el progreso?	Algunas Veces	2
9	¿Realiza reuniones retrospectivos para ajustar la dirección general del proyecto?	Algunas Veces	2
10	¿Se han integrado herramientas de gestión del portafolio y procesos?	Nunca	1
11	¿Se ha integrado la gestión de riesgos en el proceso de gestión del portafolio?	Nunca	1
12	¿Las herramientas que se utilizan para apoyar el marco elegido, incluyen, el riesgo y la oportunidad de negocio umbrales presupuestarios financieros entre otras métricas para evaluar el estado de un proyecto?	Nunca	1
13	¿Se realiza un seguimiento requisitos de gobierno?	Nunca	1
		Total de Puntaje	20
		Nivel Estandarizado	

Fuente: Basado en Microsoft (2016).

Apéndice 21

Cuestionario de implementación y operaciones con datos

En la Tabla 45 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 37.

Tabla 45 Cuestionario de implementación y operaciones con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Hay una arquitectura global en el lugar de la infraestructura?	Nunca	1
2	¿Es diseñado el proceso de la infraestructura en el lugar?	Nunca	1
3	¿Hay alguna retroalimentación del equipo de operaciones para el equipo de desarrollo?	Nunca	1
4	¿IT gestiona la infraestructura de desarrollo?	Nunca	1
5	¿Son las configuraciones de entorno implementado correctamente cada vez?	Algunas Veces	2
6	¿Hay ambientes separados para el desarrollo y producción?	Algunas Veces	2
7	¿El diseño de aplicaciones debe tener en cuenta el modelo de implementación que la empresa está utilizando?	Algunas Veces	2
8	¿Existe la instrumentación efectiva de la aplicación utilizando tecnologías tales como registros de eventos, contadores de rendimiento, y WMI?	Algunas Veces	2
9	¿Están las configuraciones de aplicaciones bajo control de versiones?	Algunas Veces	2
10	¿Hay guías de operación y guías de solución de problemas bajo el control de versiones?	Nunca	1
11	¿Los requisitos son presentados y aprobados antes de hacer cualquier cambio?	Frecuentemente	3
12	¿Son los nuevos clientes y servidores instalados de forma automática?	Nunca	1
13	¿Se mantienen los documentos para el diseño e implementación de infraestructura?	Nunca	1
14	¿Es la arquitectura general de Infraestructura bien entendido por los desarrolladores y testers?	Nunca	1
15	¿Está la infraestructura para las pruebas automatizadas?	Nunca	1
16	¿Tiene las funciones y la gestión de derechos en el lugar de la gestión del entorno de desarrollo?	Algunas Veces	2
17	¿Ha realizado la planificación de capacidad para su entorno de desarrollo?	Nunca	1
18	¿Hay una solución de vigilancia que tenga todos los componentes (red, hardware, sistema operativo, aplicaciones y datos) de operaciones?	Nunca	1
19	¿Hay un proceso de recuperación de desastres en su lugar?	Nunca	1
20	¿Hay una gestión eficaz del entorno de desarrollo?	Nunca	1
21	¿Pueden volver a crear fácilmente entornos de producción?	Algunas Veces	2
22	¿Existe un proceso de escalamiento eficaz de apoyo a la ingeniería?	Nunca	1
23	¿Están cambios de base de datos bajo control de código fuente?	Siempre	4
24	¿Normalmente los cambios de base de datos son validados contra el esquema antes de la implementación?	Nunca	1
25	¿Se han automatizado los builds de su esquema de base de datos?	Nunca	1
		Total de Puntaje	10
		Nivel	Básico

Fuente: Basado en Microsoft (2016).

Apéndice 22

Cuestionario de aseguramiento de calidad con datos

En la Tabla 46 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 38.

Tabla 46 Cuestionario de aseguramiento de calidad con datos.

