"Responsabilidad con pensamiento positivo"

UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN

CARRERA: SISTEMAS INFORMÁTICOS

TEMA: MODELO DE GESTIÓN PARA MANTENIMIENTO Y MEJORAS DE SOFTWARE COMO SERVICIO.

AUTOR: FERNANDO ANDRÉS ARIZAGA PROAÑO

TUTOR: ING. WILMER VALLE

OCTUBRE 2014

QUITO - ECUADOR

INFORME FINAL DEL TRABAJO DE TITULACIÓN

Carrera:	Sistemas Informáticos
Autor/a:	Fernando Andrés Arízaga Proaño
Tema del TT:	Modelo de gestión para mantenimiento y mejoras de software como servicio.
Articulación con la línea de investigación institucional:	Tecnologías de la Información y Comunicación
Sublínea de investigación institucional:	Simulación, desarrollo y automatización de procesos industriales, empresariales y de la sociedad.
Fecha de presentación del informe final:	29 de Agosto de 2014
Profesor - Tutor	Ing. Wilmer Valle

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del presente Trabajo de Titulación, certifico:

Que el Trabajo de Titulación “MODELO DE GESTIÓN PARA MANTENIMIENTO Y MEJORAS DE SOFTWARE COMO SERVICIO”, presentado por el Sr. Fernando Andrés Arízaga Proaño, estudiante de la Carrera de Ingeniería en Sistemas Informáticos, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado que se designe, para su correspondiente estudio y calificación.

Quito D.M. Octubre de 2014

TUTOR

_____

Ing. Wilmer Valle

UNIVERSIDAD TECNOLÓGICA ISRAEL

AUTORÍA DEL TRABAJO DE TITULACIÓN

En mi calidad de autor del Trabajo de Titulación “MODELO DE GESTIÓN PARA MANTENIMIENTO Y MEJORAS DE SOFTWARE COMO SERVICIO”, requisito previo a la obtención del Grado de Ingeniería en Sistemas Informáticos; declaro que el contenido del presente es absolutamente original y auténtico, siendo de exclusiva responsabilidad legal y académica del autor.

Quito D.M. Octubre de 2014

_____

Fernando Andrés Arízaga Proaño

CC: 1714820220

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TRIBUNAL DE GRADO

Los miembros del Tribunal de Grado, aprueban el Trabajo de Titulación "MODELO DE GESTIÓN PARA MANTENIMIENTO Y MEJORAS DE SOFTWARE COMO SERVICIO", para la graduación de acuerdo con las disposiciones reglamentarias emitidas por la Universidad Tecnológica Israel para títulos de pregrado.

Quito D.M. Octubre de 2014

Para constancia firman:

TRIBUNAL DE GRADO

_____

PRESIDENTE

_____ _____

MIEMBRO 1 MIEMBRO 2

AGRADECIMIENTO

Y

DEDICATORIA

Agradezco a Dios por permitirme la vida a través de mis padres, Fernando y Patricia, eterno ejemplo de amor, respeto, honestidad, esfuerzo y perseverancia.

A mis hermanos, Francisco y Mónica, agradezco por su entrega diaria incondicional, para nuestro desarrollo familiar y profesional.

Dedico este nuevo logro de mi vida profesional, a quienes hacen posible que siempre tenga una nueva meta que superar, a mi familia.

RESUMEN

La calidad actualmente es una de las razones para el consumo de un servicio, o para elegir uno, en caso de existir similares en su naturaleza. Un servicio debe mantener su garantía y utilidad bajo el acuerdo de contratación.

Los servicios de tecnología, al igual que otros servicios, tienen un alto nivel de competencia con las empresas proveedoras, por ejemplo: servicios de internet, alojamiento web, correo electrónico, capacitación, repositorio de archivos, noticias, videos en línea, documentos, revistas, pagos en línea, banca en línea, entre otros. Como se evidencia en los ejemplos anteriores, muchos de estos servicios en otras épocas se realizaban manualmente y no se contaba con la tecnología para su automatización. Hoy en día, el acceso a muchos de estos servicios se realiza por medio de la tecnología.

Partiendo de los conceptos de servicios y tecnología, en el presente escrito se desarrollará la importancia de la gestión de la calidad en los servicios tecnológicos, y cómo mejorar la calidad mediante la medición de indicadores de gestión. La idea principal del desarrollo de este trabajo, es la socialización del concepto de calidad y la gestión de procesos para cambios y mejoras del software como servicio, considerando que aunque un servicio tecnológicamente sea único, si su calidad es baja, sus clientes no lo continuarán consumiendo.

PALABRAS CLAVE: Calidad, gestión de calidad, servicios, software, tecnología, información, tecnologías de la información.

ABSTRACT

Quality is actually one of the reasons for consuming a service, or to choose one, if there is another similar. A service must maintain the warranty and utility contracting under the agreement.

Technology services, like other services, have a high level of competition with suppliers, for example, Internet services, web hosting, email, training, repository files, news, online videos, documents, magazines, online payments, online banking, etc. As evidenced by the above examples, many of these services in the past were done manually and did not have the technology for automation. Today, the access to many of these services is done through the technology.

Based on the concepts of services and technology in this paper will develop the importance of quality management in technological services, and how to improve quality by measuring indicators. The main idea of the development of this article, is the socialization of the concept of quality in IT services.

KEY WORDS: Quality, quality management, services, software, technology, information, information technology.

CONTENIDO

Índice de Gráficos

0. INTRODUCCIÓN

Un servicio, independientemente de su naturaleza o finalidad, es un medio para cubrir una necesidad específica de un cliente o usuario, quien no asume los costos o riesgos asociados a mantener la disponibilidad, garantía y utilidad ofrecida por un proveedor[1].

El cliente o usuario es quien recibe los beneficios que ofrece una empresa proveedora de servicios. De la satisfacción del cliente, es como la empresa podrá medir su éxito o su fracaso, ya que será proporcional al retorno de su inversión.

En los últimos años, con el crecimiento de la tendencia cloud[2], las estrategias de las empresas se han orientado a ofrecer sus productos de infraestructura, plataformas de desarrollo y software, de una manera no tradicional; es decir ya no como productos sino como servicios.

La calidad de un servicio, se verá afectada por factores internos y externos a la empresa. Los factores externos deberán ser controlados y analizados en un plan de riesgo y contingencia. Sin embargo, los factores internos deben ordenarse y planificarse para obtener los más altos resultados. Para ello, con base en la gestión por procesos, se puede medir el nivel de productividad de cada etapa de creación, modificación y puesta en producción de un servicio.

El software defectuoso genera una gran cantidad de pérdidas en las organizaciones, por contener código erróneo y por no cumplir con los requerimientos de cliente; causando además de los costos por reparaciones en el mismo, el disgusto de los clientes, y la pérdida de los mismos[3].

El levantamiento óptimo de requerimientos, el alto desempeño en el desarrollo, la correcta validación del servicio, la efectiva aplicación del despliegue; y, sobretodo la ordenada y planificada gestión del mantenimiento y la producción del servicio, será el éxito para alcanzar y mantener la calidad del software[4].

[1] (ITPreneurs, 2013, pág. 15)
[2] Cloud Computing – Computación en la nube. 1.1.4.
[3] (Pressman, 2010, pág. 365)
[4] (PinkElephant, 2013, pág. 16)

0.1. Problema Investigado

Se determinan las causas identificadas de la mala calidad del software como servicio en una organización, las cuales conllevan a la insatisfacción de los clientes:

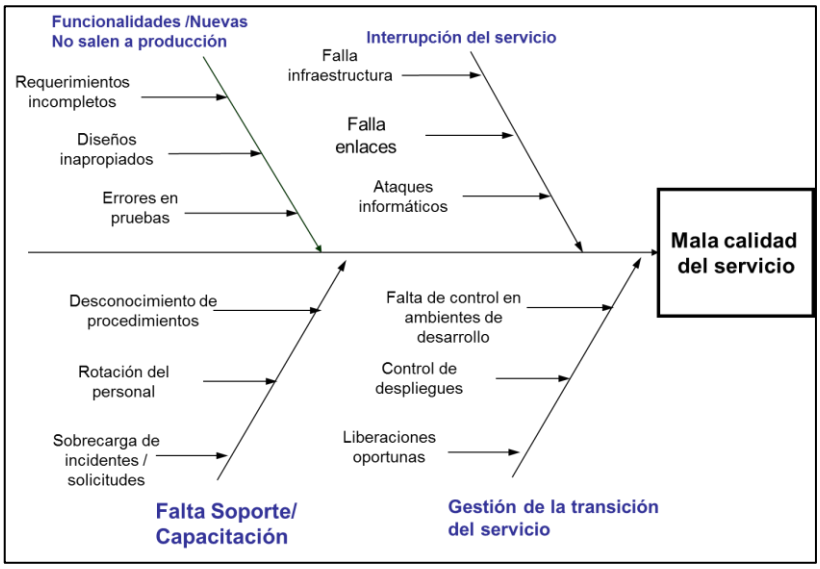

Gráfico 1: Diagrama de Ishikawa. Causa - Efecto

En la gestión de la transición de los servicios, al no existir la definición de un modelo de gestión, en este caso, para la gestión de cambios y de versiones; el servicio compromete su disponibilidad, garantía y utilidad. Los tres aspectos indicados se relacionan directamente con la calidad del servicio.

El problema de la investigación centra en que el actual proceso de producción del software está provocando mala calidad en el servicio

0.2. Definición del Alcance

El desarrollo de un modelo de gestión para mejorar la calidad de software como servicio, abarca una gran cantidad de componentes, tanto en los aspectos de definición de procesos de gestión, así como en las actividades y procedimientos que contienen estos procesos. El modelo propuesto tendrá la capacidad de gestionar los procesos de cambios y liberaciones que se realizan para las correcciones, el mantenimiento y las mejoras del servicio.

El trabajo propuesto involucra la definición de flujos y manuales de los procesos mencionados, así como la descripción de las actividades y roles que apoyan a la gestión.

Adicionalmente, se integran a los procesos, los manuales y procedimientos para poner en marcha las actividades de control de cambios y versiones a

nivel operativo. Para ello se desarrollarán estos documentos en base a las actividades definidas y el uso de la herramienta Subversion – SVN, para control de versiones.

De igual forma, los lineamientos para criterios de evaluación, gestión de ambientes controlados para pruebas, políticas de calidad y despliegues en producción, serán alineados según el marco de referencia de ITIL, como recopilación de las mejores prácticas; y, de acuerdo a la definición del servicio y su disponibilidad que se establece en la estrategia y diseño del mismo.

0.3. Objetivo general:

Desarrollar un modelo de gestión de los procesos de cambios y versiones, para las mejoras y mantenimiento del software como servicio.

0.4. Objetivos específicos:

- Realizar la investigación y recopilación bibliográfica de las mejoras prácticas para gestión de calidad y servicios de las tecnologías de la información.
- Definir los modelos de proceso para gestión de cambios y gestión de versiones, incluyendo los indicadores para medir el rendimiento de la gestión, e incluir el manual de los procesos y los instructivos de las actividades.
- Establecer las políticas para: la gestión de los ambientes controlados para validación de los cambios, calidad y parámetros de Q&A en la validación de los cambios, despliegue y control de versiones en producción, de acuerdo a las definiciones de disponibilidad del servicio.
- Validar la solución propuesta en el servicio web de IFSA, sitio web informativo y de servicios para estudiantes, desarrollado para el Instituto de Estudios en el Extranjero – IFSA.mx.

0.5. Hipótesis

Implementar la gestión por procesos de cambios y versiones en la producción del software, favorece a mantener y mejorar la calidad del servicio.

1. CAPÍTULO I

1.1. Marco Teórico

1.1.1. Servicio

Medio por el cual los clientes pueden obtener resultados sin que éstos incurran en costos y riesgos específicos[5]. Las características de un Servicio de TI incluyen, entre otros elementos:

- Aplicaciones de software que necesita para funcionar
- Ambiente de operación
- Datos
- Procesos de Gestión del Servicio (Gestión de Cambios, Liberaciones)
- Acuerdos de Nivel de Servicio (SLA's)

Un servicio de tecnología está conformado por personas, procesos e información. El proveedor de servicios de tecnología puede ser interno o externo, así como un beneficiario con las mismas características. Un proveedor es interno cuando dentro de la organización existe el servicio y se lo desarrolla internamente, así como sus beneficios y productos. Se considera un proveedor externo cuando la organización consume o se beneficia de un servicio prestado por un tercero; con el cual tiene un acuerdo o contrato preestablecido, y por el cual retribuye al proveedor con un valor, o costo, que consta en el acuerdo. El valor que recibe la organización, usuario o cliente, se define como servicio.

Desde la perspectiva de una organización, ésta puede consumir sus propios servicios, en este caso se enmarca como un proveedor interno (su propio proveedor de servicios); sin embargo, si el giro del negocio es la producción de un servicio, en este caso será un proveedor de servicios para terceros.

[5] (ITPreneurs, 2013, pág. 15)

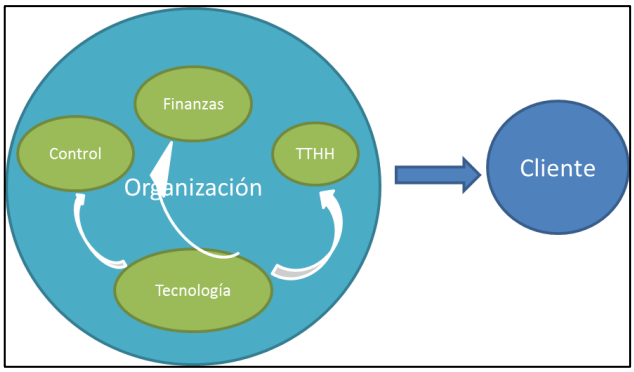

Gráfico 2: Servicio - Relación Organización y Clientes

En la ilustración se observa al área de tecnología como el área que presta los servicios internos a la organización, entre otras, a las áreas de finanzas, control y talento humano. Sin embargo, desde la perspectiva del cliente externo, la organización es un proveedor externo. Aunque el servicio se desarrolle en una o más áreas de la organización, el cliente externo no tendrá el contacto directo ni la percepción del servicio directamente de las áreas sino de la organización, que para efectos de la operación del servicio definirá previamente las áreas de contacto para el cliente y entregar el valor acordado.

El servicio también se divide en principal y complementario. El primero es básicamente la entrega del valor que los clientes necesitan y están de acuerdo a pagar, se proporciona los resultados mínimos del acuerdo entre la organización y el cliente, el cual debe cumplir con la garantía y utilidad contratada. Los servicios complementarios se utilizan para la captación de clientes, sin embargo son un extra al servicio principal, los cuales pueden ser temporales, ya que no son imprescindibles. Estos últimos pueden ser considerados como un factor de entusiasmo dentro de la clasificación de requerimientos de Kano[6].

1.1.2. Software

Programa o conjunto de programas de cómputo y documentación asociada. Los productos de software se desarrollan para un cliente en particular o para un mercado en general[7].

[6] (Kano, 1984, págs. 39-48)
[7] (Sommerville, 2011, pág. 6)

Un sistema de información es un conjunto relacionado de personas, datos, procesos y tecnología de la información que interactúan para recopilar, procesar, guardar y proporcionar como salida la información necesaria para brindar soporte a una organización[8].

La interacción necesaria para la creación de software, se describiría en grandes rasgos con las siguientes etapas:

- Especificación del Software, donde se realiza el levantamiento de los requerimientos, una de las principales etapas. Del correcto levantamiento de requerimientos dependerá la calidad del software.
- Desarrollo, donde se diseña y programa las funciones requeridas.
- Validación, donde se verifica que la funcionalidad sea la solicitada por los clientes o usuarios.
- Evolución, etapa postproducción en la cual se evidencia los requerimientos cambiantes del software, solicitados por el cliente o inducidos por el medio.

El software, por su naturaleza de distribución puede ser genérico o personalizado. El producto genérico es un sistema independiente que se desarrolla por parte de una organización, la cual busca vender en un mercado abierto, tomando en cuenta factores similares de requerimientos en un mismo ambiente o contexto similares. Estos casos tendrán éxito en clientes que comparten características similares.

Los productos personalizados son sistemas que están desarrollados para un cliente particular. La organización realiza el desarrollo del producto únicamente con el objetivo de cumplir las necesidades del cliente requirente. En ocasiones, las organizaciones que desarrollan software, tienen un producto base que podrá ser personalizado de acuerdo a las necesidades del cliente. No toda la funcionalidad será modificada, pero se mantiene un producto base parametrizable el cual facilita la reutilización de recursos y la implementación de un sistema prácticamente a la medida del cliente, con la base de un producto genérico. La reutilización de recursos permite abarcar la totalidad del proyecto en menor tiempo, manteniendo el alcance y costo del mismo.

[8] (Jeffrey L. Whitten, Lonnie D. Bentley, 2008, pág. 6)

1.1.3. Software Como Servicio - SaaS

En la definición del concepto de cloud computing, existen tres tipos de servicios definidos por sus alcances, nivel de abstracción y utilidad para el usuario. El primer nivel es ofrecer la infraestructura como servicio – Infraestructure as a Service (IaaS), en la cual el cliente tiene acceso a la configuración de los servidores o equipos para virtualización, y utilizará los recursos arrendados, de acuerdo a su necesidad. En este caso el cliente alquila por tiempos limitados o por capacidades, siendo el objeto unidades de procesamiento virtuales, memoria de almacenamiento y memoria RAM, como principales elementos. También puede solicitar, en caso de que el proveedor disponga, de seguridad perimetral y otros servicios similares.

El siguiente nivel de servicio se refiere a la Plataforma - Platform as a Service (PaaS). En este caso, el cliente solicita servidores con ciertas características, además de especificar hardware, también puede solicitar software para el desarrollo de aplicaciones o servicios intermedios.

El Software como Servicio - Software as a Service (SaaS), es un servicio[9] que adquiere el cliente, luego de llegar a un acuerdo con el proveedor, o al aceptar las condiciones del servicio prestado, es decir, que no adquiere el software como su propiedad, sino que lo arrienda o utiliza bajo los parámetros de un servicio, no de un bien no tangible. Las transacciones, el tiempo, o las prestaciones que tenga el software, determinarán los costos para el cliente.

1.1.4. Cloud Computing

Computación en la nube. El concepto de nube[10] inicia desde que los sistemas de información distribuidos se han aplicado de manera masiva. El concepto de virtualización ha dado paso a que los servidores físicos contengan varios servidores lógicos o virtuales en su “interior”. Este último concepto se refiere al uso plataformas que permiten generar varios ambientes lógicos, con características determinadas de procesamiento, almacenamiento y memoria, los mismos que se comportan como

[9] (Sommerville, 2011, pág. 13)
[10] (Sommerville, 2011, pág. 13)

servidores físicos, con ciertas limitaciones que dependiendo de la finalidad y del objetivo son altamente manejables. Estos servidores lógicos comparten las mismas características físicas del equipo del cual obtienen los recursos, sin embargo, es la acción precisa del profesional quien configura los servidores, que permite la optimización de los recursos y la mayor efectividad del equipo físico y los servicios que ofrece.

