Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Código 31105128 — Itinerario de Ingeniería de Software

Proceso de Desarrollo de Software basado en metodologías ágiles y CMMI-DEV

Estudiante: Luis Arturo Rodriguez Treviño

Directora: Dra. María Magdalena Arcilla

Curso 2014 convocatoria Junio.

Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Código 31105128 — Itinerario de Ingeniería de Software

Proceso de Desarrollo de Software basado en metodologías ágiles y CMMI-DEV

Tipo Trabajo B: Trabajo específico propuesto por el alumno

Estudiante: Luis Arturo Rodriguez Treviño

Directora: Dra. María Magdalena Arcilla

Impreso TFdM05_Autor. Autorización de publicación

y difusión del TFdM para fines académicos

Autorización

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

Firma del/los Autor/es

Juan del Rosal, 16
28040, Madrid

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

www.issi.uned.es

RESUMEN DEL TRABAJO

Es irónico como las empresas dedicadas al desarrollo de software de tamaño reducido se enfrentan a un posible fracaso al encontrarse con el éxito. Comúnmente, estas empresas nacen de los esfuerzos colectivos de profesionales del software, que emprenden el camino de trabajar por cuenta propia; ya sea uniendo talentos y comenzando la empresa con un par de socios o iniciando en solitario contratando personal.

Por su tamaño, les es perfectamente posible realizar el desarrollo de software sin procesos establecidos y gracias a su flexibilidad logran proyectos que satisfacen las necesidades de sus clientes; en muchos casos logrando un entendimiento mayor con el cliente que el logrado por las grandes empresas de software con procesos estandarizados. El reto aparece cuando por su éxito, estas empresas se ven en la necesidad de crecer y adoptar sistemas de gestión de procesos y estándares para evitar el caos. Su ritmo de trabajo –comúnmente frenético– les hace imposible sacrificar uno o más recursos productivos –personal de desarrollo– para dedicarlo a la implantación de metodologías y estándares, ya que seguramente esto significaría el incumplimiento de compromisos con los clientes. Esto sumado al poco o nulo acceso a medios de financiamiento que les permitiera invertir en recursos para la implantación de los procesos de gestión; convierten su éxito en una bomba de relojería esperando a que el esfuerzo extra requerido para controlar una organización creciente sin gestiones de proceso cobre su cuota.

Para facilitar la transición de una empresa "artesanal" hacia una empresa con gestión de procesos se crea este documento que puede ser utilizado como una guía para implantar una metodología de desarrollo basada en Scrum; que permite mantener la flexibilidad inherente de empresas pequeñas. Esta guía de implementación basada en Scrum ha sido complementada con los procesos necesarios para cumplir las buenas prácticas especificadas en el modelos CMMI-DEV para lograr un nivel 2 de madurez; con el propósito de pavimentar el camino hacia el futuro de la empresa al permitirle competir por los grandes contratos, los cuales muchas veces suponen la necesidad de contar con certificaciones que avalen los procesos de la empresa.

Esta guía propone el uso de herramientas de gestión de procesos y configuración de Microsoft.

Por último, tomando en cuenta el ambiente económico actual en España, que aumenta la competitividad de la mano de obra Española sobre otros países y la facilidad que dan las nuevas tecnologías en el desarrollo de software distribuido; se incluye en la creación de la guía, técnicas que permiten el desarrollo de software a distancia potenciando las modalidades de subcontratación fuera de las instalaciones del cliente, además del *Nearshore* y *Offshore*.

Como resultado se obtiene una guía para implementar una metodología de desarrollo ágil que cumple las buenas prácticas propuestas por el nivel 2 de madurez de CMMI-DEV, utilizando herramientas de Microsoft para su ejecución y potenciando el teletrabajo como medida de reducción de costes.

Esta guía no es rígida y permite remplazar las herramientas y algunos procesos por otros similares siempre que cumplan la misma función y proporcionen resultados similares.

PALABRAS CLAVES: CMMI-DEV, Agile, Proceso de Desarrollo de Software, Ingeniería de Software, Team Foundation Server, Teletrabajo, Pequeñas Empresas de Desarrollo de Software, Metodologías Agiles, Scrum.

CONTENIDO

Tabla de Figuras

1 INTRODUCCIÓN

1 INTRODUCCIÓN

La gestión de procesos y el uso de metodologías de desarrollo de software documentadas y estructuradas son esenciales para el correcto funcionamiento de las empresas dedicadas al desarrollo de software, entre otras cosas debido a la complejidad inherente a los proyectos de desarrollo de software. Los recursos invertidos en llevar esta gestión pueden ser altos, pero siempre redituables en empresas que por su tamaño se pueden permitir procesos tradicionales de desarrollo de software que conllevan una carga fuerte de recursos invertidos en la documentación.

El problema se presenta en las empresas de desarrollo de software pequeñas, sobre todo en las que se encuentran en crecimiento. Inicialmente estas empresas no siguen procesos estandarizados debido a que por su poco personal les es imposible crear y mantener toda la documentación requerida, que a su vez no resulta indispensable para lograr proyectos exitosos; debido a que comúnmente las empresas pequeñas basan su éxito en un par de personas claves que llevan la compañía a sus espaldas y dirigen al resto de equipo de desarrollo de forma correcta.

El éxito de estas pequeñas empresas se puede convertir fácilmente en su fracaso. Esta ironía se debe a que con el crecimiento y aumento de personal se pierde el control sobre los procesos y el trabajo heroico de su personal clave se vuelve insuficiente para mantener la calidad en los proyectos de la empresa.

Aliviar la dificultad de la transición entre una empresa pequeña "artesanal" y una empresa mediana con estándares y certificaciones, es lo que se pretende lograr con la creación de esta guía. La cual permitirá implantar una metodología de desarrollo ágil que a su vez deje a la compañía en posición de comenzar su certificación en CMMI-DEV.

1.1 OBJETIVOS DEL TFM

La creación de una guía para implantar una metodología de desarrollo ágil que cumpla las buenas prácticas propuestas por el nivel 2 de madurez de CMMI, utilizando herramientas de Microsoft para su ejecución y potenciando el teletrabajo.

Los requisitos de la guía son:

- Que mantenga la flexibilidad de los procesos de desarrollo comunes en empresas pequeñas.
- Ser lo suficiente detallada y completa para poder implantar los procesos y metodologías en una empresa con personal experimentado en el desarrollo de software, mas no así en la gestión de procesos.

- Permitir encaminar a la empresa que lo implanta, en la certificación de CMMI-DEV nivel 2 de madurez; pero nunca siendo la intención, poder hacerlo sin la ayuda de un profesional en certificaciones de CMMI.
- Detallar la guía de implementación hasta el punto de definir el uso de herramientas de software y plantilla para llevar la gestión.
- Ser intercambiable, es decir que los procesos, herramientas y plantillas puedan ser sustituidos por otros similares, pero sin descuidar el cumplimiento de la función definida en la guía para ellos.
- Que permita el desarrollo de software a distancia, para potenciar los contratos de subcontratación fuera de las instalaciones del cliente o de *Nearshore* a empresas Europeas.

1.2 ESTRUCTURA DEL TFM

El TFM se divide en 5 secciones: La introducción, el planteamiento del problema, el estado de la cuestión, la resolución y por último las conclusiones y líneas de trabajo futuro.

Siendo la resolución la sección en donde se concentra el grueso del trabajo, ya que la guía se encuentra descrita en esta sección.

La guía de implantación de la metodología está dividida a su vez en secciones, la primera a modo de introducción y descripción de los elementos que forman la guía; las siguientes 3 secciones son las fases del ciclo de vida del proceso de desarrollo descrito, la cuarta sección enuncia los procesos que no forman parte de la estructura central del ciclo de vida pero que son requeridos para cumplir con las buenas prácticas de CMMI, seguido de las plantillas que apoyan el proceso y por último la documentación de la prueba piloto de la metodología propuesta.

En la sección “4.1 Aspectos generales” se enumeran las prácticas relacionadas con CMMI nivel 2 de madurez y en cada una de las prácticas se hace referencia a la sección del proceso que asegura el cumplimiento con la práctica en cuestión; además del Ciclo de vida de Desarrollo y los elementos bases usados para su creación.

En las siguientes 3 secciones: 4.3, 4.4 y 4.5 que son las relativas a las fases del ciclo de vida, se incluye una sección describiendo el proceso, una sección de las herramientas a utilizar para apoyar la fase y las recomendaciones correspondientes a la fase para el desarrollo distribuido o teletrabajo.

Después de la fase de transición tenemos una sección con las plantillas que servirán para estandarizar la documentación de procesos involucrados y al final los resultados obtenidos de una prueba piloto que se encuentra en proceso en una empresa de desarrollo de software ubicada en Houston, Texas. Incluyendo las conclusiones obtenidas de su aplicación.

2 PLANTEAMIENTO DEL PROBLEMA

2 PLANTEAMIENTO DEL PROBLEMA

Las empresas pequeñas de desarrollo de software se enfrentan al problema de contar con recursos muy limitados que truncan su crecimiento aun y teniendo una buena demanda de sus servicios. Las empresas grandes no tienen el problema de los recursos limitados y por lo tanto pueden planear y utilizar metodologías tradicionales de desarrollo y gestión de procesos meticulosos y documentados; ya que pueden contar con un batallón de personas dedicadas a la gestión de procesos y auditorias.

¿Cómo puede una empresa pequeña "artesanal" hacer la transición a una empresa mediana con procesos de desarrollo estándares?

Si además lo que se busca es hacer esta transición para poder competir en un mercado global, con clientes ubicados en diferentes localidades dentro y fuera de España –principalmente mercados con salarios superiores– aplicando el teletrabajo; entonces lo que se requiere es una metodología que maximice las virtudes del teletrabajo.

Desde hace mucho tiempo existen metodologías de desarrollo de software. Metodologías como el método de cascada, V-Model y RUP son llamadas tradicionales y han sido usadas por compañías de desarrollo de software de forma más o menos exitosa por mucho tiempo. El éxito de los proyectos que desarrollados usando estas metodologías se basa en conocer todos los requerimientos con antelación al inicio del desarrollo, lo cual implica que los cambios durante el ciclo de vida del proyecto se vuelven problemáticos. Además de la cantidad de documentación que es requerida vuelve estas metodologías complicadas de llevar para empresas pequeñas.

Más recientes son las metodologías de desarrollo ágiles, basadas en la idea de un desarrollo incremental e iterativo. Entre estas se encuentran métodos como Crystal, desarrollo de software *Lean, Scrum,* entre otras. Estas metodologías se acercan más a la forma natural de trabajar de las empresas pequeñas, las cuales tienen comúnmente relaciones más cercanas con sus clientes, desarrollos con tiempos comparativamente cortos y poca documentación. Sus principales desventajas serian: la dificultad de aplicarse en proyectos muy grandes y su escaso nivel de documentación lo cual vuelve complicado el trabajo con equipos de desarrollo distribuidos. Por otro lado si lo que se busca son certificaciones para competir por los grandes clientes, las metodologías agiles carecen de los niveles de gestión y documentación requeridos.

Aunque existen metodologías y guías de buenas prácticas que permiten solventar algunos de estos requerimientos de forma individualizada, no existe una metodología que a la vez de ser ágil y fácil de gestionar, tenga el nivel de estandarización y documentación necesario lograr certificaciones, y lo más importante que sea fácil de implantar en empresas pequeñas con poco personal. Lo que se busca es una guía que permita aliviar la dificultad de implementar la gestión de procesos y metodologías de software en empresas pequeñas en crecimiento que favorezca

el teletrabajo y el uso de herramientas de software para reducir el tiempo necesario para su gestión.

3 ESTADO DE LA CUESTIÓN

3 ESTADO DE LA CUESTIÓN

3.1 TRABAJOS RELACIONADOS.

Existe no poca literatura sobre el tema de la gestión de procesos y las metodologías de desarrollo ágiles, pero ninguna que resuelva la cuestión con el nivel y el alcance planteado en este documento.

Las primeros documentos a los que se tendría que hacer referencia son los que definen el modelo CMMI-DEV y la Metodología de Desarrollo Scrum, *CMMI for Development, Version 1.3* (CMMI Product Team, 2010) y *The scrum guide* (Schwaber & Sutherland, 2011) respectivamente.

Encontramos una cantidad respetable de trabajos que abordan el tema del uso de CMMI-DEV con Metodologías Ágiles, entre ellas Scrum; entre éstos encontramos: *CMMI® or Agile: Why Not Embrace Both!* (Glazer, Dalton, Anderson, Konrad, & Shrum, 2008), *Integrating CMMI and agile development: case studies and proven techniques for faster performance improvement* (McMahon, 2010), *Implementing Srum (Agile) and CMMI® toghether* (Potter & Sakry, 2009) y *A Scrum-based approach to CMMI maturity level 2 in Web* (Torrecilla, Escalona, & Mejias, 2012) . Si bien algunas dan pistas de cómo se pudiera realizar este proceso que integrara los dos elementos, ninguna de ellas profundizan en el cómo hacerlo con el detalle o alcance suficiente.

También existen numerosos estudios y bibliografía acerca del uso de Metodologías Ágiles y Trabajo Distribuido, entre ellos: *Agile Software Development with Distributed Teams: Staying Agile in a Global World* (Eckstein, 2013), *Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum* (Larman & Vodde, 2010) y *Distributed Agile, DH2A: The Proven Agile Software Development Approach and Toolkit for Geographically Dispersed Teams* (Venkatesh, 2011).

Por último sobre el uso de herramientas de Microsoft, particularmente Team Foundation Server para gestionar el proceso de desarrollo encontramos: *Visual Studio Team Foundation Server 2012: Adopting Agile Software Practices* (Guckenheimer & Loje, 2012), *Professional Scrum Development with Microsoft® Visual Studio® 2012* (Hundhausen, 2012), *Pro Team Foundation Service* (Olausson, Rossberg, Ehn, & Sköld, 2013) y *Professional Scrum with Team Foundation Server 2010* (Resnick, Bjork, & Maza, 2011).

3.2 METODOLOGÍAS ÁGILES DE DESARROLLO

Las metodologías se basan en los principios mencionados en el manifiesto para el Desarrollo ágil, el cual dice:

Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:

Individuos e interacciones *sobre procesos y herramientas*

Software funcionando *sobre documentación extensiva*

Colaboración con el cliente *sobre negociación contractual*

Respuesta ante el cambio *sobre seguir un plan*

Aunque valoramos los elementos de la derecha, valoramos más los de la izquierda.

(Agile Alliance, 2001)

Principios del Manifiesto Ágil son los siguientes:

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.
  (Agile Alliance, 2001)

Scrum es una implementación de metodología de desarrollo ágil, la cual permite a las profesionales del desarrollo de software trabajar sobre problemas complejos y cambiantes; a la vez de entregar productos de forma productiva y creativa con el más alto valor posible.

La definición de *Scrum* se compone de los roles *Scrum*, los eventos, los artefactos y las reglas para unir todo esto. *Scrum* emplea un acercamiento iterativo e incremental para optimizar la previsibilidad y el control de riesgo.

Los tres pilares sobre los que se soporta la implementación empírica de los procesos de control son:

Transparencia: Aspectos del proceso deben der ser visibles para todos aquellos responsables del resultado.

Inspección: Los usuarios *Scrum* deben de inspeccionar los artefactos y el progreso hacia las metas del Sprint para detectar cualquier variación no deseada.

Adaptación: Si una inspección da como resultado una desviación fuera de los límites aceptables y de esta forma el producto resultante sea inaceptable, el proceso o material que está siendo procesado debe de ser ajustado.

El ciclo de vida simplificado de un proyecto *Scrum* sería el siguiente: El propietario de producto crea el *Backlog* con todos los elementos que son requeridos para el funcionamiento del producto deseado. Estos elementos son descritos en forma de historias que serán descompuestos en tareas durante el ciclo de vida. Todos los miembros del equipo *Scrum* se reúnen en lo que es llamado junta de Planificación del Sprint, en esta junta se deciden que elementos del *Backlog* serán comprometidos por el equipo para su construcción durante el *Sprint*, estos elementos son descompuestos en tareas conforme se van descubriendo los elementos necesarios para satisfacer la elaboración del elemento. Diariamente los desarrolladores trabajan para crear los elementos del *Incremento* y acuden a las juntas llamadas *Scrum Diario* en el cual se resuelven las dudas y los impedimentos que existen para poder continuar con el trabajo, estas juntas deben de durar alrededor de 15 min.

Al final del *Scrum* se realizan las juntas de revisión y de retrospectiva. En las juntas de revisión el propietario del producto evalúa los elementos del producto realizado y en la junta de retrospectiva se evalúa el trabajo realizado por el equipo de desarrollo para optimizar el proceso.

El producto obtenido al final de cada *Sprint* es llamado *Incremento*, el cual deberá tener la calidad y funcionalidad para ser usado. Si se decide que será usado, se implementa en producción para su uso.

Inmediatamente después de esto se regresa a la planificación del siguiente *Sprint* y se inicia otro ciclo para crear un nuevo *Incremento*.

Se puede encontrar más información sobre el marco de desarrollo *Scrum* en *The scrum guide* (Schwaber & Sutherland, 2011).

3.3 CMMI-DEV

Integración de Modelos de Madurez de Capacidades o Capability Maturity Model Integration (CMMI) es un modelo de formación y evaluación de procesos de mejora y servicios gestionado y comercializado para la Universidad de Carnegie Mellon y requerido por muchas de las organizaciones del departamento de defensa del gobierno de los Estados Unidos de América.

CMMI tiene como propósito ayudar a las empresas a mejorar sus procesos de desarrollo y mantenimiento de productos y servicios. Existen 3 constelaciones que se sostienen en el marco del modelo v1.3: desarrollo, adquisición y servicios. Una constelación es una colección de componentes CMMI que incluyen un modelo, sus materiales de formación y documentos de evaluación pertenecientes a un dominio en particular. La constelación de interés para esta guía es CMMI para desarrollo –CMMI-DEV.

CMMI para Desarrollo es un modelo de referencia que abarca las actividades para desarrollar tanto productos como servicios. CMMI para Desarrollo es utilizado por organizaciones de muchos sectores, incluyendo aeroespacial, banca, hardware, software, defensa, automoción y telecomunicaciones.

Entre las practicas cubiertas por CMMI para Desarrollo se encuentran: la gestión de proyectos, la gestión de procesos, la ingeniería de sistemas, la ingeniería de hardware, la ingeniería de software y otros procesos de soporte utilizados en el desarrollo y mantenimiento.

Cabe notar que CMMI deja a interpretación el modelo e invita a su adecuación a la cada organización como así lo menciona en su guía: *"Use su juicio profesional y el sentido común para interpretar el modelo en su organización. Es decir, aunque las áreas de proceso descritas en este modelo representen comportamientos considerados buenas prácticas para la mayoría de los usuarios, las áreas de proceso y las prácticas se deberían interpretar usando un conocimiento profundo de CMMI-DEV, las restricciones de su organización y su entorno de negocio"* (CMMI Product Team, 2010)

CMMI-DEV contiene 22 áreas de proceso, de las cuales 16 son áreas de proceso principales, 1 es una área de proceso compartida y 5 son áreas de proceso específicas. Las Áreas son las siguientes:

- Análisis Causal y Resolución (CAR).
- Gestión de Configuración (CM).
- Análisis de Decisiones y Resolución (DAR).
- Gestión Integrada del Proyecto (IPM).
- Medición y Análisis (MA).
- Definición de Procesos de la Organización (OPD).
- Enfoque en Procesos de la Organización (OPF).
- Gestión del Rendimiento de la Organización (OPM).
- Rendimiento de Procesos de la Organización (OPP).

- Formación en la Organización (OT).
- Integración del Producto (PI).
- Monitorización y Control del Proyecto (PMC).
- Planificación del Proyecto (PP).
- Aseguramiento de la Calidad del Proceso y del Producto (PPQA).
- Gestión Cuantitativa del Proyecto (QPM).
- Desarrollo de Requisitos (RD).
- Gestión de Requisitos (REQM).
- Gestión de Riesgos (RSKM).
- Gestión de Acuerdos con Proveedores (SAM).
- Solución Técnica (TS).
- Validación (VAL).
- Verificación (VER).

CMMI-DEV define 4 niveles de capacidad que son: 0. Incompleto, 1. Realizado, 2. Gestionado y 3. Definido. La capacidad se refiere al cumplimiento de la mejora de procesos de una organización en áreas de proceso individuales.

Consta de 5 niveles de madurez, los cuales definen la consecución de la mejora de procesos de la organización en un conjunto de áreas que cubren cada uno de los niveles de madurez. Estos niveles son los siguientes: Nivel 1: Inicial, Nivel 2: Gestionado, Nivel 3: Definido, Nivel 4: Gestionado Cuantitativamente, Nivel 5: En optimización.

El nivel de madurez al que se enfoca este documento es el Nivel 2, Gestionado: Un proceso gestionado es un proceso desarrollado que es planeado y ejecutado de acuerdo a las políticas de la empresa, que utiliza personal que tiene los recursos adecuados para producir resultados controlados, involucra al personal interesado, es monitorizado, controlado y revisado; además de evaluado para asegurar que se apega a la descripción del proceso. La disciplina de proceso reflejado por el nivel 2 ayuda a asegurar que las prácticas existentes sean seguidas en los tiempos de estrés (CMMI Product Team, 2010).

Las áreas de proceso a tratar en este documento serán: Gestión de Configuración (CM), Medición y Análisis (MA), Monitorización y Control del Proyecto (PMC), Planificación del Proyecto (PP), Aseguramiento de la Calidad del Proceso y del Producto (PPQA), Gestión de Requisitos (REQM) y Gestión de Acuerdos con Proveedores (SAM).

Para más información sobre CMMI-DEV se puede recurrir a la guía CMMI para Desarrollo, Versión 1.3 (CMMI Product Team, 2010).

3.4 POR QUÉ USAR CMMI Y METODOLOGÍAS ÁGILES

Tomando en cuenta las siguientes diferencias entre CMMI y Agile se llega a la conclusión que son dos elementos que pueden ser usados de forma simultánea.

- CMMI es un modelo que establece metas a alcanzar, pero no es un estándar o metodología que explique cómo alcanzarlos.
- CMMI define áreas de procesos pero no procesos, con el único requerimiento de que las metas de cada área deben de ser satisfechas.
- Las técnicas ágiles no carecen de disciplina o control, sino por el contrario para entregar resultados en tiempos cortos es necesario tener gran disciplina.
- Las tecnologías ágiles pueden ser escaladas.

Como se menciona en *CMMI® or Agile: Why Not Embrace Both!* (Glazer, Dalton, Anderson, Konrad, & Shrum, 2008), CMMI y los métodos ágiles son compatibles. A nivel de proyecto CMMI es un modelo que se enfoca en lo que los proyectos hacen en un nivel alto de abstracción, no en que metodología de desarrollo utilizar; mientras que los métodos ágiles se enfocan en como los proyectos desarrollan los productos, por lo tanto CMMI y los métodos ágiles puede coexistir.

CMMI y los métodos ágiles se complementan entre sí, creando sinergias que benefician a las empresas que los utilizan. Los métodos ágiles proveen un "¿Cómo?" que funcionan bien principalmente con equipos pequeños y que se encuentran en la misma ubicación, cosa que no hacen las buenas prácticas de CMMI. CMMI aporta las prácticas de ingeniería de software que permiten el uso de los métodos ágiles en proyectos grandes. CMMI también aporta la gestión del proceso y las prácticas que ayudan a implementar, sostener y mejorar continuamente la implementación de los métodos ágiles en cualquier organización.

3.5 RIESGOS Y BENEFICIOS DEL TRABAJO DISTRIBUIDO CON METODOLOGÍAS ÁGILES.

Venkatesh (2011) menciona en su libro *Distributed Agile, DH2A: The Proven Agile Software Development Approach and Toolkit for Geographically Dispersed Teams,* que si bien existen riesgos y sobrecargas en comparación con los equipos ágiles centralizados, hay beneficios relacionados con el trabajo distribuido. Entre éstos encontramos:

Reducción de Costes: Los costes pueden ser optimizados teniendo la mayoría del equipo de trabajo en centros ubicados en áreas de bajo coste. Venkatesh calcula una reducción del 40%-50% en el ahorro a largo plazo vs equipos centralizados en el país demandante del servicio. (Beneficio a destacar en el caso de España)

Bolsa de Talento Global: Permite el acceso a una base más amplia de talento para el desarrollo. Esta flexibilidad de obtener el talento necesario en el tiempo correcto permite a la organización

cumplir sus demandas de personal, aumentando o disminuyendo su plantilla de forma rápida; permitiendo soluciones rentables.

Aumento en la productividad: En la experiencia de Venkatesh ha observado que en el largo plazo la productividad puede ser comparable con un equipo centralizado y en algunos casos superan la productividad de equipos centralizados.

de Calidad: Comparados con el desarrollo en cascada, los métodos ágiles ya sean centralizados o distribuidos han demostrado resultados superiores en coste de calidad. Esto se debe a que el método de cascada al ser un desarrollo secuencial pospone la resolución de defectos para las últimas fases del ciclo de vida y esto supone costes más elevados de calidad. (Venkatesh, 2011)

4. RESOLUCIÓN: GUÍA PARA IMPLANTAR UN PROCESO DESARROLLO ÁGIL CUMPLIENDO EL NIVEL 2 DE MADUREZ CMMI-DEV

4 RESOLUCIÓN: GUÍA PARA IMPLANTAR UN PROCESO DESARROLLO ÁGIL CUMPLIENDO EL NIVEL 2 DE MADUREZ CMMI-DEV

A continuación se detallan los elementos usados de base para la generación de la guía de implantación de un proceso de desarrollo ágil, basado en *Scrum* y que cumple con las prácticas establecidas por el modelo CMMI-DEV nivel 2 de madurez. Así como su ciclo de vida y el listado de las prácticas específicas de CMMI-DEV con la referencia hacia el apartado de la metodología de desarrollo en donde su cumplimiento es alcanzado.

4.1 ASPECTOS GENERALES

4.1.1 Elementos Base

La guía esta creada a partir del modelo CMMI-DEV, metodologías de desarrollo ágiles (principalmente *Scrum*) y el concepto de equipos distribuidos. Está compuesta por la definición del proceso de desarrollo de software, una esquema con las referencias entre las practicas CMMI-DEV y las metodología ágiles, plantillas para apoyar la documentación de los procesos, el uso de herramientas para reducir el tiempo de gestión y recomendaciones para los equipos distribuidos. La siguiente figura nos muestra la composición de la guía.

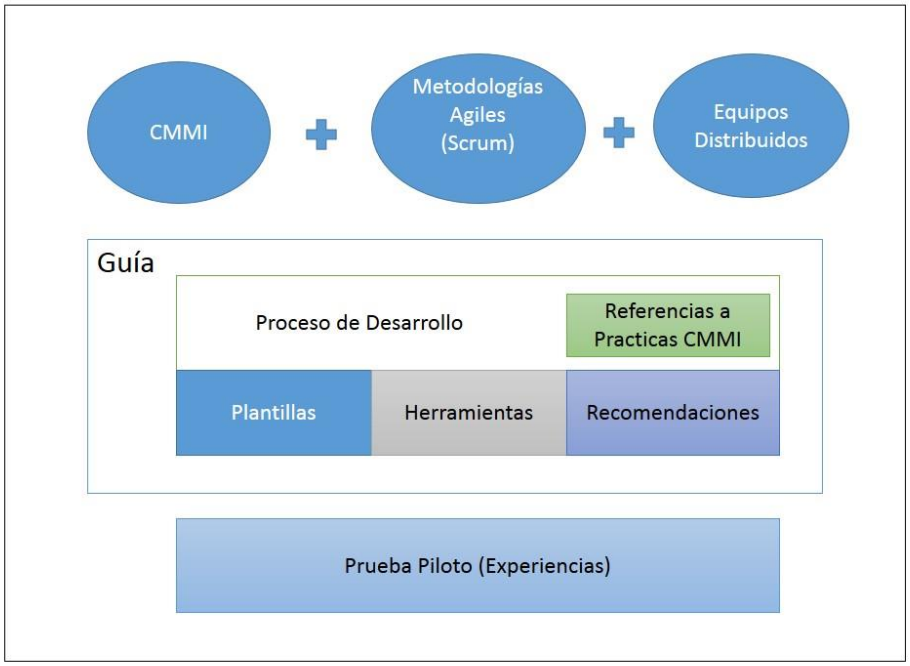

Figura 1 Elementos de los que está compuesto la Guía de Desarrollo

También cuenta con las experiencias obtenidas de una prueba pilotos realizada en un proyecto real de una compañía que cumple el perfil de empresa mencionado en este documento.

4.1.2 Estructura de la guía

Al integrar el modelo CMMI-DEV y metodologías ágiles se hace posible leer la guía en 2 direcciones, dependiendo de la intención del lector. Ya sea implantar los procesos, para lo cual una lectura secuencial del documento permitiría una fácil conceptualización de los procesos a realizar para implantar la metodología. O si por lo contrario se busca cumplir las prácticas para la certificación CMMI-DEV lo recomendable seria leerla utilizando como guía las referencias encontradas en el apartado "4.1.3 Relación entre modelo CMMI y Scrum".

Las direcciones de lectura son las siguientes:

1. Partiendo del proceso de desarrollo ágil, la cual contiene anotaciones que permiten navegar a las buenas prácticas de CMMI. El formato de las anotaciones es el siguiente:

[XXXX SP 0.0.0] en donde XXXX es el Área, SP define que es una práctica específica y 0.0.0 es el número de la práctica que se cumple; de tal forma que [CM SP 1.1.6] nos dice que este párrafo permite el cumplimiento de la sub-práctica "*1.1.6. Especificar las relaciones entre los elementos de configuración"*, que pertenece a la práctica específica "*SP 1.1 Identificar los Elementos de Configuración"* que a su vez es parte de la meta *"SG 1: Establecer las Líneas Base"* y se encuentra dentro del área "CC –Gestión de Configuración".