#	Pregunta	Respuesta	Puntaje
1	¿Hay un líder de prueba dedicada en su lugar?	Siempre	4
2	¿El líder de prueba participa en las fases de planificación y estimación?	Siempre	4
3	¿Están disponibles en el equipo de prueba todas las habilidades de prueba que se requieren?	Algunas Veces	2
4	¿Hay suficientes instalaciones de laboratorio para entornos de prueba disponible para la prueba?	Nunca	1
5	¿Existen herramientas adecuadas disponibles para realizar pruebas manuales (herramientas de depuración, las herramientas de pruebas de carga, etc.)?	Algunas Veces	2
6	¿Existen herramientas adecuadas disponibles para realizar pruebas automatizadas?	Nunca	1
7	¿Existe un sistema de gestión de pruebas en el lugar para realizar un seguimiento de elementos de trabajo, defectos y solicitudes de cambio?	Frecuentemente	3
8	¿Hay alguna manera de realizar un seguimiento de las pruebas de nuevo a los requisitos?	Algunas Veces	2
9	¿Hay un plan de prueba o estrategia en su lugar antes del comienzo de la prueba?	Algunas Veces	2
10	¿Es el equipo de prueba representado en la fase de revisión del diseño?	Frecuentemente	3
11	¿Son los casos de prueba y diseños creados de acuerdo con el diseño?	Algunas Veces	2
12	¿Se han identificado las áreas de mayor riesgo y las pruebas priorizado en consecuencia?	Algunas Veces	2
13	¿Los planes de prueba incluyen la configuración y actualización de escenarios de prueba?	Algunas Veces	2
14	¿Los planes de pruebas consideran la simulación de diferentes entornos de usuario?	Frecuentemente	3
15	¿Los planes de prueba consideren las pruebas de integración con otros sistemas y productos de terceros?	Algunas Veces	2
16	¿Los planes de prueba consideran laboratorios o eventos de pruebas de usabilidad?	Algunas Veces	2
17	¿Se rastreó la ejecución de pruebas en contra del plan de pruebas?	Algunas Veces	2
18	¿Se sigue el plan de pruebas?	Frecuentemente	3
19	¿Hay reportes de procesos correctos en el lugar?	Nunca	1
20	¿Son los criterios de salida de prueba bien definidos y evaluados?	Frecuentemente	3
21	¿Son los criterios para el usuario final o cliente-aceptación bien definidos y evaluados?	Algunas Veces	2
22	¿Se ha asignado suficiente tiempo para la estabilización antes de cada lanzamiento de la solución o producto?	Nunca	1

#	Pregunta	Respuesta	Puntaje
23	¿Tienen en cuenta los casos de pruebas los siguientes requisitos no funcionales : rendimiento, escalabilidad, seguridad, accesibilidad, la regresión, la localización, y la carga / estrés	Algunas Veces	2
24	¿Se ha considerado en su caso de prueba la cobertura de código?	Nunca	1
25	¿Hay herramientas para medir las pruebas pasadas que se deben ejecutar de nuevo cuando una nueva versión de la aplicación se	Nunca	1
26	¿Se utiliza Pruebas de aceptación del usuario (UAT)?	Nunca	1
27	¿Se utiliza pruebas exploratorias?	Siempre	4
28	¿Se utilizan pruebas de regresión formales?	Nunca	1
29	¿Se realizan pruebas de interfaz de usuario automatizados?	Nunca	1
30	¿Se utilizan pruebas de secuencia o pruebas de orden?	Algunas Veces	2
31	¿Se construyen pruebas de validación (prueba de humo)?	Siempre	4
32	¿Se automatizan las pruebas de integración	Nunca	1
33	¿Cualquier herramienta de generación de datos?	Algunas Veces	2
34	¿Alguna herramienta de pruebas de estrés?	Nunca	1
35	¿Cualquier herramienta de análisis de rendimiento?	Nunca	1
36	¿Se han automatizado las pruebas de sus bases de datos?	Nunca	1
37	¿Se prueba distintos elementos del esquema de la base de datos (procedimientos almacenados, funciones, etc.) antes de la	Nunca	1
38	¿Tiene un conjunto formal de las pruebas que se ejecutan después de cada cambio de esquema?	Nunca	1
39	¿Se ejecuta pruebas de base de datos con los datos de producción?	Nunca	1
40	¿Hay datos de prueba adecuados para asegurar las pruebas de aplicación son válidas?	Nunca	1
41	¿Se ha establecido un conjunto de datos repetibles para la prueba?	Nunca	1
Total de Puntaje			20
Nivel Estandarizado			

Fuente: Basado en Microsoft (2016).