La virtualización, ha permitido reducir la cantidad de equipos físicos necesarios para la producción de un servicio, y aumentar la oferta de servicios con la optimización de los recursos. En consecuencia, muchos productos que se vendían como bienes, hoy en día sin servicios de tecnología que se encuentran al alcance de un óptimo ancho de banda para consumirlos. Entre otros, se puede tomar el ejemplo de los discos de almacenamiento externo, los cuales, aunque no han detenido su producción, han sido reemplazados por los servicios de almacenamiento en línea; que permiten el acceso a la información, sin necesidad de llevar el dispositivo físico, pero se requiere una conexión a la red.

De similar manera existen otros servicios de cloud, empresariales en su mayoría, a gran escala, los cuales comparten los conceptos de virtualización, y ofrecen plataformas a medida de las necesidades. Estas plataformas a medida, se convierten en un software que consume un usuario o una organización, sin tener que asumir los costos o los riesgos por mantenerlo en funcionamiento y disponible. Otro servicio muy conocido y muy común es el correo electrónico. Muchos proveedores centralizan el correo electrónico en una nube, privada o pública, para el uso desde distintos lugares y dispositivos, los cuales requieren únicamente conexión a una red.

1.1.5. Calidad

La garantía de que un producto o servicio cumpla con lo esperado o contratado se conoce como calidad. Un bien adquirido es de calidad cuando tiene las características que el cliente buscaba y perdura en el tiempo que se indicó por parte del proveedor.

Los principios de calidad son relativos, respecto al espacio de tiempo, época o ambiente en el que se desarrolla un análisis. Por ejemplo, para un operador de una organización que produce un bien o servicio, la calidad es el hecho de cumplir con sus obligaciones totalmente, con los objetivos alcanzados en el tiempo acordado. Sin embargo, en la constante búsqueda por la mejora del producto, aparecen los conceptos de productividad, que no es sino el producto de mejorar la calidad de un bien o servicio. Al mejorar la calidad se transfieren las horas-hombre y las horas-máquina malgastadas a la fabricación de un producto bueno y dar un servicio mejor[11].

La calidad de un producto o servicio tiene varias escalas, que serán variables desde la perspectiva del productor y del consumidor. La perspectiva de la calidad también involucra que debemos conocer más a fondo el requerimiento del cliente, tanto los actuales como los futuros, ya que dada la competencia, habrá necesidades que el cliente aún no las demanda, pero la organización se puede adelantar a producir[12]. Garantizar la utilidad y cumplir con la calidad prometida, no basta para la fidelidad de un cliente, se debe también llenar las nuevas expectativas e incluir nuevos servicios; es decir, que para obtener brindar mayor calidad, se debe cumplir con el producto o servicio esperado y además incluir y cubrir otros requerimientos que el cliente asume que son parte del servicio adquirido; los cuales son realmente un esfuerzo agregado en el desarrollo del producto o servicio.

1.2. Marco Conceptual

1.2.1. Producción de Software

1.2.1.1. Metodologías de desarrollo

La ingeniería de software, a diferencia de la ingeniería de sistemas o las ciencias de la computación, se enfoca exclusivamente en la producción de productos o servicios que satisfacen una necesidad o un requerimiento determinado. La forma o técnica que se utilice para

[11] (Deming, 1989, pág. 2)
[12] (Deming, 1989, pág. 133)

este desarrollo será la estrategia fundamental para lograr los objetivos de un proyecto de producción de software.

Las actividades generalizadas de una estructura práctica para el desarrollo de software son: entender el problema, planear la solución, ejecutar el plan y examinar la exactitud del resultado[13].

Los modelos de desarrollo utilizados generalmente para la producción de software pueden variar de acuerdo a la visión del líder de proyecto, decisión que se adopta por factores propios del proyecto, por solicitud del cliente, o por la experiencia que tiene el equipo de trabajo y su líder.

El modelo en cascada, es una serie consecutiva de actividades, de las cuales se pasa al siguiente nivel, solamente luego de culminar la etapa previa, es decir, que la secuencia de construcción será: levantamiento de requerimientos, planeación, modelado, construcción y despliegue del sistema. La secuencia de actividades del flujo descrito dará como resultado el producto esperado. Una variante de este modelo es el modelo en "V", en el cual, cada una de las actividades de la secuencia corresponde a una validación de pruebas de distintos niveles para asegurar la calidad del producto esperado. De esta forma, desde el nivel más bajo del modelo, el desarrollador ejecuta las pruebas unitarias, hasta el nivel más alto que son las pruebas de aceptación por parte del usuario, pasando por las pruebas de integración y sistema realizadas por las áreas de aseguramiento de la calidad.

Los modelos de procesos incrementales son utilizados cuando el usuario exige la implementación inmediata de funcionalidades básicas del sistema, agregando con el tiempo otros módulos o características requeridas posteriormente. Como se indica, la funcionalidad se incrementa con este modelo, muchas veces iniciado como un prototipo. Al utilizar este modelo, no se tiene claro el alcance del proyecto totalmente, sin embargo, se requiere que luego de las

[13] (Pressman, 2010, pág. 15)

primeras liberaciones se determine los límites del sistema y contexto, de lo contrario no se podrá determinar el final del proyecto.

Los modelos evolutivos se basan en mecanismos iterativos. Cada iteración o hito, tiene una funcionalidad específica de todos los módulos previstos en el alcance inicial, sin embargo su aplicación es baja. Cada prototipo incrementa la funcionalidad del sistema hasta completar con los requerimientos del cliente. También forman parte de una secuencia, comparando con un modelo de cascada, sin embargo, luego del despliegue inicia nuevamente en los requerimientos y cada una de las etapas hasta una nueva entrega. De similar forma a los incrementales, se debe definir el alcance en las primeras iteraciones, de lo contrario será una tarea complicada finalizar el proyecto.

Con base en los modelos anteriores, se han desarrollado e implementado varios modelos y procesos de desarrollo de software que han tenido éxito de acuerdo a la naturaleza del producto y de las estrategias y políticas de las organizaciones que los desarrollaron.

1.2.1.2. Pruebas y Validación

La validación de los requerimientos, por parte de las áreas de tecnología, respecto a la solicitud de cambios o mantenimiento para los servicios de tecnología, es indispensable para la analizar la factibilidad del desarrollo e implementación del cambio.

Cada solicitud se convierte en una serie de actividades que conllevan al empleo de recursos, tiempo y costos; que no se recuperarán si la factibilidad del cambio no ha sido validada desde distintas perspectivas. Estas evaluaciones se orientan en la visión legal, operativa, cultura o política, técnica y económica[14]. La validación del requerimiento será un criterio muy importante en la decisión de los comités de cambio para la toma de decisiones.

[14] (Kenneth E. Kendall, Julie E. Kendall, Keneth C. Laudon, Jane P. Laudon, 2010, págs. 437-441, 793, 856)

Por otra parte, las pruebas que se realicen a los productos obtenidos, tanto funcionales, de calidad o restrictivas, serán el paso indispensable para el despliegue de los cambios en producción. Las pruebas que se realizan deberán mantener una concordancia con las políticas pre-establecidas en la organización.

2. CAPÍTULO II

2.1. Metodología de Investigación

Los elementos que conforman las metodologías de investigación se enlistan de distintas maneras y contenidos, dependiendo del enfoque de la metodología que se utiliza en el proyecto o desarrollo investigativo.

El enfoque de la *metodología cualitativa* lleva a cabo un proceso en el cual se determina de manera iterativa, repetitiva y recurrente, ciertos parámetros determinados para posterior evaluación o análisis estadístico o de medición.

Los puntos principales para el planteamiento de un objetivo cualitativo son: objetos de investigación, preguntas de investigación, justificación de la investigación, viabilidad, evaluación de las deficiencias y la definición inicial del ambiente o contexto como línea base[15].

El planteamiento base del proyecto es obtener y mantener la excelencia y la mayor calidad en la prestación de servicios, específicamente en el software como servicio. La disponibilidad, garantía, utilidad y cumplimiento será la medida del crecimiento o mantenimiento de las políticas de calidad.

La iteración o repetición de procesos definidos y sus respectivas actividades generan resultados que deben ser obtenidos bajo los mismos criterios de evaluación. La calidad del servicio o del proceso ejecutado se determina por los indicadores definidos y la comparación de los mismos en los periodos de tiempo determinados.

El ámbito y contexto en el cual se desenvuelve el escenario del software como servicio, puede parecer indeterminado, conociendo de antemano que este servicio es un software publicado para acceso por medio del internet en una página web; sin embargo, el ámbito del servicio determina al sistema operado por un usuario y otros servicios determinados previamente en la arquitectura. Por lo tanto, aunque la cantidad de usuarios y la demanda del servicio sea variable en cada periodo de tiempo, el contexto del sistema no

[15] (Hernández S. Roberto, Fernández-Collado Carlos, Baptista L. Pilar, 2008, págs. 523-529)

cambiará a menos que sea una decisión del giro del negocio o bajo el criterio de optimización y mejora continua.

A diferencia de la metodología cuantitativa, la metodología utilizada requiere menor escala de investigación del marco teórico y conceptual debido a que los resultados de la ejecución o desarrollo del proyecto tendrán como resultado los insumos o entradas para investigación cualitativa. Adicionalmente, como parte de la metodología utilizada para el presente desarrollo, se ha empleado a las definiciones de las mejores prácticas y recomendaciones de la *Biblioteca de Infraestructura de las Tecnologías de Información – ITIL*.

El conocimiento generado y basado en la experiencia y las mejores prácticas recopiladas en ITIL, forman parte de un conjunto de conceptos aplicados a las tecnología de la información, los cuales son aplicados por mayor número de organizaciones de ésta naturaleza desde las primeras versiones de esta biblioteca en los años 90, cuando empezó a ser adoptada como una guía de servicios tecnológicos.

El conocimiento generado a través de los años en las mejoras de las versiones de ITIL, ha enfatizado la aplicación de sus procesos del ciclo de vida de los servicios, en vista de los resultados de calidad y mejora continua adoptados en las últimas versiones. El fundamento de una teoría no es un conjunto de hechos unidos entre sí, sino un conjunto ordenados de principios, hipótesis y conclusiones que se conjugan mediante un método lógico y coherente que le da racionalidad y validez. Este tipo de sistemas son el producto del descubrimiento de nuevos métodos y del reemplazo de teorías que se van interrelacionando por su propia naturaleza.

“El conocimiento científico con respecto a su generalidad, es fundamentar sus postulados encuadrándolos de modo particular dentro de una cantidad ilimitada de casos específicos” [16].

[16] (Muñoz, 1998, págs. 177-178)

2.2. Metodología ITIL

La metodología ITIL se basa en el modelo del ciclo de vida del servicio. Esta metodología está enfocada para los servicios de tecnología. Las organizaciones que producen y prestan servicios tecnológicos, han adoptado una serie de estrategias y métodos para aprovechar sus recursos y ser más productivos. El ciclo de vida del servicio, no es un estándar definido, sin embargo, las organizaciones lo pueden adoptar con la implementación y adecuación de los procesos, respecto a su realidad y objetivos empresariales.

El ciclo de vida del servicio, que actualmente lo define ITIL en su versión 3 – 2011, se puede resumir en la siguiente ilustración:

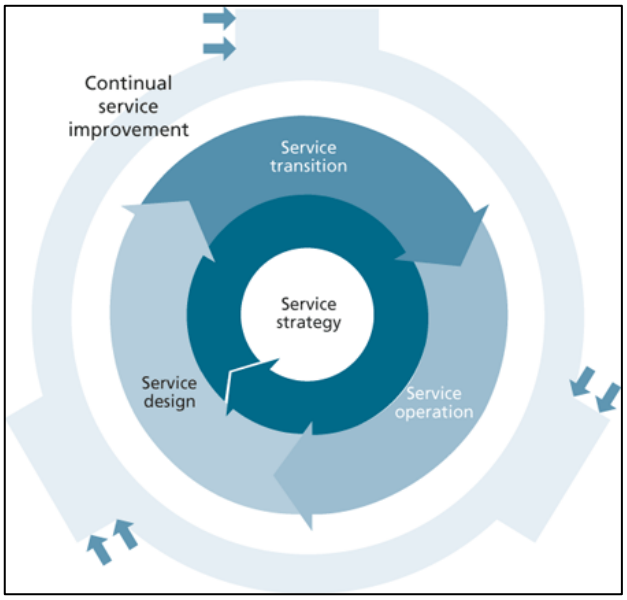

Gráfico 3: Ciclo de Vida del Servicio[17]

En la ilustración se puede observar que el objetivo de todo del ciclo de vida del servicio, inclusive como parte del mismo, es la mejora continua. En los conceptos de calidad revisados en el presente documento, se indica que para obtener calidad, se requiere una línea base, una medición de lo que se tiene actualmente, o una apreciación de la realidad en un determinado momento.

[17] (ITPreneurs, 2013, pág. 3)

Los lineamientos de la mejora continua describen una serie de procesos en las distintas etapas del ciclo de vida del servicio. Las etapas definidas son: estrategia, diseño, transición, operación y mejora continua. El cumplimiento de las etapas definidas es consecutivo, a partir de la estrategia del negocio, ya que son las áreas del negocio las que dictan las directrices para la implementación de nuevos servicios o la modificación de algunos existentes.

Cada proceso determinado por un flujo de actividades y tareas, debe incluir en su manual de ejecución a los factores críticos del éxito de la ejecución de los procesos. Estos factores se definen como indicadores cuantificados con el objetivo de analizar en periodos determinados de tiempo. Esta medida mostrará en relación a la línea base, o periodos anteriores, estados cualitativos de los procesos del ciclo de vida del servicio. Es decir, que los índices obtenidos podrán ser comparados estadísticamente para medir la calidad de los productos parciales de cada proceso y de similar forma para los productos o servicios finales.

Dentro de los procesos de la etapa de transición del servicio, se encuentra la gestión de cambios y gestión de versiones.

2.2.1. Gestión de Cambios

El propósito de la gestión de cambios es controlar el ciclo de vida de todos los cambios, permitiendo que se realicen cambios benéficos con mínimas interrupciones a los servicios de TI[18].

El proceso inicia con los requerimientos del negocio para realizar alguna variante en uno de sus servicios. La capacidad de ejecución de cambios de una organización debe ser tan dinámica como para considerar inicialmente en sus procesos de transición del servicio a la gestión del cambio, ya que mientras más maduro se encuentre el estado de sus procesos podrán mantener la vigencia de sus servicios. La estrategia de la organización definirá nuevos servicios o la modificación de algunos para mantenerse en el mercado competitivo o para atraer la fidelidad de sus clientes.

[18] (PinkElephant, 2013, pág. 105)

Entre otros objetivos y valores del proceso de gestión de cambios se puede enumerar:

- Proteger al giro del negocio y a otros servicios, mientras se despliegan los cambios en producción.
- Cubrir los requerimientos de los clientes, según lo acordado, con la optimización de recursos.
- Dar cumplimiento a los lineamientos o requerimientos externos de aspectos legales, contractuales, regulatorios o de cumplimiento.
- Reducir el número de cambios no autorizados.

2.2.2. Gestión de Versiones

El propósito de la gestión de versiones es planear la agenda y controlar la construcción, prueba y despliegue de liberaciones e implementar nuevas funcionalidades requeridas por el negocio, mientras se protege la integridad de los servicios existentes[19].

La gestión de versiones tiene una estrecha relación con la gestión de cambios ya que dentro de sus objetivos y valores, se espera optimizar la entrega de los servicios al menor costo y riesgo para el negocio. Se requiere una correcta planificación para no afectar a la disponibilidad del servicio que se cambia u otros servicios relacionados.

La planificación de despliegues de cambios demanda conocimiento del giro del negocio y una constante comunicación con las áreas interesadas y las áreas de implementación.

La gestión de versiones además de los planes y cronogramas de liberaciones, también deberá tomar las decisiones para el tipo de despliegue, ya que los servicios son distintos en las organizaciones. Sin embargo, para el caso del presente proyecto, se realizará casi en su totalidad por actualizaciones del software en el servidor de producción o con la implementación de parches en los equipos de red de seguridad perimetral.

[19] (PinkElephant, 2013, pág. 152)

2.3. Descripción del producto a desarrollar

El presente proyecto desarrolla un modelo de gestión por procesos con el objetivo de mejorar la calidad del software como servicio. Este modelo involucra una serie de procesos, actividades, procedimientos, políticas e instructivos; que serán la guía para determinar la gestión de la calidad en las mejoras y mantenimiento de un software existente.

El ámbito en el cual se desarrolla, es para un sistema web basado en un lenguaje de desarrollo sin compilación. Este sistema se publica a través del internet para los usuarios suscritos al servicio. El propósito de la gestión de cambios y versiones es mitigar todo tipo de causa para la indisponibilidad del sistema y reaccionar ante cualquier eventualidad. Además, para los mismos fines, realizar la gestión de mantenimiento y mejoras con el menor impacto en la disponibilidad del servicio.

Todos los documentos descritos en el presente proyecto, se han determinado de tal manera que se podrán utilizar como una guía para la aplicación de un sistema en un ámbito y contexto similares al caso de análisis.

Los procesos de gestión de cambios y versiones determinan en sus definiciones índices claves de desempeño – KPI (*Key Performance Indicator*), los cuales son la principal fuente de medición para el mejoramiento de la calidad. La línea base de estos índices podrá ser una referencia a los resultados anteriores del software existente, basado en hechos históricos o en registros hallados. La definición y medición de los índices será parte del éxito de la evaluación de la aplicación del modelo de gestión.

3. CAPITULO III

3.1. Análisis de Requerimientos

Propósito

Desarrollar un modelo de gestión para mejorar la calidad del software como servicio. El modelo de gestión se realiza en base a las mejoras y mantenimientos que se realizan en el servicio. No se refiere al desarrollo de software base.

El modelo de gestión incluye los flujos de procesos de cambios y liberaciones, el manual de procesos y descripción de actividades. Adicionalmente, se incorpora los procedimientos e instructivos para el control de versiones mediante Subversion – SVN.

Como complemento del modelo, se requiere establecer las políticas de despliegue de cambios, validaciones y pruebas, y gestión de ambientes controlados; de acuerdo a las estrategias del negocio y la disponibilidad o niveles de servicio.