Ejemplo de anotaciones dentro del texto:

TFS permite la documentación de las relaciones entre los elementos de configuración, entre los que se encuentra la relación de padre/hijo, predecesor/sucesor, Caso de Prueba, de que comparten los mismos pasos, o simplemente una relación diferente mediante la posibilidad de detallarlo con texto libre. Se deberá de usar estas relaciones cuando sean visualizadas en los elementos de configuración, aun y no ser de uso obligatorio por la herramienta. [CM SP 1.1.6] La Herramienta registra de forma automática el usuario que está efectuando el ingreso del elemento de gestión. [CM SP 1.1.5]

2. La segunda opción es en sentido contrario, partiendo de las buenas prácticas de CMMI y navegando a la sección de la metodología de desarrollo en donde se cumple la práctica. Del lado izquierdo se enuncia la práctica específica a cumplir y del lado derecho la sección de la metodología definida en donde se cumple la práctica.
   NOTA: Si la guía es usada en formato electrónico, la referencia a la sección es un hipervínculo que permite ir directamente a la sección con solo presionarlo con el puntero del ratón.

Ej.:

CC –Gestión de Configuración	**Sección***

SG 1: Establecer las Líneas Base	
SP 1.1 Identificar los Elementos de Configuración	
1.1.1. Seleccionar los elementos de configuración y los productos de trabajo que los componen, basándose en criterios documentados.	4.3.9.1
1.1.2. Asignar identificadores únicos a los elementos de configuración.	4.3.9.1
1.1.3. Especificar las características importantes (autor, tipo de archivo, etc.) de cada elemento de configuración.	4.3.9.1
1.1.4. Especificar cuándo se pone bajo gestión de configuración cada elemento de configuración.	4.3.9.1
1.1.5. Identificar al propietario responsable de cada elemento de configuración.	4.3.9.1
1.1.6. Especificar las relaciones entre los elementos de configuración.	4.3.9.1

4.1.3 Relación entre modelo CMMI y Scrum

No es el propósito de este documento el explicar las metas de CMMI Nivel 2, sino el identificar como lograr el cumplimiento de estos objetivos. Para más información sobre CMMI se puede consultar "*CMMI for Development, versión 1.3*" (CMMI Product Team, 2010).

En el siguiente apartado se encuentra la referencia entre las prácticas de CMMI Nivel 2 de Madurez- Gestionado y el proceso de desarrollo elaborado a partir de Metodologías Ágiles y *Scrum*.

4.1.3.1 Planificación de Proyecto (PP)

PP -Planificación del Proyecto	**Sección***
SG 1 Establecer estimaciones	
SP 1.1 Determinar el alcance del proyecto	
1.1.1 Desarrollar una estructura de descomposición del trabajo (WBS)	4.3.5.1, 4.7.1(1)
1.1.2. Definir los paquetes de trabajo con el detalle suficiente para que se puedan especificar las estimaciones de las tareas, las responsabilidades y el calendario del proyecto.	4.3.5.1, 4.7.1(1)
1.1.3. Identificar los productos y los componentes del producto a adquirir externamente.	4.3.5.1, 4.7.1(1)
1.1.4. Identificar los productos de trabajo a reutilizar.	4.3.5.1, 4.7.1(1)
SP 1.2 Establecer estimados de Productos de Trabajo y Atributos de Trabajos	
1.2.1. Determinar la aproximación técnica para el proyecto.	4.3.6

1.2.2. Usar métodos apropiados para determinar el tamaño de los productos de trabajo, y el esfuerzo de las tareas para estimar los requisitos de recursos.	4.3.5.2
1.2.3. Estimar el esfuerzo de las tareas.	4.3.5.2
1.2.4. Estimar el tamaño de los productos de trabajo.	4.3.5.2
SP 1.3 Definir el ciclo de vida del proyecto.	4.1.3.7, 4.7.1(3)
SP 1.4 Estimar Esfuerzo y Costes	
1. Recopilar modelos o datos históricos a utilizar para transformar los tamaños de los productos de trabajo y de las tareas en estimaciones de horas de trabajo y de coste.	4.3.5.2
2. Incluir las necesidades de infraestructura de soporte al estimar el esfuerzo y el coste.	4.3.5.2, 4.3.9
3. Realizar estimaciones de esfuerzo y coste.	4.3.5.2
4. Estima el esfuerzo y el coste utilizando modelos, datos históricos o una combinación de ambos.	4.3.5.2
SG 2 Desarrollar un plan de proyecto	
SP 2.1 Establecer presupuesto y Calendario	
1. Identificar los hitos principales.	4.3.5.3
2. Identificar los supuestos del calendario.	4.3.5.3
3. Identificar las restricciones del proyecto como por ejemplo de recursos, duración de tareas.	4.3.5.3
4. Identificar las dependencias entre las tareas.	4.3.5.3
5. Establecer y mantener el presupuesto y el calendario del proyecto.	4.3.5.3
6. Establecer los criterios para determinar qué constituye una desviación significativa del plan de proyecto.	4.3.5.3
SP 2.2 Identificar los Riesgos del Proyecto	
1. Identificar los riesgos.	4.3.8
2. Documentar los riesgos.	4.3.8
3. Revisar y obtener el acuerdo con las partes interesadas relevantes sobre la completitud y exactitud de los riesgos documentados.	4.3.8

4. Modificar los riesgos según sea apropiado.	4.3.8, 4.4.5.3
SP 2.3 Planificar la gestión de Datos	
1. Establecer los requisitos y los procedimientos para asegurar la privacidad y la seguridad de los datos.	4.3.9.2
2. Establecer un mecanismo para almacenar los datos y acceder a los datos almacenados.	4.3.9.2
3. Determinar los datos del proyecto que serán identificados, recogidos y distribuidos.	4.3.9.2
4. Determinar los requisitos para proporcionar el acceso a los datos y su distribución a las partes interesadas relevantes.	4.3.9.2
5. Decir qué datos y planes del proyecto requieren control de versión u otros niveles de control de configuración y establecer los mecanismos para asegurar que los datos del proyecto se controlan.	4.3.9.2
SP 2.4 Planificar los Recursos del Proyecto	
1. Determinar los requisitos del proceso (a usar en el proyecto).	4.2
2. Determinar los requisitos de comunicación.	4.4.5.1, 4.4.5.3, 4.4.5.4, 4.5.4
3. Determinar los requisitos de personal.	4.3.2, 4.4.3, 4.7.1(2)
4. Determinar los requisitos de instalaciones, equipamiento y componentes.	4.3.9
5. Determinar otros requisitos de recursos continuos (consumibles, propiedad intelectual, transporte,…)	4.3.9
SP 2.5 Planificar el Conocimiento y las Habilidades Necesarias	
1. Identificar el conocimiento y las habilidades necesarios para realizar el proyecto.	4.3.10
2. Evaluar el conocimiento y las habilidades disponibles.	4.3.10
3. Seleccionar mecanismos (formación interna, externa, nuevas contrataciones, etc.) para proporcionar el conocimiento y habilidades necesarios.	4.3.10
4. Incorporar los mecanismos seleccionados en el plan de proyecto.	4.3.10

SP 2.6 Planificar la Involucración de las Partes Interesadas	4.3.2, 4.7.4
SP 2.7 Establecer el Plan de Proyecto	4.3.5.3
SG 3: Obtener el Compromiso con el Plan	
SP 3.1 Revisar los Planes que afectan al Proyecto	4.3.5.3
SP 3.2 Conciliar los Niveles de Trabajo y de Recursos	4.3.7
SP 3.3 Obtener el Compromiso con el Plan	
1. Identificar el soporte necesario y se negocian los compromisos con las partes interesadas relevantes.	4.4.5.2
2. Documentar todos los compromisos de la organización asegurando el nivel apropiado de firmantes.	4.4.5.2
3. Revisar los compromisos internos con la alta dirección según sea apropiado.	4.4.5.2
4. Revisar los compromisos externos con la alta dirección según sea apropiado.	4.4.5.2

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.2 Monitorización y Control del Proyecto (PMC)

PMC –Monitorización y Control del Proyecto	**Sección***
SG 1: Monitorizar el Proyecto frente al Plan	
SP 1.1 Monitorizar los Parámetros de Planificación del Proyecto	
1. Monitorizar el progreso frente al calendario.	4.4.5.4
2. Monitorizar los costes y el esfuerzo empleado en el proyecto.	4.4.5.4
3. Monitorizar el tamaño de los productos de trabajo.	4.4.5.2
4. Monitorizar el esfuerzo de las tareas.	4.4.5.4
5. Monitorizar los recursos proporcionados y los utilizados.	4.4.5.4
6. Monitorizar el conocimiento y las habilidades del personal del proyecto.	4.4.5.6

7. Documentar las desviaciones significativas en los parámetros de planificación del proyecto.	4.4.5.4
SP 1.2 Monitorizar los Compromisos	
1. Revisar regularmente los compromisos externos e internos.	4.4.5.3, 4.4.5.4
2. Identificar los compromisos que no se han cumplido o que están en riesgo significativo de no cumplirse.	4.4.5.4
3. Documentar los resultados de las revisiones de los compromisos.	4.4.5.4
SP 1.3 Monitorizar los Riesgos del Proyecto	
1. Revisar periódicamente la documentación de los riesgos en el contexto del estado y de las circunstancias actuales del proyecto.	4.4.5.3
2. Modificar la documentación de riesgos, a medida que se va disponiendo de información adicional, para incorporar los cambios.	4.4.5.3
3. Comunicar el estado de los riesgos a las partes interesadas relevantes.	4.4.5.3
SP 1.4 Monitorizar la Gestión de los Datos	
1. Revisar periódicamente las actividades de gestión de los datos frente a su descripción en el plan de proyecto.	4.4.5.4
2. Identificar y documentar las cuestiones significativas y sus impactos.	4.4.5.4
3. Documentar los resultados de las desviaciones de las actividades de gestión de los datos.	4.4.5.4
SP 1.5 Monitorizar la Involucración de las Partes Interesadas	
1. Revisa periódicamente el estado de la involucración de las partes interesadas.	4.4.5.1, 4.4.5.4
2. Identificar y documentan las cuestiones significativas y sus impactos.	4.4.5.1, 4.4.5.4
3. Documentar los resultados de las revisiones del estado de la involucración de las partes interesadas.	4.4.5.1, 4.4.5.4
SP 1.6 Llevar a cabo Revisiones de Progreso	
1. Comunicar regularmente a las partes interesadas relevantes el estado de los productos de trabajo y de las actividades asignadas.	4.4.5.3, 4.4.5.4
2. Revisar los resultados de la recogida y del análisis de las medidas para controlar el proyecto.	4.4.5.6
3. Identificar y documentar las cuestiones y las desviaciones significativas frente al Plan.	4.4.5.4

4. Documentar las peticiones de cambio y los problemas identificados en los productos de trabajo y en los procesos.	4.4.5.4, 4.7.1(4)
5. Documentar los resultados de las revisiones.	4.4.5.4, 4.7.1(4)
6. Seguir las peticiones de cambio y los informes de problemas hasta su cierre.	4.4.5.4
SP 1.7 Llevar a cabo Revisiones de Hitos	
1. Llevar a cabo revisiones de hitos con las partes interesadas relevantes en puntos significativos del calendario del proyecto (tales como la finalización de las fases seleccionadas).	4.4.5.4
2. Revisar los compromisos, el plan, el estado y los riesgos del proyecto.	4.4.5.4
3. Identificar y documentar las cuestiones significativas y sus impactos.	4.4.5.4
4. Documentar los resultados de la revisión, los elementos de acción y las decisiones.	4.4.5.4, 4.7.1(4)
5. Seguir los elementos de acción hasta su cierre.	4.4.5.4
SG 2: Gestionar las Acciones Correctivas hasta su cierre	
SP 2.1 Analizar las cuestiones	
1. Recopilar las cuestiones para su análisis.	4.4.5.3, 4.4.5.4, 4.7.1(4)
2. Analizar las cuestiones para determinar la necesidad de acciones correctivas.	4.4.5.3, 4.4.5.4
SP 2.2 Llevar a cabo las Acciones Correctivas	
1. Determinar y documentan las acciones apropiadas necesarias para tratar las cuestiones identificadas.	4.4.5.3, 4.4.5.4, 4.7.1(4)
2. Revisar y se obtiene el acuerdo con las partes interesadas relevantes sobre las acciones a tomar.	4.4.5.3, 4.4.5.1
3. Negociar los cambios a los compromisos externos e internos.	4.4.5.4
SP 2.3 Gestionar las Acciones Correctivas	
1. Monitorizar las acciones correctivas hasta su finalización.	4.4.5.1
2. Analizar los resultados de las acciones correctivas para determinar su eficacia.	4.4.5.6
3. Determinar y documentar las acciones apropiadas para corregir las desviaciones producidas en los resultados planificados debido a las acciones correctivas realizadas.	4.4.5.6, 4.7.1(4)

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.3 Gestión de Configuración (CM)

CM –Gestión de Configuración	**Sección***
SG 1: Establecer las Líneas Base	
SP 1.1 Identificar los Elementos de Configuración	
1.1.1. Seleccionar los elementos de configuración y los productos de trabajo que los componen, basándose en criterios documentados.	4.3.9.1
1.1.2. Asignar identificadores únicos a los elementos de configuración.	4.3.9.1
1.1.3. Especificar las características importantes (autor, tipo de archivo, etc.) de cada elemento de configuración.	4.3.9.1
1.1.4. Especificar cuándo se pone bajo gestión de configuración cada elemento de configuración.	4.3.9.1
1.1.5. Identificar al propietario responsable de cada elemento de configuración.	4.3.9.1
1.1.6. Especificar las relaciones entre los elementos de configuración.	4.3.9.1
SP 1.2 Establecer un Sistema de Gestión de Configuración	
1.2.1. Establecer un mecanismo para gestionar múltiples niveles de control en la gestión de la configuración.	4.3.13.1
1.2.2. Proporcionar control de acceso para asegurar el acceso autorizado al sistema de gestión de configuración.	4.3.13.1
1.2.3. Almacenar y recuperar los elementos de configuración en un sistema de gestión de configuración.	4.3.13.1
1.2.4. Compartir y transferir los elementos de configuración entre los niveles de control en el sistema de gestión de configuración.	4.3.13.1
1.2.5. Almacenan y recuperan versiones archivadas de elementos de configuración	4.3.13.1
1.2.6. Almacenar, actualizar y recuperar los registros de gestión de configuración	4.3.13.1
1.2.7. Crear informes de gestión de configuración a partir del sistema de gestión de configuración.	4.3.13.1
1.2.8. Preservar los contenidos del sistema de gestión de configuración.	4.3.13.1

1.2.9. Modificar la estructura de gestión de configuración según sea necesario.	4.4.5.6
SP 1.3 Crear o Liberar las Líneas Base	
1.3.1. Obtener la autorización del comité de control de configuración (CCC) antes de crear o liberar las líneas base de elementos de configuración.	4.3.9.1
1.3.2. Crear o liberan líneas base sólo desde los elementos de configuración en el sistema de gestión de configuración.	4.3.13.1
1.3.3. Documentar el conjunto de elementos de configuración que están contenidos en una línea base.	4.3.13.1
1.3.4. Poner a disposición el conjunto actual de líneas base.	4.3.13.1
SG 2: Seguir y Controlar los Cambios	
SP 2.1 Seguir las Peticiones de Cambio	
2.1.1. Iniciar y registran las peticiones de cambio en la base de datos de peticiones de cambio.	4.4.3.4
2.1.2. Analizar el impacto de los cambios y de las correcciones propuestas en las peticiones de cambio.	4.4.5.1
2.1.3. Clasificar y priorizar las peticiones de cambio.	4.4.5.1
2.1.4. Revisar las peticiones de cambio a tratar en la siguiente línea base con las partes interesadas relevantes y se llega a un acuerdo.	4.4.5.1
2.1.5. Seguir el estado de las peticiones de cambio hasta su cierre.	4.4.5.4
SP 2.2 Controlar los Elementos de Configuración	
2.2.1. Controlar los cambios a los elementos de configuración a lo largo del ciclo de vida del producto o servicio.	4.3.13.1
2.2.2. Obtener la autorización apropiada antes de que los elementos de configuración modificados sean introducidos en el sistema de gestión de configuración.	4.4.5.4
2.2.3. Realizar actividades de check-in y check-out de los elementos de configuración en el sistema de gestión de configuración para la incorporación de los cambios, de forma que se mantenga la exactitud y la integridad de los elementos de configuración.	4.3.13.1
2.2.4. Realizar revisiones para asegurar que los cambios no hayan causado efectos no deseados en las líneas base (p.ej. asegurar que los cambios no hayan comprometido la seguridad o la protección del sistema).	4.4.8.1, 4.4.8.2
2.2.5. Registrar los cambios a los elementos de configuración y las razones de los mismos según sea apropiado.	4.3.13.1

SG 3: Establecer la Integridad	
SP 3.1 Establecer Registros de Gestión de Configuración	
3.1.1. Registrar las acciones de gestión de configuración con suficiente detalle, para que se conozca el contenido y el estado de cada elemento de configuración, y para que se puedan recuperar versiones previas.	4.4.5.4
3.1.2. Asegurar que las partes interesadas relevantes tengan acceso y conocimiento del estado de la configuración de los elementos de configuración.	4.3.13.1
3.1.3. Especifica la última versión de las Líneas Base.	4.3.13.1
3.1.4. Identificar la versión de los elementos de configuración que constituyen una Línea Base particular.	4.3.13.1
3.1.5. Describir las diferencias entre Líneas Base sucesivas.	4.3.13.1
3.1.6. Modificar, si procede, el estado y la historia (es decir, cambios y otras acciones) de cada elemento de configuración.	4.3.13.1
SP 3.2 Realizar Auditorías de Configuración	
3.2.1. Evaluar la integridad de las líneas base.	4.6.1.1
3.2.2. Confirmar que los registros de gestión de configuración identifican correctamente los elementos de configuración.	4.6.1.1
3.2.3. Revisar la estructura y la integridad de los elementos en el sistema de gestión de configuración.	4.6.1.1
3.2.4. Confirmar que los elementos en el sistema de gestión de configuración son completos, correctos y consistentes.	4.6.1.1
3.2.5. Confirmar el cumplimiento con los estándares y procedimientos aplicables de gestión de configuración.	4.6.1.1
3.2.6. Seguir los elementos de acción desde la auditoría hasta su cierre.	4.6.1.1

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.4 Aseguramiento de la Calidad del Proceso y del Producto (PPQA)

PPQA – Aseguramiento de la Calidad del Proceso y del Producto	**Sección***
SG 1: Evaluar Objetivamente los Procesos y los Productos de Trabajo	

SP 1.1 Evaluar Objetivamente los Procesos	
1.1.1 Promover un entorno (creado como parte de la Gestión del Proyecto) que incentive la participación del personal en la identificación y comunicación de las cuestiones de calidad.	4.3.11, 4.4.3.3, 4.4.3.4
1.1.2 Establecer y mantener criterios claramente definidos para las evaluaciones.	4.3.11, 4.7.1(4), 4.7.7
1.1.3 Utilizar los criterios indicados para evaluar la adherencia de los procesos realizados y seleccionados frente a las descripciones de proceso, estándares y procedimientos.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6
1.1.4 Identificar cada "no conformidad" encontrada durante la evaluación.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6
1.1.5 Identificar las lecciones aprendidas que podrían mejorar los procesos.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6
SP 1.2 Evaluar Objetivamente los Productos de Trabajo	
1.2.1. Seleccionar los productos de trabajo a evaluar, en base a criterios de muestreo documentados (en caso de usar el muestreo).	4.3.11, 4.7.1(4)
1.2.2. Establecer y mantener criterios claramente establecidos para la evaluación de los productos de trabajo seleccionados.	4.3.11, 4.7.7
1.2.3. Utilizar los criterios indicados durante las evaluaciones de los productos de trabajo.	4.4.5.4
1.2.4. Evaluar los productos de trabajo seleccionados en los momentos escogidos.	4.4.5.4
1.2.5. Identificar cada caso de "no conformidad" encontrado durante las evaluaciones.	4.4.5.4
1.2.6. Identificar las lecciones aprendidas que podrían mejorar los procesos.	4.4.5.5
SG 2: Proporcionar una Visión Objetiva	
SP 2.1 Comunicar y Resolver las No Conformidades	
2.1.1. Resolver cada no-conformidad con los miembros apropiados del personal si es posible.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6
2.1.2. Documentar las no-conformidades cuando no puedan resolverse en el proyecto.	4.7.1(4)
2.1.3. Escalar las no-conformidades que no puedan resolverse en el proyecto, al nivel de gerencia apropiado designado para recibir las no conformidades y actuar sobre ellas.	4.4.5.5

2.1.4. Analizar las no conformidades para ver si existe alguna tendencia de calidad que pueda identificarse y tratarse.	4.4.5.5
2.1.5. Asegurar que las partes interesadas relevantes están al corriente de los resultados de las evaluaciones y de las tendencias de calidad de manera oportuna.	4.4.5.5
2.1.6. Revisar, periódicamente, las no conformidades abiertas y las tendencias con el gerente designado para recibir las no conformidades y actuar sobre ellas.	4.4.5.5
2.1.7. Seguir las no conformidades hasta su resolución.	4.4.5.5
SP 2.2 Establecer los Registros	
2.2.1. Registrar las actividades de aseguramiento de la calidad del proceso y del producto con suficiente detalle, de forma que sean conocidos el estado y los resultados.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6
2.2.2. Modificar el estado y la historia de las actividades de aseguramiento de la calidad, según sea necesario.	4.4.5.1, 4.4.5.3, 4.4.5.5, 4.4.5.6

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.5 Gestión de Requerimientos (REQM)

REQM – Gestión de Requerimientos	**Sección***
SG 1: Gestionar los Requerimientos	
SP 1.1 Entender los Requerimientos	
1.1.1 Establecer los criterios para distinguir los proveedores apropiados de requerimientos.	4.4.3.1
1.1.2 Establecer los criterios para la evaluación y aceptación de los requerimientos.	4.4.3.4
1.1.3 Analizar los requerimientos para asegurarse que los criterios establecidos son cumplidos.	4.4.5.1
1.1.4 Alcanzar un conocimiento del requerimiento con los proveedores del requerimiento de tal forma que los participantes del proyecto se puedan comprometer con los requisitos.	4.4.3.1, 4.4.5.1
SP 1.2 Obtener el compromiso con los Requerimientos	
1.2.1 Evaluar el impacto de los requerimientos en los compromisos existentes.	4.4.5
1.2.2 Negociar y registrar los compromisos.	4.4.5.2
SP 1.3 Gestionar los cambios en los requerimientos.	

1.3.1 Documentar todos los requerimientos y los cambios de los requerimientos que son dados o generados por el proyecto.	4.4.3.4
1.3.2 Mantener una historia de los cambios en los requerimientos, incluyendo la razón de los cambios.	4.4.3.4
1.3.3 Evaluar el impacto de los requerimientos desde el punto de vista de las principales personas interesadas.	4.4.5.1
1.3.4 Hacer la información de los requerimientos y cambios disponibles para el proyecto.	4.4.3.4
SP 1.4 Mantener la posibilidad de rastreo bidireccional entre los requerimientos y los productos de trabajo.	
1.4.1 Mantener posible el rastreo de requerimientos para asegurarse que la fuente de requerimientos de bajo nivel son documentados.	4.3.13.1
1.4.2 Mantener posible el rastreo de un requerimiento a sus requerimientos derivados y productos de trabajo asignados.	4.3.13.1
1.4.3 Generar matriz de rastreo de requerimientos.	4.4.5.6
SP 1.5 Asegurar el alineamiento entre los Requerimientos y el trabajo del Proyecto.	
1.5.1 Revisar planes de proyecto, actividades y productos de trabajo para asegurar la consistencia de requerimientos y los cambios hechos a éstos.	4.4.5.6
1.5.2 Identificar las fuentes de la inconsistencia (Si existieran).	4.4.5.6
1.5.3 Identificar los cambios que deben de ser hechos en los planes y los productos de trabajo resultantes de los cambios a la línea de base de los requerimientos.	4.4.5.1
1.5.4 Iniciar cualquier acción correctiva necesaria.	4.4.5.1, 4.4.5.6

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.6 Medición y análisis (MA)

MA –Medición y análisis	**Sección***
SG 1: Alinear las mediciones y el análisis de actividades	
SP 1.1 Establecer Objetivos de Medición.	
1.1.1 Documentar los requerimientos de información y los objetivos.	4.3.11, 4.7.1(5)
1.1.2 Priorizar los requerimientos de información y los objetivos.	4.3.11
1.1.3 Documentar, revisar y actualizar los objetivos de medición.	4.3.11, 4.7.1(5)

1.1.4 Proveer retroalimentación para refinar y clarificar los requerimientos de información y objetivos, según sea necesario.	4.4.5.6
1.1.5 Mantener la posibilidad de rastrear los objetivos de medición para identificar los requerimientos y objetivos de información.	4.3.11
SP 1.2 Especificar las mediciones.	
1.2.1 Identificar los candidatos de medición basándose en los objetivos de mediciones documentadas.	4.3.11, 4.7.1(5)
1.2.2 Mantener el rastreo de las mediciones hacia los objetivos de medición.	4.3.11
1.2.3 Identificar las mediciones existentes que resuelven objetivos de mediciones.	4.3.11
1.2.4 Especificar las definiciones operacionales de las mediciones.	4.3.11
1.2.5 Priorizar, revisar y actualizar las mediciones.	4.4.5.6
SP 1.3 Especificar la recolección de datos y procedimientos de almacenamiento.	
1.3.1 Identificar las fuentes existentes de datos que son generados de productos de trabajo actuales, procesos o transacciones.	4.3.11
1.3.2 Identificar las mediciones de datos que son requeridas pero no disponibles actualmente.	4.3.11
1.3.3 Especificar como recolectar y almacenar los datos para cada uno de las mediciones requeridas.	4.3.11, 4.7.1(5)
1.3.4 Crear los mecanismos de recolección de datos y las guías de procesos.	4.3.11
1.3.5 Mantener la recolección automática de datos de forma apropiada y viable.	4.3.11
1.3.6 Priorizar, revisar y actualizar los objetivos de medición según sea necesario.	4.4.5.6
1.3.7 Actualizar las medidas y objetivos de medición conforme sea necesario.	4.4.5.6
SP 1.4 Especificar los procedimientos de análisis.	
1.4.1 Especificar y priorizar los análisis a ser realizados y los reportes a ser preparados.	4.3.11, 4.7.1(5)
1.4.2 Seleccionar los métodos y herramientas de análisis apropiados.	4.3.11, 4.7.1(5)
1.4.3 Especificar procedimientos administrativos para analizar los datos y comunicar los resultados.	4.3.11, 4.7.1(5)
1.4.4 Revisar y actualizar el contenido propuesto y el formato de los análisis especificados y los reportes.	4.4.5.6

1.4.5 Actualizar las medidas y mediciones de objetivos según sea necesario.	4.4.5.6
1.4.6 Especificar los criterios para evaluar la utilidad de los resultados del análisis y para la evaluación de la ejecución de las medidas y las actividades de análisis.	4.3.11, 4.7.1(5)
SG 2: Proporcionar los resultados de las mediciones.	
SP 2.1 Obtener datos de medición.	
2.1.1 Obtener datos para la medición base.	4.4.5.4
2.1.2 Generar datos de medidas derivadas.	4.4.5.4
2.1.3 Realizar revisiones de integridad de datos tan cercanos a la fuente de los datos como sea posible.	4.4.5.4
SP 2.2 Analizar los datos de medición.	
2.2.1 Realizar los análisis iniciales, interpretar los resultados y sacar conclusiones preliminares.	4.4.5.4
2.2.2 Realizar medidas y análisis adicionales según sean necesarias para preparar los resultados para su presentación.	4.4.5.4
2.2.3 Revisar los resultados iniciales con las personas involucradas relevantes.	4.4.5.5
2.2.4 Refinar los criterios para análisis futuros.	4.4.5.6
SP 2.3 Almacenar datos y resultados.	
2.3.1 Revisar los datos para asegurar su integridad, precisión, actualidad y completitud.	4.4.5.6
2.3.2 Almacenar los datos de acuerdo a los procedimientos de almacenamiento de datos.	4.4.5.4
2.3.3 Poner el contenido almacenado a disposición de solamente los grupos y el personal apropiado.	4.3.13.1
2.3.4 Provenir que la información almacenada sea usada de forma inapropiada.	4.3.13.1
SP 2.4 Comunicar los resultados.	
2.4.1 Mantener a las personas involucradas informadas de los resultados de las mediciones en el tiempo indicado.	4.4.5.5
2.4.2 Asistir a las personas involucradas a entender los resultados.	4.4.5.5

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.7 Gestión de Acuerdos con Proveedores (SAM)

SAM –Gestión de Acuerdos con Proveedores	**Sección***
SG 1: Establecer los acuerdos con los proveedores	
SP 1.1 Determinar los tipos de Adquisición.	4.3.5.1
SP 1.2 Seleccionar proveedores.	
1.2.1 Establecer y documentar los criterios para evaluar a los potenciales proveedores.	4.6.2.2
1.2.2 Identificar los potenciales proveedores y entregarles una solicitud de material y requerimientos.	4.6.2.2
1.2.3 Evaluar las propuestas de acuerdo a los criterios de evaluación.	4.6.2.2
1.2.4 Evaluar los riesgos asociados con cada proveedor propuesto.	4.6.2.2
1.2.5 Evaluar las habilidades para desarrollar el trabajo de cada uno de los proveedores propuestos.	4.6.2.2
SP 1.3 Establecer los Acuerdos con los Proveedores	
1.3.1 Revisar los requerimientos que serán solucionados por el proveedor que reflejan las negociaciones con el proveedor, cuando así sea necesario.	4.6.2.3
1.3.2 Documentar lo que el proyecto entregara al proveedor.	4.6.2.3
1.3.3 Documentar los acuerdos con el proveedor.	4.6.2.3
1.3.4 Revisar periódicamente los acuerdos con el proveedor para asegurarse que reflejan fielmente las relaciones del proyecto con el proveedor, los riesgos y condiciones de mercado.	4.6.2.3
1.3.5 Asegurarse que todos las partes involucradas en el acuerdo de proveedor entiendan y estén de acuerdo en todos los requerimientos antes de implementar el acuerdo o cambiarlo.	4.6.2.3
1.3.6 Revisar el acuerdo con el proveedor como sea necesario para reflejar los cambios en los procesos del proveedor o sus productos.	4.6.2.3
1.3.7 Revisar los planes y compromisos, incluyendo cambios a los procesos del proyecto o productos de trabajo, según sea necesario para reflejar los acuerdos con el proveedor.	4.4.5.1
SG 2: Satisfacer los acuerdos con el Proveedor	
SP 2.1 Ejecutar el acuerdo con el proveedor.	
2.1.1 Monitorizar el progreso y el desempeño del proveedor como se definió en el acuerdo de proveedor.	4.6.2.4

2.1.2 Seleccionar, monitorizar y analizar los procesos usados por el proveedor, tal y como se define en el acuerdo de proveedor.	
2.1.3 Seleccionar y evaluar los productos de trabajo, tal y como se definen en el acuerdo de proveedor.	
2.1.4 Realizar revisiones con el proveedor según acordado en el acuerdo de proveedor.	4.4.5.5
2.1.5 Realizar revisiones técnicas con el proveedor según lo estipulado en el acuerdo de proveedor.	4.4.5.5
2.1.6 Realizar revisiones de gestión con el proveedor, tal y como está acordado en el acuerdo de proveedor.	4.4.5.5
2.1.7 Usar los resultados de las revisiones para mejorar el desempeño del proveedor para establecer y alimentar relaciones de largo término con los proveedores preferidos.	4.4.5.6
2.1.8 Monitorizar los riesgos relacionados con el proveedor y tomar acciones correctivas según sea necesario.	4.4.5.5
SP 2.2 Aceptar el producto adquirido	
2.2.1 Definir los procedimientos de aceptación.	4.6.2.4
2.2.2 Revisar y obtener acuerdo con las partes involucradas principales sobre la aceptación de los procedimientos antes de la revisión de aceptación o pruebas.	4.6.2.4
2.2.3 Verificar que los productos adquiridos satisfacen sus requerimientos.	4.5.3, 4.5.6
2.2.4 Confirmar que los compromisos no técnicos asociados con los productos de trabajo adquirido sean satisfechos.	4.5.3, 4.5.6
2.2.5 Documentar los resultados de las revisiones de aceptación y pruebas.	4.5.3, 4.5.6, 4.5.7.1
2.2.6 Establecer planes de acción y obtener acuerdo con el proveedor para tomar acciones correctivas sobre los productos de trabajo que no pasen la revisión de aceptación o pruebas.	4.5.3, 4.5.6, 4.5.7.1
2.2.7 Identificar, documentar y rastrear los elementos de acción hasta su cierre.	4.5.6, 4.5.7.1
SP 2.3 Asegurar la transición de los productos.	
2.3.1 Asegurar que existan las instalaciones para recibir, almacenar, integrar y mantener los productos adquiridos de forma apropiada.	4.5.3
2.3.2 Asegurarse que en las personas involucradas en recibir, almacenar, integrar y mantener los productos adquiridos, recibieron una formación adecuada.	4.5.4

2.3.3 Asegurarse que los productos son almacenados, distribuido e integrados de acuerdo a los términos y condiciones especificadas en el acuerdo de proveedor o licencia.	4.5.4

* La referencia que se encuentra al lado derecho de las tablas con el listado de los objetivos de CMMI son enlaces que pueden ser presionados para navegar hacia la sección misma.