Apéndice 23

Cuestionario de planificación y administración de proyectos con datos

En la Tabla 47 se muestran las respuestas que se obtuvieron al aplicar el cuestionario de la Tabla 39.

Tabla 47 Cuestionario de planificación y administración de proyectos con datos.

#	Pregunta	Respuesta	Puntaje
1	¿El equipo del proyecto ha sido educado o entrenado en metodologías de desarrollo estándar de la empresa y debe ser capaz de realizar en niveles compatibles con otros equipos?	Algunas Veces	2
2	¿Ha habido algún análisis de recursos realizado en base a los requerimientos del proyecto? Por ejemplo, ¿hay un recurso asignado ETL si hay importaciones / Movimiento datos necesarios?	Nunca	1
3	¿Los miembros del equipo saben dónde encontrar la documentación del proyecto, tales como los requisitos y planes de proyecto y tienen el acceso adecuado a estos documentos? ¿Hay alguna cuenta a los miembros del equipo para revisar y entender estos documentos?	Algunas Veces	2
4	¿Su proyecto sigue una metodología formal para la gestión de proyectos tales como CMM, ágil,?	Nunca	1
5	¿Se revisó el plan del proyecto con frecuencia y se ajusta a las necesidades del cliente?	Algunas Veces	2
6	¿Son los niveles individuales de habilidades considerados cuando se determinan las estimaciones de recursos y otras responsabilidades de estos individuos (Por ejemplo: reuniones, apoyo, otros proyectos)	Algunas Veces	2
7	¿Es la visión y el alcance del proyecto definidos, documentados, y adoptaron como una visión compartida por todo el equipo?	Nunca	1
8	¿Son actividades de control de calidad (seguridad, rendimiento, facilidad de uso, la capacidad de prueba) y Gestión de la Entrega (implantación en la web y base de datos, migraciones) previstas para llevar a cabo simultáneamente con las actividades de desarrollo?	Nunca	1
9	¿Tiene plantillas de documentos con instrucciones para llenar, y ejemplos de documentos para todos los artefactos utilizados en su metodología de gestión de proyectos almacenados en un repositorio único?	Nunca	1
10	¿Tiene el equipo a identificado y acordado los factores de éxito internos que ayuden a alcanzar el objetivo del proyecto?	Nunca	1
11	¿Se realiza un seguimiento del estado del proyecto en contra de la programación del proyecto y refleja si el proyecto se encuentra detrás o en la fecha prevista?	Algunas Veces	2
12	¿La asignación de recursos se gestionan con una forma de calcular la capacidad en comparación con la cantidad de trabajo que se les asigna?	Nunca	1
13	¿Están todos los entregables definidos y entendidos por todos en el equipo y son estos entregables revisados y actualizados con regularidad?	Nunca	1
14	¿¿Están los roles de los miembros del proyecto y las líneas de trabajo claramente definidas y documentadas por el equipo?	Nunca	1