Referencias

Esta información fue redactada en base a los siguientes documentos:

- Plan de Tesis: Modelo de gestión para mantenimiento y mejoras de software como servicio.
- Librería de Mejores Prácticas de TI. ITIL v3 2011
- Procesos. PMBOK. Análisis del problema. Mejorar la calidad del software.

Requerimientos Funcionales

En base al análisis del problema, con la meta de mejorar la calidad de software, se planifica establecer un modelo de gestión para mantenimiento y mejoras de software como servicio.

Del análisis realizado, se determinan varios requerimientos por parte de las áreas operativas y de gestión de TI, las cuales se priorizan a continuación.

Priorización de Requerimientos:

Los requerimientos serán clasificados de acuerdo a su Importancia según corresponda como:

- **C Crítico:** Que en caso de no estar implementado el sistema no podrá funcionar.
- **L Leve:** Que en caso de no estar implementado el sistema si podrá continuar funcionando.

Lista de Requerimientos Funcionales		
N°	**Descripción**	**Prioridad**
1.	Definir un modelo de proceso para gestión de cambios	C
2.	Definir un modelo de proceso para gestión de versiones	C
3.	Medir la calidad de los procesos operativos	L
4.	Medir los resultados de la gestión por procesos	L
5.	Medir la calidad del servicio	L
6.	Definición de procedimientos, instructivos y manuales para la gestión y operación del modelo	C

Requerimientos no Funcionales

Lista de Requerimientos No Funcionales		
N°	**Descripción**	**Prioridad**
1.	Establecer políticas para la gestión de ambientes controlados	C
2.	Establecer las políticas de calidad y validación (Pruebas – Q&A)	C
3.	Establecer las políticas para despliegue en Producción y procedimientos de gestión de versiones	C

3.2. Definición de Procesos

Un proceso se define como una serie de actividades relacionadas, que a recibir una entrada o acción externa, interactúan generando un resultado único y específico.

A continuación se presentan los flujos de los procesos diseñados para la gestión de cambios y versiones del mantenimiento y mejoras del software como servicio. Estos diseños se relacionan con otros procesos que intervienen en el desarrollo de un servicio, en este caso, para una aplicación o herramienta web.

Cada diseño de flujo de procesos está acompañado de su respectivo manual, que es la guía que determina la descripción de las actividades y el alcance de cada una de las etapas para obtener un producto determinado. Adicionalmente, se agregan a la descripción de las actividades, las políticas e instructivos para la ejecución de estas acciones o tareas.

En el Gráfico 4, de la página 21, se presenta el diagrama de flujo del proceso de gestión de cambios. En el Gráfico 5 de la página 22, se presenta el diagrama de flujo del proceso de versiones. El desarrollo de los flujos se realiza en base a las recomendaciones del ITIL v3, y del análisis de requerimientos. Entre los documentos de manuales e instructivos se desarrollan también los manuales de ambos procesos.

3.2.1. PROCESO DE GESTIÓN DE CAMBIOS

PRO-001-PIC-FAP

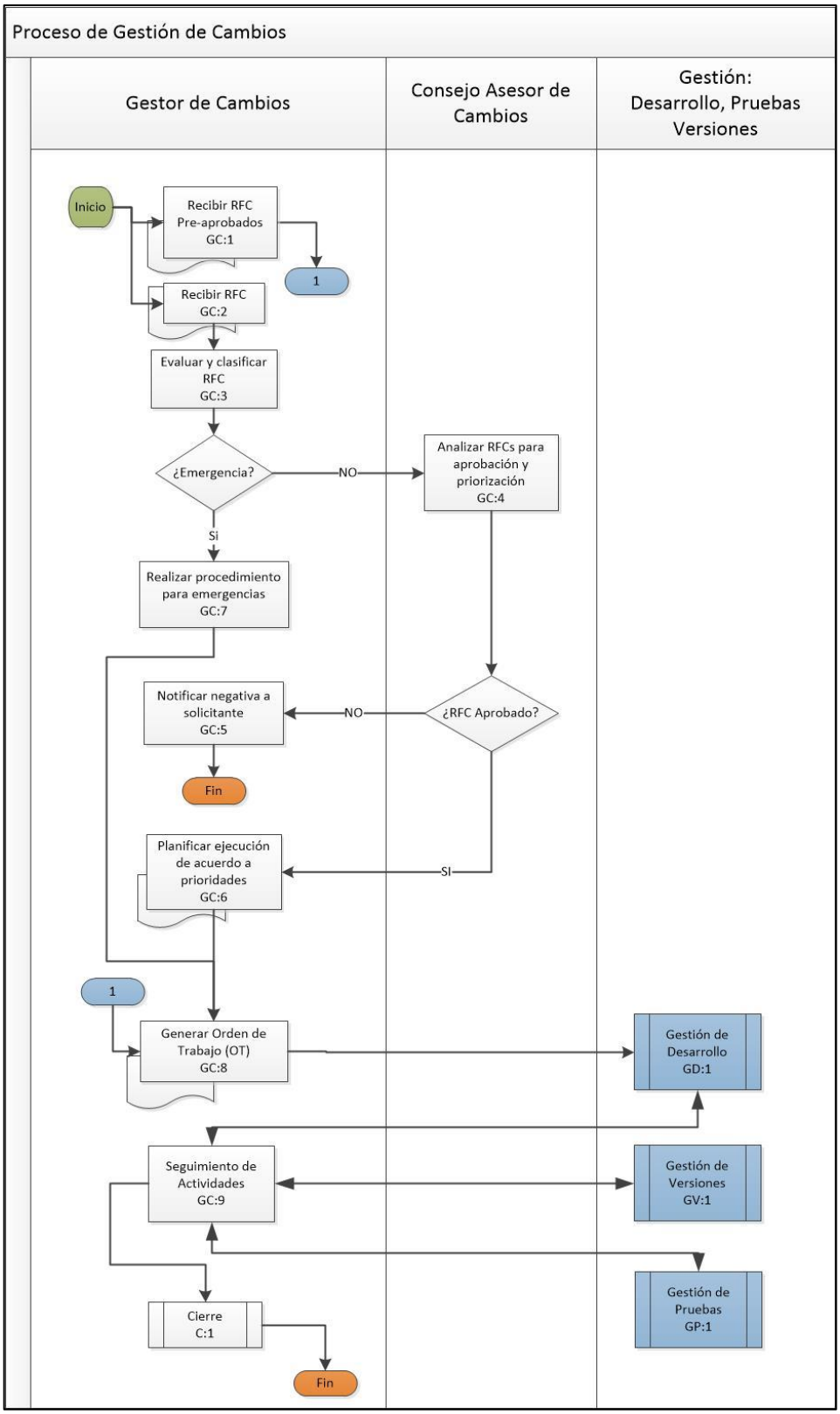

Gráfico 4: Diagrama de flujo – Proceso de gestión de cambios

3.2.2. PROCESO DE GESTIÓN DE VERSIONES

PRO-002-PIC-FAP

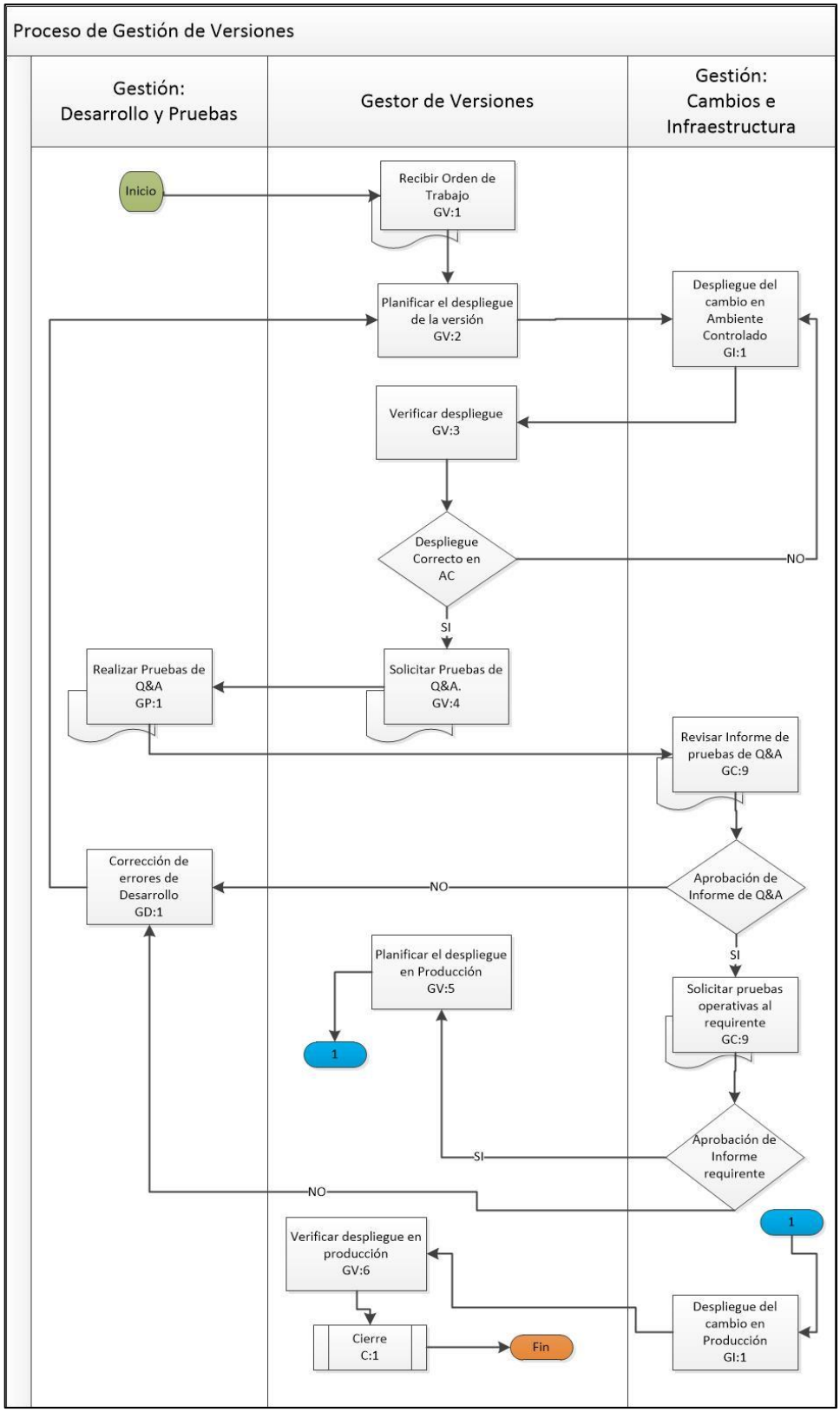

Gráfico 5: Diagrama de flujo – Proceso de gestión de versiones

3.3. Definición de Políticas

Las políticas para la gestión de los procesos, son los lineamientos y criterios necesarios para alcanzar los objetivos estratégicos del giro del negocio. Es decir, que para el cumplimiento de las metas fijadas por la organización, no solo basta con el cumplimiento de los flujos establecidos y la mejora de los factores claves de éxito, también son necesarios ciertos aspectos o criterios que determinan la confiabilidad, garantía y utilidad del servicio prestado.

3.3.1. POLITICAS DE GESTIÓN DE CAMBIOS Y AMBIENTES CONTROLADOS

POL-001-PIC-FAP

1. La gestión de cambios será administrada por el rol del gestor de cambios, quien será el encargado de organizar y exigir la aplicación del proceso establecido para mejoras y mantenimiento de los sistemas. Las áreas del negocio y las áreas de tecnología, deberán solicitar sus requerimientos por medio del formulario publicado, formulario de requerimiento de cambio - RFC-001.

2. Los requerimientos que hayan ingresado en el periodo de tiempo indicado en el proceso, serán analizados por el comité asesor de cambios con la finalidad de aprobar los requerimientos que se encuentren alineados con los objetivos estratégicos del negocio, y en caso de aprobar esta instancia también recibirá la correspondiente prioridad.

3. Los cambios se desarrollan de acuerdo al proceso establecido, PRO-001-PIC-FAP, y se desplegarán en cada uno de los ambientes destinados para los fines. Estas implementaciones no deberán afectar a la disponibilidad de los servicios en los ambientes controlados, ya que los usuarios podrán hacer uso de los ambientes dentro de los horarios establecidos en los catálogos de servicios de la organización.

4. Los ambientes controlados: desarrollo, pruebas, preproducción y capacitación; deben considerarse como servicios internos de la organización, por lo tanto deberán contar con la configuración necesaria en el catálogo de servicios, indicando su disponibilidad y comunicando a

los interesados las ventanas de mantenimiento programadas y los cambios realizados.

5. Los cambios de emergencia, luego de su comprobación de la categoría por el comité asesor de cambios de emergencia, serán desarrollados y desplegados de manera inmediata, siempre y cuando se considere este estado a la indisponibilidad total o crítica del servicio. La formalización del cambio deberá ser como la de un requerimiento de cambio, sin ingresar al comité asesor de cambios.

3.3.2. POLÍTICAS DE CALIDAD, VALIDACIÓN Y Q&A

POL-002-PIC-FAP

1. Los parámetros de calidad y aceptación de los desarrollos realizados, deberán ser incluidos desde el inicio del levantamiento de requerimientos. Las áreas de verificación, validación y aseguramiento de la calidad tomarán como referencia lo descrito en los requerimientos para dar paso a la aprobación o aceptación de un cambio o desarrollo.

2. El levantamiento de requerimientos debe ser documentado en el formato Análisis de requerimientos - AR-001, el cual contiene la siguiente estructura:

a. Objetivos
b. Alcance
c. Referencias
d. Descripción
e. Funciones del producto
f. Restricciones
g. Suposiciones y dependencias
h. Requisitos específicos
i. Otros requisitos
j. Parámetros de calidad
k. Criterios de evaluación

3. El responsable de aseguramiento de la calidad deberá revisar y aprobar el requerimiento realizado, verificando que se encuentre completo y guarde concordancia con lo solicitado.

4. Las pruebas funcionales deberán ser documentadas y detalladas en un informe de respaldo previo a la solicitud de despliegue en producción. Adicionalmente, la aprobación de despliegue deberá ser revisada y aceptada por las áreas del negocio interesadas, para proceder con la implementación.

5. Luego de la puesta en producción del cambio, se debe notificar a las áreas interesadas para recibir notificaciones de las pruebas postproducción. Cualquier información de eventualidades deberá ser notificada al área de incidentes para analizar las posibles asociaciones al cambio. En caso de no recibir notificaciones se tomará como exitoso el despliegue.

6. La trazabilidad de los elementos modificados deberá ser detallada por el área de producción, ya que a pesar de la modularidad del sistema, en ocasiones los cambios de configuración pueden afectar a uno o más cambios. En caso de detectar uno de estos eventos, se deberá coordinar para que los cambios se desplieguen de forma simultánea, o de manera organizada para no afectar la disponibilidad del servicio.

3.3.3. POLITICAS PARA DESPLIEGUE EN PRODUCCIÓN

POL-003-PIC-FAP

1. Para programar el despliegue en producción de un cambio, mejora o mantenimiento, se debe recibir la aprobación de las áreas del negocio interesadas y de la gerencia de tecnología.

2. Se solicitará al área de operaciones la última actividad programada en el servicio para el día establecido para el despliegue, y las actividades con menor impacto del cambio a liberar. El tiempo en el cual el sistema estará parcial o totalmente fuera de servicio, se establece como una “ventana de

mantenimiento”; la cual deberá ser informada y publicada para todos los interesados del cambio y los usuarios del servicio.

3. El área de monitoreo de servicios e infraestructura deberá manifestar cualquier alerta o evento fuera de los límites considerados como aceptables, mientras dure la ventana de mantenimiento y posteriormente hasta que se considere validadas las pruebas postproducción.

4. Los cambios que por razones demostradas y justificadas no hayan cumplido con lo esperado en el ambiente de producción, deberán ser retirados con la aprobación de la gerencia de tecnología o de las áreas del negocio. Este procedimiento se denominará una reversa del cambio en producción. Si un mismo cambio se ha reversado más de una vez consecutiva, deberá devolverse el requerimiento a las áreas del negocio para un nuevo análisis de requisitos.

3.4. Manuales e Instructivos

Los manuales e instructivos de los procesos y actividades, son documentos necesarios para el correcto desempeño y obtención de los productos esperados. La ejecución de actividades en un proceso puede ser desarrollada por distintas personas con un mismo rol, por lo tanto, se debe asegurar que las personas que realizan una tarea o una actividad asignada en un proceso entreguen los mismos resultados.

Entre otros motivos, la rotación o inclusión de personal, no debe afectar a la ejecución de los procesos establecidos. Los manuales de procesos y procedimientos deben ser presentados a todas las áreas involucradas en los flujos para el mantenimiento y mejoras del software. Todas las actividades se deben realizar de manera similar por las personas que desempeñan un mismo rol en la organización.

3.4.1. MANUAL DE PROCESO DE GESTIÓN DE CAMBIOS

MAN-001-PIC-FAP

1. **Objetivos y Alcance**

- Reducir el número de incidentes y problemas potencialmente asociados a todo cambio.
- Retornar a configuraciones estables de manera sencilla y rápida en caso de que el cambio tenga un impacto negativo en la estructura TI.
- Reducir el número de reversas o "back-outs" (retiro del cambio) necesarios.
- Desarrollar procedimientos de cambio estándar que permitan la rápida actualización de infraestructura y sistemas no críticos.

2. Glosario de Términos

Requerimientos: Todos aquellos eventos donde un usuario solicita asignaciones, Información o Cambio sobre el servicio brindado.

Urgencia: Medida de la criticidad para el negocio de un incidente o problema basada en el impacto sobre las necesidades de negocios del Cliente.

Impacto: Medida de la criticidad sobre el negocio de un incidente. A menudo, igual al grado con que un Incidente distorsiona el nivel de servicio acordado o esperado.

Prioridad: Es el resultado de la evaluación del impacto y la urgencia de los incidentes.

Requerimientos de Primer Nivel: Solicitudes que pueden ser atendidas, diagnosticadas y resueltas por los analistas de soporte.

Especialistas de Producto: Son expertos con conocimientos, recursos específicos en la infraestructura, aplicaciones, dominios tecnológicos para la solución de incidentes y atención de requerimientos.

Solicitud de Cambio (RFC): La Solicitud de Cambio (Request for Change, RFC) es un requerimiento formal de cambio en espera de ser implementada. Incluye detalles del cambio propuesto y estará documentada en un formulario específico.

Orden de Trabajo (OT): La Orden de Trabajo (OT) es un requerimiento formal de ejecución, desarrollo o implementación de un trabajo, donde se detalla claramente: alcance, actividades, cronograma, horario de ejecución y restricciones que deben ejecutar uno o varios especialistas de un área específica.