4.1.3.8 Metas Genéricas y Prácticas Genéricas

Las metas genéricas establecidas por CMMI-DEV van directamente ligadas con institucionalizar los procesos y afecta a todas las áreas involucradas en las prácticas de CMMI-DEV.

Al tener las metas genéricas un alcance mayor al nivel 2 de madurez y su poca relación con la metodología *Scrum*, base de este proceso; solo se establecerán las relaciones entre las prácticas genéricas y el área de proceso en donde esta práctica tendría cabida. Esto se debe a que sería posible utilizar las mismas prácticas establecidas extendiéndolas a las demás áreas de proceso.

A continuación se documentan las relaciones entre las prácticas genéricas y las áreas de proceso. Estas relaciones son extraídas de la especificación de CMMI-DEV (CMMI Product Team, 2010, p. 122).

GG 1 Proceso Realizado
GP 1.1 Realizar las prácticas específicas.
El cumplimiento de esta práctica se logra de forma automática al cumplir con las prácticas específicas definidas en esta guía para las áreas contenidas en el nivel 2 de madurez.

GG 2 Institucionalizar un proceso gestionado	**Área/Práctica**
GP 2.1 Establecer una política de la organización	
La alta dirección establecerá y comunicara las directrices y las expectativas que se tienen sobre cada una de las áreas. Contando con el apoyo del encargado o experto en cada una de las áreas del proceso para su elaboración. Estas políticas deberán de estar visibles para todos los miembros de la organización. Para esto se utilizara la wiki de *Sharepoint*.	Alta Dirección
GP 2.2 Planificar el proceso	
Planificar el proceso se implementara como parte de las prácticas especificas del Área de Planificación de Proyecto (PP). Con la excepción de planificar el proceso de área PP misma, su planificación se realiza antes de la planificación del proyecto utilizando la plantilla de Plan de Gestión de Proyecto (PMP).	PP

GP 2.3 Proporcionar recursos	
Proporcionar recursos para todas las áreas de procesos se puede realizar como parte de la planificación de proyectos (PP) en la práctica SP 2.4. La excepción es el proporcionar recursos para el área PP, la cual deberán ser proporcionados con antelación a la planificación y documentados en el PMP.	PP
GP 2.4 Asignar responsabilidad	
La identificación de procesos, roles y responsabilidades necesarias para asegurar que están garantizados el personal, las instalaciones, el equipamiento y otros activos necesarios apropiados para el proyecto se podrán realizar durante la práctica específica SP 2.4 del área de planificación de proyectos (PP).	PP SP 2.5
GP 2.5 Formar al personal	
La parte del proceso de planificación del proyecto que implementa PP SP 2.5 "Planificar el conocimiento y las habilidades necesarias" y el proceso de formación de la organización, da soporte a la implementación de GP 2.5 en su totalidad para todas las áreas de proceso relacionadas con el proyecto. También se da soporte a la formación de personal en el área de Formación en la Organización (OT), pero al ser un nivel de madurez 3 está fuera del alcance de esta guía.	PP SP 2.5
GP 2.6 Controlar los productos de trabajo	
El proceso de gestión de configuración (CM) implementara el proceso de Controlar los productos de trabajo en su totalidad para todas las áreas de proceso relacionadas con el proyecto. Además de poder ayudar a lograr estas prácticas para ciertas áreas de proceso de la organización.	CM
GP 2.7 Identificar e involucrar a las partes interesadas relevantes	
La parte del proceso de planificación del proyecto que implementa PP SP 2.6 "Planificar la involucración de las partes interesadas" puede implementar la parte de la identificación de las partes interesadas relacionadas con la identificación de las partes interesadas a cada proceso y sus relaciones con ella; además de compartir las identificaciones con los encargados de planear el proyecto. Esto sería en su totalidad para todas las áreas de proceso relacionadas con el proyecto.	PP SP 2.6 PMC SP 1.5 IPM SP 2.1

La parte del proceso de monitorización y control del proyecto que implementa PMC SP 1.5 "Monitorizar la involucración de las partes interesadas" puede ayudar en la implementación de la tercera subpráctica de GP 2.7 para todas las áreas de proceso relacionadas con el proyecto. También la práctica SP 2.1 de Gestión Integrada del Proyecto puede ayudar en la implementación de la tercera subpráctica de GP 2.7 para todas las áreas de proceso relacionadas con el proyecto. Pero esa área está fuera del alcance de esta guía.	
GP 2.8 Monitorizar y controlar el proceso	
La práctica genérica de monitorizar y controlar el proceso se implementa en la prácticas generas de los procesos del área de monitorización y control del proyecto, en su totalidad para todas las áreas de proceso relacionadas con el proyecto. Los procesos del área de Medición y Análisis proporcionan orientación general sobre la medición, el análisis y el registro de información que pueden usarse para establecer medidas para monitorizar el rendimiento del proceso. Esto es para todas las áreas.	PMC MA
GP 2.9 Evaluar objetivamente la adherencia	
La evaluación objetiva de la adherencia es implementada en su totalidad para todas las áreas del proceso en el proceso de Aseguramiento de la calidad del proceso del producto (PPQA). Para cumplir con el área misma de PPQA será necesario establecer un proceso extra fuera del área misma.	PPQA
GP 2.10 Revisar el estado con el nivel directivo	
Los proceso SP 1.6 y 1.7 del área de Monitorización y Control del Proyecto permite la implementación de la práctica genérica 2.10 para todas las áreas de proceso relacionadas con el proyecto.	PMC SP 1.6 y 1.7

GG 3 Institucionalizar un proceso definido	**Sección***
GP 3.1 Establecer un proceso definido	
Se realizan en áreas fuera del nivel 2 de madurez, por lo tanto está fuera del alcance de esta guía.	IMP SP 1.1 OPD

GP 3.2 Recoger experiencias relativas al proceso	
Se realizan en áreas fuera del nivel 2 de madurez, por lo tanto está fuera del alcance de esta guía.	IMP SP 1.7 OPF SP 3.4 OPD

4.2 CICLO DE VIDA DE LA METODOLOGÍA PROPUESTA

El ciclo de vida de la metodología de desarrollo propuesto en este documento está dividido en 3 fases: Planificación Inicial, Desarrollo/Construcción y Transición/Implementación; además de otros procesos que corren en paralelo con la fase de desarrollo del proceso.

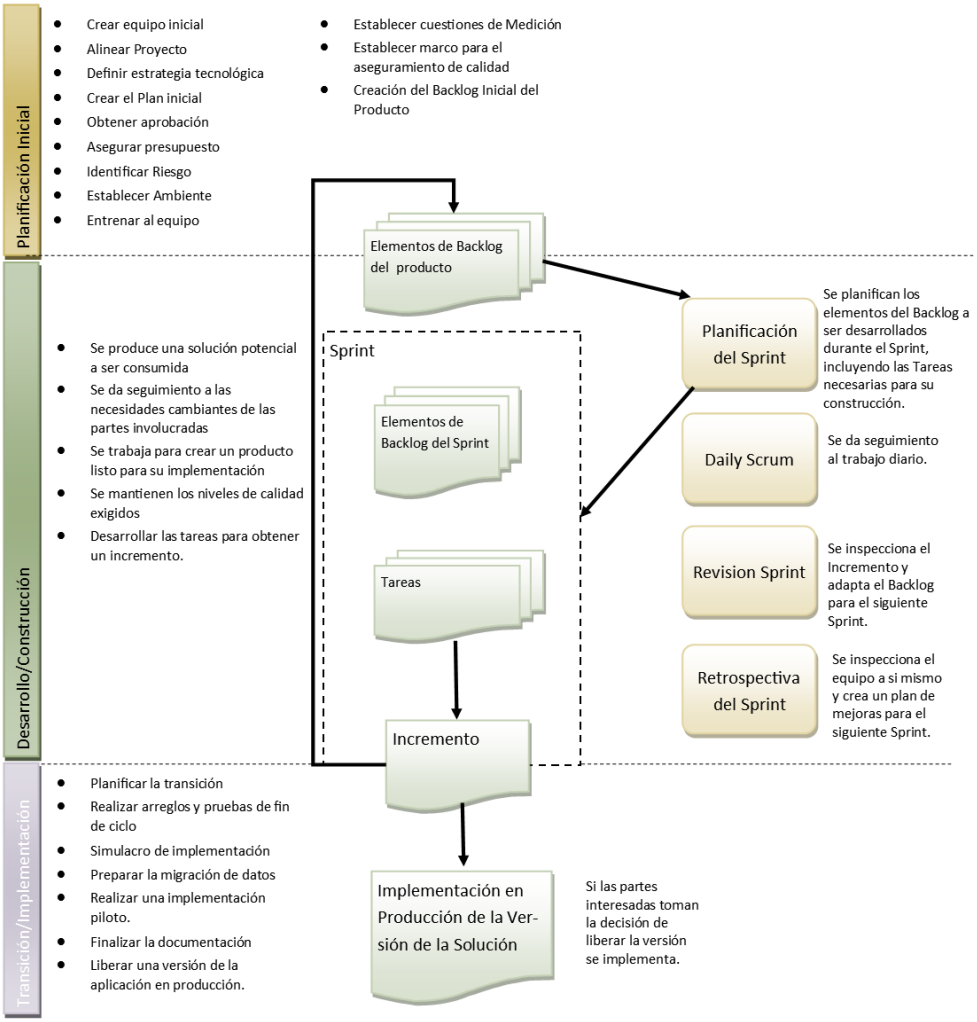

Figura 2 Ciclo de Vida del proceso de desarrollo

Los requisitos de este proceso de desarrollo se pueden apreciar a lo largo de las fases que incluyen el plan inicial, la construcción y la transición, por lo que es necesario leer todas las fases para determinar los requisitos necesarios del proceso. [SP PP 2.4.1]

4.3 PLANIFICACIÓN INICIAL

La fase de planificación inicial de esta metodología de desarrollo está basado en la fase llamada "*Inception"* descrita en *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise* (Ambler & Lines, 2012).

4.3.1 Objetivos.

Los objetivos de esta fase se pueden definir en 3 categorías: coordinación, colaboración y conclusión. Dentro de la categoría de **coordinación** se realizan las siguientes actividades: iniciar el equipo y planificar las sesiones de visualización con las partes involucradas. En la categoría de **colaboración** se realiza la creación del equipo, la conceptualización de los requerimientos, la conceptualización de la arquitectura, la consideración de la factibilidad, la alineación del proyecto con la estrategia de la compañía, la liberación del plan inicial, el desarrollo de una visión compartida y el establecimiento del ambiente. Por último en la categoría de **conclusión**, se realiza la revisión de hitos ligeros y la comunicación de la visión a las partes involucradas.

A continuación se describe en que consiste cada una de estas actividades.

4.3.2 Crear el equipo inicial e identificar partes interesadas.

El equipo a formar en esta fase será la base del equipo que trabajará de forma continuada en la fase de construcción. Es conveniente que esto suceda así y que todo el equipo involucrado en el inicio se encuentre en la construcción, pero puede no ocurrir así.

Un equipo inicial típico consiste en un propietario del producto, un propietario de la arquitectura y el líder del equipo (*Scrum Master*). Este equipo debe de contar con las habilidades necesarias para entender todos los aspectos de la solución candidata a implementar y de los requerimientos necesarios para implementarla. Igual de importante es que el equipo formado tenga acceso a las partes interesadas en el proyecto. [PP SP 2.4.3, SP 2.6]*

*Como se lee en las instrucciones de uso de la guía, estos párrafos hacen referencia a las prácticas específicas PP SP 2.4.3 y 2.6. Como ejemplo se muestra a continuación un extracto del apartado 4.1.3 mostrando la práctica específica. En el resto de las anotaciones de este inciso se omitirán las referencias de ejemplo, las cuales deberán de ser consultadas en el apartado 4.1.3.

PP -Planificación del Proyecto	**Sección***
SG 2 Desarrollar un plan de proyecto	
SP 2.4 Planificar los Recursos del Proyecto	
3. Determinar los requisitos de personal.	4.3.2, 4.4.3, 4.7.1(2)

Existe la posibilidad de que ya exista un equipo formado con anterioridad al proyecto y se decida mantenerlo unido y usarlo en el proyecto nuevo, de ser así la diferencia entre la creación del equipo centralizado a uno distribuido no sería relevante; de lo contrario se intentara que el equipo inicial sea centralizado para evitar la inversión en tiempo que conlleva la formalización de un equipo distribuido para un proyecto que aún no se sabe si pasara o no a etapa de construcción.

Normalmente se crea el equipo de la fase inicial y se va agregando personal para la fase de construcción, es importante ir preseleccionando a los miembros que se unirán al equipo de continuar el proyecto hacia la fase de construcción.

Se identifican todas las partes interesadas en la solución, entre ellos podemos enumerar: clientes y usuarios del proyecto, patrocinadores del proyecto, personal encargado de operar la solución, el departamento de Marketing, organismos reguladores, organismos de gestión de cambio y Oficina de Gestión de Proyectos (PMO). No es necesario identificar todas las posibles partes involucradas en esta fase temprana, pero los principales deben de estar identificados y se irán agregando conforme se acerque la fase de Transición.

La información recopilada del equipo de desarrollo y el personal involucrado se documentaran usando la plantilla "4.7.4 Matriz de Partes Involucradas". [PP SP 2.6]

4.3.3 Alinear el proyecto con la filosofía de la empresa.

Para lograr una meta a largo plazo es necesario alinear los objetivos del proyecto con los objetivos generales de la organización, entre los cuales están las guías de desarrollo, el uso de plantillas comunes y reusar infraestructura existente.

Usar guías comunes de desarrollo. Existirá un repositorio que contendrá las guías relativas a las convenciones de codificación, convenciones para la creación de interfaces de usuario, convenciones para el nombramiento de objetos de base de datos, etc. Estas convenciones son dependiente de la arquitectura y tecnologías a utilizar, por ejemplo si se tiene una arquitectura basada en Microsoft IIS y servicios web entonces las guías de Microsoft C# y de ASP.MVC serán las utilizadas.

Adoptar plantillas comunes. La documentación como manuales de usuarios, manuales operativos y manuales de equipo de soporte al funcionamiento deben de ser escritos de forma consistente entre todos los proyectos de la compañía. De la misma forma que con las guía de desarrollo; estas guías deberán estar ubicadas en un repositorio en la red, disponible para todos los proyectos desarrollados para la compañía.

Reusar la infraestructura existente. Antes de iniciar cualquier desarrollo es necesario identificar que infraestructura se encuentra disponible de proyectos anteriores. Para esto se debe de mantener un diccionario de componentes y servicios en donde sea fácil identificar los

posibles recursos a reutilizar, entre los cuales se encuentran pero no se limitan a: componentes, servicios, patrones y mecanismos, código fuente, referencias de cómo implementar cierta arquitectura, etc. El plantilla de este diccionario se encuentra en la sección de plantillas "4.7.7 Diccionario de infraestructura".

Entender la forma de gobernanza del departamento de TI de la organización. Al ser necesario en los proyectos la interacción con partes externas al proyecto se hace indispensable conocer la forma correcta de relacionarse con éstos. La guía debe establecer que debe de ser revisado, por quien y cuando; que información debe de ser entregada, mediante qué medios, a quien y cuando. Estas revisiones y reportes deben de ser identificados como un elemento de trabajo para ser añadido en los *Sprints* correspondientes.

4.3.4 Identificar la estrategia tecnológica.

Durante la fase inicial el equipo deberá modelar la arquitectura inicial del proyecto como elemento de soporte para la planificación inicial.

Mediante el uso de la práctica llamada Conceptualización Arquitectónica (*Architecture Envisioning*) se crean modelos arquitectónicos ágiles de alto nivel, estos modelos son creados en horas o días en lugar de semanas o meses como seria requerido por los modelos arquitectónicos usados en los metodologías tradicionales.

De forma abreviada el proceso seria el siguiente: se reúnen los desarrolladores en un lugar, comúnmente con una pizarra y discuten y crean un esquema de la potencial solución arquitectónica. El propósito es identificar la estrategia arquitectónica y no crear un modelo arquitectónico detallado. Este modelo detallado será desarrollado según vaya siendo requerido en las fases de construcción usando técnicas de *model storming* y desarrollo guiado por pruebas (TDD).

Ventajas: Reduce el riesgo de proyecto porque los posibles problemas técnicos son pensados a conciencia, permite al equipo estar de acuerdo y hacer suya una visión técnica, permite flexibilidad, las decisiones técnicas detalladas pueden ser dejadas para cuando la toma de esta decisión sea más apropiada.

Desventajas: Requiere gran colaboración entre los miembros del equipo y los miembros deben de tener grandes habilidades de diseño y arquitectura, se debe estar conscientes en todo momento de las guías de dirección arquitectónica de la empresa y puede motivar la sobre-construcción de la solución en una fase temprana del ciclo de vida.

Más información sobre Conceptualización Arquitectónica en el Capítulo 12 de *Disciplined Agile Delivery* (Ambler & Lines, 2012)

Se mantiene un manual de arquitectura, el cual debe ser integrado en un *wiki* en *Sharepoint;* que contiene los diagramas principales de arquitectura, las decisiones claves sobre la

arquitectura, el pensamiento detrás de las mismas. Esto es con el propósito de soportar el mantenimiento futuro y los esfuerzos de mejora.

4.3.5 Desarrollar el plan inicial.

4.3.5.1 Definir estructura de desglose de procesos (WBS)

Se define la estructura de desglose de procesos (WBS) utilizando el *add-in WBS Modeler* de Visio. Este *add-in* permite la posterior exportación a *MS Project* de los productos de trabajo para seguir con la planificación del proyecto. [PP SP 1.1.1]

La estructura de desglose de trabajo se realiza mediante elementos de S*crum*, como lo son: las Historias, las Épicas y los Temas.

El nivel más alto del WBS es ocupado por todos los Temas involucrados en el proyecto, debajo de los temas se colocan las historias Épicas y por último las Historias de Usuario.

Cada Historia cuenta con un número único de identificación. Estas historias se convierten en productos que serán usados para crear el *backlog* del producto. [PP SP 1.1.2]

Se analiza y establece si los elementos necesarios para satisfacer las historias serán desarrollados, serán elementos reutilizados de antiguos proyectos o si serán elaborados por un proveedor externo. Si serán adquiridos de un proveedor es necesario definir qué tipo de adquisición será, por ejemplo: productos que se pueden comprar empaquetados, obtenidos con acuerdos de proveedor, desarrollados por otra área de la compañía y adquiridos a ésta, obtenidos de un proveedor preseleccionado por la compañía, etc. Esta información deberá de ser explicada dentro de las notas del elemento de la estructura de desglose de procesos [SAM SP 1.1]. La evaluación de los proveedores y su selección será llevada a cabo en el proceso de Gestión de Proveedores, sección 4.6.2 de este documento.

Los elementos del diagrama de desglose de trabajo tienen diferentes colores, los cuales ayudan a definir el tipo de elemento que representan; los colores a utilizar son los siguientes:

Azul – Temas

Amarillo – Historias Épicas

Naranja – Historias de Usuario

Marrón – Para componentes que serán desarrollados externamente [PP SP 1.1.3].

Violeta – Para Productos de Trabajo que serán reutilizados [PP SP 1.1.4].

Cada uno de estos Productos de Trabajo deberá contener información de su estimación en horas y costes.

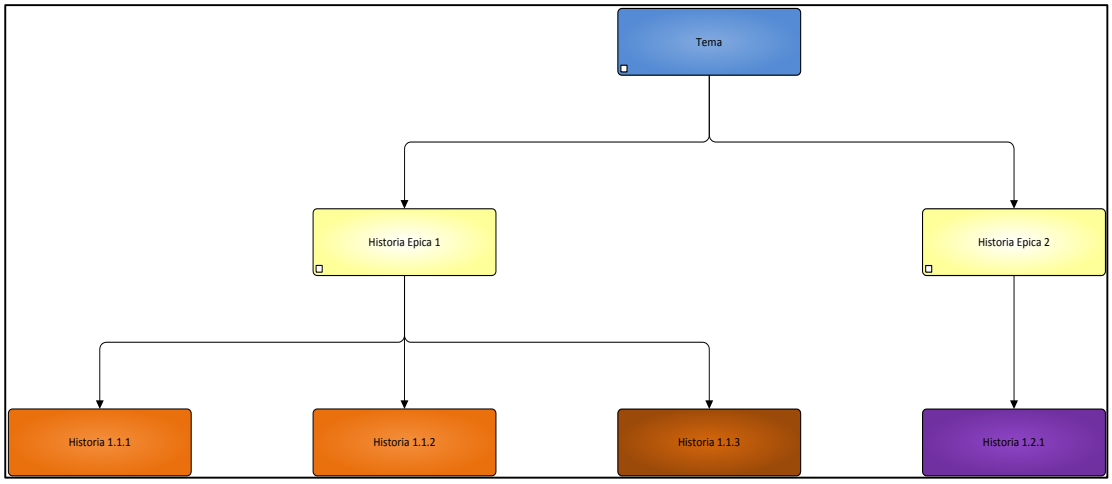

Figura 3 Ejemplo de Estructura de desglose de trabajo.

Cada uno de los elementos de esta estructura puede contener la información que se muestra en la **Error! Reference source not found.**. Los datos que son estrictamente necesarios al momento de definir el diagrama serán: el coste, la duración estimada, nombre de elemento y un identificador único. Los demás elementos pueden ser usados a criterio de autor del diagrama.

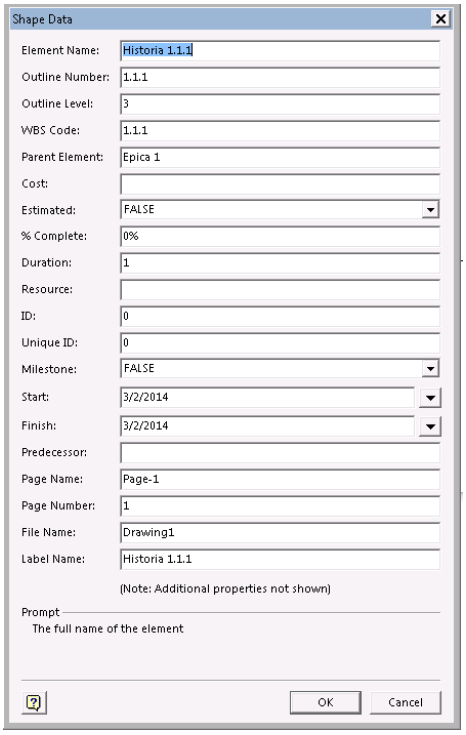

Figura 4 Detalle de las propiedades de los elementos de WBS

4.3.5.2 Estimar Tiempos y Coste

Las estimaciones iniciales se hacen utilizado datos históricos de proyectos elaborados en la compañía, los cuales serán almacenados en un repositorio creado en *Team Foundation Server*. [PP SP 1.4.1, SP 1.4.4] Dicho repositorio será actualizado con los tiempos reales utilizados en cada producto y ponderado la medida de sus atributos al finalizar cada proyecto en la etapa de Revisión del Sprint. [PP SP 1.2.4]

Los atributos básicos a utilizar son:

Para un elemento de desarrollo: La métrica principal de medición es “Puntos de Historia”. El *Scrum Master* podrá definir otra métrica de ser necesario. Las métricas añadidas en ningún caso permitirán el remplazo de los “Puntos de Historia” como métrica principal, solo podrán ser utilizadas de apoyo.

Para documentación: Número de páginas, número de diagramas contenidos.

Para otros tipos de elementos: la definición de los atributos a usar para su medición se decidirá por el *Scrum Master*.

Estas métricas son plasmadas en el documento de Plan de Gestión del Proyecto (PMP) en la sección de 5. Métricas.

Al estar el desarrollo de software basado en metodologías ágiles los estimados de esfuerzo y costes obtenidos en la estimación inicial no se puede tomar como una verdad absoluta, si no que éstos irán evolucionando conforme avance el proyecto. Esto es debido principalmente, a que se asume como principio que los requerimientos no son estáticos.

Los resultados de los tiempos requeridos para realizar los proyectos históricos serán almacenadas en una hoja de calcula de MS Excel y versionados en TFS, esta información será utilizada para convertir las historias en horas y costes requerido para el nuevo proyecto. [PP SP 1.4.1]

Al tener a las historias de usuario como elementos de nuestro esquema de desglose de trabajos, las estimaciones se realizaran sobre ellas. El modelo que se utilizará para calcular el esfuerzo requerido en desarrollar cada una de estas historias es el llamado "Planificación Póker" cuyo detalle esta explicado a detalle en el en capítulo 2 de (Olausson, Rossberg, Ehn, & Sköld, 2013). [PP SP 1.2.2]

El modelo nos indica que la forma en que se medirán las estimaciones será en puntos de historia. Estos puntos no son una medida exacta, sino una medida relativa. El personal involucrado y con experiencia en el producto a crear, se reúne y entre ellos asignan la cantidad de puntos de historia requeridos para cada una de las historias (productos de trabajo). La escala más comúnmente utilizada es una escala Fibonacci modificada, la cual se muestra en la siguiente secuencia: 1, 2, 3, 5, 8, 13, 20, 40 y 100.

La técnica a utilizar en las juntas de estimación de requerimientos será la llamada "Planificación Póker". Debe este nombre, al hecho de que se utilizan cartas marcadas con la secuencia Fibonacci de simulado cartas de póker; la técnica es la siguiente:

1. El *Scrum Master* lee la primera historia de usuario.

2. El equipo considera las implicaciones de la historia y selecciona una carta, sin mostrar esta carta a los demás participantes.

3. Todos los miembros del equipo enseñan sus cartas al mismo tiempo.

4. Si el resultado varía mucho, aquéllos con mayor y menor puntuación explican el porqué de sus resultados.

5. Después de una discusión corta, el equipo juega de nuevo.

6. Cuando el equipo llegue a un consenso (o que los miembros del equipo solo varíen un escalón en la escala), entonces se califica la historia.

7. Si resulta imposible llegar a un acuerdo se elige el valor más alto.

El siguiente paso sería calcular el tiempo requerido a partir de los puntos de historia, pero para ello es necesario primero conocer la capacidad del equipo de desarrollo.