#	Pregunta	Respuesta	Puntaje
15	¿Los individuos trabajan en varios proyectos al mismo tiempo, a menudo cambiando entre ellos durante una semana determinada?	Algunas Veces	2
16	Si procede: ¿Los socios tienen la visibilidad que necesitan para apoyar eficazmente el proyecto?	Nunca	1
17	¿Los proyectos requieren aprobación formal en las etapas de ajuste con el fin de avanzar o liberar presupuesto?	Nunca	1
18	¿Son todas las decisiones del proyecto documentados y fácilmente disponible para todo el equipo?	Nunca	1
19	¿Tiene la firma del Cliente formalmente en el proceso de control de cambio?	Siempre	4
20	¿Está el reporte de estado del proyecto en gran medida calculada automáticamente por indicadores clave de rendimiento predeterminados y estándares que se han acordado métricas en función de la organización?	Nunca	1
21	¿Los riesgos son conocidos y gestionan de forma activa?	Nunca	1
22	¿Se utilizan planes de acción correctiva con disparadores formales?	Nunca	1
23	¿Dependencias de terceros y el cliente son bien administrados?	Algunas Veces	2
24	¿Se sabe todos los grupos de interés internos cuando un proyecto se inicia?	Frecuentemente	3
25	¿Se conocen los grupos de interés externos cuando se inicia un proyecto?	Nunca	1
26	¿Han sido todos los entregables aceptado formalmente? ¿Las salidas se ajustan a los criterios de aceptación definidos?	Algunas Veces	2
27	¿Son los datos de la iteración y la pila de producto preparadas para cerrar una historia o tarea abierta?	Nunca	1
28	¿La documentación del proyecto ya está archivada en un lugar permanente?	Nunca	1
Total de Puntaje			10
Nivel			Básico

Fuente: Basado en Microsoft (2016).

Anexo 1

Carta de la Filóloga

San José, 10 de noviembre de 2016

Sres.
Instituto Tecnológico de Costa Rica
Área Académica de Administración de Tecnología de Información
Tribunal Evaluador del Trabajo Final de Graduación
S.D.

Estimados señores:

Hago constar que leí y corregí el Trabajo Final de Graduación elaborado por el estudiante Walter Cordero Ureña, denominado "Propuesta de una solución para la definición del proceso de entrega de *software*, basada en la gestión del ciclo de vida de las aplicaciones" para optar por el grado académico de Licenciatura en Administración de Tecnología de Información.

Se revisaron errores gramaticales, de puntuación, ortográficos, de estilo, de concordancia y otros relacionados con el campo filológico. Con base en lo anterior, se considera que dicho trabajo cumple con los requerimientos para ser presentado como requisito de conclusión por parte del postulante al grado de Licenciatura. Por último, la edición final del documento, que incluirá o excluirá los comentarios de la filóloga, queda bajo la completa responsabilidad del solicitante del servicio.

Se suscribe de ustedes cordialmente,

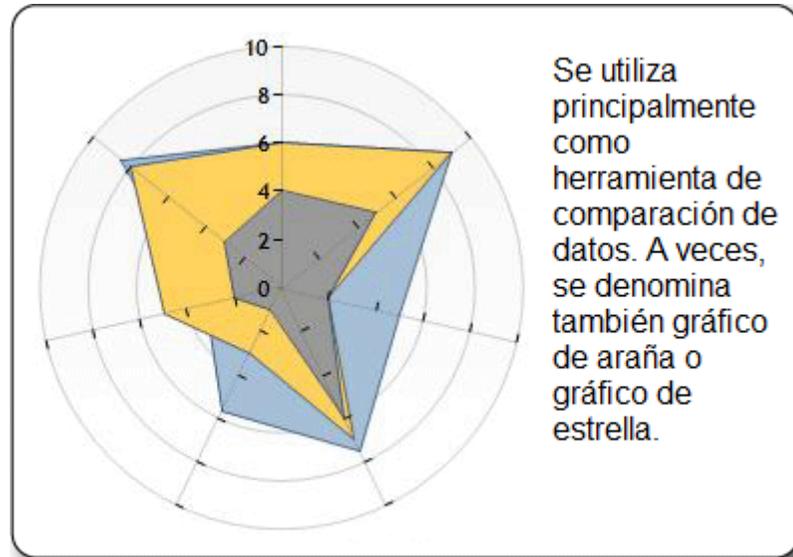
María Alejandra Brenes Rodríguez
Filóloga Española
Cédula: 1-1955-0982
Carné: 094
Teléfono: (506) 8754 8904
Correo electrónico: Alejandra.Brenes.R@gmail.com



Firma

Anexo 2
Gráfico Radial

Gráfico 20 Gráfico radial.



Fuente: Microsoft (2016).

En la Tabla 48 Glosario se definen los términos no conocidos o poco comunes mencionados en el documento.

Tabla 48 Glosario.