Consejo Asesor de Cambios (CAB): Es un órgano interno, presidido por el Gestor de Cambios, formado principalmente por representantes de las direcciones del negocio y los aprobadores de los RFC's. En algunos casos también puede incorporar: Consultores externos, Representantes de los colectivos de usuarios, Representantes de los principales proveedores de software y hardware.

Comité Asesor de Cambios de Emergencia (ECAB): Se encarga de autorizar, ajustar o rechazar una RFC de emergencia. Se recurre a este proceso cuando los procedimientos regulares de Gestión de Cambios no son aplicables, dada la acción inmediata requerida en casos de emergencia. Está formado por el Gestor de cambios, el Gerente de Tecnología, la Gerencia General y quien se considere pertinente según el caso.

Gestor de Cambios: Es el responsable del proceso del cambio y como tal debe ser el único responsable de todas las tareas asignadas a la Gestión de Cambios.

3. Roles del Proceso

3.1. Gestor de Cambios

- Registra los RFC's recibidas.
- Evalúa y clasifica RFC's
- Convoca reuniones del CAB para la aprobación de las RFC's; en caso de emergencia, convoca al ECAB.
- Viabiliza la realización de cambios beneficiosos con un mínimo de interrupciones en la prestación de servicios de TI.
- Genera Ordenes de Trabajo (OT)
- Planifica ejecución de cambios de acuerdo a prioridades.

3.2. Consejo Asesor de Cambios (CAB)

- Analiza, aprueba y establece prioridades de cambio
- Evalúa y acepta o rechaza las RFC’s recibidas.

3.3. Gestor de Desarrollo. Proceso de Desarrollo

3.4. Gestor de Pruebas. Proceso de Pruebas

3.5. Gestor de Versiones. PRO-002-PIC-FAP.

4. Descripción de Actividades

4.1. GC 01: Recibir RFC’s Pre-Aprobados

Ejecución: Se reciben los RFC’s pre-aprobados por parte del gestor de cambios. Estos documentos se encuentran en la categoría de un cambio estándar, el cual ha sido analizado por el CAB, y se considera que puede ser una función temporal. No requiere nueva aprobación, a menos que no se encuentre dentro del contexto o el gestor de cambio considere que no es conveniente.

Entradas: RFC Tipo pre-aprobado.

Salidas: Orden de trabajo. OT.

Condiciones: Se debe verificar que el RFC corresponde a un cambio pre-aprobado, de lo contrario se categoriza como una solicitud común, o se devuelve al interesado en caso de no proceder definitivamente.

4.2. GC 02: Recibir RFC’s

Ejecución: El gestor de cambios recibe los RFC’s y los registra con un código único del proyecto al que pertenece y una numeración consecutiva. Se requiere una revisión del requerimiento para considerar su validez. A su criterio, podrá consultar con el área de aseguramiento de la calidad para considerar un RFC completo.

Entradas: RFC, documento RFC-001.

Salidas: Orden de trabajo. OT.

Condiciones: Si no está completo el documento o no cumple con alguna condición de calidad, se devuelve al interesado para completar el mismo.

4.3. GC 03: Evaluar y Clasificar RFC's

Ejecución: Los RFC's serán clasificados por su naturaleza de mantenimiento o mejoras. Adicionalmente, previo a la reunión del CAB, se determina la criticidad e impacto para establecer la prioridad del mismo.

Entradas: RFC's ingresados.

Salidas: Informe de RFC's categorizados, impacto y criticidad.

Condiciones: Si son RFC's de emergencia se convocará de inmediato al ECAB (flujo GC 07), de lo contrario, se informará en las reuniones regulares del CAB (flujo GC 04).

4.4. GC 04: Analizar RFC's para aprobación y priorización

Ejecución: Los RFC's de acuerdo a la clasificación, se requiere el análisis de impacto, riesgo y urgencia, el mismo que determina la prioridad de ejecución. Adicionalmente, se analiza la disponibilidad de recursos y el tiempo aproximado de desarrollo y pruebas. Esta información es de gran utilidad previa a la reunión de CAB.

Entradas: RFC's evaluados y clasificados.

Salidas: Lista de RFC's aprobados y priorizados.

Condiciones: Si los RFC's son aprobados, se iniciará el proyecto para su desarrollo (flujo GC 06). Si los RFC's han sido rechazados, se notificará al área interesada los resultados (flujo GC 05).

4.5. GC 05: Notificar negativa al solicitante

Ejecución: Los RFC's que no han sido aprobados se devuelven a las áreas solicitantes, junto con la justificación del rechazo. El solicitante podrá ingresar nuevamente una solicitud de cambio, cuando se encuentre debidamente justificada.

Entradas: RFC's no aprobados.

Salidas: RFC rechazado, justificación.

Condiciones: Informar al solicitante el resultado del RFC analizado. Finaliza el flujo.

4.6. GC 06: Planificar ejecución de acuerdo a prioridades

Ejecución: La planificación del cronograma de cambios se realiza junto con las coordinaciones de las áreas de desarrollo, pruebas, producción e infraestructura. Se requiere conocer las actividades

actuales y la disponibilidad de los recursos para cada proyecto. En los cronogramas se debe incluir los recursos asignados para cada actividad para posterior seguimiento.

Entradas: RFC's aprobados con la información de prioridades, recursos y tiempo. Se detalla los aprobadores para despliegue en producción.

Salidas: Cronogramas, plan de proyectos.

4.7. GC 07: Realizar procedimiento para emergencias

Ejecución: Los procedimientos de emergencia se derivan de la aprobación del ECAB. Estos cambios también deben ser incluidos en los planes de proyectos, considerando el cambio de actividades y prioridades temporales. En los análisis de cronogramas para los RFC's del CAB, se debe considerar que las emergencias pueden ocurrir, de acuerdo al plan de riesgos de la organización; en donde, se considerará una holgura considerable para las emergencias. En caso de necesitar más recursos, por el factor principal del tiempo, se modificará los tiempos de las prioridades. Es necesaria una correcta gestión de proyectos y un gran conocimiento para la gestión de manera óptima.

Entradas: RFC's de emergencia. Aprobados por el ECAB.

Salidas: Solicitud de generación de orden de trabajo. OT.

Condiciones: Por la condición de la emergencia, se deberá modificar los cronogramas y asignación de recursos para que se actualice los tiempos de entrega de los cambios, ya que puede variar; los cambios de emergencia tienen prioridad alta, sobre los cambios planificados.

4.8. GC 08: Generar Orden de Trabajo. OT.

Ejecución: Generar la orden de trabajo, es el desglose de la solicitud de tareas a cada una de las áreas respectivas para el desarrollo e implementación de los cambios solicitados. La OT, debe generarse con un código único de proyecto y único dentro de la organización. En cada documento se detallan las tareas o actividades a realizar y los recursos asignados, así como también los tiempos y la interacción entre las áreas para lograr los objetivos. Las áreas de desarrollo, producción e infraestructura deberán utilizar las políticas definidas en los documentos POL-001-PIC-FAP, POL-002-PIC-FAP y POL-003-

PIC-FAP; así como los instructivos y manuales realizados para estos fines.

Entradas: RFC's aprobados, pre-aprobados y de emergencia.

Salidas: Órdenes de trabajado debidamente numeradas y codificadas para cada una de las áreas requeridas. OT.

Condiciones: Las OT, no deben contener la misma numeración ni codificación a otra orden de trabajo. Este documento[20] deberá darse a conocer a las coordinaciones de área y a los ejecutores de las tareas.

4.9. GC 09: Seguimiento de Actividades

Ejecución: El gestor de cambios, cumple con un rol central entre las áreas de servicios tecnológicos, ya que se integra con todas las actividades que se requieren ejecutar para lograr el éxito del cambio. El gestor de cambios ha elaborado previamente las órdenes de trabajo que se entregará a cada una de las áreas interesadas en el proceso del ciclo de vida del servicio. En esta etapa del flujo el gestor de cambio realiza el seguimiento de las actividades determinadas en el cronograma de ejecución. Desde los procesos de gestión de las otras áreas que también forman parte de este flujo, se emiten los resultados e informes de las tareas ejecutadas. Los resultados del proceso de desarrollo se pasarán al proceso de pruebas luego del análisis de resultados de la gestión de cambios y de la ejecución de las tareas de gestión de versiones.

Entradas: Órdenes de trabajo, productos de desarrollo, versiones e informes de pruebas.

Salidas: Reasignación de actividades en el flujo del cambio. Informe para cierre del cambio.

Condiciones: Los informes del área de pruebas deberán ser positivos para el despliegue del cambio en otros ambientes. De acuerdo al instructivo de la gestión de ambientes controlados, se requiere implementar los cambios en los ambientes de pruebas y pre-producción antes que en producción. El seguimiento del cambio permite analizar cada resultado en los ambientes previo el despliegue

[20] Se recomienda el uso de herramientas de software para la generación de las órdenes de trabajo y el respectivo seguimiento. Dependiendo de la organización y el tipo de proyecto se puede utilizar herramientas de gestión de proyectos, bug-trackers y de gestión de servicios de tecnología.

en producción. Se retornará al procedo antecesor en caso de detectar errores en los requerimientos funcionales, de acuerdo a las políticas establecidas de validación y pruebas.

4.10. Cierre.

El cierre del cambio se define como la culminación de las actividades de todos los procesos y tareas ejecutadas, hasta el informe definitivo de las pruebas post-producción.

5. Indicadores clave de desempeño. KPI.

Ítem	Nombre del indicador	Variables cuantitativas	Índice	Periodo	Formula	Unidad de medida	Objetivo	Metas	Tendencia
1	Eficacia de los cambios puestos en producción.	**NR**: Número de retiradas realizadas al mes. **TC**: Total de cambios al mes.	Índice = 0% no existen retiradas en el mes. Índice > 0% existen retiradas en el mes. Índice = 100% Todos los cambios fueron retirados.	Mensual y Anual	(NR/TC)*100	%	Disminuir el número de cambios retirados en la fase de puesta a producción debido a una mala coordinación, planificación, desarrollo del producto, pruebas o versionado.	Primer año: reducir el indicador al 30%. Segundo año: 20%.	El indicador debe tender siempre a 0%.
2	Calidad del requerimiento de cambio.	**TRFC**: Total de RFC registrado**.** **NCR:** Número de RFC rechazados.	Índice = 0% no existen rechazos. Índice >0% existen rechazos. Índice = 100% Todos los RFC fueron rechazados.	Mensual y Anual	(NCR/ TRFC)*100	%	Disminuir la cantidad de RFC rechazados debido a que la solicitud está mal planteada o incompleta.	Primer año: reducir el índice al 30%. Segundo año: 15%	El indicador debe tender siempre a 0%.

3.4.2. MANUAL DE PROCESO DE GESTIÓN DE VERSIONES

MAN-002-PIC-FAP

1. Objetivos y alcance

- Establecer una política de implementación de nuevas versiones de software y hardware.
- Implementar las nuevas versiones de software y hardware en el entorno de producción después de que la Validación y Pruebas las haya verificado en un entorno realista.
- Garantizar que el proceso de cambio cumpla las especificaciones del RFC aprobada correspondiente.
- Asegurar, en colaboración con la Gestión de Cambios y la Gestión de Configuración y Activos TI, que todos los cambios se ven correctamente reflejados en la DML y DS.
- Archivar copias idénticas del software en producción, así como de toda su documentación asociada, en la DML.
- Mantener actualizado el DML y DS.

2. Glosario de Términos

Versión: Se utiliza una versión para identificar una línea base específica de un elemento de configuración.

Generalmente se utiliza un incremento numérico en la nomenclatura para identificar una secuencia de versiones.

Ambiente de pruebas: Es un ambiente en el cual se desarrollan las pruebas de los nuevos servicios o nuevas versiones de servicios, mismo al que tiene acceso únicamente la gestión de validación y pruebas para realizar la comprobación de que el servicio cumpla con lo requerido.

Ambiente real (pre-producción): El ambiente en que se realizan las pruebas y validaciones de un servicio debe contener las características funcionales que el ambiente en el que se publica un servicio, es decir, que la validación y pruebas se realiza en un ambiente en el que más tarde el servicio será puesto en producción.

Entorno de producción: Ambiente o entorno de producción es en el cual se encuentra el servicio en producción, está descrito en el catálogo de servicios y es accesible a los usuarios.

Arquitectura: Es la estructura de un sistema o servicio de TI, incluye las relaciones de los componentes entre sí y con el ambiente en que se encuentran. La arquitectura también incluye las normas y directrices que guían el diseño y evolución del sistema.

Estructura: El diseño de un servicio define una serie de componentes y características que con las cuales debe cumplir.

Lanzamiento de versión: Uno o más cambios en un servicio de TI que se construyen, prueban e implementan de forma conjunta. Una sola versión puede incluir cambios en el hardware, software, documentación, procesos y otros componentes.

Plan de Back-out: Plan de retirada. Son las medidas que se han tomado para recuperarse después de un cambio o liberación fallida. Una rectificación puede incluir un back-out, la invocación de los planes de continuidad del servicio, u otras acciones diseñadas para permitir que el proceso de negocio continúe.

DML: Biblioteca definitiva de medios. Es uno o más lugares en los que las versiones definitivas y autorizadas de todos los elementos de configuración de software se almacenan en forma segura. La biblioteca definitiva de medios también puede contener licencias y documentación como elementos de configuración asociados. Es una sola área de almacenamiento lógica, aun tratándose de múltiples lugares. La biblioteca definitiva de medios está controlada por la gestión de activos de servicio y configuración y se registra en el sistema de gestión de la configuración.

DS: Repuestos definitivos. Repositorio que contiene piezas de repuesto para los elementos de configuración en el entorno de producción, generalmente considerados para los cambios de hardware.

3. Roles del proceso

3.1. Gestor de Versiones

- Se encarga de planificar, programar y controlar el movimiento de versiones en ambientes reales y de prueba. Su objetivo principal es salvaguardar la integridad en el ambiente real y que se utilicen los componentes correctos.
- Establece una política de planificación para la implementación de nuevas versiones.
- Diseña las versiones
- Gestiona la revisión Post Implementación, o pruebas post-producción.
- Actualiza la DML y el DS una vez que se ha puesto en producción el cambio.
- Comunica la liberación de las nuevas versiones al Gestor de Cambios

3.2. Gestor de Cambios.

- Entrega la Orden de Trabajo, con la cual inicia el proceso de gestión de versiones.
- Analiza errores o fallas reportadas en el caso en que se haya ejecutado el Plan de Back-out
- Comunica errores a los responsables de la solución.

3.3. Gestor de Pruebas

- El Gestor de Pruebas se asegura de que las versiones implementadas y los servicios resultantes cumplan las expectativas del cliente, y verifica que las operaciones de TI puedan brindar apoyo a los servicios nuevos.
- Validar las nuevas versiones.
- Poner a prueba las nuevas versiones en un entorno que simule lo mejor posible el entorno de producción.

3.4. Gestor de Desarrollo

- Es responsable del diseño de aplicaciones necesarias para la prestación de un servicio, esto incluye la especificación de tecnologías, la aplicación de arquitecturas y de estructuras de datos como base para el desarrollo o la personalización de aplicaciones.

- Desarrollar los nuevos requerimientos de acuerdo a las solicitudes de cambio y órdenes de trabajo presentadas, además de la revisión de la planificación realizada.

- Si el desarrollo no se realiza internamente, es el responsable de llevar a cabo la contratación para adquirir de terceros las nuevas versiones de servicios o nuevos servicios.

3.5. Gestor de Infraestructura

- Diseña y facilita los componentes de infraestructura y sistemas necesarios para la prestación de un servicio; esto incluye la especificación de tecnologías y productos como base para su implementación, adquisición o personalización.

- Implementar las nuevas versiones desarrolladas y validadas en el entorno de producción.

- Lleva a cabo los planes de back-out o retirada de la nueva versión si esto fuera necesario.

4. Descripción de Actividades.

4.1. GV 01: Recibir orden de trabajo - OT.

Ejecución: El Gestor de Versiones revisa que la documentación esté completa y define políticas que responden al cómo y cuándo se configura y despliega una nueva versión, los horarios de liberación, y establece las horas/hombre que serán necesarias para configurar y desplegar versiones.

Entradas: Orden de trabajo.

Salidas: Plan de despliegue de las nuevas versiones. Prepara los archivos y configuraciones para el área de desarrollo.

Condiciones: La orden de trabajo especifica lo que requiere el área de desarrollo para iniciar la programación del cambio.

4.2. GV 02: Planificar el despliegue de la versión.

Ejecución: Se detalla y se especifica los usuarios y elementos del servicio que se van a modificar o agregar. El control de versiones se llevará a cabo mediante los instructivos y manuales desarrollados por la organización.

Entradas: Solicitud y detalle de lo requerido por la gestión de desarrollo para realizar el desarrollo del cambio.

Salidas: Diseño de la versión para despliegue en los ambientes controlados.

Condiciones: Los cambios y mantenimientos del presente manual, se basan en las características del servicio como software. Las tareas y actividades del control de versiones se realizarán de acuerdo a los instructivos adjuntos.

4.3. GV 03: Verificar despliegue

Ejecución: El despliegue de las nuevas versiones se realiza directamente en los ambientes controlados. El acceso, por las políticas de seguridad, se tiene únicamente por parte del área de infraestructura. Se solicita el despliegue de los cambios desde la gestión de cambios. El gestor de versiones verifica que el despliegue se ha realizado de la manera correcta, según lo indicado en los instructivos, y se registra el número de revisión asignado al cambio.

Entradas: Confirmación de despliegue en ambiente por parte de la gestión de infraestructura.

Salidas: Confirmación de la verificación. Continúa el flujo en el seguimiento del gestor de cambios.

Condiciones: En la verificación de despliegue, se controla que todos los cambios se hayan implementado en el ambiente controlado que se dio la orden de despliegue. En caso de existir observaciones por el gestor de versiones, deberá analizar la tarea de despliegue y solicitar nuevamente a la gestión de infraestructura la ejecución (flujo GI). Si no existen observaciones y el despliegue es correcto, informa al gestor de cambio el éxito del despliegue para que se realicen las pruebas en el ambiente correspondiente (flujo GV 04).

4.4. Solicitar pruebas de Q&A

Ejecución: El gestor de versiones solicita que se realicen las pruebas de aseguramiento de la calidad por parte de la gestión de pruebas. Las actividades de seguimiento de la gestión de cambio intervienen para mediar entre las áreas de tecnología y el negocio. Las tareas de Q&A y del personal de pruebas es similar, sin embargo, existen factores no funcionales que lo determina la calidad del servicio implementado y lo determinará las áreas del negocio.

Entradas: Confirmación de despliegue en el ambiente controlado.

Salidas: Solicitud de pruebas y validaciones.

Condiciones: No se iniciará las actividades de validación y pruebas sin la confirmación de despliegue correcto.