Para realizar el cálculo de la capacidad del equipo se deberán de tomar en consideración los siguientes puntos:

- ¿Qué tan grande es el sprint?
- ¿Cuantos días hay disponibles dentro de un sprint?
- ¿Cuántos días trabaja un miembro del equipo en un sprint? (Quitando los días de vacaciones, días festivos, juntas planeadas, etc.)
- Sustraer el tiempo requerido para la planificación del Sprint, revisión o juntas de retroalimentación.
- El resultado de esto es la capacidad antes de *drag*. (*Drag* es tiempo usado en actividades desconocidas)
- Se debe de medir el *drag* en cada sprint. Es imposible saber realmente que tanto se debe calcular para el *drag* inicial. Entre más tiempo tome el proyecto, más exacto será el cálculo de *drag*.
- Si no se sabe por experiencia cuanto tiempo estimar al *drag*, 25% es un buen inicio. Éste 25% incluye un 10% de crecimiento del *backlog* del producto.
- Ahora se tiene el tiempo disponible en horas en el sprint.

Por último es necesario calcular la velocidad del equipo, esto es el número de puntos de historia que pueden ser desarrollados en un sprint. Esto se calculara en las juntas de planificación del Sprint. Durante esta junta el equipo calcula las tareas en horas y decide que tantas historias de usuario pueden ser integradas en la planificación del *Sprint*. La forma en que se desarrollaría esta junta es la siguiente.

1. Se estima la primera historia de usuario a detalle.
2. Se desglosa lo que el equipo tendrá que hacer para desarrollar y entregar la historia.
3. Se estiman las horas de cada tarea y se suman.
4. Se restan las horas estimadas del tiempo disponible del equipo en el sprint.
5. ¿Quedan horas disponibles?

6. Se toma una nueva historia de usuario y se repite el proceso hasta que ya no queden horas disponibles.

7. Se suma el número de puntos de historia de las historias incluidas en el sprint.

8. De esta forma se obtiene la velocidad teórica.

Ahora que se tiene la velocidad del equipo, se hace el cálculo de tiempo dividiendo los puntos de historia del proyecto entre los puntos de historia que se pueden realizar en cada sprint; sobre el resultado obtenido se multiplica el tiempo requerido para cada sprint por el número total de *Sprints* requeridos para realizar el proyecto y así obtenemos el tiempo total requerido. [PP SP 1.4.3]

A los tiempos de desarrollo se agrega el tiempo necesario para establecer y mantener la infraestructura técnica para realizar el proyecto, incluyendo: Tiempos para instalar y mantener servidores de bases de datos, de administración de proyectos, las herramientas de trabajo de los desarrolladores, actualización de seguridad, etc. Los tiempos serán basados en los tiempos requeridos en el pasado para establecer ambientes de trabajo similares. [PP SP 1.4.2]

4.3.5.3 Crear un Plan de Proyecto

Se revisan otros proyectos que tengan relación con el proyecto que se está elaborando, para establecer los compromisos que se deben de cumplir para no afectar negativamente a los demás proyectos. Las conclusiones de esta revisión se agregan en la sección de "Alcance y Descripción General del Sistema" del "4.7.1 Plan de Gestión del Proyecto (PMP)". [PP SP 3.3.1]

Esta información será tomada en cuanta para definir las fechas de los hitos en el calendario del proyecto.

Al trabajar de forma opuesta a los métodos tradicionales, en donde las variables de un proyecto son el tiempo y los recursos; la definición de hitos es indispensable para los planes de trabajo en proyectos ágiles. En metodologías ágiles lo que varía es la cantidad de características incluidas dentro determinada versión del proyecto. Así vemos que tendremos en nuestro proyecto las fechas en las que se liberaran una versión del proyecto fijado, pero no el detalle de las características incluidas en esa versión. [PP SP 2.1.1]

Además de los hitos, un elemento a definir con claridad deberá de ser el orden de los productos de trabajo a realizar. De esta forma se podrá priorizar la inclusión de los trabajos en los *Sprints* utilizando factores como prioridad y dependencia de los trabajos entre sí. [PP SP 2.1.4]

Con toda esta información se construirá un calendario el cual no llegara a los detalles de productos de trabajo, pero si a nivel de historias de usuario. En este calendario, que será elaborado en MS Project, se definirán los hitos de desarrollo con sus fechas y las historias que serán incluidos en cada uno de los *Sprints*. El coste en puntos de historia y el valor económico

serán identificados en el calendario. Así mismo se resumirán los costes económicos y de tiempo en la plantilla de "4.7.2 Tiempos y Costes"; que se encuentra en la sección de plantillas de este documento. Estos documentos son ingresados en TFS para controlar su mantenimiento y cambios en el tiempo. La referencia hacia estos documentos se debe de definir en el apartado de "Ciclo de vida y Calendario" del PMP, así como incluir la frecuencia de actualización del calendario. [SP PP 2.1.5]

Se deberán de plasmar la información que se presupone del proyecto, de lo que no se tiene suficiente información en el calendario realizado. Este calendario será elaborado con MS Project y referenciado en el PMP. [PP SP 2.1.2]

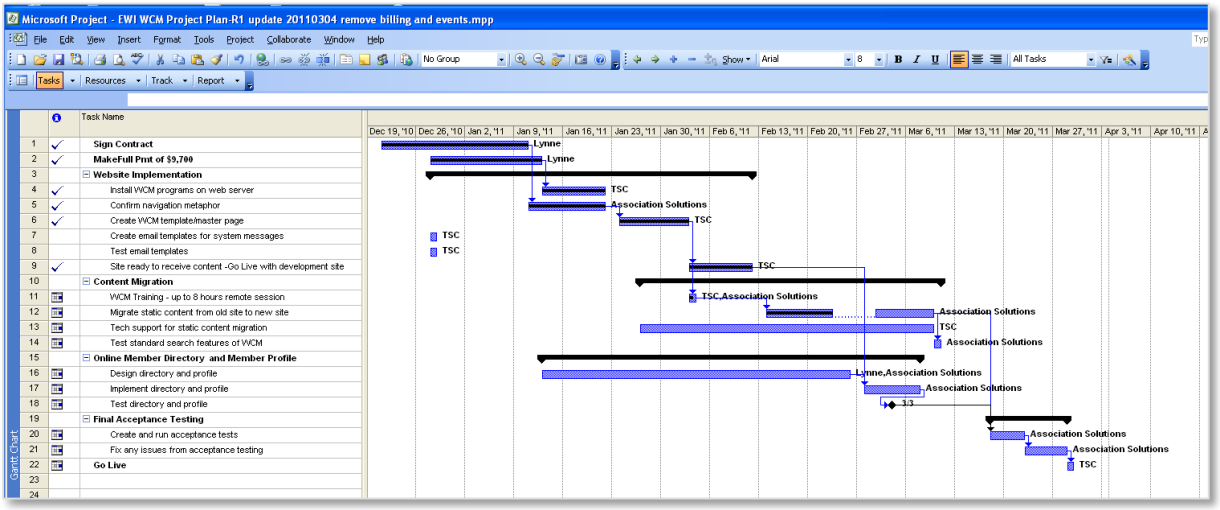

Figura 5 Ejemplo de calendario en MS Project

El plan del proyecto se establece en un Plan de Gestión de Proyecto (PMP), una plantilla de este documento se encuentra definido en la sección "4.7.1 Plan de Gestión del Proyecto (PMP)". Se integrará en un repositorio del proyecto en TFM, con lo cual será controlada su versión y con eso se mostrara la evidencia de su mantenimiento. [PP SP 2.7]

4.3.6 Identificar la visión del proyecto y obtener la aprobación de las partes involucradas en ella.

Se revisan otros proyectos que tengan relación con el proyecto que se está elaborando, para establecer los compromisos que se deben de cumplir para no afectar negativamente a otros proyectos. Las conclusiones de esta revisión se agregan en la sección "4.7.1 Plan de Gestión del Proyecto (PMP)" bajo el apartado "Alcance y Descripción General del Sistema".

Tener una visión identificada y aprobada por las partes involucradas permite incrementar las oportunidades de éxito en el proyecto. La visión debe de reflejar los requerimientos a alto nivel, las estrategias técnicas y el plan de liberación. [PP SP 1.2.1]

4.3.7 Asegurar el presupuesto del proyecto.

Un objetivo primordial en la fase inicial es conseguir el presupuesto para poder pasar a la fase de construcción.

El tipo de financiación de proyectos que se utiliza es *Stage Gate*, esta estrategia asegura el fondo por un periodo dado de tiempo y después es necesario conseguir más fondos para continuar con el proyecto. Es decir se realizan las estimaciones de tiempo y coste hasta un hito, que puede ser un prototipo funcional o una solución inicial con todos los atributos finales desarrollados; sobre eso se consiguen los fondos para su realización. Al llegar a cada uno de estos hitos se realizan la retroalimentación y se estima el tiempo y costes para el siguiente hito y su financiación.

Dependiendo del resultado de estas negociaciones se llega a un nivel de recursos y de trabajo posible para llegar al hito definido. [PP SP 3.2]

4.3.8 Identificar y abordar el riesgo.

Una estrategia importante para incrementar las oportunidades de éxito es entender y mitigar los riesgos a los que se pudiera enfrentar el proyecto.

Aplicando técnicas de *Lluvia de Ideas* y de *Grupo focal* se identifican los riegos involucrados en el proyecto. Se fomenta la creación de un repositorio de "Lecciones Aprendidas", el cual también será consultado en la identificación de riegos; para evitar caer de nuevo en los mismos problemas. [PP SP 2.2.1]

Los riesgos identificados son documentados usando la plantilla que se encuentra en la sección de plantillas "4.7.3 Bitácora de Riesgos del Proyecto", para su consideración y seguimiento. Este documento será almacenado en un repositorio TFS para mantener la información de las versiones y su seguimiento [PP SP 2.2.2, 2.2.4]

Para comprobar que se identificaron todos los riesgos de forma correcta, los documentos de riesgos deberán de ser revisados y aprobados por las partes involucradas. Deberá tener plasmada la firma de un superior del *Scrum Master* y del propietario del producto para ser marcados como terminados. [PP SP 2.2.3]

4.3.9 Establecer el ambiente.

Después de conseguir la aprobación de las partes involucradas, el equipo necesitara un área de trabajo y las herramientas necesarias para construir el proyecto (Ordenadores, software de desarrollo, ambientes de integración, de pruebas, etc.). Algunas de estos elementos pueden ya encontrarse en la compañía pero otras tendrán que ser compradas o alquiladas. [PP SP 1.4.2]

Es necesario tener instalada un servidor de *Microsoft Team Foundation Server* con la plantilla del proceso *Visual Studio Scrum*. Al ser ésta la metodología y herramienta estándar, lo normal sería que ya estuviera instalada y solo fuera necesario crear el proyecto en la colección correspondiente.

Además se deben planear los futuros requerimientos planeados durante el tiempo, como pueden ser: el ingreso planeado de otro desarrollador o el transporte requerido para llevar el equipo al lugar de trabajo del cliente para instalar el producto. [PP SP 2.4.4, 2.4.5]

4.3.9.1 Establecer Criterio de Gestión de Configuración

La Gestión de Configuración se realiza mediante la herramienta de control de versiones de TFS, los elementos de configuración que deben de ser alimentados en el control de versiones son: [CM SP 1.1.1, 1.1.3]

Hardware: La descripción del hardware en donde se realizaran las pruebas y que será definido como requerimientos mínimos para que la aplicación funcione de forma correcta; incluyendo: procesador, memoria, cantidad de almacenamiento disponible. Lo mismo será para el ambiente requerido para que el proyecto sea realizado por el equipo de desarrollo.

Software: Versión sistema operativo número de *build*, parches instalados, software de terceros que requiere estar instalado en la máquina. Esta información será definida tanto para los requerimientos del usuario final como para los requerimientos de desarrollo.

Código Fuente: Todo el código fuente generado deberá de ser gestionado por el control de versiones.

Documentación de Planificación, Técnica y de Requerimientos: Toda la documentación recibida por los cliente, la generada en el proceso de planificación o documentación técnica deberá de ser almacenada y versionada en el repositorio de control de versión.

Team Foundation Server se encarga de generar identificadores únicos para cada elemento de configuración, pero es recomendable utilizar un identificador preestablecido por el equipo de desarrollo, para los documentos de la planificación y de los requerimientos para su fácil identificación y seguimiento; este identificador será complementario al creado por TFS. [CM SP 1.1.2]

Los datos de hardware se documentaran utilizando la plantilla "4.7.6 Configuración de Hardware y Herramientas", la cual será ingresada y actualizada en el control de versiones en cuanto se tengan identificados los elementos iniciales y en cada modificación. Los documentos de requerimientos y de planificación serán ingresados al ser generados o recibidos. Por último el código fuente se ingresa de forma automática cada vez que el desarrollador finalice un elemento de trabajo y sea mandado al servidor para su la creación del *build*. [CM SP 1.1.4]

TFS permite la documentación de las relaciones entre los elementos de configuración, entre los que se encuentra la relación de padre/hijo, predecesor/sucesor, caso de prueba, de poseer los mismos pasos o simplemente una relación diferente mediante la posibilidad de detallarlo con texto libre. Se deberán de usar estas relaciones cuando sean visualizadas en los elementos de configuración, aun y no ser de uso obligatorio por la herramienta. [CM SP 1.1.6]

La herramienta registra de forma automática el usuario que está efectuando el ingreso del elemento de gestión. [CM SP 1.1.5]

Se deberá de identificar al personal técnico dentro y fuera del proyecto que fungirá como la Junta para el Control de Configuración. Esta Junta deberá de aprobar los cambios en la línea base de los elementos de configuración y será la responsable de gestionar el servidor de versiones de aplicación (*Build Server*). [CM SP 1.3.1]

4.3.9.2 Gestión de Datos

Se deberá llevar una bitácora de todas las formas de datos que son recibidas como insumo al proceso durante el desarrollo, para esto se utilizará la plantilla "4.7.5 Bitácora de Gestión de Datos". Esta bitácora contendrá el formato de los datos, su nivel de privacidad y las medidas de seguridad que deberán de ser aplicadas para preservar su privacidad. Como política general todos los documentos físicos que no sean públicos deberán ser mantenidos bajo llave por el *Scrum Master*. Los datos electrónicos serán guardados en TFS usando el nivel de acceso establecido en la bitácora, permitiendo su acceso a los desarrolladores con un nivel igual o superior al requerido y a las demás personas involucradas en el proyecto con la autoridad necesaria. [PP SP 2.3.1, 2.3.2, 2.3.4, 2.3.5]

Los datos producidos durante el desarrollo del proyecto también formarán parte de la bitácora y será definido su nivel de privacidad. [PP SP 2.3.3]

4.3.10 Formar al equipo de trabajo

La formación necesaria para que las partes involucradas puedan llevar el proyecto se realizará siempre antes de iniciar la fase de construcción.

Después de tener las historias de usuario identificadas en la *Estructura de Descomposición de Trabajo (WBS)*, se identifican las habilidades y conocimiento necesario para llevarlas a cabo. En la plantilla "4.7.1 Plan de Gestión del Proyecto (PMP)" bajo el apartado de personal, se enumerará el personal necesario para la elaboración del proyecto, incluyendo su rol dentro del proyecto, las habilidades requeridas y de ser necesario la forma en que adquirirán las habilidades para el proyecto. [PP SP 2.5.1, 2.5.2, 2.5.3, 2.5.4]

Además de la formación específica para el proyecto, es necesario que todos los miembros de los equipo de desarrollo *Scrum* tengan una formación básica en cuestiones de control de calidad, para mantener un nivel mínimo requerido y poder realizar evaluaciones a sus colegas. [PPQA SP 1.1.1]

4.3.11 Establecer las cuestiones de medición.

Se establece el ambiente necesario para desarrollarlo y mantener la capacidad de tomar mediciones para apoyar la toma de decisiones en el proyecto.

Para respaldar el tiempo invertido en la toma de mediciones, lo primero que se debe de crear son los objetivos por los cuales se deberán de tomar estas mediciones. En la plantilla "4.7.1 Plan de Gestión del Proyecto (PMP)" en la sección 5, se describirá el objetivo por el cual se tomaran las métricas; así como las métricas a recolectar, cuando serán recolectadas, como serán analizadas, almacenadas y comunicadas. [MA SP 1.1.1, 1.2.1, 1.4.3, 1.4.6]

Se revisa si existen mediciones que se estén llevando a cabo, tal vez dentro de un ámbito empresarial, que pudieran ser usadas para alcanzar alguno de los objetivos establecidos. [MA SP 1.2.3, 1.3.1] Después de establecer qué medidas ya se están generado, se establecen cuáles son las mediciones requeridas que no son generadas y recabadas. [MA SP 1.3.2]

Para cada una de las definiciones de medición especificada, se genera una definición operacional precisa y que no dé lugar a ambigüedades; la cual tiene como propósito que la medida sea repetible y que sus características sean documentadas. Entre estas características se encuentran: que debe ser medido, como será medido, que unidades de medida se usaran y que debe de ser incluido o excluido. Esto será documentado en la plantilla "4.7.9 Recolección y Análisis de Métricas". [MA SP 1.2.4]

Si bien esta plantilla ya tiene establecidos los objetivos básicos de la recolección de métricas, así como las métricas mínimas a levantar, el *Scrum Master* puede agregar las métricas extras que crea pertinentes.

Debido al coste en recursos que representa la toma de mediciones es imposible utilizar todas las mediciones disponibles, por lo que es necesario priorizar las mediciones y tomar la decisión de cuáles será recolectadas y cuáles no. [MA SP 1.1.2]

Los documentos generados a partir de las plantillas de "Métricas y del Plan de Gestión" serán almacenados en TFS, permitiendo su disponibilidad y su rastreo para futuras consultas; además de ser versionado por la herramienta para su registro de cambios. [MA SP 1.1.3, 1.1.5, 1.2.2, 1.3.3]

Se crean guías que establecen la forma en la que la recolección de datos es realizada. Existen algunas mediciones que se crean de forma automática con las herramientas utilizadas en la gestión del proyecto, por ejemplo: Los costes y tiempos de los productos de trabajo son generados de forma automática por TFS al alimentar los valores de los puntos de historia y las asignaciones de los recursos. Estas mediciones deberán de quedar registradas en el documento de Mediciones. [MA SP 1.3.1, 1.3.4, 1.3.5]

4.3.12 Establecer ambiente para el aseguramiento de la calidad

Justo ante de terminar esta fase se asignara un Entrenador de Calidad, este rol será definido en el apartado "4.4.3 Equipo Scrum". Este entrenador será responsable de dirigir el proceso del Aseguramiento de la Calidad en el proyecto.

Se crea un Plan de Calidad, el cual forma parte del "4.7.1 Plan de Gestión del Proyecto (PMP)". En éste se incluyen los productos y procesos que serán revisados para asegurar su calidad. [PPQA SP 1.2.1]

Además se crean las listas de verificación y de mínimos, utilizando las plantillas de la sección 4.7.7, las cuales serán creadas y usadas por el Entrenador de Calidad para asegurarse que el proyecto está siguiendo los procesos y construyendo sus productos de acuerdo al plan establecido. [PPQA 1.1.1, 1.1.2]

4.3.13 Herramientas

4.3.13.1 Microsoft Team Foundation Server

Microsoft Team Foundation Server (TFS) es un producto que provee gestión de código fuente, generación de informes, gestión de requerimientos, gestión de proyectos, generación automática de versiones de la aplicación *–builds–*, laboratorio, pruebas y gestión de liberaciones. Abarca completamente la gestión del ciclo de vida de la aplicación.

Está integrado con un sistema de gestión de usuario, que permite crear grupos de usuario con privilegios definidos para acceder a los diversos elementos de configuración, incluyendo informes, elementos de software y versiones compiladas de la aplicación. La información está disponible para todo el personal con los privilegios necesario a ello de forma inmediata al subir el nuevo elemento de configuración. [CM SP 1.2.1, 1.2.2, 1.2.3, 1.2.4, 1.2.5, 1.2.6, 1.2.7, 1.2.8. 3.1.2, 3.1.5, 3.1.6]

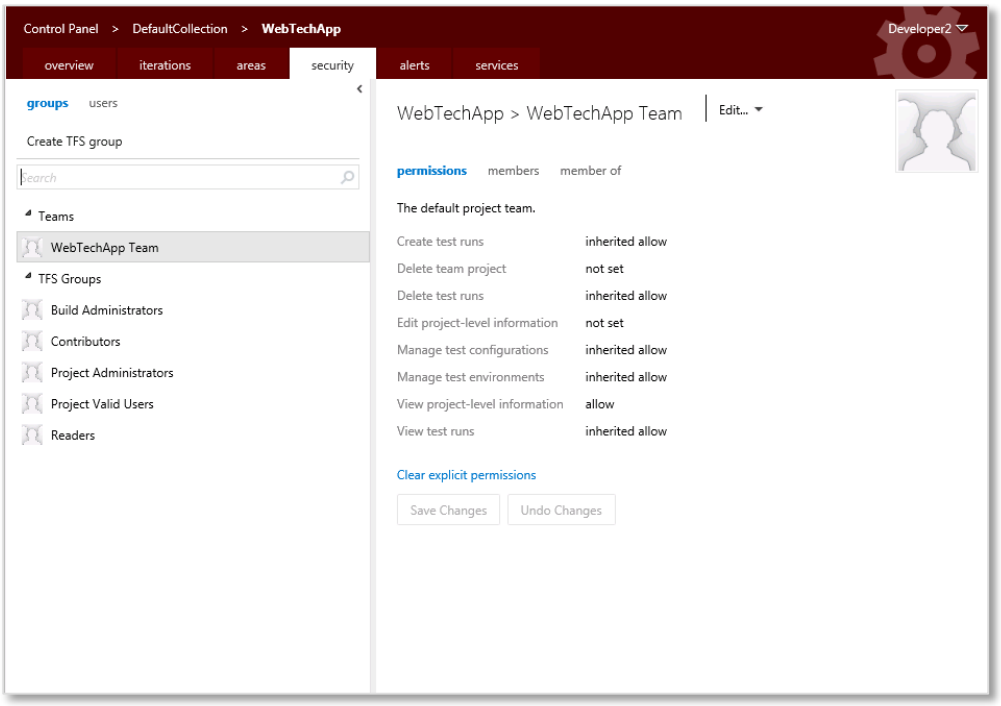

Figura 6 Estructura de permisos de TFS

Éste mismo manejo de seguridad aplica para todos los elementos ingresados en la herramienta, como los documentos con las mediciones que serán almacenados para poner a disposición de las partes involucradas. Estos documentos solo estarán disponibles para el personal apropiado, asignándoles los privilegios necesarios para acceder a los elementos en cuestión y así evitando un uso inadecuado de los mismos. [MA SP 2.3.3, 2.3.4]

Source Control: Permite mantener versiones de los elementos de configuración, incluyendo código fuente y archivos en cualquier formato (Excel, Word, JPG, PDF, etc.). Además de permitir la comparación línea por línea entre dos archivos de texto plano, con la opción de comparar 2 versiones del mismo archivo.

En esta fase será utilizado para almacenar todos los documentos de planificación que sean producidos para mantener un historial de las versiones, incluyendo: quien lo creo o modifico, cuando sucedió su última modificación y cuáles fueron los cambios exactos en el contenido. [CM SP 2.2.1, 2.2.5]

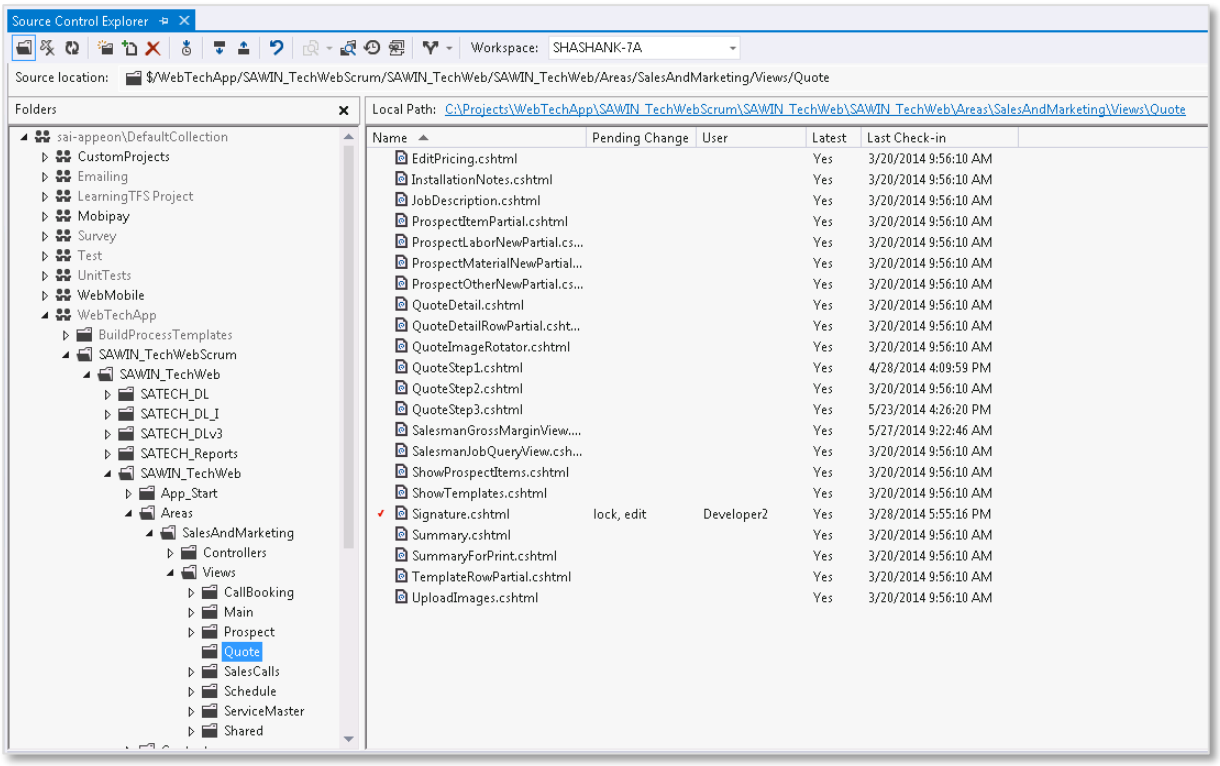

Figura 7 Herramienta de control de versión de TFS

Todos los cambios registrados en el control de versiones se hacen mediante actividades de *check-in* y *check-out* por definición de la herramienta. [CM SP 2.2.3]

Application Lifecycle Management (ALM): Esta aplicación, que forma parte de TFS, tiene como función la gestión de proyectos *Scrum*. Permite la creación y gestión del *backlog de* producto, el *Sprint* y las tareas del *Sprint*. En esta fase se usara para dar entrada a todos los elementos del *backlog* del producto que sean identificados en la planificación del proyecto.