Palabra	Significado
Interbancarias	Entre bancos o que los relaciona.
SINPE	Sistema Nacional de Pagos Electrónicos.
Corredurías	Intermediario experto que media entre la compañía y el asegurado.
Contingencia	Suceso que puede ocurrir o no, especialmente un problema que se plantea de forma imprevista.
Compilar	Traducir un código de programación a código ejecutable por la máquina.
Cimiento	Principios
<i>Ad hoc</i>	Para este propósito o para esto.
<i>Merge</i>	Unir
<i>Release</i>	Liberar
<i>Service pack</i>	Conjunto de programas informáticos que consisten en un grupo de actualizaciones que corrigen y mejoran aplicaciones y sistemas operativos.
<i>Hotfix</i>	Es un paquete que puede incluir varios archivos y que sirve para resolver un bug específico dentro de una aplicación informática.
<i>Frontend</i>	Capa de presentación.
<i>Backend</i>	Capa de acceso a los datos.
<i>CheckIn</i>	Proceso por el cual se sube el código al TFS, es el equivalente al <i>Commit</i> a nivel de SVN, una vez que se realiza el <i>checkin</i> se genera un <i>changeset</i> de los cambios que se subieron en la solución.
<i>CheckOut</i>	Proceso en el cual se toma uno o varios archivos para trabajarlos en el ambiente local, de manera que genera cambios pendientes de subir al TFS.

Palabra	Significado
<i>Changeset</i>	Conjunto de cambios que se genera a partir de un <i>checkin</i> , es el equivalente a una “revisión” a nivel de SVN, básicamente se convierte en una versión de la solución.
<i>Shelveset</i>	Corresponde a un conjunto de cambios aplazados en los cuales el desarrollador puede subir el código fuente al TFS, pero este no se sube directamente al <i>branch</i> o carpeta, sino que se sube solamente el conjunto de cambios de una manera independiente para respaldar los cambios o bien mantener el código seguro dentro del TFS y no almacenarlo localmente.
<i>Merge</i>	Proceso mediante el cual se mezclan los cambios de dos <i>branchs</i> diferentes, aunque se tome solo uno como origen y un destino.
<i>Branch</i>	Proceso mediante el cual se realiza la ramificación de una carpeta, para realizar una copia en el servidor y poder trabajarla para un fin diferente (Equipos de trabajo, QA, Producción, entre otros).
<i>Label</i>	Es el proceso en el cual se genera una etiqueta por medio de la cual se genera un entregable listo, para realizar alguna acción o pasarlo de ambiente, para generar una versión especial. Las etiquetas permiten obtener una instantánea de los archivos que podrá consultarse posteriormente. Mediante una etiqueta, se puede ver, compilar o incluso revertir un amplio conjunto de archivos al estado en el que estaban cuando se aplicó la etiqueta.
<i>Workspace</i>	Las áreas de trabajo son la copia local del código base del equipo. Aquí es donde puede desarrollar y probar su código en aislamiento hasta que esté listo para comprobar en su trabajo. En la mayoría de los casos el único espacio de trabajo que necesita se crea automáticamente para usted y usted no tiene que editarlo.

Palabra	Significado
	En algunos casos, es posible que desee modificar su espacio de trabajo o crear múltiples espacios de trabajo para aislar y cambiar entre los cambios que está haciendo en diferentes ramas.
<i>Lock / Unlock</i>	Es el proceso de bloquear los <i>branches</i> , en este momento ninguna persona del equipo pueda hacer <i>checkin</i> del código, solamente pueda descargarlo. Se utiliza, más que todo, al momento de realizar los <i>merges</i> y también cuando existe un <i>branch</i> que no debe ser modificado.
Diagrama	Dibujo en el que se muestran las relaciones entre las diferentes partes de un conjunto o sistema.
Nomenclatura	Conjunto de términos o palabras propias utilizadas en una ciencia, técnica, o especialidad, o por un autor.
<i>Storyboarding</i>	Es un conjunto de ilustraciones o ideas mostradas en secuencia con el objetivo de servir de guía para entender un caso.

Fuente: Elaboración propia, 2016.