4.5. GV 05: Planificar el despliegue en producción.

Ejecución: El gestor de cambios ha solicitado las pruebas y validaciones funcionales y no funcionales al área solicitante del negocio. La aprobación ha sido informada al área de producción para que el gestor de versiones del área de producción realice la planificación y despliegue del servicio en producción. Para esta actividad se determinará el menor impacto en el servicio y se seguirá estrictamente los lineamientos de las políticas de despliegue, que son parte integral del presente documento. El despliegue debe contar con un plan de reversa en caso de que las pruebas post-producción no tengan los resultados esperados.

Entradas: Informe del requirente aprobado. Solicitud de despliegue del gestor de cambios. Plan de reversa o “back-out”

Salidas: Programación de despliegue. Confirmación de la ventana de mantenimiento en caso de requerir. Coordinación de equipos de trabajo para el despliegue.

Condiciones: Lo especificado en las políticas de despliegues en producción. No se podrá poner en producción el cambio si no se cumple con lo establecido en las políticas, ya que los objetivos principales del proyecto son obtener la mayor disponibilidad del servicio y su calidad.

4.6. Verificar despliegue en producción.

Ejecución: Se verifica que la versión que se implementó en los ambientes controlados sea la misma que se desplegó en producción y se realiza el control de versiones. El ambiente de producción utiliza otro repositorio de versiones independiente de las versiones de desarrollo y pruebas. La verificación involucra la revisión de todos los cambios de aspectos funcionales y de configuraciones. Ejecutar los planes de back-out en caso de recibir la orden de reversa, cuando las pruebas post-producción no sean exitosas, o algún otro aspecto indicado por el gestor de cambios.

Entradas: Confirmación de despliegue en producción. Informe de pruebas post-producción.

Salidas: Confirmación de despliegue exitoso. Informe al gestor de cambios.

Condiciones: Si las pruebas son fallidas, o existe un componente defectuoso se debe esperar una orden de reversa para aplicar los planes de back-out.

5. Indicadores Clave de Desempeño. KPI.

Ítem	Nombre del indicador	Variables cuantitativas	Índice	Periodo	Formula	Unidad de medida	Objetivo	Metas	Tendencia
1	Nuevas versiones en producción	**NVP**: Número de nuevas versiones en producción **NVR**: Número total de nuevas versiones requeridas por cambios	Índice: i = 100 % Todas las versiones requeridas están en producción Índice: 0% < i < 100% Existen versiones requeridas que no se encuentran en producción.	Mensual y Anual	i=(NVP/NVR)*100	%	Obtener el mayor número de cambios solicitados en versiones nuevas en producción	Primer año: Incrementar el indicador al 50% Segundo año: 70%.	El índice debe tender al 100%.

2	Rechazo de las nuevas versiones	**NBO**: Número de Back-Out (Retorno de versiones en producción) **NVP:** Número de nuevas versiones en Producción.	Índice i = 0% no existen retornos de versión en el mes. Índice i >0% existen rechazos.	Mensual y Anual	(NBO/ NVP)*100%	%	Mejorar la calidad del proceso de versiones. Disminuir la cantidad de back-out de versiones.	Primer año: reducir el índice al 40%. Segundo año: 20%	El indicador debe tender a 0%,

3.4.3. INSTRUCTIVO: GESTIÓN DE POLÍTICAS DE LOS AMBIENTES CONTROLADOS

CODIGO: INS-003-PIC-FAP

INTRODUCCIÓN

La gestión del servicio tecnológico, en este caso el software, se encuentra orientada a la estrategia de mejorar la operación del mantenimiento y nuevos desarrollos de las aplicaciones. En este instructivo se define las políticas, diseño y análisis del requerimiento para la gestión operativa de los ambientes controlados para pre-producción del servicio.

En una organización en la cual se utilizan varios ambientes para el desarrollo y pruebas de los cambios solicitados y nuevas aplicaciones, los servidores tienen

un control inadecuado de accesos y un inexistente manejo de versiones, lo cual podría causar en reiteradas ocasiones que los servicios no se encuentren disponibles y que varios desarrollos sean retornados a versiones anteriores ya que no cumplen con las especificaciones solicitadas.

El presente instructivo describe el análisis y diseño para la implementación de varios ambientes controlados y homogéneos para las actividades de: desarrollo, pruebas y preproducción; con el objeto de garantizar el menor impacto en la puesta en producción del software como servicio.

Como complemento de los instructivos de este proyecto, se desarrollan las políticas y los procesos de gestión de versiones y gestión de cambios para los lanzamientos a producción, controlando los cambios que se realizan a las aplicaciones en cada etapa, desde desarrollo hasta producción. Estos procesos contienen planes de reversión en caso de falla, los cuales se deberán ejecutar de manera inmediata para no tener paralizaciones en los servicios.

1. ANÁLISIS

Para el servicio se requiere de al menos tres servidores, los mismos que se detallan a continuación. Se recomienda que los servidores sean provisionados de manera virtual, ya que puede requerirse una modificación en las capacidades. Por otra parte, en ocasiones puede presentarse un escenario en el cual se debe realizar desarrollos que se afectan mutuamente, cuando se usan módulos en común; para ello podría agregarse otro servidor de desarrollo y pruebas, de esta manera lograr que se ejecute el plan de pruebas independientemente por cada cambio solicitado. No se recomienda esta técnica para el ambiente de preproducción, ya que se requieren pruebas en las cuales se estabilicen las aplicaciones y, en el caso antes descrito, se necesitará pruebas de compatibilidad de los desarrollos realizados antes del lanzamiento a producción.

- DESARROLLO: Ambiente en el cual el área de programación y desarrollo realiza la programación de los cambios o nuevas herramientas, de acuerdo a los requerimientos formales ingresados. Se realizan pruebas unitarias para los cambios específicos.

- PRUEBAS: Es el ambiente en el cual el área de DESARROLLO implementa los cambios realizados para que sean aprobados en los aspectos técnicos por el área de PRUEBAS (Q & A). En estos dos ambientes se podrá solicitar la creación de ambientes similares para realizar desarrollos y pruebas de cambios aislados. Sin embrago, las pruebas también deberán unir los cambios antes de pasar al siguiente ambiente, ya que se requiere comprobar la funcionalidad de los requerimientos en conjunto. Los parámetros de calidad de las pruebas se describen en las "*Políticas de calidad y validación para el proceso de Q&A*"[21].

- PREPRODUCCIÓN: En este ambiente se implementan los cambios que han tenido éxito en el ambiente de PRUEBAS. Se realizan los controles necesarios por las áreas requirentes, a los cambios formales, los cuales deben cumplir con los objetivos del giro del negocio.

Todos los ambientes controlados formarán parte del proceso de versiones para control de los cambios realizados y de las reversas en caso de existir; por lo tanto, todos aquellos servidores que se dispongan para estos fines, deberán estar registrados al mismo y único repositorio de SVN provisto para la gestión y control de versiones.

2. DISEÑO

Los ambientes requeridos, deben ser de similares características al ambiente real de producción en arquitectura. En capacidades será de acuerdo al número de procesos o usuarios que lo utilicen. La capacidad de los servidores de desarrollo y pruebas podrá variar de acuerdo al requerimiento del módulo o herramienta que se requiera implementar. Como se indica en la redacción de este proyecto, la gestión de cambios se utiliza justamente para las modificaciones de los servicios, por lo tanto, también deberá analizar el rendimiento de las capacidades de los servidores en estas etapas. Es posible que se detecten necesidades de cambios de hardware, arquitectura, framework[22] o seguridad, que no hayan sido analizadas en el documento de requerimiento.

[21] DOC-Políticas de calidad y validación de Q&A, POL-002-PIC-FAP

[22] En desarrollo de software: Una estructura establecida de artefactos, funciones, objetos o módulos preexistentes para facilitar el proceso de desarrollo.

No se recomienda aumentar los recursos cada vez que los indicadores de monitoreo alerten de los umbrales permitidos en procesamiento o memoria, así como tampoco es recomendable reiniciar los servidores para cerrar las conexiones. Cuando estos casos se presentas reiteradamente, es preciso analizar otras causas del diseño del servicio.

A continuación, se muestra el diagrama lógico de la relación entre los ambientes a implementar. Cada uno estará vinculado al repositorio correspondiente en el servidor de versiones. Únicamente cuando se tiene éxito en las pruebas pasarán a la siguiente etapa, hasta implementar en producción el cambio, de acuerdo a las "*Políticas para despliegue en Producción*".[23] Como se ha indicado, el ambiente de producción no es considerado en este instructivo. Por seguridad y gestión de la disponibilidad del servicio se utiliza otro repositorio SVN.

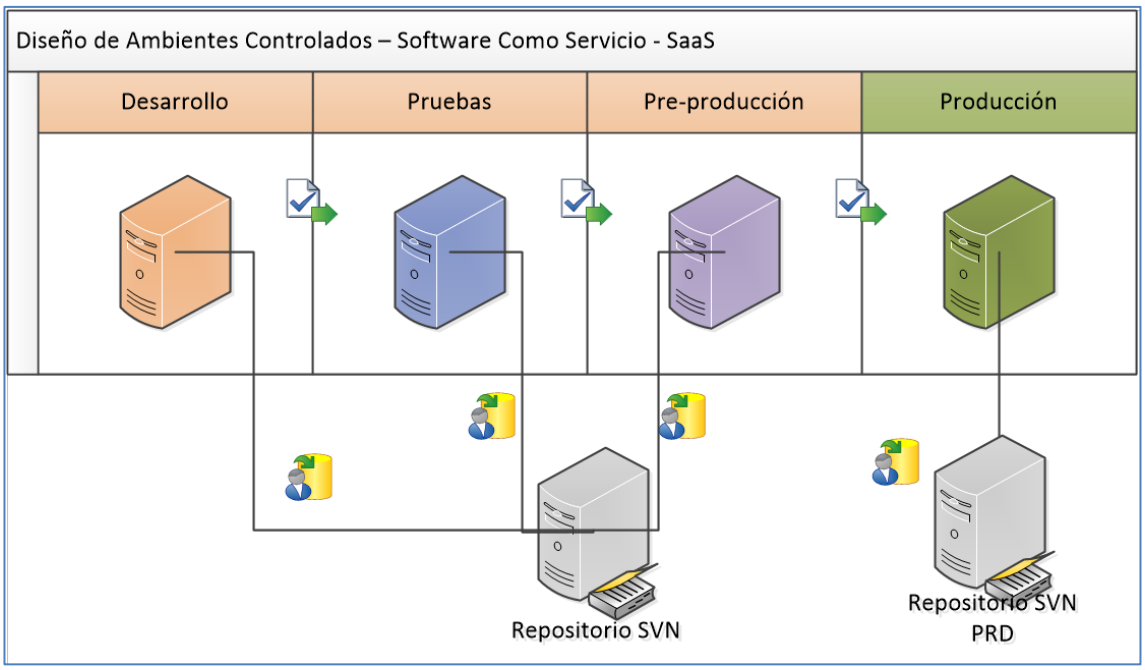

Gráfico 6: Diseño lógico de ambientes controlados y repositorio SVN

3. REQUERIMIENTOS

Los requerimientos para cada servidor, deberán indicarse al área de infraestructura, o el área indicada para la provisión de los servidores, al menos con las siguientes características:

[23] Políticas de despliegue en producción, POL-003-PIC-FAP

A.- SERVIDOR DE DESARROLLO VIRTUAL

Procesadores: 2VCPU

RAM: 4GB

DISCO: 50GB, 40GB, 15GB, 32GB

SO: Red Hat Enterprise Linux 5.9

Plataforma de Virtualización: VM

B.- SERVIDOR DE PRUEBAS VIRTUAL

Procesadores: 2VCPU

RAM: 4GB

DISCO: 50GB, 40GB, 15GB, 32GB

SO: Red Hat Enterprise Linux 5.9

Plataforma de Virtualización: VM

C.- SERVIDOR DE PREPRODUCCION:

Procesadores: 16

RAM: 32GB

DISCO: 80GB, 15GB, 40GB, 32GB

SO (Sistema Operativo): Red Hat Enterprise Linux 5.9

Plataforma de Virtualización: NINGUNA (AMBIENTE FISICO)

3.4.4. INSTRUCTIVO: INSTALACIÓN Y CONFIGURACIÓN DE LA HERRAMIENTA DE CONTROL DE VERSIONES. SVN – SUBVERSION.

CODIGO: INS-004-PIC-FAP

<table>
<tr><td rowspan="2"> 
</td><td colspan="6">INSTRUCTIVO: INSTALACIÓN Y CONFIGURACIÓN DE LA HERRAMIENTA DE CONTROL DE VERSIONES. SVN – SUBVERSION.</td></tr>
<tr><td rowspan="2">CODIGO: INS-004-PIC-FAP

REF: PIC-04-2014-FAP</td><td>No. de revisión:</td><td>Autor:</td><td colspan="3">Fecha de Elaboración y/o actualización:</td><td>Fecha de impresión:</td></tr>
<tr><td>01</td><td>FAP</td><td>Día
15</td><td>Mes
06</td><td>Año
2014</td><td>Página:
de 76</td></tr>
</table>

**Se recomienda utilizar este encabezado como parte del formato del instructivo*

INTRODUCCIÓN

Para la gestión del proceso de control de versiones, existe una gran cantidad de mecanismos o herramientas que proporcionan facilidades para el cumplimiento de los objetivos. Es así que en varias herramientas de mesa de servicios o mesa de ayuda, existen las plantillas para llevar a cabo la gestión de las versiones. Para el caso determinado, el servicio es software, el cual requiere un control a detalle de todas las variaciones y cambios que tiene el producto y sus configuraciones.

Las herramientas de la mesa de servicios tiene un control sobre el cambio y la versión de la cual se libera un aplicativo o un módulo del mismo, sin embargo, existen otras herramientas disponibles, las cuales generan un detalle paso a paso de cada cambio por mínimo que sea, en el cual se crea un registro por cada revisión.

Una revisión es una versión del repositorio o archivo modificado. Se lo puede comparar como un estado histórico del software en determinado momento, ya que se puede regresar y avanzar a cualquier punto deseado siempre y cuando así esté registrado.

La herramienta en la cual se basa este instructivo es SVN – Subversion, una conocida y popular aplicación que centraliza los repositorios de la aplicación a controlar. Cada desarrollador tiene la opción de crear una copia de trabajo total en su unidad de desarrollo y de acuerdo a los cambios realizados, registrarlos en el repositorio central para compartir una nueva versión del software. Como se ha

indicado desde un inicio, la mayor utilidad de la herramienta es para software desarrollado sin compilación, es decir, específicamente en este caso para PHP, indistintamente del framework que se utilice.

1. REQUERIMIENTOS

1.1. Software

Para la instalación de la herramienta es necesario contar con los siguientes elementos:

- Sistema operativo Linux, distribución recomendada: Fedora (SO utilizado), CentOS, Red Hat. En general, la instalación dependerá de las características del sistema operativo que se vaya a utilizar, aunque las prestaciones de cada uno no involucran el rendimiento de la herramienta. En cuanto a las configuraciones específicas del Sistema Operativo se requiere conocimiento en cada caso para el inicio de servicios y configuraciones de seguridad. La configuración propia de la herramienta se encuentra en parámetros definidos por el usuario que realiza la instalación.

- Servicio HTTP Apache: Se requiere la instalación de este servicio en el Sistema Operativo, el mismo que es utilizado como interfaz para la comunicación entre el repositorio central y los usuarios que requieren leer o escribir nuevas versiones del software. La aplicación principal de este servicio es como un servicio web, con el cual se accede a páginas web estáticas o dinámicas, que se encuentran en ambientes locales o publicados. La herramienta subversión utiliza uno de estos módulos para la interfaz entre los clientes y el servidor por medio de HTTP.

- Almacenamiento lógico: los repositorios lógicos deberán tener la capacidad necesaria para almacenar a la aplicación y sus revisiones generadas. Se recomienda una revisión periódica del espacio utilizado para proyectar un incremento posterior.

1.2. Hardware

- Los requerimientos de hardware no son específicos para la aplicación. Dependerá del análisis de cantidad de usuarios y tamaño de la aplicación desarrollada. El control de versiones se puede instalar en un computador local en caso de una aplicación pequeña, o en servidores virtuales si es pequeña o mediana. Para el caso de una gran aplicación se requiere servidores de mayor escala.

- El equipo en el que se instala la herramienta deberá contar con: tarjeta de red, espacio físico en disco, unidad de procesamiento y memoria RAM.

2. PROCEDIMIENTO

2.1. Servicio HTTP Apache

Verificación de que el servicio httpd, Apache web service, se encuentra instalado en el sistema operativo.