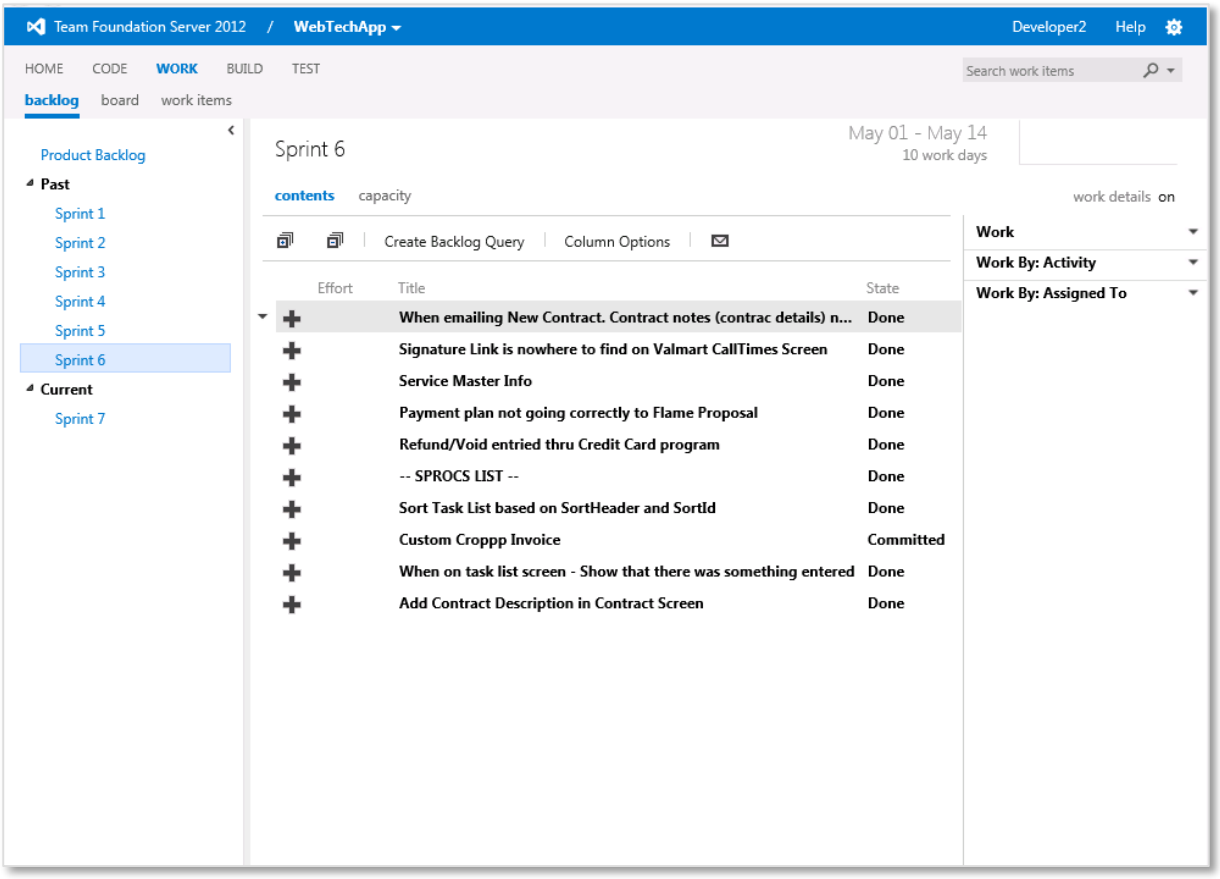

Figura 8 Contenido de un Sprint en TFS

TFS permite la documentación de las relaciones entre los elementos de *Scrum*, entre los que se encuentra la relación de padre/hijo, predecesor/sucesor, caso de prueba, elementos que comparten los mismos pasos, o simplemente una relación diferente mediante la posibilidad de detallarlo con texto libre. Se deberá de usar estas relaciones cuando sean visualizadas en los elementos de configuración, aunque la herramienta no lo obliga. [REQM SP 1.4.1, 1.4.2

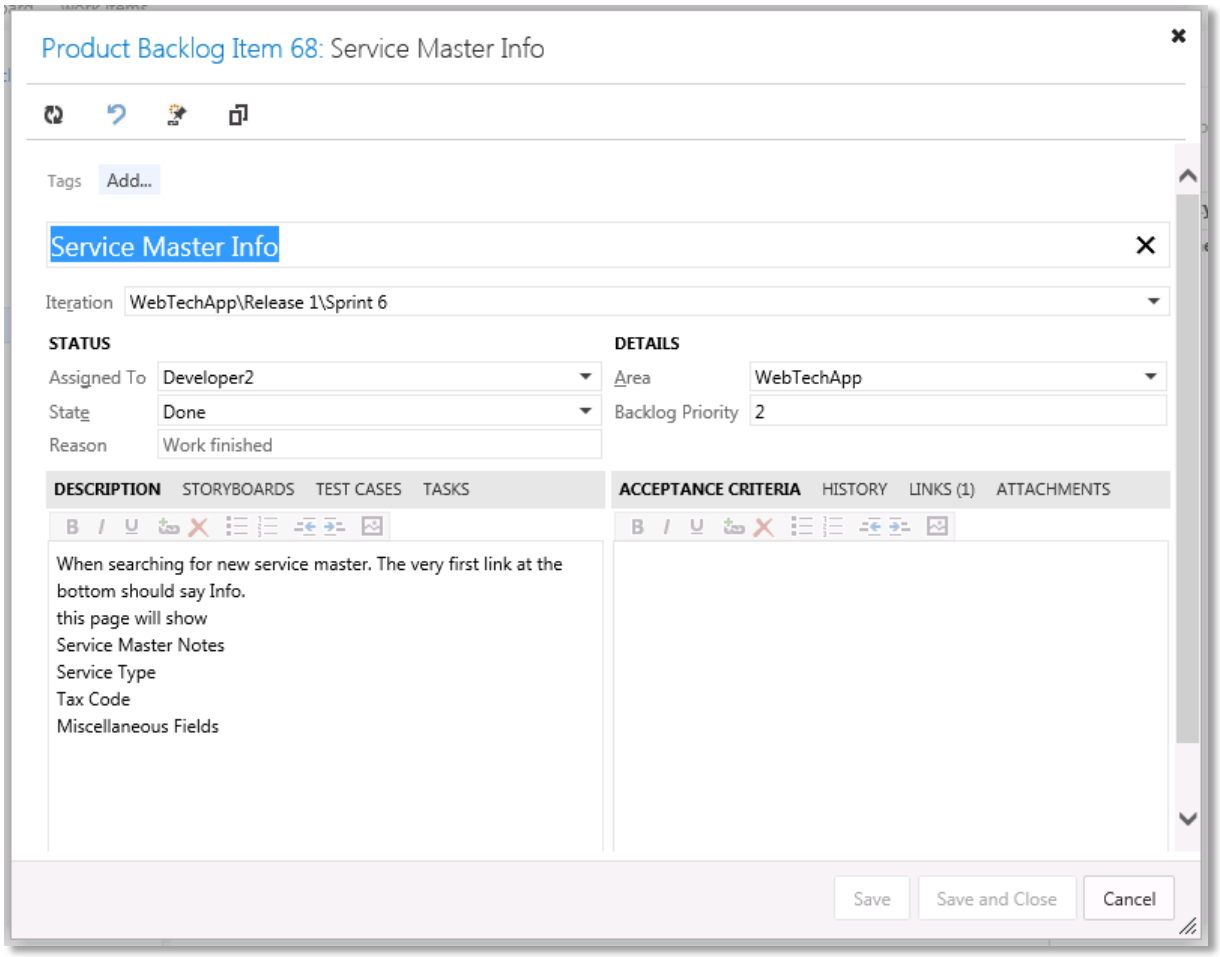

Figura 9 Elemento del backlog en TFS

Build Server: Para asegurarse de la consistencia de las versiones y que solo los elementos que están ingresados en el sistema de configuración serán usados para generar la versión del producto se utiliza el servidor de compilación versiones o *build server* en inglés. Estas versiones son almacenadas y pueden ser obtenidas en cualquier momento por el personal con los permisos necesarios para hacerlo. Al ser la única forma permitida para construir nuevas versiones, toda la documentación sobre la línea base de configuración se encuentra en TFS. [CM SP 1.3.2, 1.3.3, 1.3.4, 3.1.3, 3.1.4]

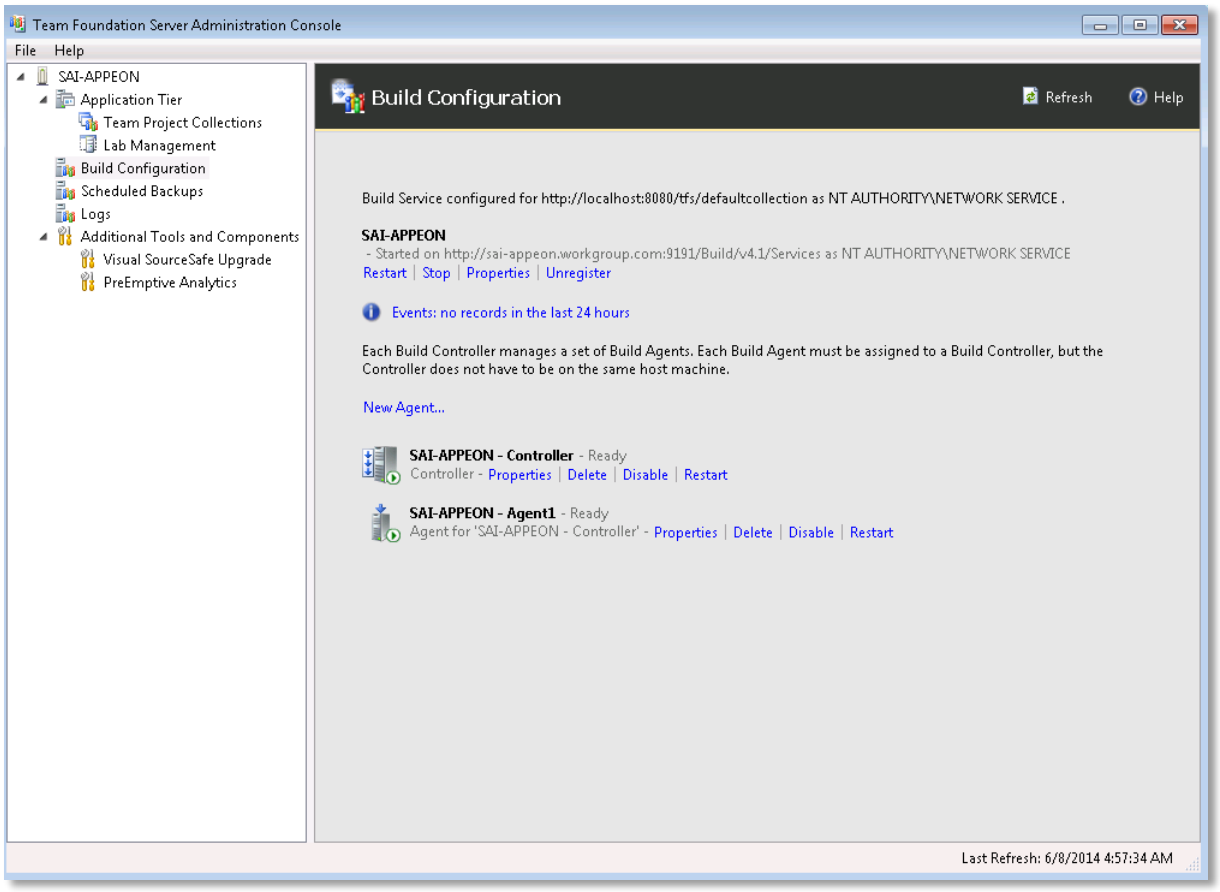

Figura 10 Visualización del Servidor de Builds de TFS

Release Managment: Esta capacidad de TFS permite a los equipo realizar implementaciones controladas, gestionados por el flujo de trabajo en los ambientes de desarrollo, pruebas y producción; además de proporcionar tableros de control para monitorizar el progreso de las versiones del producto.

Sharepoint: Es un marco para aplicaciones de web, que permite la creación de blogs y wikis que se pueden usar para publicar y compartir la información relacionada con el proyecto.

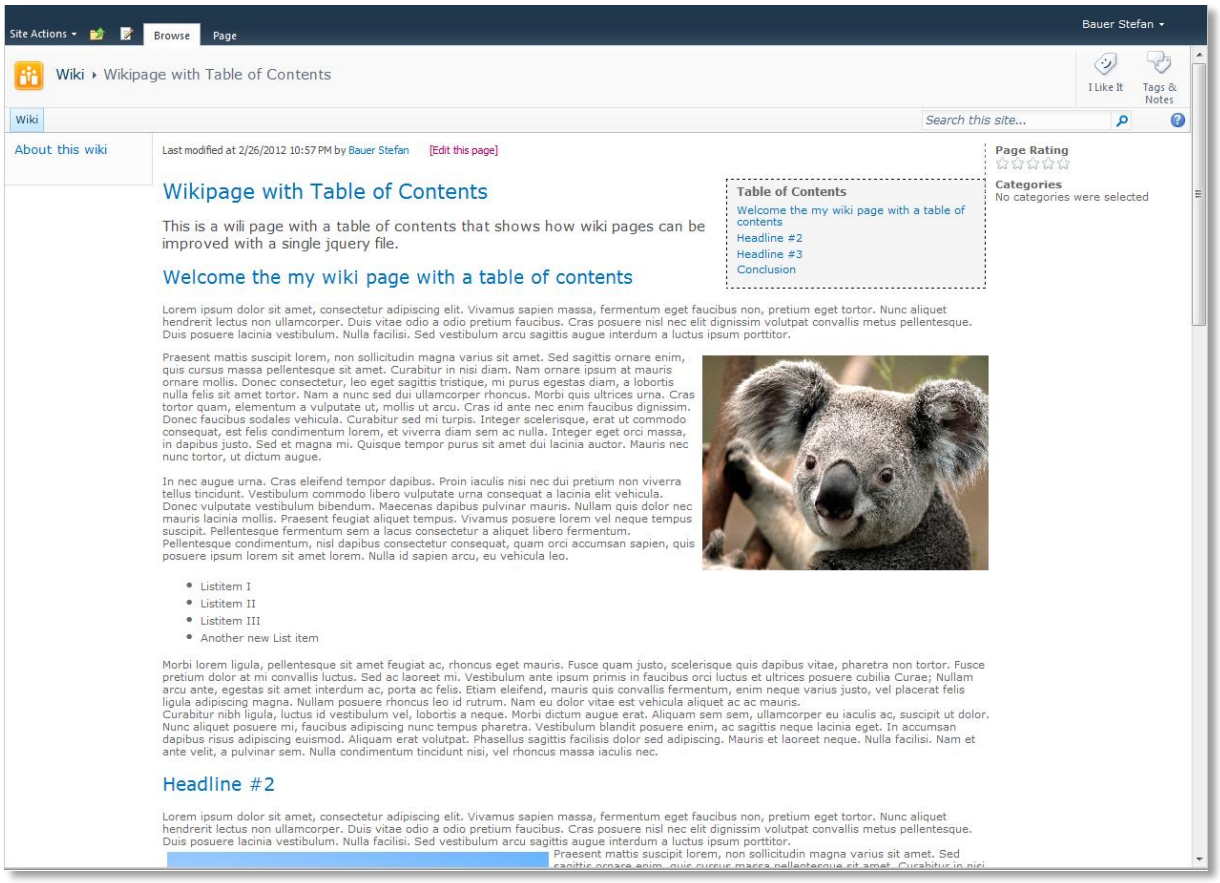

Figura 11 Wiki sobre Sharepoint

4.3.14 Recomendaciones para equipos distribuidos

Si el equipo es distribuido la forma de gestionar el *Scrum* será utilizando equipos *Scrum* totalmente integrados.

Según explica Woodward, Surdek, & Ganis (2010) en este modelo cada equipo *Scrum* tiene miembros en múltiples lugares. Pudiera ser que todos los encargados de las pruebas se encuentran en una localidad mientras que todos los desarrolladores se encuentran en otra localidad o que los miembros de los equipos multifuncionales cambian de ubicación constantemente. Cuando múltiples equipos *Scrum* están trabajando juntos para entregar un proyecto, todos los equipos tienen el requerimiento de un único incremento con pruebas al final de cada *Sprint*.

Los equipos *Scrum* totalmente integrados son comunes en IBM. En donde equipos *Scrum* multifuncionales tienen miembros en 3 o más localidades, con múltiples zonas horario y con miembros cuyo idioma nativo no es el idioma usado en las juntas de equipo.

Más información de equipos *Scrum* totalmente integrados se puede encontrar en *A Practical Guide to Distributed Scrum* (Woodward, Surdek, & Ganis, 2010).

4.4 DESARROLLO/CONSTRUCCIÓN

Esta fase está basada en la metodología de desarrollo *Scrum* (Schwaber & Sutherland, 2011).

Para hacer la distinción entre los procesos *Scrum* originales y los agregados o modificados para esta guía se utilizara el término de "*Scrum* extendido".

4.4.1 Objetivos

Adaptado de (Ambler & Lines, 2012). Los objetivos de esta fase son los siguientes: producir una solución potencial a ser utilizada, abordar las necesidades cambiantes de las partes involucradas, acercarse a un producto listo para implementación, mantener o mejorar los niveles existentes de calidad y probar la arquitectura en las fases tempranas.

Producir una solución potencial a ser consumida. Es necesario tener un producto de software funcionando tan rápido como sea posible. Enseñar los avances en forma de un producto funcionando permite a los usuarios establecer si se va en la dirección correcta o no. En ocasiones la retroalimentación será que no se va en la dirección correcta, pero esto es un resultado muchas veces mejor que esperar al final del proyecto, ya que permite adaptar el producto lo más pronto posible y evita perder tiempo desarrollando un producto que no es lo que espera el cliente.

Abordar las necesidades cambiantes de las partes involucradas. El *backlog* del producto es una lista de elementos de trabajo que necesitan ser completados para entregar la solución al cliente. Está pensada para ser dinámica y reflejar las prioridades actuales del cliente, de tal forma que todo lo que no esté construido y se encuentre en la lista puede ser cambiado en cualquier momento. A diferencia de los métodos tradicionales donde es necesario especificar los requerimientos de forma exhaustiva desde un inicio y los cambios son castigados con procedimientos de control de cambios. De tal forma, que permitiendo a los propietarios del producto cambiar requerimientos se entrega a los usuarios una solución que realmente es valiosa para ellos. Para dar una idea a los clientes de lo que pueden esperar al final de la última iteración planeada, la lista de elementos de trabajo debe de mostrar grupos de elementos que serán desarrollados en cada una de las iteración por realizar; de tal forma que los clientes entiendan que los elementos que se encuentren debajo de cierto punto en la lista están fuera del alcance.

Acercarse a un producto listo para implementación. Al final de cada iteración se tiene una solución que es potencialmente consumible por los clientes. Por consumible se entiende que el producto no solo se puede entregar sino que puede ser usado por el cliente. Solo cuando los usuarios están realmente utilizando la solución pueden apreciar el retorno de la inversión.

Mantener o mejorar los niveles existentes de calidad. Al tener una solución consumible en las fases tempranas del ciclo de vida, se obtiene como valor añadido un alto nivel de calidad. Esto es debido a que este nivel de calidad es requerido para que una solución pueda ser utilizada, por lo tanto este nivel es obtenido desde la primera entrega hasta la última. Esto es muy difícil

de lograr con los métodos tradicionales ya que la solución no se utiliza hasta su entrega al final del proyecto.

Probar la arquitectura en las fases tempranas. Es necesario probar las arquitecturas en una etapa temprana del proyecto, ya que las arquitecturas fallidas son una fuente común de fracasos en los proyectos. Las aplicaciones grandes son muy difíciles de ser cambiadas de arquitectura, por eso se sugiere que los elementos de trabajo que toquen puntos clave de la arquitectura sea tratado con prioridad alta en el *backlog* del producto.

4.4.2 Proceso

En la siguiente figura se puede apreciar la secuencia de eventos que componen esta fase del proceso de desarrollo.

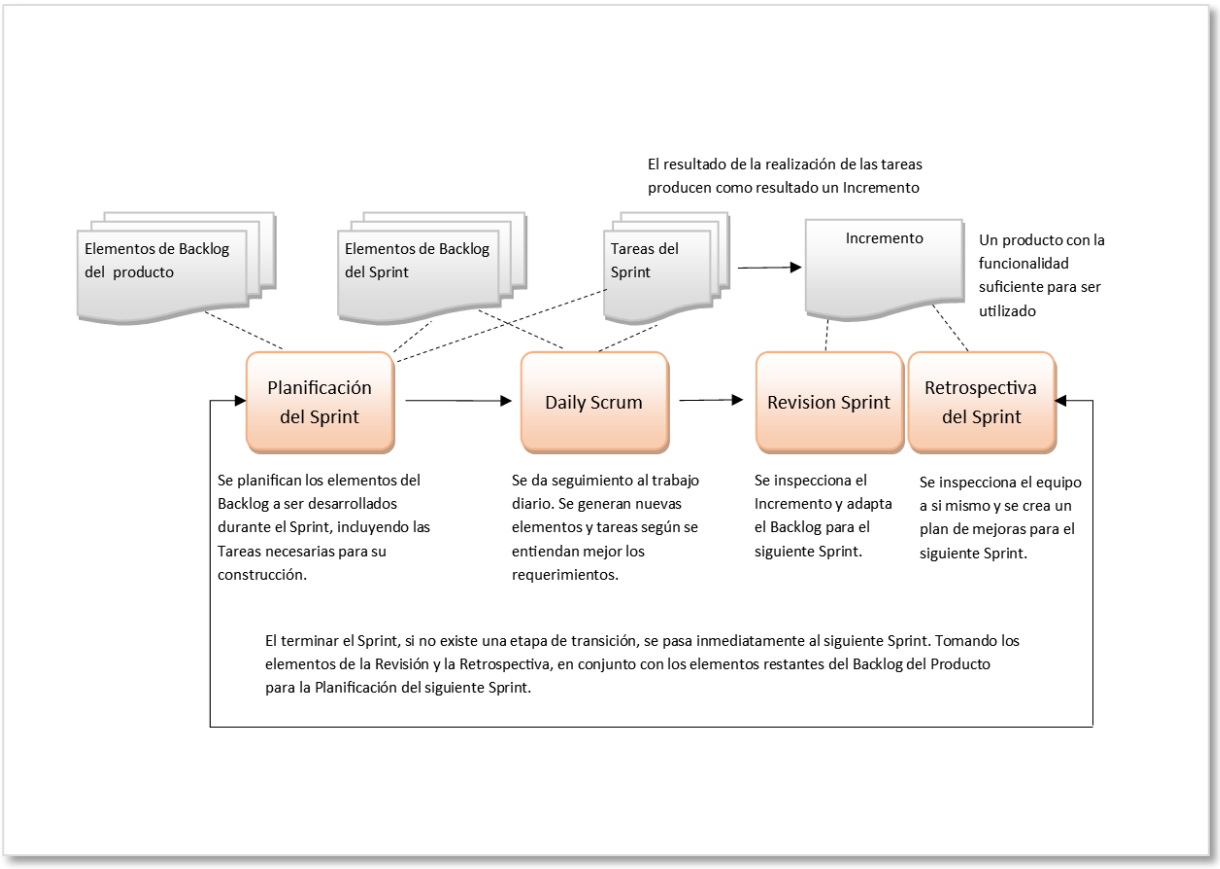

Figura 12 Secuencia de actividades de la fase de Desarrollo

4.4.3 Equipo Scrum

Adaptado de (Schwaber & Sutherland, 2011).

Los equipos *Scrum* son auto-organizados y multifuncionales, por lo tanto conocen la mejor manera de realizar su trabajo y no dependen de la dirección de personas fuera del equipo. Al ser multifuncionales poseen las habilidades necesarias para realizar su trabajo sin depender de gente externa al equipo.

Las roles de un equipo *Scrum* extendido son: el propietario del producto, el equipo de desarrollo, el *Scrum* master y el entrenador de calidad. [PP SP 2.4.3]*

*Como se lee en las instrucciones de uso de la guía, estos párrafos hacen referencia a las prácticas específicas PP SP 2.4.3. Como ejemplo se muestra a continuación un extracto del apartado 4.1.3 mostrando la práctica específica. En el resto de las anotaciones de este inciso se omitirán las referencias de ejemplo, las cuales deberán de ser consultadas en el apartado 4.1.3.

PP -Planificación del Proyecto	**Sección***
SG 2 Desarrollar un plan de proyecto	
SP 2.4 Planificar los Recursos del Proyecto	
3. Determinar los requisitos de personal.	4.3.2, 4.4.3, 4.7.1(2)

4.4.3.1 Propietario de producto

El propietario del producto es el responsable de maximizar el valor del producto y el trabajo del equipo de desarrollo. El propietario del producto es el responsable de gestionar el *backlog* del producto, haciendo entre otras cosas:

- Ingresar los elementos de trabajo al *backlog* del producto.
- Ordenar las prioridades de los elementos de trabajo del *backlog* del producto para lograr las misiones y metas.
- Optimizar el valor del trabajo realizado por el equipo de desarrollo.
- Asegurarse que el *backlog* del producto es visible, transparente y claro para todos, y mostrar el *backlog* en el siguiente trabajo del equipo de *Scrum*.
- Asegurarse que el equipo de desarrollo entiende los elementos del *backlog* del producto hasta el nivel requerido. [REQM SP 1.1.4]

El propietario del producto es el responsable de las tareas anteriores, aunque no sean realizadas por él y que tenga a un miembro del equipo de desarrollo haciendo parte de ellas.

El propietario del producto es una persona no un comité, pero puede representar los deseos de un comité en el *backlog* del producto, si alguien desea cambiar una elemento de trabajo del *backlog* deben de dirigirse al propietario del producto. [REQM SP 1.1.1]

El propietario del producto es respetado por toda la organización, el contenido y la prioridad que él le asigna al *backlog* es visible por toda la compañía.

Nadie tiene permitido decirle al equipo de desarrollo que trabaje en un conjunto de requerimientos diferente al establecido por el propietario, y el equipo de desarrollo no tiene permitido trabajar en peticiones de alguien diferente al propietario del producto. [REQM SP 1.1.1]

4.4.3.2 Equipo de desarrollo.

El equipo de desarrollo está formado por los profesionales que hacen las labores necesarias para entregar una potencial *Incremento* de trabajo, convertido en versión, al final de cada *Sprint*. Solo los miembros del equipo de desarrollo pueden crear un *Incremento*.

Los equipos de desarrollo tienen las siguientes características:

- Auto-organizados, ni siquiera el *Scrum Master* dice al equipo de desarrollo como convertir el *backlog* del producto en Incrementos de funcionalidad potencialmente liberable.
- Multifuncionales, el personal que forma el equipo de desarrollo tiene todas las habilidades necesarias para crear los *Incrementos* del producto.
- *Scrum* no reconoce títulos para los miembros del equipo de desarrollo.
- *Scrum* no reconoce equipo dentro del equipo.
- El equipo puede tener miembros con habilidades especializadas y áreas de especialización, pero las obligaciones pertenecen al equipo como un todo.

El equipo debe de ser lo suficientemente pequeño para permanecer ágil y lo suficientemente grande para completar un trabajo significativo en cada Sprint, deseablemente entre 3 y 9 miembros. El *Scrum Master* no cuenta en esta sugerencia a menos que realice además del rol de *Scrum Master* labores relacionadas con la creación de los *Incrementos*.

4.4.3.3 Scrum Master

El *Scrum Master* es responsable de asegurar que la metodología *Scrum* es entendida y representada, asegurándose que el equipo sigue la teoría, las prácticas y reglas del *Scrum*. [PPQA SP 1.1.1]

El *Scrum Master* ayuda a las personas fuera del equipo *Scrum* a entender que interacciones con el equipo son benéficas y cuáles no, ayuda a todos a cambiar las interacciones negativas para maximizar el valor creado por el equipo *Scrum*; sirviendo un rol de líder-sirviente al equipo.

Servicio del *Scrum Master* al propietario del producto.

- Encontrar técnicas efectivas de gestión del *backlog* del producto.
- Ayudar al equipo a entender las necesidades de los elementos del *backlog* del producto de forma clara y concisa.
- Entender la planificación de los productos en un ambiente empírico.
- Asegurarse que el propietario del producto entienda como ordenar los productos del *backlog* para maximizar el valor.
- Entender y practicar agilidad.
- Facilitar eventos *Scrum* según sea pedidos o necesitados.

Servicio del Scrum Master al equipo Scrum.

- Formar al equipo de desarrollo a auto-organizarse y ser multifuncional.

- Ayudar al equipo de desarrollo a crear productos de alto valor.
- Remover impedimentos al progreso del Equipo de desarrollo.
- Facilitar eventos *Scrum* como vayan siendo pedidos o necesitados.
- Formar al equipo de desarrollo en ambientes organizacionales en donde *Scrum* no está adoptado y entendido al 100%.

Servicio de Scrum Master a la Organización.

- Liderar y formar a la organización en su adopción de *Scrum*.
- Planear implementaciones *Scrum* dentro de la organización.
- Ayudar a los empleados y partes involucradas a entender y seguir *Scrum*, así como el desarrollo empírico del producto.
- Causar cambios que incrementen la productividad del equipo *Scrum*.
- Trabajar con otros *Scrum Masters* para incrementar la efectividad de una aplicación *Scrum* dentro de la organización.

4.4.3.4 Entrenador de Aseguramiento de Calidad

El Entrenador será un miembro regular del equipo de desarrollo que realizara labores de aseguramiento de calidad en proyectos puntuales. La idea es que todos los desarrolladores se concienticen de la importancia del aseguramiento de la calidad y a su vez sirvan de mentor a los demás miembros del equipo. El entrenador no podrá ejercer el rol de desarrollador dentro del proyecto en el cual está capacitando, para asegurar la independencia de los resultados, no obstante puede ejercer labores simultáneas de desarrollador en otro proyecto.

La diferencia principal entre el entrenador de aseguramiento de calidad y un auditor, es que la responsabilidad del entrenador es el asegurarse que la calidad del proyecto sea la requerida por las normativas de calidad de la empresa y no el de encontrar incumplimientos y reportarlos. Solo en el caso específico que sea imposible resolver la variación dentro del equipo mismo, serán ventiladas las carencias fuera del equipo al nivel de gerencia para facilitar la solución.

Los desarrolladores experimentados serán los que tomen el rol de Entrenador de Calidad, siendo la intención que todos los desarrolladores de la compañía pasen por este rol en algún momento.

4.4.4 Backlog

El *backlog* del producto es una lista ordenada con todo lo que pudiera ser necesario en el desarrollo del producto y es la única fuente de requerimientos y cambios a realizar sobre el producto. [CM SP 2.1.1; REQM 1.3.1]

El *backlog* del producto es creado por primera vez en la fase inicial, pero debe ser mantenido por el propietario del producto durante todo el tiempo del proyecto. Como se vio en la descripción de las responsabilidades del propietario en el apartado 4.4.3.1.

Siempre se encuentra en constante desarrollo y solo será completado si el producto al que pertenece deja de existir. Los primeros elementos que se añaden al *backlog* del producto son los conocidos y mejor entendidos inicialmente, pero al ser dinámico, cambia constantemente para identificar que necesita el producto para ser apropiado, competitivo y útil.

El *backlog* del producto enumera todas las cualidades, funciones, requerimientos, mejoras y arreglos que constituyen los cambios por hacer al producto en versiones futuras. Los atributos esenciales de un elemento del *backlog* del producto son: descripción, orden, estimado y valor. [REQM SP 1.1.2]

El refinamiento del *backlog* del producto es la acción de agregar detalles, estimaciones y orden a los elementos del *backlog*. Éste es un proceso continuo en el cual el propietario del producto y el equipo de desarrollo colaboran para detallar los elementos del *backlog* del producto. Durante la refinación los elementos son revisados y repasados.

Los elementos con un orden superior –mayor prioridad– son usualmente más claros y detallados que los elementos de orden inferior. Entre más claridad y detalle tengan los elementos, se obtienen estimados más precisos; de tal forma que los elementos a ser trabajados en el siguiente *backlog* por lo general serán los que tienen más detalle y por lo mismo la estimación del *Sprint* será muy precisa.

El equipo de desarrollo es el responsable de todos los estimados, el propietario del producto puede influenciar en el equipo de desarrollo ayudándolos a entender y seleccionar las convenciones, pero el estimado final lo debe de hacer las personas que realizan el trabajo.

Monitorizar progreso hacia un objetivo.

En cualquier momento el tiempo restante para alcanzar cierto objetivo puede ser consultado. El Propietario de Producto rastrea este tiempo restante cuando menos en cada *Revisión de Sprint*, compara la cantidad de trabajo restante contra el trabajo realizado en las pasadas revisiones de *Sprint* para ponderar el progreso hacia la finalización del trabajo estimado para alcanzar el objetivo. Esta información debe de hacerse transparente a todas las partes interesadas.