Para verificar, se debe ejecutar el siguiente comando:

```
[root@localhost$]# service httpd status
```

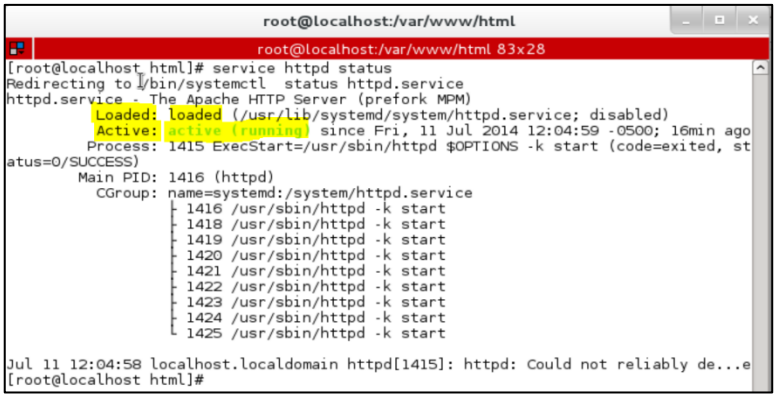

Gráfico 7: Verificar servicio Apache

2.2. Instalación de Subversion y módulos para Apache

Se requiere la instalación de Subversion y el módulo de svn para Apache. Dependiendo de la distribución de Linux, se requiere la instalación de los mismos. Para el caso en Fedora, lo siguiente:

```
[root@localhost$]# yum install subversion
```

```
[root@localhost$]# yum install mod_dav_svn
```

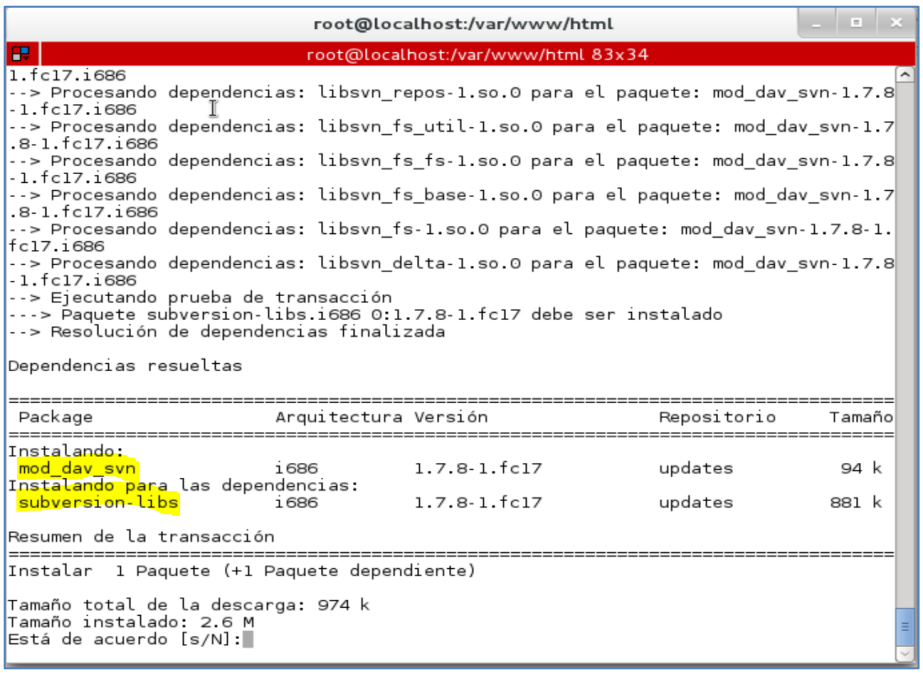

Gráfico 8: Instalación de Módulos SVN

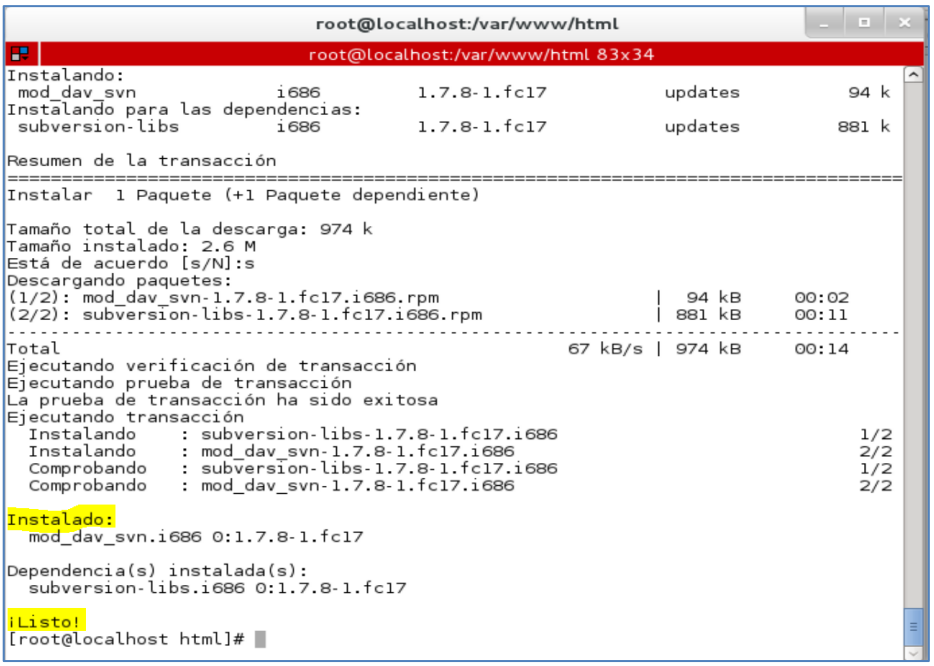

Gráfico 9: Verificación de instalación módulo SVN

2.3. Configuración de Apache para subversión

En el directorio destinado para la instalación del servicio Apache, se requiere la edición y configuración de acuerdo al requerimiento del fichero subversion.conf, el cual se encuentra en la ruta /etc/http/conf.d/subversion.conf.

En este fichero se encuentra la configuración del módulo svn de apache, en el cual se indican los parámetros para el acceso de esta herramienta. Se define a continuación:

- Ruta del directorio (repositorio) que se utilizará para el ambiente de desarrollo: desarepos
- Se debe indicar también la ruta del archivo para usuarios y contraseñas: /etc/svn/svn-auth-users
- Nombre del repositorio: “SVN DESARROLLO”

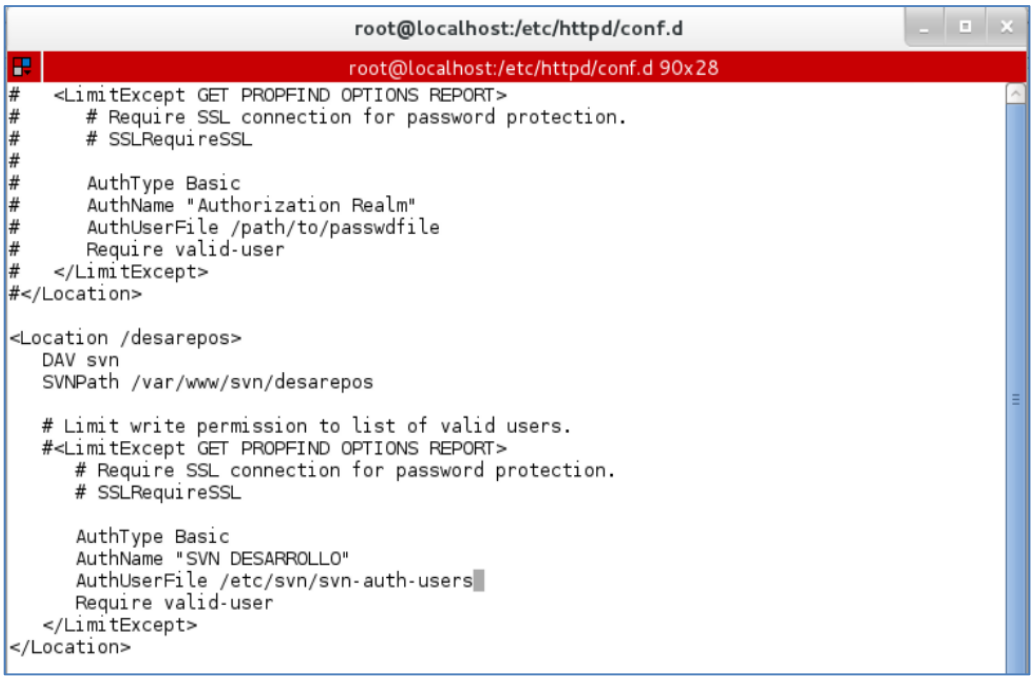

Gráfico 10: Configuración subversion.conf

2.4. Creación de un Repositorio para control de versiones

La herramienta SVN permite que se realice la creación de varios repositorios para distintas aplicaciones, o para el control de versiones de una misma aplicación pero de distintos ambientes. En el documento de políticas de ambientes controlados se indica la aplicación de los repositorios.

Para crear un nuevo repositorio del ambiente de desarrollo, se requiere realizar lo siguiente (con el usuario root):

- Ir a la ruta de los directorios públicos del servicio httpd: /var/www/
- Crear el directorio svn: mkdir svn
- En el directorio “svn” creado:
    - svnadmin create desarepos (nombre elegido para el repositorio)
    - chown –R apache:apache desarepos/
    - chcon –R –t httpd_sys_content_t desarepos[24]

En esta etapa se puede crear otros repositorios, en caso de requerirlo, ya que debe tener la misma configuración. Se cambia el propietario del directorio a apache, para que el servicio realice la comprobación de usuario

[24] Algunos repositorios solo podrán ser de lectura para el usuario apache. Para habilitar la escritura se requiere habilitar: `setsebool -P httpd_unified=1`

y contraseña, y no se realice a nivel del sistema operativo. Para ello, a continuación configuramos el acceso de los usuarios.

```
[root@localhost svn]# pwd
/var/www/svn
[root@localhost svn]# svnadmin create desarepos
[root@localhost svn]# chown -R apache:apache desarepos/
[root@localhost svn]# chcon -R -t http_sys_content_t desarepos/
```

Gráfico 11: Creación de nuevo repositorio

2.5. Creación de usuarios para acceso al repositorio

En la configuración de accesos al repositorio creado, revisar el punto 2.3. del presente instructivo, se ha definido a la ruta /etc/svn/svn-auth-users como el contenedor de los usuarios permitidos para acceder a escribir o leer en el repositorio.

Para la creación de un nuevo usuario y el fichero (únicamente en caso de que sea nuevo el fichero), se debe emplear el siguiente comando:

- `[root@localhost$]# htpasswd –cm /etc/svn/svn-auth-users "Nuevo_usuario_nuevo_archivo"`

En el caso de que el archivo de usuarios ya exista, se requiere crear más usuarios en el mismo archivo. Se debe notar que no se utiliza el parámetro "-c" ya que esto eliminaría al archivo existente, se requiere únicamente la modificación agregando un usuario nuevo.

- `[root@localhost$]# htpasswd –cm /etc/svn/svn-auth-users "Nuevo_usuario_agregado_archivo_antiguo"`

Se agrega el usuario "usuario_test":

- `[root@localhost$]# htpasswd –cm /etc/svn/svn-auth-users usuario_test`

2.6. Configuración de Firewall y acceso remoto al repositorio

Para comprobar que el repositorio creado para control de versiones ha sido creado y publicado correctamente, se puede realizar el acceso mediante un navegador de internet. Es importante recalcar que además de la prueba

local del servicio y el repositorio se realice desde otro terminar que tenga acceso al recurso en la red.

Para verificar que en el servidor que se ha instalado esté habilitado el firewall y que permite el acceso desde externos:

```
[root@localhost$]# system-config-firewall-tui

Habilitar los servicios http y https para acceso externo
```

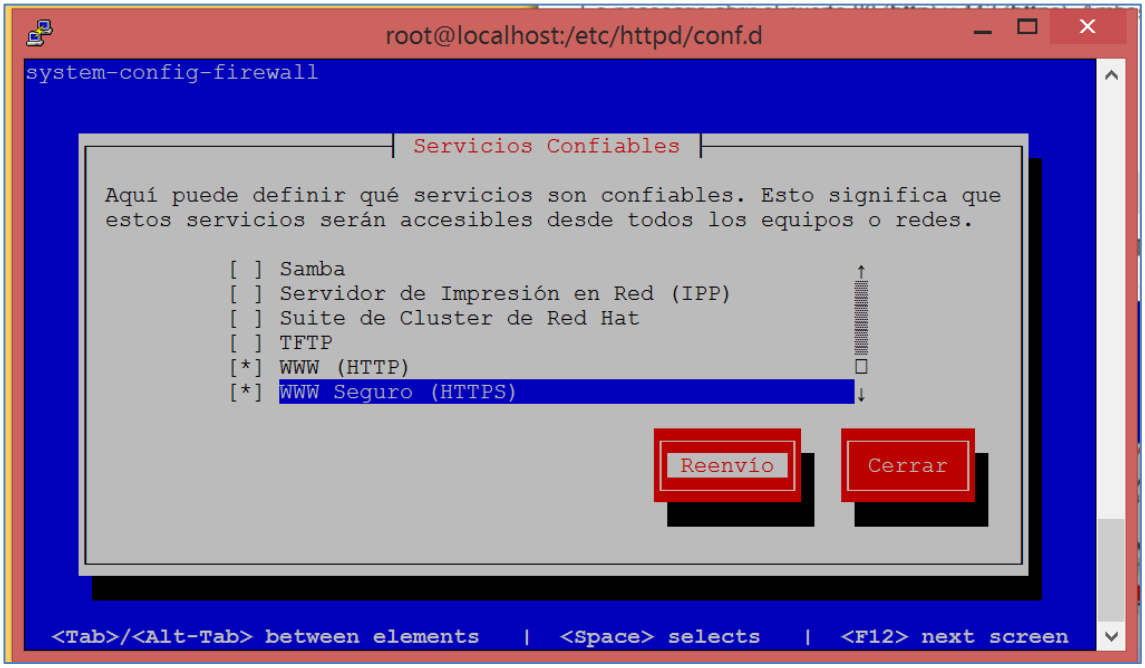

Gráfico 12: Configuración de Firewall

- Acceder mediante un navegador de internet, tanto desde el servidor local (en caso de ser posible) como desde un terminar de la red.

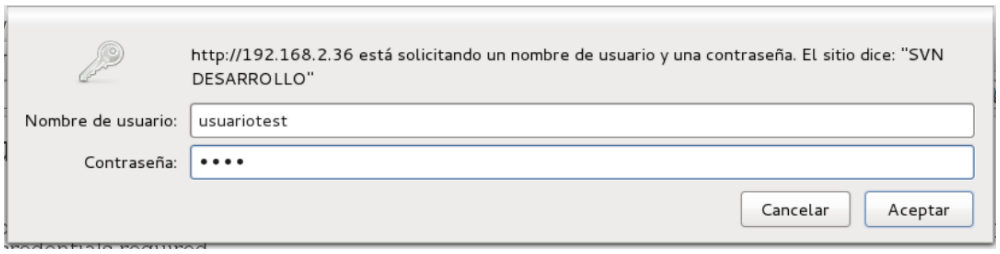

Gráfico 13: Ingreso de usuario y contraseña definidos

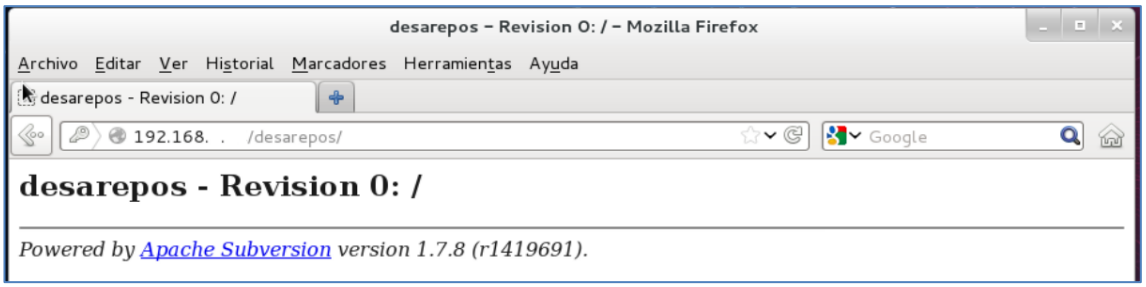

Gráfico 14: Acceso local al repositorio SVN

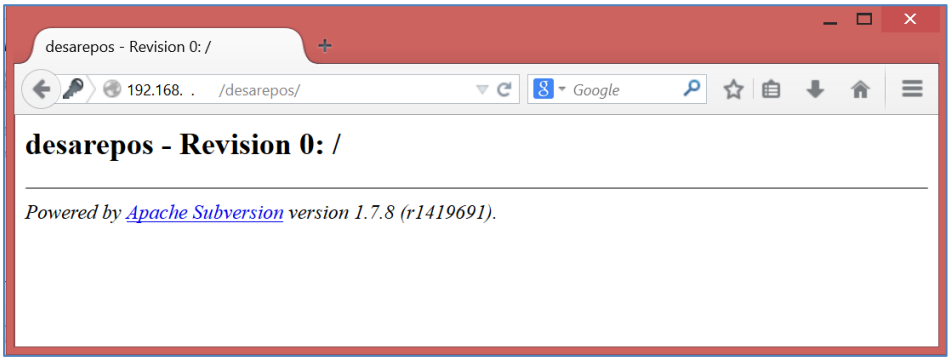

Gráfico 15: Acceso remoto al repositorio SVN

El acceso a los repositorios se comprueba con la visualización del nombre y el número de la última revisión, en este caso, la revisión 0 se muestra ya que no se ha ingresado ningún cambio en el control de versiones.

3.4.5. INSTRUCTIVO: GESTIÓN DE VERSIONES CON SVN

CODIGO: INS-005-PIC-FAP

<table>
<tr><td rowspan="2"> 
</td><td colspan="6">INSTRUCTIVO: GESTIÓN DE VERSIONES CON SVN</td></tr>
<tr><td rowspan="2">CODIGO: INS-005-PIC-FAP

REF: PIC-04-2014-FAP</td><td>No. de revisión:</td><td>Autor:</td><td colspan="3">Fecha de Elaboración y/o actualización:</td><td>Fecha de impresión:</td></tr>
<tr><td>01</td><td>FAP</td><td>Día
05</td><td>Mes
07</td><td>Año
2014</td><td>Página:
de 76</td></tr>
</table>

**Se recomienda utilizar este encabezado como parte del formato del instructivo*

INTRODUCCIÓN

Los instructivos INS-004-PIC-FAP y INS-005-PIC-FAP se desarrollaron en función de la aplicación de la herramienta para control de versiones SVN – Subversion. Esta herramienta, como se indica en los documentos mencionados, requiere de una instalación en servidor y el acceso desde los clientes o usuarios que realizan las tareas de desarrollo. Además, los ambientes controlados para desarrollo y pruebas, utilizan una copia de trabajo del repositorio SVN para actualizar la aplicación.

Los procesos para gestión de cambios y versiones determinan un flujo y serie de actividades. Las relacionadas con desarrollo y lanzamiento a producción requieren de registros de versiones y variaciones de las aplicaciones, para llevar un control de lo realizado. Para ello, en este modelo de gestión, se utiliza la herramienta para control de versiones Subversion.

En el instructivo de gestión de ambientes controlados, se indicó el propósito de cada uno de los ambientes de la siguiente manera:

- DESARROLLO: Ambiente en el cual el área de programación y desarrollo realiza la programación de los cambios o nuevas herramientas, de acuerdo a los requerimientos formales ingresados. Se realizan pruebas unitarias para los cambios específicos.

- PRUEBAS: Es el ambiente en el cual el área de DESARROLLO implementa los cambios realizados para que sean aprobados en los aspectos técnicos por el área de PRUEBAS (Q & A). En estos dos ambientes se podrá solicitar la creación de ambientes similares para

realizar desarrollos y pruebas de cambios aislados. Sin embrago, las pruebas también deberán unir los cambios antes de pasar al siguiente ambiente, ya que se requiere comprobar la funcionalidad de los requerimientos en conjunto. Los parámetros de calidad de las pruebas se describen en las "*Políticas de calidad y validación para el proceso de Q&A*"[25].

- PREPRODUCCIÓN: En este ambiente se implementan los cambios que han tenido éxito en el ambiente de PRUEBAS. Se realizan los controles necesarios por las áreas requirentes, a los cambios formales, los cuales deben cumplir con los objetivos del giro del negocio.

Además de las características de los servidores, se debe revisar con el gestor de cambios que los ambientes de pruebas cumplan con lo requerido en cada RFC, ya que puede existir una variación de requerimientos no funcionales que no serán evidentes en las pruebas de negocio. El gestor de cambios deberá coordinar con el preparador de los ambientes para asegurar que las pruebas se realicen bajo óptimas condiciones.