Existen varias prácticas que permiten pronosticar el progreso como *burn-downs, burn-ups*[1] *y flujos acumulativos*[2], pero ninguna de ellas remplaza la importancia de la experiencia. En ambientes complejos lo que va suceder es desconocido, solo lo que ya ha pasado puede ser utilizado como referencia para tomar decisiones del futuro.

Los elementos de trabajo que forman parte del *backlog* del producto serán registrados en TFS, lo cual permita su consulta por cualquier personal que tenga los privilegios necesarios. Lo cual

[1] http://brodzinski.com/2012/10/burn-up-better-burn-down.html

[2] http://agilesherpa.org/agile_coach/metrics/cumulative_flow/

también permitirá mantener un historial de los elementos que componen el *backlog* del producto y la razón de la creación del elemento. [REQM SP 1.3.2, 1.3.4]

4.4.5 Sprint

El *Sprint* es un lapso de tiempo de un mes o menos en el cual un *Incremento* de producto es creado, este *Incremento* debe de ser funcional, usable y potencialmente pudiera ser liberado al cliente. Lo óptimo es que todos los *Sprint* tengan una duración y esfuerzo de desarrollo equivalentes. Un *Sprint* empieza inmediatamente al terminar el anterior, con la variante de un *Sprint* de implementación cuando por la labor requerida para poner en marcha el producto si así lo requiera. El *Sprint* de implementación será detallado en el apartado "4.54.5 Transición/Implantación proyecto".

Los *Sprints* están formados por la planificación del *Sprint*, los *Scrum Diarios*, el trabajo de desarrollo, las *Revisiones de Sprint* y la *Retrospectiva Sprint*.

Durante el *Sprint* sucede lo siguiente: no se realiza ningún cambio que pueda poner en peligro la meta del *Sprint*, las metas de calidad no se reducen y el alcance puede ser renegociado entre el propietario del producto y el equipo de desarrollo si existiesen nuevos conocimientos sobre las actividades a realizar.

Durante el *Sprint* no se realizan nuevos compromisos de los que se hicieron al iniciar el *Sprint* por lo que no existe el riesgo de que un nuevo compromiso impacte a otro compromiso anterior. Los nuevos compromisos deben de esperar a la apertura del siguiente *Sprint*. [REQM SP 1.2.1]

La razón de limitar el *Sprint* a un mes calendario es debido a que con horizontes muy largos la definición de lo que se está construyendo puede cambiar, aumentar en complejidad e incrementar su riesgo. Esta corta duración permite inspeccionar y adaptar el progreso hacia la meta del *Sprint* cuando menos una vez al mes. Por lo anterior los *Sprints* además apoyan el propósito de reducir los riesgos al coste de un mes calendario.

4.4.5.1 PLANIFICACIÓN DEL SPRINT

La planificación del *Sprint* consiste en planear el trabajo a ser realizado durante el *Sprint*, se realiza mediante la colaboración de todo el *Equipo Scrum* [PP SP 2.4.2]. Esta planificación contesta las siguientes preguntas: ¿Qué puede ser entregado en el *Incremento* como resultado del próximo *Sprint*? Y ¿Cuál es el trabajo necesario para lograr el *Sprint*? [CM SP 2.1.2, 2.1.3, 2.1.4]

Cualquier cambio en la composición del *Equipo de Desarrollo* que participara en *Sprint* deberá ser documentado en la "4.7.4 Matriz de Partes Involucradas". [PMC SP 1.4.1, 1.4.2, 1.4.3]

Parte del trabajo a realizar en el *Sprint* vendrá de cuestiones que quedaron abiertas en el *Sprint* anterior. Estos elementos, de seguir siendo importantes para la aplicación, serán prioritarios y serán agregados al *Sprint* junto con las acciones correctivas que fueron definidas al ser analizados. [PMC SP 2.2.2, 2.3.1]

Igualmente las inconsistencia detectadas en la retrospectiva del *Sprint* anterior y formalizadas en eventos de *backlog* serán analizadas y sus acciones correctivas transformadas en tareas para el *Sprint*. De esta forma la planificación y los requerimientos generales se verán afectados al ser usado tiempo a resolver conflictos del *Sprint* anterior en este *Sprint*. [REQM 1.5.3, 1.5.4]

Lo mismo ocurre con las tareas que dependan de proveedores, los acuerdos con el proveedor deberán de ser tomado en cuenta en el momento de crear el plan y compromisos para el *Sprint*. [SAM SP 1.3.7]

Es responsabilidad del *Scrum Master* que la planificación del *Sprint* se realice respetando las guías establecidas por *Scrum*, no obstante es posible que el entrenador del aseguramiento de calidad asista a este evento y apoye al *Scrum Master* haciéndole saber de algún elemento que pudiera estar omitiendo o llevando de forma incorrecta. Para realizar esto el entrenador se apoya en los listados de verificación y de mínimos creados en la fase inicial. El entrenador toma notas para documentar las "no conformidades" y sus resoluciones. [PPQA SP 1.1.3, 1.1.4, 1.1.5, 2.1.1, 2.2.1, 2.2.2]

¿Qué puede ser hecho en el Sprint?

Se realiza una junta de planificación en donde el propietario del producto muestra el objetivo que el *Sprint* debe lograr y los elementos del *backlog* del producto que de ser completados en el *Sprint* lograrían el objetivo de propuesto.

Todo el equipo *Scrum* colabora para entender el trabajo del *Sprint*, las entradas de esta junta son el *backlog del producto*, el último *Incremento*, la capacidad proyectada del equipo de desarrollo durante el *Sprint* y el rendimiento pasado del equipo de desarrollo. [REQM SP 1.3.3] El número de elementos seleccionados del *backlog* del producto para el *Sprint* son decisión única del equipo de desarrollo, solo él puede ponderar lo que puede ser logrado en el próximo *Sprint*.

Cuando los elementos del *backlog* del producto a ser entregados en el próximo *Sprint* son elegidos, se pasa al siguiente paso que es la creación de la meta del *Sprint*. La meta del *Sprint* está compuesta por los objetivos que serán cumplidos en el *Sprint* y aportan una guía al equipo de desarrollo del porqué de la construcción del *Incremento*.

¿Cómo será desarrollado el trabajo elegido?

Los elementos del *backlog* del producto seleccionado para el *Sprint* junto con el plan de entrega de éstos son llamados *backlog de Sprint*.

El equipo de desarrollo empieza por diseñar el sistema y el trabajo necesario para convertir el *backlog* del producto en un *Incremento* funcional. Suficiente trabajo es planeado durante la planificación del *Sprint* para que el equipo de desarrollo pueda pronosticar lo que él cree puede ser entregado en el siguiente *Sprint*. El trabajo planeado para los primeros días de trabajo es descompuesto antes de finalizar la junta, comúnmente en unidades de un día o menos. El equipo de desarrollo se debe de organiza a sí mismo para realizar el trabajo del *backlog* del *Sprint*, tanto durante la planificación del *Sprint* como durante el *Sprint*.

El propietario del producto puede ayudar a aclarar los elementos del *backlog* del producto y hacer algunas concesiones, si el equipo de desarrollo cree que es demasiado o muy poco trabajo, éste puede ser renegociado con el propietario del producto. [REQM SP 1.1.3, 1.1.4]

También se puede invitar personal ajeno al equipo a aportar consejos técnicos o de dominio.

Al finalizar la planificación del *Sprint* el equipo de desarrollo debe de ser capaz de explicar al propietario del producto y al *Scrum Master* como pretenden organizarse para realizar el trabajo requerido para lograr la meta del *Sprint* y crear el *Incremento*.

La meta del Sprint proporciona un ideal colectivo sobre el que todos los miembros del equipo trabajan para lograr, en lugar de trabajar cada uno para lograr su trabajo de forma individual.

Si durante el trabajo en el Sprint se encuentra que el trabajo a realizar es diferente al visualizado en la planificación del *Sprint*, se renegocia con el propietario del producto un cambio de alcance.

4.4.5.2 Backlog del Sprint

Los elementos seleccionados del *backlog* del producto para ser desarrollados en el *Sprint*, además del plan para entregar el *Incremento* del producto y realizar la meta del *Sprint* son llamados *backlog del Sprint*. El *backlog* del *Sprint* se pronostica por los miembros del Equipo de desarrollo a partir de los elementos que estarán contenidos en el siguiente *Incremento* y el trabajo que será necesario para que esto ocurra. El detalle del *backlog* del *Sprint* deberá ser suficiente para que el progreso de las tareas sean entendidas en la junta del *Scrum diaria*.

El equipo de desarrollo puede modificar el *backlog del Sprint* durante el *Sprint* conforme se va aprendiendo más sobre el trabajo a realizar necesario para alcanzar la meta del *Sprint*. Estas modificaciones pueden ser de varios tipos: agregar elementos al *backlog del Sprint* cada que un nuevo trabajo es requerido, cada que un nuevo trabajo es realizado o completado el trabajo faltante es actualizado, y cuando un elemento se vuelve innecesario es removido. Solo el equipo de desarrollo puede modificar el *backlog del Sprint* durante el Sprint. [PMC SP 1.1.3]

Es posible ver el trabajo planeado a realizar por el equipo de desarrollo, además de lo que se está trabajando gracias a la transparencia alcanzada por el *backlog del Sprint*. En cualquier momento es posible visualizar el total de trabajo faltante para finalizar el Sprint Backlog, el equipo de desarrollo hace un seguimiento de estos faltantes cuando menos en cada junta de *Scrum Diario* y de esta forma se aumenta la posibilidad de lograr la meta del *Sprint*. Haciendo este seguimiento el equipo de desarrollo puede gestionar su progreso.

El desarrollo de software utilizando *Scrum* se basa en el compromiso a cumplir con el trabajo que cada involucrado determina que puede entregar a tiempo y en forma en cada sprint. Por lo tanto el compromiso del equipo de desarrollo se realiza durante la definición de cada sprint, en donde cada uno de los desarrolladores se comprometerá a realizar los trabajos que ellos mismos se saben capaces de lograr. [REQM SP 1.2.2] Los elementos comprometidos serán las tareas agregadas al *backlog del Sprint*, siempre haciendo constar la relación de las tareas con el

elemento del *backlog* (Historia) para cuyo cumplimiento es necesario la creación de la tarea en cuestión. [PP SP 3.3.1, 3.3.2, REQM SP 1.4.1, 1.4.2]

Será necesario involucrar a la alta dirección, tanto en la revisión de compromisos internos como en compromisos externos; principalmente en aquéllos en donde la alta dirección tenga la autoridad para hacer que estos compromisos sean cumplidos por partes externas al equipo de desarrollo y así buscar minimizar los riesgos de incumplimiento no controlables por el equipo de desarrollo. Estas revisiones se harán durante las juntas de definición de cada *Sprint* y en casos especiales cuando sea necesario fuera de estos hitos. [PP SP 3.3.3, 3.3.4]

Si el proyecto es parte de un proyecto más grande o tiene integraciones con otros proyectos, las personas responsables de la creación de las interfaces deberán de comprometerse a entregar en los tiempos establecidos. Este compromiso será firmado en el PMP en la sección de Organización. [PP SP 3.3.5]

4.4.5.3 SCRUM DIARIO (DAILY SCRUM)

El *Scrum Diario* es una reunión con el equipo de desarrollo de 15 minutos en donde se sincronizan las actividades y se crea un plan para las siguientes 24 horas [PP SP 2.4.2]. Se toma de punto de partida los avances del *Scrum Diario* anterior y se pronostica el trabajo que será realizado antes del siguiente *Scrum Diario*. Para reducir complejidad lo conveniente es realizarlo siempre en el mismo lugar y a la misma hora, durante la junta los miembros del equipo de desarrollo deben explicar lo siguiente:

- ¿Qué se hizo el día anterior que ayudo a acercarse a la meta del *Sprint*?
- ¿Qué se hará el día de hoy para acercarse a la meta del *Sprint*?
- ¿Existe algún impedimento que me impida a mí o al equipo a alcanzar la meta del *Sprint*?

Si existe algún impedimento éste será recompilado y analizado para su solución. De ser posible las acciones a tomar serán determinadas al momento, en caso de que por su tamaño o complejidad no puedan ser resultas al momento serán registradas como nuevos elementos del *Sprint* o en el *backlog* del producto. [PMC SP 2.1.1, 2.1.2, 2.2.1, 2.2.2]

Diariamente el Equipo de Desarrollo entiendo cómo se trabaja como un equipo que se organiza a sí mismo para alcanzar las metas del *Sprint*. Es probable que individuos del Equipo de Desarrollo se reúnan justo después de la junta para discutir asuntos puntales de interés para exclusivo de los dos.

El Equipo de Desarrollo utiliza el *Scrum Diario* para inspeccionar el progreso, de esta forma se optimiza la probabilidad del Equipo de Desarrollo de alcanza los objetivos del *Sprint*. [PMC SP 1.2.1]

Los *Scrum Diarios* mejoran la comunicación, eliminan otras juntas, identifican los impedimentos de desarrollo a remover, enfatizan y promueven la toma de decisiones, y mejoran el nivel de conocimiento del equipo de desarrollo; es una junta clave de inspección y de adaptación. [PMC SP 1.6.1]

Durante estas reuniones los miembros del equipo son motivados a traer a la luz los riegos y problemas con los que se van topando. El *Scrum Master* analizará estos riesgos y tomará la decisión de llevarlos a un nivel más alto para trabajar en ellos. [PP SP 2.2.4, PMC SP 1.3.1, 1.3.2, 1.3.3]

Es responsabilidad del *Scrum Master* que el *Sprint Diario* se realice respetando las guías establecidas por *Scrum*, no obstante es posible que el entrenador de aseguramiento de calidad asista a este evento y apoye al *Scrum Master* haciéndole saber de algún elemento que pudiera estar omitiendo o llevando de forma incorrecta. Para realizar esto el entrenador se apoya en los listados de verificación y de mínimos creados en la fase inicial. El entrenador toma notas para documentar las "no conformidades" y sus resoluciones. [PPQA SP 1.1.3, 1.1.4, 1.1.5, 2.1.1, 2.2.1, 2.2.2]

4.4.5.4 Trabajo Diario

Además de la junta de *Scrum Diario*, diariamente el equipo de desarrollo realiza actividades enfocadas en crear los elementos de la aplicación que permitan lograr la meta del *Sprint*.

Cuando el elemento en el que estén trabajando esté completo y listo para ser integrado en la versión será ingresado al sistema de gestión de configuración para ser agregado en la integración continua y pruebas automatizadas. Además del código fuente, se deben añadir las anotaciones relevantes para facilitar la integración y posible solución de errores con el elemento. [CM SP 3.1.1]

Se actualiza la información en TFS relativa a los avances y clarificación de tareas, es decir: Se agregan tareas que van surgiendo necesarias para realizar una *Historia Scrum*, el número de horas reales requeridas para realizar un elemento de trabajo, los *bugs* que se vayan surgiendo, entre otras cosas. De esta forma se utilizara esta herramienta como principal fuente de medidas para su evaluación. Entre las medidas que son posibles obtener de TFS encontramos: Fecha inicial y final de las tareas, el esfuerzo real vs el estimado, coste real vs estimado, número de requerimientos, número de defectos (bugs), tamaño del producto, tiempo invertido resolviendo defectos. Es necesario también obtener y registrar cualquier otra medición que haya sido definida que no sea posible su obtención automática utilizando la herramienta de TFS. [MA SP 2.1.1, 2.1.2]

Un par de días antes de que la *Revisión del Sprint* tenga lugar, el entrenador de aseguramiento de calidad recopila una muestra de los Productos de Trabajo del Sprint y hace una evaluación de éstos. Los criterios para la creación de esta muestra se encuentran en el documento de "4.7.1 Plan de Gestión del Proyecto (PMP)". El entrenador utilizara las listas de verificación y mínimos para asegurarse que se cumplen los criterios mínimos de calidad en cada uno de los productos de trabajo analizados. Las "no conformidades" serán documentadas para ser compartidas con los miembros apropiados. [PPQA SP 1.2.3, 1.2.4, 1.2.5]

Como preparación para la revisión del *Sprint*, el *Scrum Master* realiza la revisión de integridad de las mediciones registradas basándose en exploraciones de datos faltantes, valores fuera de los rangos establecidos para la métrica. Patrones inusuales y correlación entre mediciones. Después de tener la integridad de las mediciones realizada, comienza a analizar los resultados, interpretarlos y sacar conclusiones, de ser necesario realiza o solicita mediciones adicionales para completar el análisis y preparar una presentación con los resultados. Estas evaluaciones y resultados, así como las métricas que no se registran de forma automática en TFS son almacenadas en un documento versionado que se mantiene en el repositorio de TFS. [MA SP 2.2.1, 2.2.2, 2.3.2]

También durante la preparación de la revisión, el Scrum Master revisa las actividades de Gestión de Datos que se están realizando y las compara contra lo definido en el PMP. De existir cuestiones significativas o desviaciones las documenta en el PMP y las expome en la Revisión de Sprint para buscar su solución. [PMC SP 1.4.1, 1.4.2, 1.4.3]

4.4.5.5 REVISIÓN DEL SPRINT

La revisión del *Sprint* se realiza al final de cada *Sprint* para inspeccionar el *Incremento* y adaptar el *backlog* del producto si fuera necesario. El equipo *Scrum* y las partes involucradas se reúnen para discutir, basados en los resultados el *Sprint*, sobre los posibles cambios que se pudieran realizar para optimizar el valor de los *Sprints*. Ésta es una junta informal y la presentación del *Incremento* es con la intención de obtener retroalimentación y fomentar colaboración [PP SP 2.4.2].

Cuando la revisión del *Sprint* coincide con un Hito en el calendario, la revisión del *Sprint* es a su vez la revisión del Hito. [PCM SP 1.7.1]

El *Scrum Master* se asegura que los eventos se sucedan y que los participantes entiendan las reglas y propósito de la reunión, además de mantener la reunión dentro de la duración pre-establecida. [PPQA SP 1.1.1]

Es posible que el entrenador de aseguramiento de calidad asista a este evento y apoye al *Scrum Master* haciéndole saber de algún elemento que pudiera estar omitiendo o llevando de forma incorrecta. Para realizar esto el entrenador se apoya en los listados de verificación y de mínimos creados en la fase inicial. El Entrenador toma notas para documentar las "no conformidades" y sus resoluciones. [PPQA SP 1.1.3, 1.1.4, 1.1.5, 2.1.1, 2.2.1, 2.2.2]

Los elementos incluidos en la revisión del *Sprint* son los siguientes:

- Los participantes: el equipo *Scrum* y partes involucradas invitadas por el propietario del producto (administradores, personal, clientes, usuarios finales, proveedores y otras partes involucradas; a criterio del propietario del producto y si es revisión de hito).
- El propietario del producto explica los elementos del *backlog* del producto que fueron establecidos en el *Sprint* y cuales no han sido realizados. [PMC SP 1.7.2]

Se analizan las causas de los elementos que no fueron realizados, se determinan las acciones a tomar para el siguiente *Sprint* relativas a estos elementos y serán documentados en el *backlog* para su seguimiento. [PMC SP 2.1.1, 2.1.2, 2.2.1]

- El equipo de desarrollo discute lo que estuvo bien, que problemas se presentaron y como fueron resueltos. Se deberá crear una bitácora con esta información para su consulta. [PMC SP 1.6.4]
- *Scrum Master* presenta la evaluación de las mediciones y resultados para hacerlos de conocimiento de las partes involucradas y usadas para toma de las decisiones de la revisión del *Sprint*. Cualquier duda de las personas involucradas será esclarecida por el *Scrum Master*. El *Scrum Master* pondrá a disposición en *Sharepoint* esta información para que pueda ser consultada por personal involucrado externo al equipo *Scrum*. [MA 2.2.3, 2.4.1, 2.4.2]
- El propietario del producto discute el estado actual del *backlog* del producto y realiza un pronóstico de las posibles fechas de terminación del proyecto basado en el progreso a la fecha. Incluyendo el estado de los riegos identificados con anterioridad o los nuevos riesgos encontrados. [PMC SP 1.1.1, 1.6.1, 1.6.3, 1.7.2]
- El equipo acuerda que será lo siguiente por hacer, lo cual servirá como una entrada valiosa a la siguiente planificación de *Sprint*.
- Se revisa como el mercado o el uso potencial del producto pudo haber cambiado, para entender que sería lo más valioso para hacer ahora.
- Se revisa la línea de tiempo, presupuesto, capacidad potencial y mercado para la siguiente versión del producto. [PMC SP 1.2.1]
- El propietario del producto, deberá de decidir qué elementos deberán serán aceptados y retenidos para ser parte de la nueva versión de la aplicación (*Build*) o *Incremento*. [CM SP 2.2.2]
- El entrenador de aseguramiento de calidad comparte las "no conformidades" encontradas en su evaluación de procesos y productos con los demás miembros del equipo *Scrum*, se analizan las acciones a tomar para solucionar estas "no conformidades", se analizan las tendencias en cuestiones de calidad y se identifican las lecciones aprendidas. [PPQA SP 1.2.6, 2.1.4, 2.1.5] Los resultados son almacenados en un repositorio versionado por TFS para su seguimiento. Se usa la misma estructura de *backlog* en un proyecto independiente de aseguramiento de calidad. [PPQA SP 1.2.7]
- Las cuestiones de Calidad que no puedan ser resultas por el equipo *Scrum* son escaladas por el Entrenador del Aseguramiento de Calidad, así como las "no conformidades" que aún no han sido resultas por el equipo *Scrum* después de ser informado de éstas. [PPQA SP 2.1.3, 2.1.6]
- Si existen productos de trabajo que están siendo trabajados por un proveedor, un embajador del proveedor se agregara a estas juntas como un desarrollado al

estilo de los *Scrum of Scrums*. Este embajador realiza las mismas funciones que los desarrolladores del *Scrum* en esta junta. [SAM 2.1.4, 2.1.5, 2.1.6]

El resultado de la revisión del *Sprint* es un *backlog* del producto revisado, el cual define los elementos probables del *backlog* del producto que se realizar en el siguiente *Sprint*. De esta forma se da un seguimiento a los cambios y problemas que aún queda sin resolver en el *Sprint* actual y serán resueltos con mucha probabilidad en el siguiente *Sprint*, en todo caso en algún *Sprint* posterior. [PMC SP 1.6.6, 1.7.4, 1.7.5; CC SP 2.1.5]

El *backlog* del producto puede ser ajustado para cumplir las nuevas oportunidades y reflejar los cambios negociados en los compromisos externos e internos. [PMC SP 2.2.3]

Se deberá de actualizar las gráficas *burndown* para medir el esfuerzo faltante para terminar la aplicación. [PMC SP 1.6.4]

En este punto del *Sprint*, el *Scrum Master* debe de asegurarse que los planes de trabajo y el calendario del proyecto se encuentren actualizados, de no ser así, actualizarlos para reflejar los avances alcanzados en el Sprint. En estos planes se debe de reflejar los costes, esfuerzo utilizado y el esfuerzo requerido, medido en forma de Puntos de Historia; así como los recursos utilizados vs los disponibles. [PMC SP 1.1.2, 1.1.4, 1.1.5, 1.2.3, 1.6.3].

Las desviaciones significativas serán documentadas para ser analizadas posteriormente en juntas con las principales partes involucradas. [PMC SP 1.1.7, 1.2.2, 1.7.3]

Las personas involucradas de primer nivel son requeridas para esta junta, por lo cual su grado de involucración en el proyecto se revalida; por otro lado el propietario del producto será el encargado hacer llegar las resoluciones y cuestiones obtenidas en esta revisión a las demás partes involucradas de las cuales es el representante. [PMC SP 1.4.1, 1.4.2]

Si el *Scrum Master* detecta un riesgo nuevo con el trabajo realizado por el proveedor será responsable de dirigir esto a la persona involucrada conveniente y al gestor de proveedores. [SAM SP 2.1.8]

4.4.5.6 RETROSPECTIVA SPRINT

La retrospectiva permite al equipo *Scrum* inspeccionarse a sí mismo y crear un plan con mejoras para implementar durante el siguiente *Sprint*. Este evento se sucede después de la revisión del *Sprint* y antes de la siguiente planificación del *Sprint*, tiene una duración aproximada de 3 horas para *Sprints* de 1 mes.

El *Scrum Master* se asegura de que los eventos tomen lugar y los participantes entiendan el propósito de la junta, además de guiar al equipo para mantener la junta dentro de la duración preestablecida. En esta junta en particular, el *Scrum Master* participa como un miembro más del equipo como responsable del proceso *Scrum*.

El entrenador de aseguramiento de calidad puede asistir a este evento y apoyar al *Scrum Master* haciéndole saber de algún elemento que pudiera estar omitiendo o llevando de forma incorrecta. Para realizar esto el entrenador se apoya en los listados de "4.7.8 Lista de verificación y mínimos" creados en la fase inicial. El entrenador toma notas para documentar las "no conformidades" y sus resoluciones. [PPQA SP 1.1.3, 1.1.4, 1.1.5, 2.1.1, 2.2.1, 2.2.2]

El propósito de la retrospectiva *Sprint* es: inspeccionar como el último *Sprint* se sucedió conforme a gente, relaciones, procesos y herramientas; identificar y ordenar los elementos principales que se sucedieron bien y sus mejoras potenciales; crear un plan de mejoras a implementar en la forma en que el equipo *Scrum* hace su trabajo. [PMC SP 1.6.2]

Durante la retrospectiva del *Sprint* el equipo *Scrum* planea formas de incrementar la calidad afinando la definición de lo que debería ser un elemento terminado o "Hecho". Al finalizar la retrospección el equipo tendrá identificados las mejoras que se pueden aplicar en el siguiente *Sprint*. Aunque las mejoras pueden ser implementadas en cualquier momento, la retrospección presenta una oportunidad formal para enfocarse en la inspección y adaptación del proceso.

Si existen productos de proveedor que se integran en este *Sprint* un embajador del proveedor hará las veces de desarrollador *Scrum* y tendrá el mismo rol que ellos, llevando las mejoras que pueden ser implementadas en el proceso del proveedor. [SAM SP 2.1.7]

Entre estas mejoras a integrar se incluyen las relacionadas con el conocimiento y habilidades del personal. [PMP SP 1.1.6]

Las acciones correctivas realizadas a los elementos del *backlog* del producto que provienen de una cuestión o problema que debía ser resulto, son evaluadas en términos de eficacia y son usadas para tomar nuevas decisiones a problemas similares basados en estas lecciones aprendidas. Estos documentos serán almacenados en el repositorio de lecciones aprendidas. [PMC SP 2.3.2, 2.3.3]

La estructura de gestión de configuración será revisada y modificada de ser necesario, para responder a la nueva información adquirida. [CM SP 1.2.8]

Utilizando la información registrada en TFS sobre los requerimientos y los trabajos realizados se puede crear una Matriz de Rastreo exportando estos datos a Excel. De esta forma se tendrá una idea clara de los elementos que forman parte de uno o varios requerimientos. Con esta información se deberá validar la consistencia de las tareas realizadas, productos de trabajo y planes con los requerimientos. [REQM SP 1.4.3, 1.5.1]

En caso de existir inconsistencia deberán de ser identificado el elementos o elementos que causan la inconsistencia y crear los elementos necesarios en el *backlog* del producto con la prioridad adecuada para ser resueltos, de ser posible, en el siguiente *Sprint*. [REQM SP 1.5.2, 1.5.4]

De ser necesario, después de haber utilizado las medidas obtenidas de las actividades realizadas para evaluar el *Sprint*, se realiza una retroalimentación para clarificar los requerimientos de

información, se identifican si es necesario revisar y/o actualizar las mediciones, se actualizan las medidas y objetivos de medición y se actualiza el formato y el contenido propuesto de la toma de mediciones. Además de refinar los criterios para análisis futuros. [MA SP 1.1.4, 1.2.5, 1.3.6, 1.3.7, 1.4.4, 1.4.5, 2.2.4, 2.3.1]

4.4.6 Incremento

La suma de todos los elementos del *backlog* del producto completados durante el *Sprint* y el valor de los incrementos de todos los *Sprints* anteriores tiene el nombre de *Incremento*. Al final del *Sprint* el nuevo *Incremento* debe de estar en condición de ser usado y cumplir la definición de "Hecho" creada por el equipo *Scrum*; debe de estar en condiciones de ser usado sin importar si al final es liberado o no.

Si las partes interesadas toman la decisión de liberar la versión, entonces se pasa a la fase de transición/implantación.

4.4.7 Herramientas

- **Source control**: Para mantener el código y los documentos versionados y almacenados en sus repositorios correspondientes.
- **Build Server**: Necesario para mantener la política de integración continua, de tal forma que inmediatamente después de que un elemento del *backlog* de producto es terminado por un desarrollador, éste pasa a la cola de integración; de la cual se tomaran los fuentes para crear una nueva versión de pruebas. Estas versiones de prueba son generadas con un lapso de tiempo no mayor a un día.
- **Visual Studio Unit Tests**: Las unidades de pruebas que pueden ser creadas en Visual Studio son utilizadas para la automatización de pruebas y para mantener un desarrollo guiado por pruebas, como se define a continuación en las recomendaciones para equipos distribuidos.

4.4.8 Recomendaciones para Equipos Distribuidos

Las siguientes 3 prácticas propuestas por Woodward, Surdek, & Ganis (2010), son en la medida de lo posible utilizadas para proyectos con equipos distribuidos. Estas buenas prácticas aportan beneficios a cualquier tipo de equipo *Scrum*, pero son excelentes para equipos distribuidos.

4.4.8.1 Desarrollo guiado por Pruebas

Usando el desarrollo guiado por pruebas, los desarrolladores circulan entre la escritura de las pruebas, la escritura del código para pasas esas pruebas y el reacondicionamiento de su código. Esto ayuda al equipo de desarrollo a trabajar rápido y efectivamente porque el desarrollador entrega código de alta calidad probado. El resultado es que el trabajo avanza con menos defectos, lo que significa menos tiempo depurando y recreando los problemas. [CM SP 2.2.4]

El desarrollo guiado por pruebas es un método de diseño de bajo nivel que permite a los equipos evolucionar el diseño con el tiempo. Encontrar defectos en una etapa temprana del desarrollo es menos costoso y consume menos tiempo que encontrarlos después.

Dada la disminución en la calidad de la comunicación de los equipos distribuidos, las buenas prácticas que reducen los problemas más adentrados al *Sprint* son aún más importantes.

4.4.8.2 Integración Continua

La integración continua es una práctica que permite que los miembros del equipo integren su trabajo de forma frecuente. Se utilizan pruebas automatizadas para detectar los problemas de integración de forma rápida y así resolver los problemas que pudieran aparecer en un tiempo cercano a su agregado a la cola de compilación de la versión. [CM SP 2.2.4]

Se puede encontrar más información de Integración continua en (Duvall, Paul M, Matyas, Steve, & Glover, Andrew, 2007)

4.4.8.3 Automatización de pruebas

Una práctica muy importante para equipos distribuidos, las pruebas automatizadas ayudan al equipo a realizar las pruebas de código de forma más eficiente; además de una implementación y ejecución de pruebas del nuevo código. Para la automatización de las pruebas se puede utilizar las unidades de prueba –*unit tests*– de Visual Studio.

4.5 TRANSICIÓN/IMPLANTACIÓN PROYECTO

Esta fase está basado en la fase de "Transición" de *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise* (Ambler & Lines, 2012)

4.5.1 Objetivo

Esta fase de proyecto es un tipo especial de iteración dedicada a implementar la solución al cliente y dejarla lista para su uso, si ya no existen elementos en el *backlog* del producto entonces sería la finalización del proyecto.

Los objetivos de esta fase son: asegurarse que la solución está lista para producción, asegurarse que las partes involucradas están preparadas para recibir la solución e implementar la solución en producción.

4.5.2 Planificación

Al acercarse la fase de Transición es necesario que el propietario del producto empiece a agregar al *backlog* del producto las tareas relacionadas con poner el producto en manos de los usuarios y que el producto pueda ser usado adecuadamente. El *Scrum Master* ayuda al propietario del producto a colaborar con las partes involucradas externas para asegurarse que el trabajo requerido en la implantación está identificado.

Entre las actividades que deben de ser planificadas se encuentran: planificación de la implementación, instalación de los entregables, conversión de datos, comunicación con las partes involucradas, actividades de marketing, formación y finalización de la documentación.

Al igual que en la fase de Construcción esta iteración requiere la planificación de actividades, con la diferencia de que en la construcción los elementos del *backlog* del producto eran

desarrollados al cien por ciento por el equipo de desarrollo y aquí la colaboración con personal externo del equipo es inevitable; entre ellos: especialistas de seguridad, gente de implementación y técnicos de servidores. Es importante recalcar que al depender de personal externo que dan servicio a varios equipos, puede ocurrir un conflicto de prioridades. De esto que sea tan importante planear con antelación y tener informadas a las partes involucradas en la instalación, ya que el no tenerlos disponibles en las fechas esperadas traerían un retraso en el proyecto.

4.5.3 Preparar la Transición

Hay una serie de elementos que son necesarios preparar antes de efectuar la implementación del producto, que de ser tomados en serio reducen en gran medida el riesgo relacionado con una implementación fallida o retrasada en el tiempo. Cuando se trata de un sistema crítico cada minuto de retraso puede ser extremadamente costoso. Entre las buenas prácticas a seguir en la transición encontramos las siguientes:

Planear la transición: En ambientes de producción de misión crítica es necesario preparar un plan detallado de los requerimientos necesarios para efectuar la implementación. Una buena práctica es realizar esta implementación fuera del horario de uso de la aplicación, comúnmente durante la noche y entre semana; de esta forma si ocurre algún imprevisto el personal de Tecnologías de Información se encontrara disponibles al día siguiente para resolverlo.

Es conveniente, sobre todo para los sistemas que deben de ser implementados en tiempo crítico tener los pasos de la instalación detallada, incluyendo: tiempo esperado de inicio y fin, dependencias y responsabilidades. También las instrucciones para hacer un *rollback* –deshacer la instalación y volver la aplicación a su estado anterior– deberán de ser documentados para un caso extremo en donde sea imposible avanzar en la implementación y se tenga que posponer. Si además de la implementación es necesario una conversión de datos o migrar de plataforma tecnológica, los planes deben de ser más detallados y precisos.

Arreglos y pruebas de fin de ciclo. Los últimos días antes de la transición el Equipo de Desarrollo no deberá de intentar agregar nuevas funcionalidad o características y enfocarse en corregir los defectos que pudieran tener las funciones que ya están codificadas. En otras palabras, se busca solidificar y volver más robusta la solución enfocándose en arreglar defectos en lugar de nuevas funcionalidades.

Una métrica importante para evaluar la efectividad de las prácticas de calidad es el monitorizar el número de defectos encontrados en la fase de Transición ordenados por severidad, también los defectos encontrados una vez que la solución está siendo usada por los clientes en el ambiente de producción.

Simulacro de Implementación: Una técnica que reduce el riesgo y desconocimiento de lo que pudiera pasar en la implementación es realizar un simulacro de implementación en un ambiente

lo más parecido posible a producción. De esta forma el riesgo y la incertidumbre de lo inesperado se reducen en gran medida.

Preparar la migración de Datos: La preparación de la migración involucra la creación de programas de extracción de datos de los viejos sistemas, transformación de ellos para que puedan ser almacenadas en la nueva plataforma, y la carga de ellos en el sistema nuevo. La creación y prueba de estos programas son una parte importante de la transición, afortunadamente estos programas probablemente ya fueron utilizados y probados en la fase de Desarrollo/Construcción ya que es necesario para las pruebas unitarias de los elementos desarrollados en la fase Desarrollo/Construcción. Importante prestar atención especial a la migración del modelo de seguridad, incluyendo los usuarios y sus credenciales.

Uso de Piloto o versión Beta de la solución: Las versiones Beta o pruebas Piloto son comúnmente implementadas a través de la cual se da un subconjunto de los usuarios base para obtener retroalimentación basada en el uso real en producción. Esta estrategia mitiga muchos riesgos relacionados con los siguientes puntos:

- Saca a la luz nuevos defectos en el ambiente de producción en situaciones que no puede ser probadas adecuadamente en ambientes de no-producción.
- Se visualizan los requerimientos y procesos de soporte al cliente, reduciendo el riesgo al tener solo un pequeño grupo de usuarios requiriendo soporte.
- Afinar aspectos del ambiente de producción como servidores, bases de datos, redes y seguridad sin el riesgo de afectar a toda la base de usuarios a la vez.
- Si existe un problema grave en producción es más sencillo de revertirlo haciendo un *rollback*.

Finalizar Documentación: La documentación de soporte pudiera necesitar ser terminados, en caso de que no se haya hecho anteriormente. El material a terminar incluye manuales de usuario, manuales de operación, guías de instalación, documentación de recuperación de desastres, documentos de vista general de los sistemas, entre otros.

Asegurarse que el hardware a usar será el adecuado: Los cambios realizados en el incremento pudiera requerir algún cambio en la estructura de hardware instalado para el proyecto. De ser así es necesario adecuar el hardware para prepararlo a recibir el nuevo incremento. [SAM SP 2.3.1]

*Como se lee en las instrucciones de uso de la guía, estos párrafos hacen referencia a las prácticas específicas SAM SP 2.3.1. Como ejemplo se muestra a continuación un extracto del apartado 4.1.3 mostrando la práctica específica. En el resto de las anotaciones de este inciso se omitirán las referencias de ejemplo, las cuales deberán de ser consultadas en el apartado 4.1.3.

SAM –Gestión de Acuerdos con Proveedores	**Sección***
SG 2: Satisfacer los acuerdos con el Proveedor	

SP 2.3 Asegurar la transición de los productos.	
2.3.1 Asegurar que existan las instalaciones para recibir, almacenar, integrar y mantener los productos adquiridos de forma apropiada.	4.5.3

4.5.4 Preparar a las Partes Involucradas.

No es aconsejable implantar la solución si los usuarios no están listos para ello. La comunicación debe de iniciarse mucho antes de llegar al punto de liberar la versión de la solución, de hecho la comunicación debería mantenerse desde el inicio del proyecto. Acercándose el final del proyecto la comunicación debe de incrementarse. Los componentes de la comunicación y formación a considerar para preparar a las partes involucradas hacia la transición son las siguientes: [PP SP 2.4.2]

Comunicar la implementación a las partes involucradas. Las partes involucradas deben de ser informadas antes de liberar la solución, algunas soluciones requieren que se formaliza la comunicación tanto a usuarios internos como clientes externos. Para a cada uno de los tipos de usuario se les debe entregar comunicaciones específicas, enumeradas a continuación:

- **Operaciones**: Ellos necesitan soportar la aplicación manteniéndola funcional, corriendo trabajos de mantenimiento, realizando respaldos y recuperaciones, y manteniendo el hardware funcionando. Es necesario producir documentación describiendo las actividades para mover la solución a producción, además de información necesaria para que se puedan efectuar los respaldos y mantenimiento técnico a la solución. También es necesario que operaciones estén listos para gestionar el licenciamiento de los productos adquiridos y asegurarse que no se incumplen los compromisos de licencias adquiridas. [SAM SP 2.3.3]
- **Marketing**: Si la solución es un producto, se debe de informar a los encargados de distribuir la información correspondiente a los potenciales y existentes usuarios con antelación.
- **Otros empleados internos**: En algunos casos la solución puede cambiar el proceso de negocio o la descripción de los trabajos a realizar, si esto sucede los empleados necesitan estar preparados para estos cambios.
- **Organismos reguladores**: En algunos casos es necesario comunicar a los organismos reguladores de la implementación antes de que ésta ocurra.
- **Patrocinadores del proyecto**: Los patrocinadores necesitan estar informados de los planes de implementación de la solución.
- **Organismos de gestión de cambio**: En empresas grandes las implementaciones sin imposibles sin la coordinación con otros proyectos con los que compartirán

datos y servicios. Debieran existir comités que se reúnen regularmente para revisar los cambios nuevos que serán implementados en producción para medir el impacto que pudiera afectar a otros sistemas en producción. En esta etapa de Transición un responsable del proyecto debería de estar asistiendo a estas juntas para validar los impactos que pudiera tener la nueva solución con el resto de soluciones en producción.

- **Oficina de Gestión de Proyectos (PMO)**: Al llegar a esta fase la oficina de Gestión de Proyectos ya debería estar al tanto del itinerario de implementación como parte del estar continuamente involucrado en lo que respecta al proceso durante todo el ciclo de vida. Ellos estarán involucrados con la decisión de proceder con la fase de Transición como parte de los hito de revisión de que la funcionalidad alcanzada en la Construcción sea suficiente para continuar con la Transición.

Formar a las partes involucradas: Formar y educar a los usuarios finales, personal de operación y personal de soporte es parte de la preparación de las partes involucradas para aceptar la solución. Esta formación puede darse en cualquier momento, pero es mejor hacerlo durante la Transición para que lo aprendido esté fresco. Es conveniente solicitar retroalimentación sobre la utilidad de la solución durante la formación del equipo piloto. [SAM SP 2.3.2]

Preparar el ambiente de soporte para las partes involucradas: Es necesario tener preparado el equipo que va a encargarse de dar soporte a los usuarios, comúnmente habrá personal dedicado a dar este tipo de soporte a todas las soluciones de la compañía. Habrá también un sistema de soporte como un *help desk* para apoyar a los clientes. Para asegurar un buen servicio es necesario preparar a las siguientes partes involucradas para apoyar el sistema:

- *Help Desk*: La primera línea de soporte para los clientes, su función es capturar estos incidentes y dirigirlos al personal calificado para su resolución. Las hojas de tiempos de personal disponible para resolver los problemas de la solución deben de generarse.
- Recuperación de Desastres: Se crean procedimientos de recuperación contra repentinas caídas del sistema. Los procedimientos serán probados periódicamente para verificar que el sistema puede ser recobrado y puesto nuevamente en marcha en un tiempo aceptable. La documentación para realizar una instalación del sistema desde cero debe de estar actualizada y probada, para asegurarse que ésta puede ser usada por el personal de operaciones aunque no estén familiarizados con la los detalles de la arquitectura de la solución.

Se tiene un hito en esta fase en donde se decidirá si finalmente se hace la implementación o no. Se tiene un documento con las condiciones necesarias para que se acepte la solución, de esta

forma la decisión de aceptar la implementación o no, se pueda tomar de forma objetiva. La lista de aceptación puede incluir el número máximo de defectos aceptables y su grado de severidad, los resultados esperados de pruebas de desempeño, la documentación mínima requerida, si los elementos para la implementación están listos y si se tiene la aprobación del patrocinador de la solución.

Si un elemento rechazado pertenece a un producto comprado a un proveedor, será necesario iniciar las acciones correctivas en conjunto con el encargado de gestión con proveedores para su solución. [SAM SP 2.2.3, 2.2.4, 2.2.5, 2.2.6]

4.5.5 Implementación

En algún momento de la fase de Transición se elegirá la implementación de la solución por las partes interesadas. Pudiera darse el caso que más de una implementación sea necesaria, como en el caso de aplicaciones de escritorio con múltiples ubicaciones geográficas. Es típico tener personal independiente dedicado a realizar las implementaciones. Si esto sucede la importancia de la planificación y la creación de documentos con los pasos de la implementación y las instrucciones para regresar a la versión anterior en caso de fallo, son indispensables. Para implementación con tiempos críticos, el plan debe contener los pasos uno por uno, los tiempos, duraciones, dependencias y responsabilidades. A groso modo las actividades a realizar son:

Correr programas por lotes sobre el sistema existente: Es necesario correr procesos para cerrar las transacciones de los programas anteriores antes de migrar los datos al nuevo sistema.

Respaldar el sistema existente: Si se diera el caso de una implantación infructuosa es necesario contar con los respaldos de la aplicación anterior.

Migrar los datos fuente a los nuevos formatos o tecnologías. Los posibles problemas con la migración debieron presentarse en la etapa de Desarrollo/Construcción, pero de igual forma es importante tener en cuenta la planificación de estos procesos de migración; así como los horarios a los que será necesario hacer la migración. Es importante dejar abierta la posibilidad de deshacer los cambios a los datos en caso de una implementación fallida.

Implementar los artefactos de la solución al ambiente destino: Los artefactos para correr la nueva solución debieron haber sido construidos con anterioridad a la decisión de poner la solución en vivo. Ahora necesitan ser implementados en la plataforma destino y hacer las configuraciones necesarias.

Realizar prueba de la implementación. Es necesario hacer pruebas con la solución en el ambiente de producción para asegurarse que esté funcionando como se espera. En esta prueba se realizan transacciones reales en el ambiente de producción y después se deben de deshacer los movimientos que se realizaron. Esta prueba es llamada también “Prueba de Humo”. Aunque pueda haber resistencia en realizar pruebas con datos reales y que posteriormente se requiera deshacer las transacción, resulta una mejor opción o pasar por el problema que un usuario encuentre el error y sea necesario regresar al sistema anterior, con todo el tiempo

perdido que esto implica. Estas pruebas deben de ser identificadas por el personal de operación y de soporte, y pueden ser consideradas como una estrategia de desarrollo y operaciones.

Respaldar la nueva solución. La nueva solución debe de ser respaldada y de ser posible almacenar el respaldo un una ubicación física diferente a la solución, para prevenir catástrofes como por ejemplo inundaciones e incendios y que puedan afectar los equipos de una ubicación.

Poner la nueva solución a disposición de los clientes. Puede ser algo tan sencillo como cambiar la dirección del URL, para el caso de aplicaciones web, para que la solución sea liberada a los clientes.

Activar el equipo de soporte al sistema. Esto incluye el *help desk*, que desde ahora estará listo para soportar al sistema según lo requieran las partes involucradas.

Comunicar que la implantación fue exitosa. Las comunicaciones planeadas internas y externas serán enviadas al llegar este momento. También es una buena práctica tener una comunicación de que a la implementación no se logró realizar, en caso de necesitarlo.

4.5.6 Aceptación

Contrario a lo que algunos equipos de desarrollo pudieran pensar que el proyecto está terminado en el momento en que se libera a los clientes, esto no es así. Algunas ocasiones es necesario realizar arreglos rápidos –*hot fixes*– para arreglar defectos encontrados por los clientes. Es un error deshacer el equipo de desarrollo el día después de la implementación ya que posibles arreglos pueden ser requeridos. Existen actividades que necesitan completarse antes de que el equipo pueda dedicarse a otros proyectos o empezar la siguiente iteración.

A la comunidad de desarrolladores les gusta decir que la meta es "deleitar" a las partes involucradas, ya que no es suficiente que están satisfechos. El hito llamdo "Deleitar a las partes involucradas" se debe lograr:

Aceptación de las partes involucradas. Las partes involucradas del negocio deben de estar deleitadas con el sistema y aceptarlo.

Aceptación de Operaciones. El personal responsable de operar el sistema una vez que está en producción está deleitado con los procedimientos relevantes, la documentación de soporte y el funcionamiento del sistema mismo.

Aceptación de Soporte. El personal responsable de soportar el sistema una vez que está en producción está deleitado con la solución, particularmente la documentación de soporte y los procedimientos de soporte relevantes.

Aceptación de coste y estimados. El gasto actual generado es aceptado, y un estimado razonable ha sido hecho para costes futuros.

Con los productos adquiridos depende del método de pago definido en el contrato con el proveedor, si es un tipo de contrato con precios fijos la aceptación se realizara contra los

criterios de aceptación estipulados en el contrato. Si por otro lado es de tipo Tiempos y Materiales (T&M) la aceptación se realiza con el mismo tipo de criterio que los productos internos, en otra palabras obtener el "deleite" de las partes involucradas. [SAM SP 2.2.3, 2.2.4]

Las incidencias serán registradas y seguidas utilizando la herramienta de ALM que forma parta de Team Foundation Server. [SAM SP 2.2.5, 2.2.6, 2.2.7]

4.5.7 Herramientas

4.5.7.1 Application Lifecycle Management (ALM)

Los incidencias encontradas en la aceptación de los productos serán dados de alta como nuevos elementos del *backlog* o como *bug* dependiendo del tipo de incidente para ser resueltos en el siguiente *Sprint*. De ser necesario, ya sea por el tamaño o por que impiden el cumplimiento de objetivos de tiempo, se crea un *Sprint* de emergencia para solucionar esta incidencia. Esto aplica de igual forma para los productos internos como para los productos adquiridos a proveedores. [SAM SP 2.2.5, 2.2.6]

El tener las incidencias registrados en ALM se asegura su seguimiento hasta su cierre. [SAM SP 2.2.7]

4.6 PROCESOS PARALELOS AL DESARROLLO

4.6.1 Auditorias

4.6.1.1 Auditoria de Configuración

Las auditorias de configuración se realizan para comprobar que las líneas bases y la documentación siguen el estándar o el requerimiento definido.

Para asegurar la eficacia de estas auditorías, se realizan por personal externo al equipo de desarrollo y de forma aleatoria, como mínimo se realizara una vez para los proyectos pequeños. El número máximo de auditorías a realizar será a criterio de los auditores, prestando atención a los resultados de las autorías anteriores para establecer el número de auditorias subsecuentes.

Los tipos de auditorías a realizar serán:

- **Auditorias Funcionales**: Se realizan para verificar que el desarrollo de la funcionalidad de elementos de configuración ha sido completado de forma satisfactoria y que sus atributos funcionales tienen la calidad especificada en la línea base, además de validar la existencia de documentación de soporte está completa. [CM SP 3.2.4]
- **Auditorias de configuración física**: Se asegura que el elemento de configuración exista y haya sido construido conforme su definición en la documentación técnica. (Esta documentación se encuentra en los elementos del *backlog*). [CM SP 3.2.2]

- **Auditorias de Gestión de Configuración**: sirven para confirmar que los registros de gestión de configuración y los elementos de configuración son completos, consistentes y precisos. [CM SP 3.2.1, 3.2.3, 3.2.5]

Se llevara un sistema de seguimiento, para asegurarse que las acciones pertinentes a realizar por el incumplimiento o discrepancia en la gestión de configuración sean corregidas de forma aceptable. [CM SP 3.2.6]

4.6.2 Gestión de Proveedores

4.6.2.1 Propósito

El propósito de la Gestión de Proveedores es el gestionar la adquisición de productos y servicios de los proveedores.

Sus principales objetivos son:

- Asegurarse que los contratos y acuerdos están alineados con las necesidades del negocio.
- Obtener valor adecuado por el dinero invertido en proveedores y contratos.
- Gestionar las relaciones con los proveedores.
- Gestionar el desempeño de los proveedores.

4.6.2.2 Seleccionar Proveedores

En empresas pequeñas el encargado de realizar el rol de adquirir los productos y servicios de los proveedores lo podrá realizar un miembro del equipo *Scrum*, pero es recomendado que exista una persona encargada exclusivamente de este rol si el tamaño de la empresa lo permite.

Al recibir una petición de adquisición lo primero que se realiza es la plantilla de "4.7.10 Requisición de Productos y Servicios", la cual contiene los requisitos mínimos generales que forman parte de las políticas de la empresa. El rellenado de la plantilla se hará en conjunto con el solicitante del producto o servicio para aclararle las dudas y que los criterios de selección requeridos por el proyecto sean claros. [SAM 1.2.1]

Teniendo perfectamente claros y documentados los criterios de selección de los proveedores, el encargado de su gestión se pondrá en contacto con los proveedores para solicitar los productos y servicios proporcionándoles la lista de requerimientos y materiales. [SAM 1.2.2] Al recibir respuesta de los proveedores se procederá a evaluarlos de acuerdo a los criterios establecidos, por parte del Gestor de Proveedores para asegurarse que cumplen las normativas mínimas exigidas por la empresa para con sus proveedores, y por el solicitante del producto o servicio con respecto a si se solventan de forma satisfactoria los requerimientos del producto dentro del proyecto. [SAM 1.2.3]

El gestor evalúa los riesgos relativos al proveedor, por ejemplo: Si la empresa tiene poco reconocimiento, si no cuenta con oficinas físicas en el país, si es proveedor nuevo o muy pequeño lo que pudiera implicar que desapareciera en el futuro y se perdiera el soporte, el proveedor no dominan el idioma del cliente, etc.

Por último y para pasar a tomar la decisión, se evalúan las habilidades del proveedor para desarrollar el trabajo, incluyendo:

- Si es un proveedor preferido o no, es decir si es un proveedor con el que ya se ha trabajado anteriormente y se tiene una experiencia positiva.
- Si ya ha realizado trabajos similares al solicitado.
- Si cuenta con certificados de gestión y desarrollo.
- Personal capacitado y certificado.
- Evaluación de sus recursos humanos y oficinas.

Éstos son solo algunos de los factores que son usados para tomar la decisión. [SAM SP 1.2.4]

Después de esta evaluación el encargado del proyecto y el gestor de proveedores seleccionarán al proveedor. [SAM SP 1.2.5]

4.6.2.3 Establecer los acuerdos

Sin ir en contra de la idea que la relación entre el proveedor y el cliente debe de ser basado en confianza, es de suma importancia crear contratos claros y legalmente válidos. Por lo anterior es indispensable contar con la asesoría de un abogado de contratos, para evitar caer en la creación de acuerdos sin validez legal.

Antes de crear el contrato es necesario revisar y aclarar tanto los requerimientos del proyecto como los requerimientos que tendrá el proveedor del proyecto. Los primeros incluyen: los requerimientos del producto, el nivel de servicio requerido, etc. Mientras los segundos incluyen: la documentación que será entregado al proveedor, el lugar físico y mobiliario que deberá ser proporcionado por el proyecto y los servicios. [SAM SP 1.3.1, 1.3.2]

Los acuerdos serán documentados en forma de contrato. El tipo de contrato a utilizar será contrato ágil, la excepción será con los productos completos que no requieren adecuaciones o que su desarrollo es tan mínimo que estarían terminados en el lapso de un *Sprint*.

Los contratos ágiles siguen la misma filosofía que los proyectos ágiles: poseen ciclos de vida cortos (*Sprint),* el alcance del proyecto no está definido exactamente y puede variar, existe manejo del cambio además de tener los siguientes elementos clave:

- Los entregables son identificados como objetivos de negocios que son medibles.
- El proceso es detallado, pero no la solución.
- Los costes son estructurados por iteración.
- La flexibilidad es primordial, debe contener la flexibilidad suficiente para poder variar en: entregables, costes, fechas de entrega y mecanismos de informar estatus.

El precio de los contratos se adapta al concepto de iteraciones, teniendo pagos al final de cada iteración. La recomendación seria: si es un proveedor preferido del cual ya se tiene constancia de eficiencia y responsabilidad se puede optar por un variaciones de (T&M) Tiempo y material, el cual se paga por hora trabajada. Si es un proveedor nuevo antes de pasar a un estilo de precios T&M, se comenzaría con uno de costes fijos para el primer *Sprint* y después de evaluar el resultado del *Sprint* se podría pasar a uno de Tiempos y Materiales. Los contratos basados en T&M pueden ser de los siguientes tipos:

- T&M limitado por iteración
- T&M limitado por proyecto o versión
- T&M limitado por iteración con ajuste. Si se logran todos los objetivos de la iteración en menos coste de T&M se le da una bonificación al proveedor (por ejemplo la mitad del ahorro en coste de T&M).

Ya sean contratos tradicionales o ágiles éstos deben de incluir claramente lo siguiente: [SAM SP 1.3.4]

- Establecer claramente el trabajo a realizar, especificaciones, términos y condiciones, lista de entregables, calendario, presupuesto y proceso de aceptación. Si es contrato ágil, se definirá en el contrato en donde se podrán encontrar esta información relativa a cada una de las iteraciones, que es como se llevara el proceso.
- Criterios de evaluación, forma de monitorizar al proveedor y como serán evaluados los productos de trabajo.
- Identificar quien será el responsable en el proyecto de realizar cambios al contrato de ser requerido.
- Definir los estándares y procedimientos que se deben seguir.
- Identificar las dependencias entre el proyecto y el proveedor.
- Las responsabilidades de soporte del proveedor.
- La garantía que se ofrece, quien es el dueño del producto y los derechos de uso de los productos adquiridos.
- Definir los criterios de aceptación.

Para más detalle en la creación de contratos ágiles consultar el capítulo 16 de *Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum* (Larman & Vodde, 2010).

Después de la creación del contrato deberá de ser firmado por los responsables del proyecto y el representante del proveedor como símbolo de aceptación de todo el contenido del contrato. [SAM 1.3.5]

4.6.2.4 Gestionar que los acuerdos se cumplan

A los proyectos o servicios que son proporcionados por los proveedores se les dará seguimiento con la misma herramienta ALM de TFS que se usa para llevar los proyectos *Scrum*, pero en este caso el detalle será mucho menor ya que solo se agregaran las historias y no el desglose de tareas requeridos para ser realizada. Esto permitirá el dar el seguimiento y monitorizar al trabajo realizado por el proveedor y su comparación con el contrato de acuerdos. [SAM SP 2.1.1]

Los productos generados por los proveedores pasar a formar parte del *Scrum* del proyecto, si el producto se puede manejar como una unidad como en el caso de los productos comerciales empaquetados (COTS), se agregara un solo elemento al Sprint para su monitorización desde la planificación hasta su aceptación y transición. Si el producto consta de múltiples elementos y su tamaño lo amerita se podrá crear un *Scrum* individual y un *Scrum de Scrums* para la integración de éste con el proceso de desarrollo del proyecto maestro. Ya sea un caso o el otro el proceso de desarrollo de los productos de proveedores seguirá el mismo proceso que se utiliza para la gestión de proyectos internos. Más información en la sección “4.4 Desarrollo/Construcción” y “4.5 Transición/Implantación proyecto”. [SAM SP 2.2.1]

Los procedimientos de aceptación son los mismos que los definidos en el *Scrum* mantenido por la empresa, por lo que la aceptación por las partes involucradas se entiende por asentada. [SAM SP 2.2.2]

El rol de entrenador de aseguramiento de calidad se encargara de revisar los procesos y productos sean desarrollados conforme a lo definido en el contrato de acuerdos del proveedor, que deberá de cumplir cuando menos las mismas normativas que se exigen en los desarrollos internos de la compañía. [SAM SP 2.1.2, 2.1.3]

Si al revisar que se están cumpliendo los procesos se encuentra que no es así, pero que el cambio en el proceso es deseando entonces se deberá de proceder a adecuar el contrato de acuerdos para reflejar esta variación en los procesos o productos del proveedor. [SAM SP 1.3.6]

4.7 PLANTILLAS

4.7.1 Plan de Gestión del Proyecto (PMP)

PLAN DE GESTIÓN DE PROYECTO (PMP)

[Fecha de Creación]

[Versión*]

Autor: _____

Aprobado por: _____

*La versión exacta será controlada por la herramienta de versiones (TFS), por lo cual solo se cambiara la versión cuando sea un cambio significativo en el documento y no solo incremental.

1. ALCANCE Y DESCRIPCIÓN GENERAL DEL SISTEMA

Descripción general del sistema

[Escribir aquí una descripción general del sistema, su propósito, componentes físicos, usuarios y una declaración de la visión a largo plazo y la visión de liberaciones tempranas. Incluir un diagrama de referencia e identificar en donde se puede encontrar el documento con la Estructura de Desglose de Trabajo (*Work Breakdown Structure - WBS*)]

Productos incluidos en esta sección:

Alcance, Estructura de desglose de Trabajos (WBS).

Relación con otros proyectos o entes externos al proyecto.