Se implementa una herramienta para control de versiones Subversion – SVN, con lo cual se debe informar a las áreas de Desarrollo e Implementación de los nuevos procedimientos para ejecución de cambios en los ambientes controlados. Del resultado de la evaluación y análisis de la gestión de procesos de cambio y liberaciones, se podrán optimizar los procedimientos definidos.

En los documentos mencionados al inicio del presente, no se incluye al ambiente de producción, el mismo que a pesar de manejar la misma herramienta y formar parte de los procesos, requiere otro tratamiento por seguridad, registro, seguimiento y aseguramiento del control de versiones. La herramienta para este control y el proceso es el mismo, sin embargo, se debe considerar las políticas de despliegues en producción. En el presente documento se detalla el procedimiento para incorporar el ambiente de producción a la herramienta de control de versiones.

[25] DOC-Políticas de calidad y validación de Q&A, POL-002-PIC-FAP

1. PROCEDIMIENTO

1.1. Crear un nuevo repositorio para el ambiente de producción. El URL es: http://subversion.pruebaslocales/prod_repos.

En el servidor de SVN, en var/www/svn/:

```
mkdir prod_repos
svnadmin create prod_repos
chown -R apache.apache prod_repos
chcon -R -t httpd_sys_content_t prod_repos
```

1.2. El directorio de la aplicación que se ingresa al control de versiones debe ser desde el inicio de los archivos de htdocs o www, exclusivos de la aplicación a controlar. Para ello, en este instructivo se tomará en cuenta a la ruta: /www/appweb/exe/

1.3. En el repositorio SVN, se debe crear tres directorios principales para la gestión de versiones, etiquetas y ramas.

La recomendación de las mejores prácticas para estos procedimientos es la creación de la siguiente estructura de directorios, con su respectiva utilidad:

- Trunk: el directorio /trunk será el tronco de la aplicación, es decir que contiene la aplicación completa desarrollada y todas las configuraciones necesarias. En este directorio se registrará todo cambio o versión que se realice desde una copia de trabajo.

- Branch: el directorio /branch almacena las ramas creadas para los nuevos desarrollos y modificaciones a la aplicación. Una rama es una copia del tronco en determinado momento para continuar o iniciar un cambio. La herramienta SVN se comporta como una “máquina del tiempo” para la aplicación. Es decir que se puede crear una rama de la aplicación como ésta fue tiempo atrás, e inmediatamente realizar una nueva rama con la aplicación actual. La creación de ramas es muy útil

para realizar modificaciones y desarrollos paralelos, de esta manera, el área de desarrollo podrá ejecutar sus actividades en ambientes distintos. Posterior a la ejecución del desarrollo se deberá unir los cambios para pasar el ambiente de pruebas. Todos los ambientes podrán tener estos tres directorios para los distintos propósitos, sin embargo, en el ambiente de producción no se recomienda el uso de ramas.

- Tags: El directorio /tags contiene las versiones liberadas de la aplicación. Se recomienda que los incrementos se lleven en un registro nuevo, distinto al de las revisiones que controla svn ya que la decisión de incremento de versiones del comité de cambios será de acuerdo a la siguiente política de tres niveles:

  Ejemplo: Appweb v1.4.2

  - Cambios Mayores: El incremento será en el primer nivel, Appweb v2.0.0

  - Cambios Menores: El incremento será en el segundo nivel, Appweb v1.5.0

  - Cambios de Emergencia: El incremento será en el tercer nivel: Appweb v1.4.3

  La creación de los directorios detallados en los repositorios no son mandatorios. La aplicación puede trabajar y gestionar el control de versiones sin ellos, sin embargo, se recomienda al menos la creación de los directorios sin contenido para una futura aplicación.

  Para realizar este procedimiento, primero se requiere crear una copia de trabajo a la base del repositorio creado en la revisión 0.

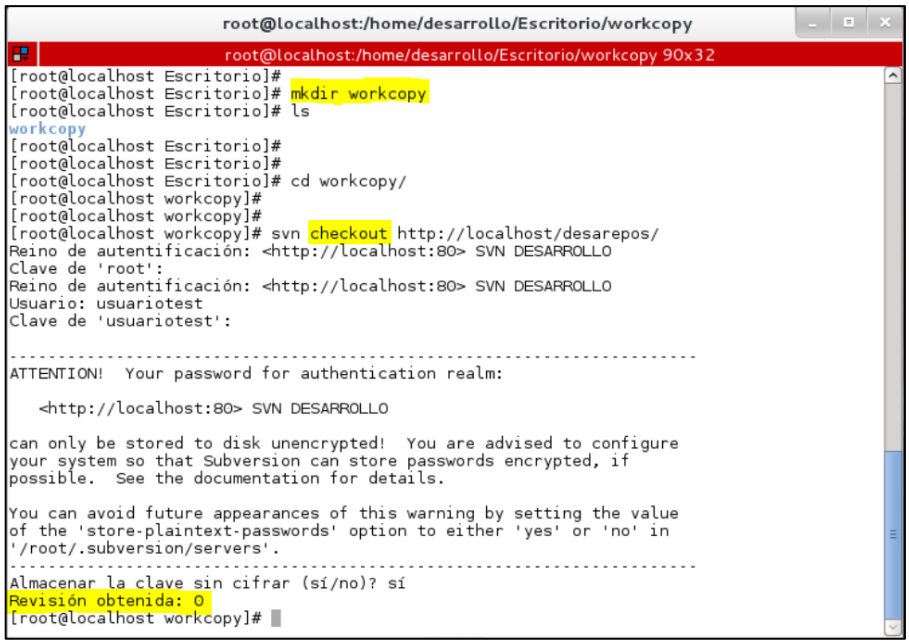

Gráfico 16: Copia de trabajo, revisión 0

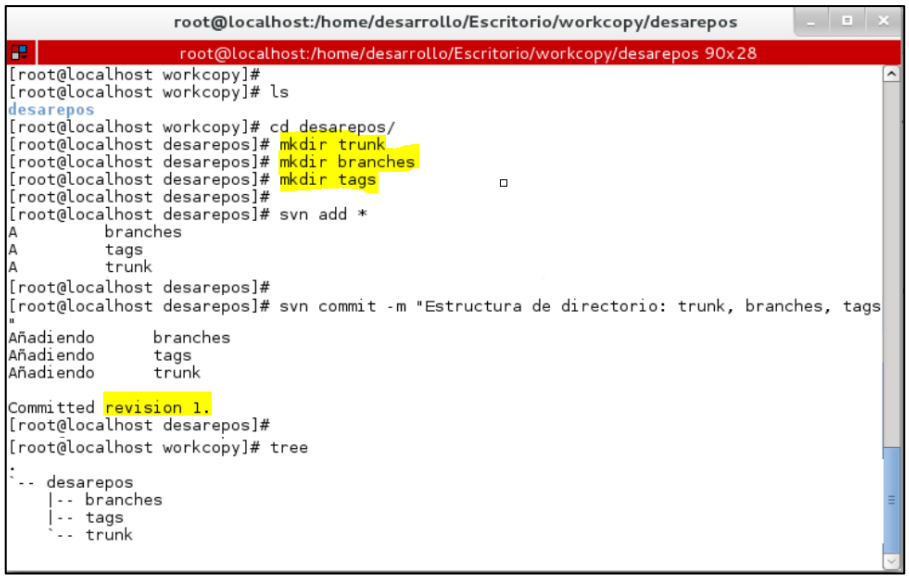

Gráfico 17: Creación de estructura: trunk, branches, tags

1.4. Crear el path /exe/ en el directorio trunk creado en el repositorio SVN para el ambiente de producción. (/www/appweb/exe/)

```
svn mkdir –m “Crear repositorio exe”
http://subversion.pruebaslocales/desarepos/trunk/exe
```

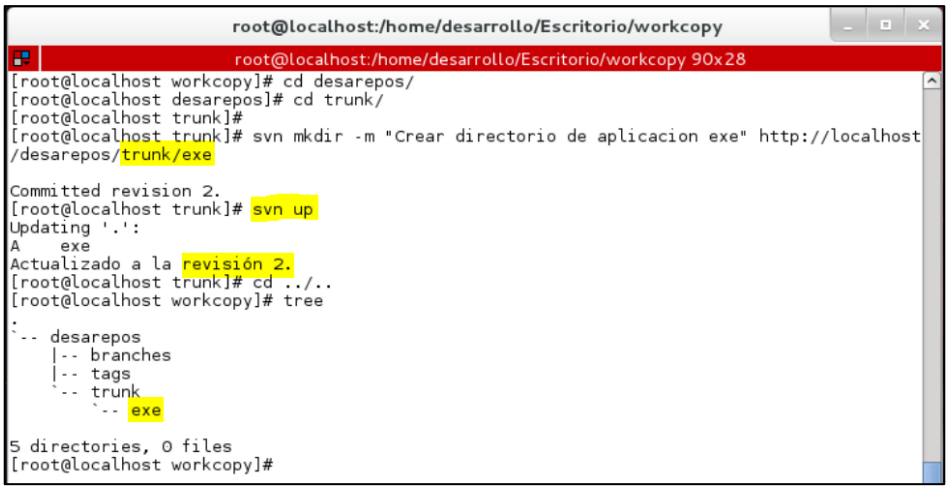

Gráfico 18: Creación de la ruta para la aplicación

1.5. El servidor de SVN, en la ruta /etc/svn/, verificar la configuración del archivo svn-auth-users en el cual constan los usuarios que tienen acceso a este repositorio.

Para crear nuevo archivo y usuario:

```
htpasswd -cm /etc/svn/svn-auth-users usuario
```

Para nuevo usuario en archivo existente:

```
htpasswd -m /etc/svn/svn-auth-users usuario
```

1.6. Solicitar una ventana de mantenimiento para agregar la aplicación o servicio en una copia de trabajo en el servidor de producción. Se requiere revisar las "*Políticas para despliegue en Producción*".

1.7. Verificar el espacio disponible del servidor de SVN. Se debe realizar una estimación del 100% para los siguientes 6 meses. Pasado este tiempo se analiza el recurso en disco utilizado para la estimación en los periodos futuros.

1.8. En el ambiente de producción, se requiere realizar el respaldo de todo el directorio /www/appweb/exe/. Registrar el archivo .bkp con fecha y hora de respaldo.

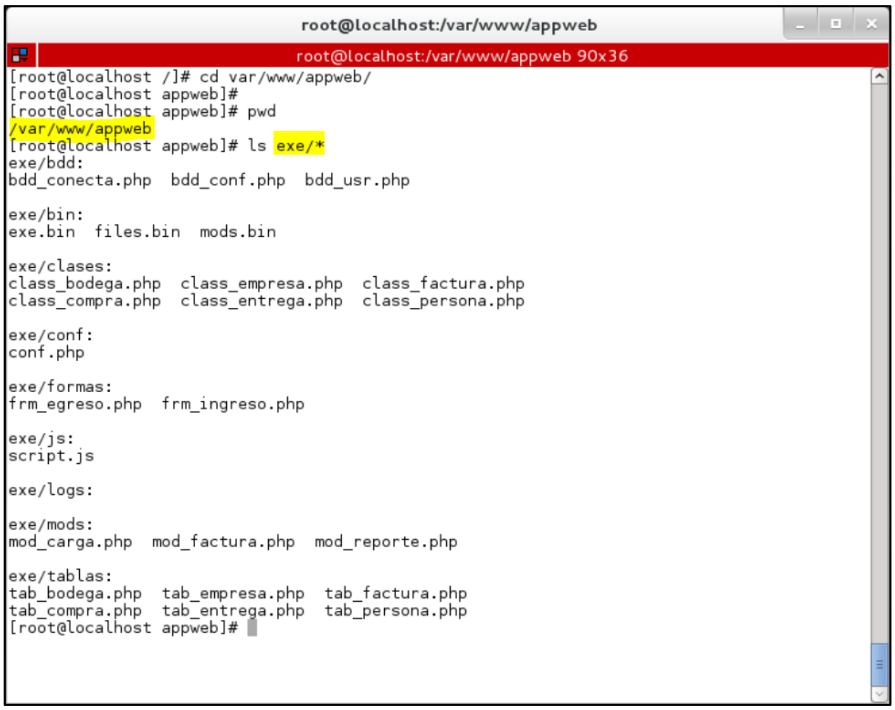

Gráfico 19: Contenido de la aplicación de ejemplo, Appweb

1.9. En el ambiente de producción, se requiere cambiar de nombre a la carpeta /exe/, ubicada en la ruta www/appweb/, a “exe.000”

Gráfico 20: Cambio de nombre del directorio de la aplicación Appweb

1.10. En el ambiente de producción, se requiere crear una carpeta con el nombre “exe” en el directorio www/appweb/

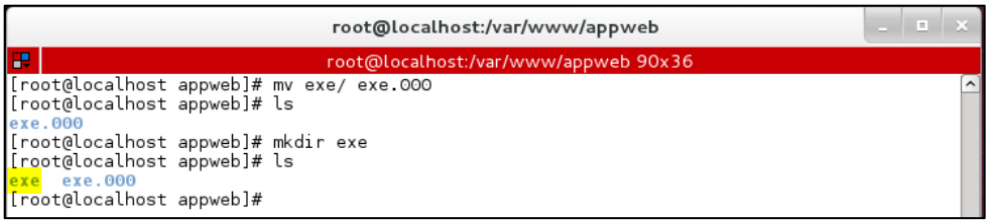

Gráfico 21: Creación del nuevo path para la aplicación Appweb

1.11. Se requiere convertir en una copia de trabajo al directorio de la aplicación: exe (Appweb)

```
svn checkout http://subversion.pruebaslocales/desarepos/trunk/exe exe
```

Gráfico 22: Copia de trabajo para el servidor de aplicación

1.12. Importar todo el contenido de exe.000/ a exe/ (copia de todo el contenido de la carpeta exe.000 al repositorio trunk/exe).

```
svn import –m "Carga inicial de la aplicación Appweb" http://
subversion.pruebaslocales/desarepos/trunk/exe/
```

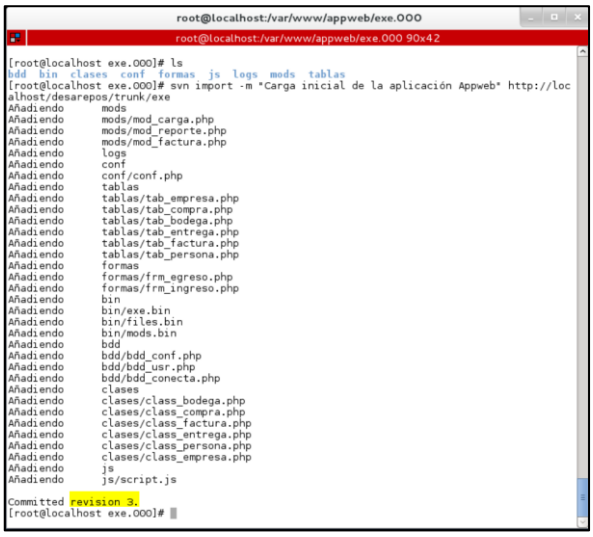

Gráfico 23: Importar la aplicación Appweb al repositorio svn

1.13. Actualizar el directorio exe/ de la aplicación Appweb. Indicar número de revisión. Ejemplo: Appweb v1.0.0 r3[26]

[26] Verificar que la carga inicial de la aplicación se encuentre en la r3. Esto confirma que se ha realizado el procedimiento correctamente.

```
svn up
```

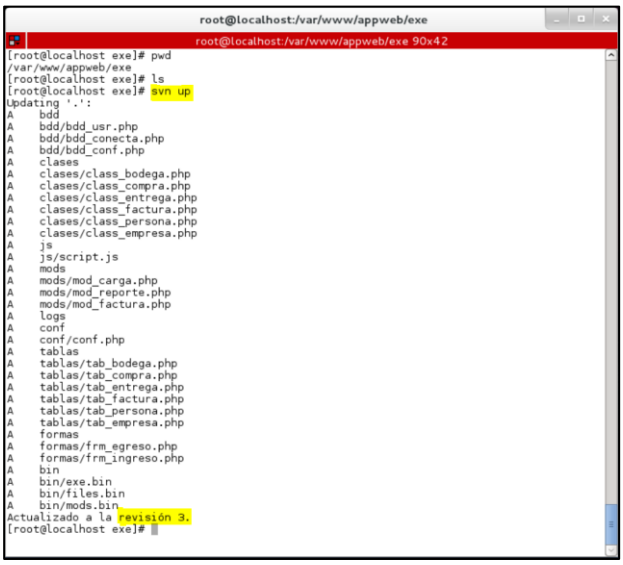

Gráfico 24: Actualización de la copia de trabajo. Aplicación en producción Appweb.

1.14. Para comprobar el acceso desde la red, o desde algún otro terminal, se puede realizar la creación de una copia de trabajo. También es posible esta comprobación con el acceso desde un navegador. (Solo lectura)

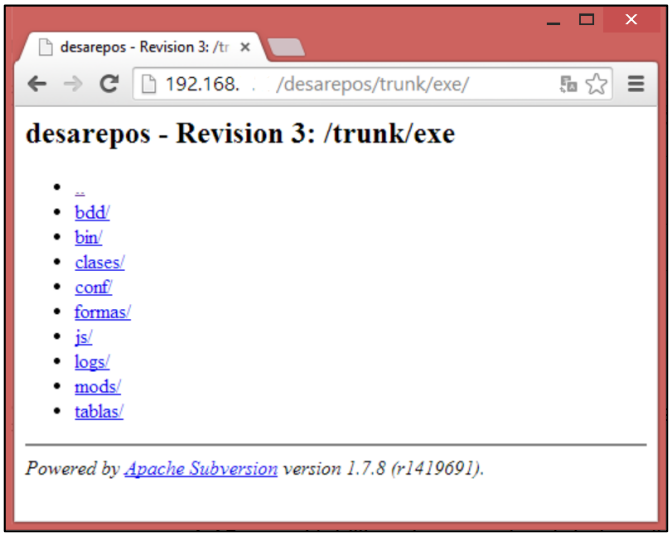

Gráfico 25: Acceso al repositorio SVN vía navegador.

1.15. Habilitar el acceso a la aplicación y servicio. Realizar pruebas de acceso y funcionamiento. Se agrega la página inicial index.html. Revisión 4.