Proyecto/Entidad	**Relación / Interfaz**	**Datos de Contacto del Responsable**	**Fecha Compromiso**
[Nombre y descripción del proyecto o entidad que tiene compromisos con este proyecto]	[Se describe la relación o la interfaz que se tendrá entre el proyecto y la entidad externa involucrada]	[Información de Contacto de la persona responsable de la entidad externa]	[Fecha de entrega comprometida]

2. ORGANIZACIÓN Y PERSONAL

Organización

[Escribir aquí una descripción de la organización del proyecto. Incluir los roles principales y las responsabilidades, como: Propietario del Producto, *Scrum Master* y Equipo *Scrum*. Incluir la descripción de los equipos claves del proyecto]

Personal

[Escribir aquí un plan del personal esperado o en su caso la referencia hacia el documento externo en donde se encuentra este plan]

Rol	Personal Asignado	Habilidades Requeridas *

*Si la habilidad mencionada es única dentro del equipo deberá de ser subrayada, ya que remover un recurso que posee una habilidad que no puede ser compensada por otro miembro del equipo trae consecuencias mayores.

Partes Involucradas

[Agregar una matriz de partes involucradas o hacer referencia a la ubicación de este documento]

Capacitación

[Proporcionar aquí una declaración acerca de los planes de capacitación para el personal]

3. CICLO DE VIDA Y CALENDARIO

Modelo de Ciclo de Vida

[Proporcionar una descripción del ciclo de vida del proyecto. Lo normal será utilizar el ciclo de vida *scrum*, por lo cual ya contiene el ciclo de vida *scrum.* Es posible modificarlo para reflejar las particularidades del proyecto]

1. Planificación general.
2. Análisis general (Creación de *Backlog*)
3. Iteración 1
    a. Planificación
    b. Iteración
        i. Análisis
        ii. Diseño
        iii. Desarrollo
        iv. Pruebas
    c. Liberación
4. Iteración 2
    a. …..
5. ….
6. Iteración X
7. Implantación

Calendario del Proyecto

[Describir el plan para desarrollar, revisar, aprobar y mantener el calendario del proyecto. Incluir la ubicación en donde el Calendario* del proyecto será mantenido y la frecuencia de su actualización]

* El calendario del proyecto será creado utilizando *MS Project*.

4. MONITORIZACIÓN Y CONTROL DEL PROYECTO

Organización de Tareas

[Describir el proceso mediante el cual las tareas serán comunicadas a los responsables de las mismas. Incluir técnicas escritas y verbales, además de la ubicación en donde serán mantenidas las descripciones de las tareas]

Revisiones Periódicas del Proceso

[Describir las revisiones periódicas planeadas del proceso. Incluir información como el día, tiempo y frecuencia de las reuniones, los temas discutidos, las minutas tomadas y el procesamiento de los elementos de acción]

Revisión de Hitos

[Describir cuando las revisión de los hitos son planeados y el propósito de cada revisión. La revisión de hitos se realiza para comunicar avances y problemas a las principales partes interesadas. Estas revisiones están enfocadas en mantener el interés de las partes involucradas.]

Revisiones de Colegas

[Describir las juntas planeadas del equipo de desarrollo y los productos a ser revisados por los colegas. Identificar el personal requerido para atender las juntas]

Ejemplo:

Producto a Revisar	**Personal Requerido**
Documento de Requerimientos del Sistema	Autor, Jefe de Ingeniería, *Scrum Master*, Propietario del Producto
Documento de Arquitectura del Sistema	Autor, Jefe de Ingeniería, *Scrum Master*

Gestión de Riesgo

[Describir o referenciar el proceso por el cual los riesgos son identificados, documentados, administrados y el enfoque de mitigación de riesgo (bajo qué criterios un riego dispara el proceso definido para ser mitigado)]

[De existir una bitácora de Riesgos, hacer referencia a ella.]

Ejemplo: Los miembros del equipo de trabajo son motivados para sacar a la luz los riegos y los problemas durante los *Stand Up Meetings*. El equipo usa el siguiente criterio: _____ para ayudar a determinar cuando los riesgos deben de ser elevados al siguiente nivel para ser atendidos.

Ambiente de Trabajo y Herramientas

[Identificar las herramientas utilizadas en el proyecto]

Herramienta	Propósito de la Herramienta

Gestión de Configuración

[Identificar los productos que serán controlados en el Sistema de Gestión de Configuración]

Aseguramiento de Calidad

[Identificar como la administración lograra el objetivo de obtener una visión interna del proceso y la calidad de producción]

Mantenimiento de este Plan

[Describir el proceso mediante el cual será mantenido este Plan. Incluyendo quien es el responsable del mantenimiento]

Ejemplo: Este Plan de Gestión del Proyecto (PMP) será mantenido por el líder de Proyecto. La versión original será revisada y aceptada por el Equipo de Desarrollo y aprobada por el administrador del proyecto. El documento es actualizado según sea necesario al inicio de cada incremento mayor. Las versión de este documento serán mantenidos en el sistemas de control de versión de *Team Foundation Server*. Las revisiones menores serán….

5. METRICAS

[Describir las métricas planeadas para ser recolectadas en el proyecto. Incluir como las métricas serán obtenidas, la frecuencia de obtención, como serán analizadas, almacenadas y comunicadas]

Ejemplo:

Métrica	Descripción
Requerimientos	Estado real y cambios desde el último periodo reportado.

Calendario	Actualizado semanalmente. El calendario es mantenido en el repositorio del proyecto en TFS.
Coste	Tasa de gasto actual
Personal	Planificación actual de personal y personal real.

FUENTE: MODIFICADO DE (McMahon, 2010, p. Anexo 2)

4.7.2 Tiempos y Costes

TIEMPOS Y COSTES

[Fecha de Creación]

[Versión*]

Autor: _____

Aprobado por: _____

*La versión exacta será controlada por el manejador de versiones (TFS), por lo cual solo se cambiara la versión cuando sea un cambio significativo en el documento y no solo incremental.

Se hace un resumen de las horas y coste necesarios para completar el proyecto, además de las fechas esperadas de entrega de cada uno de los elementos.

Descripción	Horas	Coste	Fecha Entrega
Tema 1			
Historia de Usuario 1	10	1,000 €	12/12/2013
Historia de Usuario 2	10	1,000 €	12/12/2013
Historia de Usuario 3	20	2,000 €	12/12/2013
Tema 2			
Historia Épica 1	40	4,000 €	12/12/2013
TOTAL:	80	8,000 €	

4.7.3 Bitácora de Riesgos del Proyecto

BITÁCORA DE RIESGOS DEL PROYECTO

[Fecha de Creación]

[Versión*]

Autor: _____

Aprobado por: _____

*La versión exacta será controlada por la aplicación de versiones (TFS), por lo cual solo se cambiara la versión cuando sea un cambio significativo en el documento y no solo incremental.

Descripción del Riego	Posibilidad Ocurrencia (1-5)	Impacto (1-5)	Calificación del Riesgo (Alto/Med/Bajo)	Medidas de Contención	Fecha Identificación	Última Actualización	Estatus

Posibilidad Ocurrencia:

1. Muy Escasa
2. Escasa
3. Posible
4. Muy Posible
5. Casi Seguro

Impacto:

1. Insignificante
2. Menor
3. Moderado
4. Mayor
5. Catastrófico

4.7.4 Matriz de Partes Involucradas

MATRIZ DE PARTES INVOLUCRADAS

Autor: _____ [Fecha de Creación]

Aprobado por: _____ [Versión*]

*La versión exacta será controlada por la aplicación de versiones (TFS), por lo cual solo se cambiara la versión cuando sea un cambio significativo en el documento y no solo incremental.

Personal Involucrado/ Etapa Ciclo Vida	**Planificación General (Creación *Backlog* y Estimados)**	**Habilitar Herramientas y Área Trabajo**	**Planificación del *Sprint***	**Desarrollo *Sprint***	**Pruebas *Sprint***	**Instalar y liberar *Sprint***	**…**
Scrum Master (LRS)		-	Moderar la Junta de Planificación	Moderar *Stand Up Meetings*, resolver problemas a los miembros del equipo	Dirigir pruebas	-	…
*Equipo Desarrollo (JKL, LRT, SPR)**	Estimación de tiempos.	-	Compromiso con los elementos a desarrollar	Codificación de las tareas del sprint	Pruebas de unidad	-	…
Propietario del Producto. (JRT)	Definir las historias de usuario…	-	Definir prioridades para productos	-	Pruebas generales del *Sprint*	Aprobar la liberación.	…
Encargado Gestión Configuración .(LRG)	Estimación tiempos de creación de Configuración	Creación de Configuración.	-	-	-	Instalación en máquinas de producción.	…

4.7.5 Bitácora de Gestión de Datos

BITACORA DE GESTIÓN DE DATOS

Autor: _____ [Fecha de Creación]

Aprobado por: _____ [Versión*]

*La versión exacta será controlada por la aplicación de versiones (TFS), por lo cual solo se cambiara la versión cuando sea un cambio significativo en el documento y no solo incremental.

Id. de los Datos	Medio	Tipo de dato	Nivel de Autorización	Dueño de los datos	Ubicación	Responsable	Fecha Recepción	Fecha Regreso o Borrado.

Id Datos: Un identificador único para referenciar los datos.
Medio: F - Físico, D – Digital
Tipo Datos: M-Manual, G-Guía, P-Publicidad, E-Especificación, T-Técnico.
Nivel de Autorización: Se define la autoridad requerida para acceder a los datos, a nivel rol o usuario individual.
*La ubicación de los datos físicos debe de estar siempre bajo llave.

4.7.6 Configuración de Hardware y Herramientas

CONFIGURACIÓN DE HARDWARE Y HERRAMIENTAS

Proyecto:	
Tipo Ambiente:	Versión:
Cambios en la última versión:	

Arquitectura de la aplicación

Id de Hardware:	
Descripción	
Requerimientos mínimos hardware CPU: Memoria: Almacenamiento: Sistema Operativo:	Software requerido: - - -

Arquitectura de Desarrollo

Perfil de puesto de desarrollo 1:	
Descripción	
Número de puestos:	
Requerimientos mínimos hardware CPU:	Herramientas de Software: - -

Memoria: Almacenamiento:	-

Perfil de puesto de desarrollo 2:	
Descripción	
Número de puestos:	
Requerimientos mínimos hardware CPU: Memoria: Almacenamiento:	Herramientas de Software: - - -

4.7.7 Diccionario de infraestructura

DICCIONARIO DE INFRAESTRUCTURA

Nombre del Elemento:	
Área Funcional:	
Propósito:	
Tipo de elemento:	
Áreas Relacionadas:	
Forma de uso:	
Referencia hacia la documentación del elemento:	

Nombre del Elemento:	
Área Funcional:	
Propósito:	
Tipo de elemento:	
Áreas Relacionadas:	
Forma de uso:	
Referencia hacia la documentación del elemento:	

4.7.8 Lista de verificación y mínimos

LISTA DE VERIFICACIÓN Y MÍNIMOS

Proyecto: <Nombre del Proyecto>	Versión:

LISTA DE MINIMOS DE CALIDAD GENERAL.

1. Mínimo 1
2. Mínimo 2

LISTA DE MINIMOS DE CALIDAD PARTICULAR

Elemento:	<Nombre del Elemento 1>
Tipo:	<Tipo de Elemento (Documento, Código, Ejecutable, Informe, etc.)>

Mínimos

1. <Mínimo 1>
2. <Mínimo 2>

Elemento:	<Nombre del Elemento 2>
Tipo:	<Tipo de Elemento (Documento, Código, Ejecutable, Informe, etc.)>

Mínimos

1. <Mínimo 1>
2. <Mínimo 2>

Elemento:	<Nombre del Elemento n>
Tipo:	<Tipo de Elemento (Documento, Código, Ejecutable, Informe, etc.)>

Mínimos

1. <Mínimo 1>
2. <Mínimo 2>

4.7.9 Recolección y Análisis de Métricas

ESPECIFICACIÓN DE RECOLECCIÓN Y ANÁLISIS DE MÉTRICAS

Proyecto: <Nombre del Proyecto>	Versión:

Id Métrica:	
Objetivo de Negocio:	
Información Requerida:	
Objetivo de Medida:	
Categoría:	
Como se realiza la medición:	
Cuando se tomara la medida:	
Mediciones Derivadas:	

Id Métrica:	
Objetivo de Negocio:	
Información Requerida:	
Objetivo de Medida:	
Categoría:	
Como se realiza la medición:	
Cuando se tomara la medida:	
Mediciones Derivadas:	

4.7.10 Requisición de Productos y Servicios

REQUISICIÓN DE PRODUCTOS Y SERVICIOS

Este formato será rellenado cuando sea requerido solicitar la compra de productos y/o servicios a un proveedor.

Proyecto: <Nombre del Proyecto>	Versión: <Versión>
Responsable del Proyecto:	
Fecha Limite de Recepción de Propuestas:	

REQUERIMIENTOS

1. **Requerimientos de Interfaces**
    a. Interfaz de usuario
    b. Interfaz de Hardware
    c. Interfaz de Software
    d. Interfaz de Comunicación

2. **Requerimientos Funcionales**
    a. <Requerimiento Funcional o Característica #1>
        i. Introducción
        ii. Entradas
        iii. Proceso
        iv. Salidas
        v. Manejo de Errores
    b. <Requerimiento Funcional o Característica #n>
        i. Introducción
        ii. Entradas
        iii. Proceso
        iv. Salidas
        v. Manejo de Errores

…

3. Casos de Uso
    a. Caso de Uso 1
    b. Caso de Uso 2
4. Requerimientos No Funcionales
    a. Desempeño
    b. Confiabilidad
    c. Seguridad

d. Mantenimiento
e. Portabilidad

5. Restricciones de Diseño

<Especificar las restricciones de diseño que pueden afectar el proyecto de software, impuestos por otros estándares, políticas de la compañía, limitaciones de hardware, etc.>

6. Otros Requerimientos

4.8 PRUEBA PILOTO

4.8.1 Descripción Prueba

Se realiza la implementación de la metodología propuesta en este documento en una empresa de desarrollo de Software establecida en Houston, TX. USA.

La empresa tiene un personal que ronda los 20 elementos, de ellos 12 personas están dedicadas al desarrollo de Software y el resto a labores administrativas y comerciales. No cuenta con ningún tipo de metodología de desarrollo y cada equipo de desarrollo trabaja de acuerdo a las circunstancias y los procesos que el líder del proyecto cree convenientes. Tiene trabajando así por más de 10 años de forma más o menos exitosa, pero el último par de años la desorganización se ha ido incrementando; el número de clientes ha aumentado y la cantidad de trabajo es mayor. La necesidad de contratar más personal ha complicado aún más las cosas, ya que existe un cuello de botella en un par de personas claves en la empresa por donde pasan todas las decisiones de desarrollo; dando como resultado que el contratar nuevo personal solo implique más personas esperando a recibir asignación de trabajo o autorizaciones para proseguir con sus labores.

Debido a la necesidad de mejorar los procesos de desarrollo de la compañía, ésta accedió a implementar la metodología propuesta en un proyecto piloto. Este proyecto tiene un equipo que está compuesto por cuatro desarrolladores y un *Scrum Master*, además del propietario del producto que será el enlace entre el cliente y el equipo de desarrollo. El proyecto tiene una duración estimada de 1 año, pero si el resultado es del agrado del cliente se extendería indefinidamente.

La metodología se ira adoptando de forma gradual, comenzando por lo relacionado directamente con la planificación y el desarrollo. Es decir se inicia con la fase de planificación y construcción, buscando cumplir la mayoría de los requerimientos CMMI de las áreas de Planificación de Proyecto (PP), Monitorización y Control del Proyecto (PMC), Gestión de Configuración (CM) y Gestión de Requerimientos (REQM). Por decisión de la empresa, relacionada con los recursos y necesidades del cliente, las áreas de Aseguramiento de la Calidad del Proceso y del Producto (PPQA) y Medición y análisis (MA) serán implementadas en el proceso 6 meses después del inicio del proyecto con el ingreso de un nuevo recurso a la empresa que tomara los roles de Entrenador de Calidad –se asumen los riesgos y trabajo extra de esa decisión. El área de Gestión de Acuerdos con Proveedores (SAM) no es necesaria al no haber proveedores involucrados en el proyecto, se agregara al proceso si fuese necesario.

4.8.2 Experiencias

Al día de hoy se ha pasado por tres *Sprints* utilizando esta metodología. Se observó una disminución muy considerable a la sensación de caos gobernante en los proyectos, en donde los cambios de prioridades eran cosa diaria. Aunque estos cambios de prioridades siguen siendo posibles, el no permitir cambios de requerimientos dentro de un Sprint ayudó a disminuir la

sensación de navegar sin rumbo. Un beneficio muy apreciado por los lideres es la disminución de la dependencia que existía hacia ellos en la toma de decisiones, pasando la responsabilidad de la toma de decisiones a cada uno de los miembros del equipo de desarrollo *Scrum* sobre lo referente a el trabajo que está realizando cada uno de ellos. El tamaño de los *Sprints* que es de 6 semanas y el uso de los *Scrum Diarios* permite a los líderes –*Scrum Masters*– un grado de tranquilidad al poder realizar un seguimiento diario, pero limitado, a los trabajos de los miembros del equipo de desarrollo. Si todavía así se escapa alguna decisión errónea por parte de los desarrolladores, el alcance de 6 semanas del Sprint evita que este error crezca a proporciones alarmantes.

La otra preocupación de la empresa era el tiempo extra que era requerido invertir en documentación y en el uso de las herramientas necesarias para la metodología. Se encontró que el trabajo necesario es irrelevantes al calcularse que se está invirtiendo menos 5% del tiempo total de desarrollo en estas actividades. La falta de costumbre del equipo a realizar estas tareas los obliga a regresar a actualizar algún documento cuando el momento establecido para hacerlo ya ha pasado. Por ejemplo: sigue siendo común que se realicen tareas sin tener creado el registro de la tarea en la herramienta de gestión del ciclo de vida. Gracias a que existe un candado que evita que los desarrolladores puedan mandar su código a el Servidor de *Builds* sin tener el registro creado se conserva la integridad, esto hace ver la importancia de tener candados en determinadas etapas del desarrollo para mantener la integridad del proceso. Aun así el tiempo invertido en llevar la documentación del proceso no se ve como un problema por parte de la empresa y el equipo de desarrollo.

Por último el tener documentación de la toma de decisiones y trabajos ha disminuido mucho las discusiones referentes a la diferencia de interpretación de los requerimientos entre las personas involucradas al momento de resolver alguna tarea.

La experiencia en general ha sido positiva para la empresa y los planes de implementarla al 100% en este proyecto se mantienen. Si después de un año se puede apreciar la diferencia entre este proyecto y el resto de los proyectos, la metodología se implementara en todos los proyectos de la compañía. Cabe mencionar que tal y como el propósito de esta guía lo indica, la metodología se irá adecuando a las necesidades de la empresa hasta hacerla suya. Nunca ha sido el objetivo que se siga esta guía sin efectuar cambios en algunas partes de ella al ser implementada, la única restricción es que estos cambios no signifiquen el incumplimiento de los objetivos específicos de CMMI-DEV nivel 2 de madurez.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5 CONCLUSIONES Y LÍNEAS FUTURAS

Como era esperado, una buena cantidad de las prácticas específicas requeridas por CMMI nivel 2 de madurez son cumplidas sin necesidad de realizar cambio alguno a la metodología de desarrollo *Scrum*; tal como lo estipulan los trabajos existentes en el tema del uso de metodologías ágiles con CMMI-DEV. Esto se comprobó identificando directamente en el proceso *Scrum* las actividades que permitían cumplir con estos requerimientos.

En cuanto a las prácticas especificas cuyo cumplimiento no se logra con la metodología Scrum, se agregaron dentro de la misma secuencia de procesos *Scrum* las actividades que permiten el cumplimiento al 100% de los requerimientos CMMI-DEV nivel 2 de madurez, sin alterar los principios de filosofía ágil de desarrollo.

Pocos resultados se obtuvieron de la prueba que se realizó utilizando esta metodología en el proyecto piloto en donde se implementó parte de ella, principalmente debido a que solo fue posible correr 3 *Sprints* utilizando la metodología y que no se implementaron al cien por ciento todos los procesos establecidos en la metodología.

Partiendo de una empresa en donde los proyectos eran llevados completamente a criterio del líder de proyecto, se observó una disminución muy considerable en la incertidumbre dentro de los proyectos, en donde los cambios de prioridades eran cosa diaria. El tiempo invertido en documentación y en usar las herramientas necesarias para la metodología fueron irrelevantes, aunque la falta de costumbre obligó constantemente a los miembros del equipo a dar un paso atrás y regresar a actualizar algún documento cuando el momento estipulado para ello ya había pasado.

La percepción de la metodología es positiva, se continuara su uso y su integración hasta lograr el total de las actividades establecidas. De seguir siendo positivo el sentimiento, se implementará en el resto de la empresa.

Como líneas futuras se proponen, para obtener una mejor sensibilidad de los beneficios y deficiencias de esta metodología propuesta sería necesario llevar un proyecto real a término, desde su planificación hasta su transición, el cual pueda ser analizado y evaluado. Así mismo como evaluar que la metodología propuesta cumpla con las Metas y Prácticas Genéricas establecidas por CMMI-DEV, de ser necesario hacer las adecuaciones necesarias en la metodología para su cumplimiento.

Pudiera aportar gran valor a esta propuesta el lograr la certificación correspondiente con CMMI-DEV nivel 2 de madurez en una empresa implantando esta metodología o en su defecto la validación por parte de un profesional en certificaciones CMMI de la validez de la metodología.

De tal forma que se puedan hacer las adecuaciones o complementar la metodología con las actividades pertinentes.

Después de lograr la validación práctica de esta guía en un proyecto real, lo lógico sería continuar con los siguientes niveles de madurez comenzando con el nivel 3 y extender la metodología para lograr el cumplimiento de CMMI-DEV en todos sus niveles. No sin antes realizar una evaluación sobre si sería posible crear procesos genéricos para el cumplimiento de las Prácticas Especificas de CMMI-DEV nivel 3 de madurez o si estas prácticas están demasiado ligadas con las estructuras específicas de cada empresa y es necesario resolverlas de forma particular en cada empresa, lo cual haría imposible una guía genérica para niveles de madurez mayores al 2.

Referencias

Agile Alliance. (2001). *Agile manifesto.* Obtenido de http://www. agilemanifesto.org

Ambler, S. W., & Lines, M. (2012). *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise.* IBM Press.

CMMI Product Team. (2010). *CMMI for Development, Version 1.3.* Software Engineering Institute.

Duvall, Paul M, Matyas, Steve, & Glover, Andrew. (2007). *Continuous integration: improving software quality and reducing risk.* Pearson Education.

Eckstein, J. (2013). *Agile Software Development with Distributed Teams: Staying Agile in a Global World.* Addison-Wesley Professional.

Glazer, H., Dalton, J., Anderson, D., Konrad, M. D., & Shrum, S. (2008). CMMI® or Agile: Why Not Embrace Both! *Software Engineering Institute*.

Guckenheimer, S., & Loje, N. (2012). *Visual Studio Team Foundation Server 2012: Adopting Agile Software Practices: From Backlog to Continuous Feedback, Third Edition.* Addison-Wesley Professional.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994). *Benefits of CMM-Based Software Process Improvement: Initial Results.* Pittsburgh: Carnegie Mellon University.

Hundhausen, R. (2012). *Professional Scrum Development with Microsoft® Visual Studio® 2012.* Microsoft Press.

Larman, C., & Vodde, B. (2010). *Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum.* Pearson Education.

McMahon, P. E. (2010). *Integrating CMMI and agile development: case studies and proven techniques for faster performance improvement.* Pearson Education.

Olausson, M., Rossberg, J., Ehn, J., & Sköld, M. (2013). *Pro Team Foundation Service.* Apress.

Potter, N., & Sakry, M. (March de 2009). IMPLEMENTING SCRUM (AGILE) AND CMMI® TOGETHER. *The Process Group - Post, 16*(2).

Resnick, S., Bjork, A., & Maza, M. d. (2011). *Professional Scrum with Team Foundation Server 2010.* Wrox.

Schwaber, K., & Sutherland, J. (2011). *The scrum guide.* Scrum.org. Obtenido de Scrum.org.

Torrecilla, J., Escalona, J., & Mejias, M. (2012). A Scrum-based approach to CMMI maturity level 2 in Web. *iiWAS2012.* Bali.

Venkatesh, U. (2011). *Distributed Agile, DH2A: The Proven Agile Software Development Approach and Toolkit for Geographically Dispersed Teams.* Technics Publications.

Woodward, E., Surdek, S., & Ganis, M. (2010). *A Practical Guide to Distributed Scrum.* IBM Press.

Glosario

Add-in: Add-in o plug-in es un componente de software que añade una característica específica a una aplicación de software existente.

Agile: Relativo al desarrollo ágil de software se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinarios

Backlog: lista de elementos de trabajo que necesitan ser completados para entregar la solución al cliente.

Build: Es una versión compilada procedente de un conjunto de códigos fuentes.

Build Server: Servidor encargado de recibir los códigos fuentes de la aplicación y compilarlos para crear una versión o build a partir de ellos. En este proceso realiza de forma automática la ejecución de unidades de pruebas de forma periódica o después de cada *commit* y reporta los resultados a los desarrolladores.

Burn-downs: Un diagrama burn down o diagrama de quemado es una representación gráfica del trabajo por hacer en un proyecto en el tiempo. Usualmente el trabajo remanente (o backlog) se muestra en el eje vertical y el tiempo en el eje horizontal

Burn-up: Diagrama prácticamente igual a Burn-downs pero que en lugar de rastrear que tanto trabajo falta por hacer, se rastrea que tanto ya fue hecho.

CMMI: Integración de modelos de madurez de capacidades o Capability maturity model integration (CMMI) es un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software.

COTS: Componente tomado fuera del estante, tal cual o Commercial Off-The-Shelf (COTS) es un término del Reglamento Federal de Adquisiciones (FAR), que define un elemento no-desarrollativo (NDI) de suministro, que es a la vez comercial y se vende en grandes cantidades en el mercado comercial, y que puede ser adquirido o utilizado bajo contrato gubernamental de la misma forma exacta a como está disponible al público en general.

Flujos acumulativos: Diagrama utilizado en teoría de colas. Es una gráfica de área que muestra la cantidad de trabajo en un estado dado, mostrando llegadas, tiempo en cola, cantidad en cola y salidas.

Grupo focal: También conocido como grupo de discusión (focus group en inglés) es una técnica cualitativa de estudio de las opiniones o actitudes de un público, utilizada en ciencias sociales y en

estudios comerciales. Consiste en la reunión de un grupo de personas, entre 6 y 12, con un moderador, investigador o analista; encargado de hacer preguntas y dirigir la discusión.

Hotfix*:* Paquete que puede incluir varios archivos y que sirve para resolver un bug específico dentro de una aplicación informática.

Lluvia de Ideas: La lluvia de ideas, también denominada tormenta de ideas, es una herramienta de trabajo grupal que facilita el surgimiento de nuevas ideas sobre un tema o problema determinado

MS Project: Software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

Nearshore: es un tipo de subcontratación o externalización de una actividad con salarios más bajos que en el propio país, que se encuentra relativamente cerca en la distancia o la zona horaria.

Oficina de Gestión de Proyectos (PMO): También conocida por sus siglas OGP o PMO (del inglés project management office), es un departamento o grupo que define y mantiene estándares de procesos, generalmente relacionados a la gestión de proyectos, dentro de una organización.

Offshore: designa la actividad por parte de empresas con sede en un determinado país de trasladar o construir fábricas o centros de producción en otro país, donde por lo general enfrentarán menores costes en mano de obra, menor presión en leyes laborales, menor cantidad de normativas gubernamentales, reducción de otro tipo de costes, u otros beneficios cualesquiera desde el punto de vista del lucro económico para la empresa.

PMP: Plan de Gestión de Proyecto, en ingles Project Managment Plan, es el documento de la planificación, captura el proyecto entero de principio a fin, abarca todas las fases desde la iniciación, planificación, ejecución y finalización.

Plan de Gestión de Proyecto: ver PMP.

Rollback: Regresar al estado exacto anterior en el que se encontraban un conjunto de elementos antes de ser modificados.

Scrum de Scrums: Son juntas con el mismo formato que las juntas de Scrum Diario con la única diferencia, que debido al tamaño de los equipos, solo un miembro de cada equipo Scrum asiste en representación de los miembros de su equipo. No remplaza la junta de Scrum Diario, sino que se realiza a un nivel diferente.

Sharepoint: También conocido como Microsoft SharePoint Products and Technologies, es una plataforma de colaboración empresarial, formada por productos y elementos de software que incluye, entre una selección cada vez mayor de componentes, funciones de colaboración, basado

en el Explorador web, módulos de administración de proceso, módulos de búsqueda y una plataforma de administración de documento.

Team Foundation Server: Es un producto de Microsoft que proporciona gestión de código fuete, generación de informes, gestión de proyectos, compilación de versiones automáticos, gestión de laboratorios, pruebas y gestión de implementaciones.

TFS: ver Team Foundation Server.

Visio: Microsoft Visio es un software de dibujo vectorial para Microsoft Windows. Visio comenzó a formar parte de los productos de Microsoft cuando fue adquirida la compañía Visio en el año 2000.

WBS Modeler: Add-in de Visio que permite la creación de diagramas WBS que pueden ser exportados posteriormente a MS Proyecto.

WBS: Una Estructura de Descomposición del Trabajo o EDT, también conocida por su nombre en inglés Work Breakdown Structure o WBS, es en gestión de proyectos una descomposición jerárquica orientada al entregable, del trabajo a ser ejecutado por el equipo de proyecto, para cumplir con los objetivos de éste y crear los entregables requeridos, con cada nivel descendente de la EDT representando una definición con un detalle incrementado del trabajo del proyecto. La EDT es una herramienta fundamental en la gestión de proyectos

Wiki: Es un sitio web cuyas páginas pueden ser editadas por múltiples voluntarios a través del navegador web.