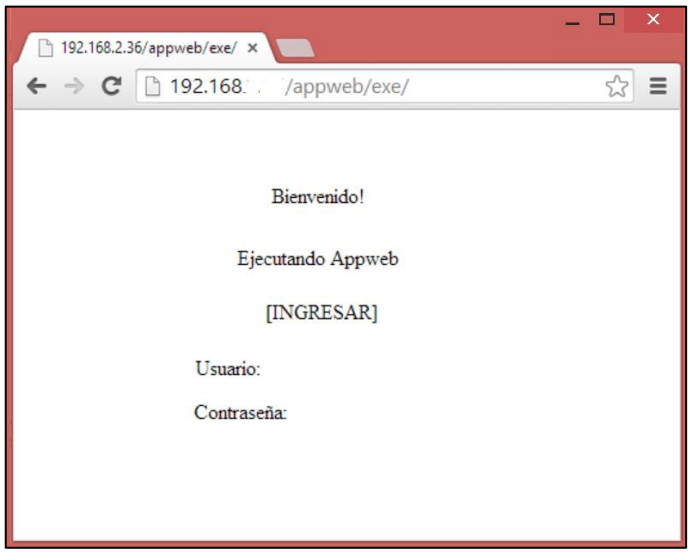

Gráfico 26: Verificación del servicio en producción

3.4.6. INSTRUCTIVO: GESTIÓN DE DESPLIEGUE EN LOS AMBIENTES CONTROLADOS

CODIGO: INS-006-PIC-FAP

LOGO INSTITUCIONAL	**INSTRUCTIVO: GESTIÓN DE DESPLIEGUE EN LOS AMBIENTES CONTROLADOS**					
CODIGO: INS-006-PIC-FAP **REF:** PIC-04-2014-FAP	No. de revisión:	Autor:	Fecha de Elaboración y/o actualización:			Fecha de impresión:
	01	FAP	Día 05	Mes 07	Año 2014	Página: de 76

**Se recomienda utilizar este encabezado como parte del formato del instructivo*

INTRODUCCIÓN

La definición de los procesos de gestión de cambios y versiones, no aseguran que la automatización o aplicación de herramientas informáticas mejoren la calidad de los servicios o productos que se están implementando. Estos flujos colaboran con las mediciones de resultados con los cuales luego de respectivos

análisis se puede determinar las mejoras de los servicios. Sin embargo, además del cumplimiento de los procesos, se debe definir las políticas, procedimientos, instructivos y documentación necesaria para que sea efectiva la gestión de cambios y versiones.

En los instructivos anteriores se ha revisado la instalación y manejo de la herramienta de control de versiones SVN, la cual es la aplicación elegida para el control de versiones de la aplicación o servicio.

El presente documento se desarrolla con la finalidad de aplicar la herramienta subversion de una forma ordenada, y de acuerdo al flujo determinado en los procesos de cambios y versiones, para conseguir los mejores resultados, mitigar errores y bajar los tiempos de interrupción al servicio.

En el instructivo de "*Gestión de versiones con SVN*", se detalla cómo se debe aplicar la herramienta para los casos de inconvenientes por desarrollos paralelos. Uno de los inconvenientes que se puede presentar con este método, es que existan desarrollos de nuevas funcionalidades que no se han concluido todavía. Las razones para ello, entre las principales: prioridad de desarrollo, errores presentados en pruebas, calendarios y cronogramas predefinidos, complejidad del desarrollo, entre otros; estas razones expuestas, provocan que en los ambientes de pruebas y preproducción existan archivos y configuraciones que se utilizan para el desarrollo de más de una funcionalidad. Por lo tanto, al realizar las pruebas en estos ambientes, se puede obtener resultados favorables, pero al desplegar el cambio o versión en producción, se evidencian algunos errores que no fueron previstos, causando inestabilidad, lentitud, bloqueos y hasta la suspensión del servicio.

1. PROCEDIMIENTO PARA LAS COPIAS DE TRABAJO EN LOS AMBIENTES CONTROLADOS

1.1. Crear un repositorio SVN para los ambientes de desarrollo, pruebas y preproducción.

El URL es: http://subversion.pruebaslocales/repos/ambiente/trunk/exe/

En el servidor de SVN, en var/www/svn/:

```
mkdir repos

svnadmin create repos

chown -R apache.apache repos

chcon -R -t httpd_sys_content_t repos
```

1.2. El directorio que se ingresa al control de versiones, será desde /exe/

1.3. Crear el path /exe/ en el directorio trunk creado en el repositorio SVN para el ambiente de producción.

```
svn mkdir compras –m "Crear trunk"
http://subversion.pruebaslocales/repos/ambiente/trunk/exe exe
```

1.4. El servidor de SVN, en la ruta /etc/svn crear el archivo svn-auth-users en el cual constan los usuarios que tienen acceso a este repositorio.

Para crear nuevo archivo y usuario:

```
htpasswd -cm /etc/svn/svn-auth-users usuario
```

Para nuevo usuario en archivo existente:

```
htpasswd -m /etc/svn/svn-auth-users usuario
```

1.5. En los ambientes de preproducción, se requiere crear un repositorio SVN en la carpeta /exe/, ubicado en la ruta /www/html/appweb.

```
svn checkout
http://subversion.pruebaslocales/repos/ambiente/trunk/exe
```

1.6. Para actualizar el ambiente:

```
svn up
```

Si se requiere una revisión anterior:

```
svn up – r
```

Siendo r el número de revisión requerida

1.7. Para registrar los cambios en el repositorio

```
svn commit –m “mensaje referente al cambio solicitado – RFC, orden de trabajo - OT”
```

2. PROCEDIMIENTO PARA LAS COPIAS DE TRABAJO LOCALES (USUARIOS)

2.1. Descarga del repositorio, parcial o total.

```
svn checkout
http://subversion.pruebaslocales/repos/ambiente/trunk/exe
```

Para nuevos desarrollos, se requiere crear una nueva copia de trabajo local dentro del directorio ‘branches’. Este procedimiento aplica para la generación de ramas de desarrollo. Como se revisó en los instructivos anteriores, si re requiere desarrollar cambios paralelos, o que utilicen módulos o configuraciones comunes, se necesita la creación de ramas de desarrollo. Se debe agregar el contenido y realizar la respectiva validación.

```
svn copy http://subversion.pruebaslocales/repos/ambiente/trunk/exe
http://subversion.pruebaslocales/repos/ambiente/branches/BR_cod_RFC_OT –m “Creacion de rama – nombre de cambio – RFC - OT”
```

La creación de la rama deberá registrar con la codificación de los documentos oficiales de solicitud aprobada de cambio y la respectiva orden de trabajo. Estos formatos se encuentran definidos en el proceso de gestión de cambios. No se puede ejecutar un desarrollo, a menos que sea emergente, sin un RFC aprobado por el comité de cambios. Se puede generar una orden de trabajo – OT, cuando sea un cambio emergente y luego formalizar el requerimiento. Una orden de trabajo también se puede generar para cambios de alertas o mensajes en la aplicación, para comunicar errores, disponibilidad e información de nuevas funcionalidades.

2.2. Para unificar los cambios realizados, en el ambiente de desarrollo, se requiere actualizar la copia de trabajo principal, o “trunk”.

`svn log –stop-on-copy :` devuelve el número de revisión cuando fue creada la rama (XX)

`svn up (en trunk):` devuelve el número de la revisión actual del trunk (YY)

```
svn merge –r YY:XX
http://subversion.pruebaslocales/repos/ambiente/branches/BR_cod_RFC_OT
```

Se requiere la comprobación y solución de conflictos, en caso de desplegar cambios con archivos comunes en ramas y tronco.

Nota Importante:

Para la solución de conflictos, se debe tomar en cuenta desde el ambiente o repositorio que se utiliza ese momento, ya que la solución es relativa para la perspectiva del repositorio. Es decir, que se toma en cuenta en dónde está el conflicto y cuál es la acción que se realizará en los cambios. La solución depende de “mi lado” o “del otro lado”. Para aclarar los casos: el primer caso, “mi lado”, se refiere a la copia de trabajo en la que se encuentra el usuario realizando la actualización de cambios. Para el segundo caso, “del otro lado”, se tomará en cuenta al repositorio svn al que apunta la copia de trabajo. Los dos casos se podrían invertir cuando se trabaja desde el repositorio directamente, es decir, “mi lado” será el repositorio y “del otro lado” será la copia de trabajo.

Las soluciones se resolverán con las siguientes opciones por cada fichero modificado:

```
Conflict discovered in 'ejemplo.php'
Select: (p) postpone, (df) diff-full, (e) edit,

        (mc) mine-conflict, (tc) theirs-conflict,

        (s) show all options:
```

No se recomienda el uso de (p) posponer, ya que eso permite que se quede en conflicto el fichero y causaría problemas en la aplicación. Se recomienda el uso de (mc) mine-conflict, (tc) theirs-conflict; los cuales corresponden a la nota expuesta en el párrafo anterior a “mi lado” o “del otro lado”, respectivamente. Si se requiere ver todas las opciones se debe presionar “s”.[27]

[27] (Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato, 2014, pág. /en/1.6/svn.tour.cycle.html)

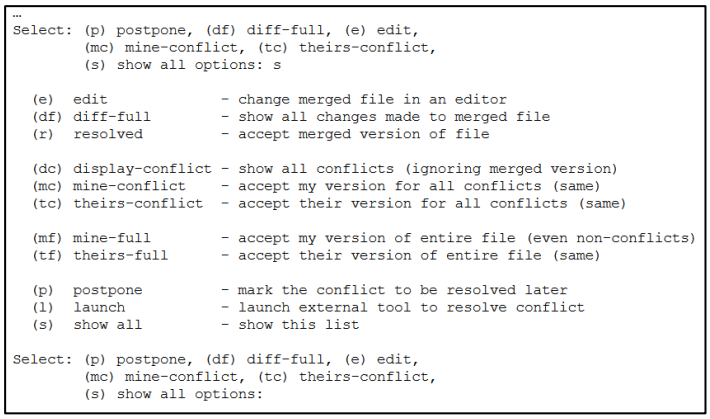

Gráfico 27: Solución de conflictos en SVN

2.3. Desde el momento de la ejecución del procedimiento, cuando exista el despliegue de archivos en las copias de trabajo, se debe ejecutar el siguiente comando para registrar los cambios en el repositorio y relacionar a la orden de trabajo y requerimiento de cambio:

```
svn commit –m “No. de OT - RFC, No. Solicitud de despliegue en Producción”
```

2.4. Luego, en el ambiente controlado, actualizar la copia de trabajo para colocar los cambios registrados en la acción anterior.

```
svn up
```

2.5. IMPORTANTE: Cuando se requiera desplegar en un ambiente controlado un directorio completo y su contenido, se recomienda utilizar el comando svn import; si se realiza una copia completa del directorio, se copiarán también las carpetas ocultas .svn del control de versiones. En caso de realizar la copia normal, se quedará atado el directorio a otro repositorio y no al del amiente controlado. Se requiere utilizar una exportación del directorio y la copia del mismo al path indicado. Luego agregar el directorio al repositorio SVN.

Ejemplo: DESARROLLO a PRUEBAS

a) En DESARROLLO: svn export path_requerido path_temporal
b) Copiar el path_temporal a PRUEBAS
c) Copiar path_temporal a path_requerido en PRUEBAS
d) Establecer los permisos y usuario del directorio y contenido respectivos
e) Agregar el path_requerido al control de versiones:

   En el directorio padre del path_requerido: svn add path_requerido

   svn commit –m “No. OT - RFC, No. Nota solicitud”

   En el path_requerido: svn add *

   svn commit –m “No. OT - RFC, No. Nota solicitud”

f) Indicar números de revisión

2.6. A continuación, se muestra un glosario de comandos variados, que son útiles para el uso de subversión por parte del grupo de desarrollo.

a) svn checkout [URL] –r NUM

Realiza una copia de trabajo local, de una revisión anterior indicada con NUM.
NUM es un número entero que determina un número de revisión al que se requiere actualizar la copia de trabajo, total o parcialmente. Es útil para realizar inmediatamente una reversa en el ambiente por errores o pruebas. Se puede volver inmediatamente a la última revisión omitiendo la opción –r NUM, o a su vez, para control de reversas y actualizaciones utilizar – r HEAD. El parámetro HEAD es la revisión de cabecera o la última revisión realizada en el repositorio.

b) svn commit –m “Mensaje”

Se ejecuta obligatoriamente para registrar los cambios realizados en la copia de trabajo, en el repositorio registrado.

c) svn log –vr NUM

Muestra el log de archivos cambiados en una revisión determinada por NUM.

d) svn log –l LIM

Muestra el log de cambios de las revisiones, con un límite de numérico LIM.

e) svn log –l LIM | grep “usuario”

Muestra el log de cambios de las revisiones realizadas por un determinado “usuario”, con un límite de numérico LIM.

f) svn diff ARCHIVO.php –r XX:YY

Muestra las diferencias de un mismo ARCHIVO (directorio) entre dos números de revisiones XX, YY

g) svn up ARCHIVO –r NUM

Actualiza un ARCHIVO determinado a una revisión anterior. Similar al comando a), actualizando únicamente un archivo.

4. CONCLUSIONES

- La definición de los modelos de gestión para control de cambios y versiones favorecen a mantener y mejorar la calidad del servicio. Los modelos establecidos y las políticas definidas para la gestión de calidad conducen a mantener la garantía, utilidad y disponibilidad del servicio.

- La validación de los procesos, mediante su aplicación en el servicio del portal web del Instituto para Estudios Extranjeros – IFSA.mx, demuestra que las implementaciones de los cambios deberán provocar el menor impacto en el usuario, quien a su vez recibe las mejoras en los servicios.

- Es necesaria la aplicación de cada instancia de autorización en el flujo de los procesos, como por ejemplo en el Comité Asesor de Cambios – CAB; sin la autorización previa, los cambios no podrán ser priorizados, no se considerará el impacto en el servicio y el levantamiento de requerimientos no será óptimo para la producción del servicio.

- Los servicios tecnológicos en la actualidad, son un eje importantísimo en las organizaciones alrededor del mundo aunque el giro del negocio, años atrás, no tenía las intenciones ni la visión de adoptar uno de los activos más complejos, como son los servicios tecnológicos. La disponibilidad y fiabilidad de un servicio tecnológico, como la calidad del mismo, favorecen a la satisfacción de los usuarios o clientes y al crecimiento de las oportunidades tecnológicas.

- El software como servicio, en ocasiones, se pude considerar como una estrategia del giro del negocio para mejorar la atención a sus clientes o usuarios, o con fines de control de transacciones, recopilación de información, actualización de datos, entre otros. En nuestro país, muchas instituciones y empresas públicas, han establecido algunas estrategias para mejorar los servicios públicos, recaudación de impuestos, control de la contratación pública, referencia cruzada de información, registros públicos y varios propósitos similares; los cuales se realizan por medio de servicios web. Estos sitios no son más que sistemas de gran escala que requieren una gestión de servicios tecnológicos apropiada para cumplir y mantener ciertas políticas de disponibilidad y de calidad.

- La perspectiva de la tecnología desde los inicios tempranos de un proyecto, puede ser un aspecto clave en la definición de las estrategias y diseños de nuevos servicios, así como también en las etapas de transición y modificación de servicios existentes, mediante la gestión de cambios y versiones.

- Los procesos propuestos para la gestión de cambios y versiones, están basados en las definiciones generales del ITIL, sin embargo, esta metodología no es un estándar que pueda adoptar una organización, es el personal de las áreas de tecnología quienes lo aplicarán como parte de la definición de actividades que adopte la organización de servicios.

- En cada proceso desarrollado en el presente proyecto, se definen los indicadores claves del éxito, los cuales serán útiles para evaluar la gestión de los cambios ejecutados en los servicios. Estos factores pueden ser agregados o modificados de acuerdo a las necesidades de la organización, sin embargo la definición inicial y mantener un periodo de análisis, será el mecanismo de validar el proceso para determinar si se obtiene la calidad buscada en los servicios.

5. RECOMENDACIONES

- Para la gestión del ciclo de vida del servicio, especialmente para la transición y operación, existen varias herramientas informáticas para realizar la gestión. El uso de una herramienta es de gran utilidad para llevar a cabo las actividades.

- La utilización de las herramientas de control de cambios y versiones dependerán de los objetivos de la organización. Se recomienda realizar un análisis detallado para elegir la mejor herramienta.

- El presente proyecto recopila y define varios documentos que han sido desarrollados de acuerdo a la investigación y aplicación de mejores prácticas y conceptos generales bajo los lineamientos de la búsqueda de la calidad en los servicios tecnológicos. Cada organización puede adoptar

y modificar los procesos de cambios y versiones, así como adoptar otros procesos.

- Los factores clave de éxito se pueden modificar o agregar, sin embargo, se recomienda realizar una evaluación de al menos un semestre, y comparar con la línea base o el mes anterior para obtener información de la calidad obtenida.

- La gestión por procesos, como se ha mencionado anteriormente, no es un estándar para ITIL, sin embargo, las organizaciones podrían optar por una certificación del estándar ISO 9001 para la gestión por procesos. Se recomienda también analizar otros marcos de referencia como COBIT o TOGAF. Aunque las opciones descritas son muy utilizadas, el estado de madurez de la organización respecto a la gestión de sus servicios, determinará cuáles serían las mejores estrategias.

6. REFERENCIAS BIBLIOGRAFICAS

6.1. LIBROS

Deming, E. (1989). *Calidad, Productividad y Competitividad.* Madrid, España: Díaz de Santos S.A.

Hernández S. Roberto, Fernández-Collado Carlos, Baptista L. Pilar. (2008). *Metodología de la Investiación* (4 ed.). México D.F., México: McGraw Hill.

ITPreneurs. (2013). *ITIL Foundation.*

Jeffrey L. Whitten, Lonnie D. Bentley. (2008). *Análisis de Sistemas. Diseño y Métodos* (7 ed.). México D.F., México: Mc Graw Hill.

Kano, N. (1984). *Miryokuteki Hinshitsu to atarimae Hinshitsu (Calidad atractiva y calidad obligatoria).* Japón: Japanese Society for Quality Control.

Kenneth E. Kendall, Julie E. Kendall, Keneth C. Laudon, Jane P. Laudon. (2010). *Sistemas de Información: Fundamentos y Análisis* (6 ed.). Naucalpan de Juárez, México: Pearson.

Muñoz, C. R. (1998). *Cómo Elaborar y Asesorar una Investigación de Tesis* (1 ed.). Naucalpan de Juárez, México: Prentice Hall.

PinkElephant. (2013). *ITIL Liberación, Control y Validación. R 5.4.2.* Ontario, Canadá: Pink Elephant.

Pressman, R. S. (2010). *Ingeniería del Software - Un enfoque práctico.* México DF: McGRAW-HILL.

Sommerville, I. (2011). *Ingeniería de Software* (9 ed.). Naucalpan de Juárez, México, México: Pearson.

6.2. INTERNET

Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato. (24 de JUL de 2014). *http://svnbook.red-bean.com/*. Obtenido de Control de versiones con Subversion: http://svnbook.red-bean.com/en/1.6/svn.tour.cycle.html