Estructura de Alto Nivel Del Desarrollo del ciclo de vida de Software y sus componentes para certificación ISO 29110 – 4-1

Ingeniería de Software. Perfiles del Ciclo de Vida para pequeñas entidades (PES). Parte 4-1: Especificaciones del perfil genérico.

Bogotá 21 de Febrero de 2017.

Contenido

Marco Metodológico

El marco metodológico nos brinda una guía de "cómo hacer las cosas" tanto internamente como externamente, refiriéndonos a nuestros clientes (externo), pero también a la forma de trabajo interna con nuestros pares y superiores en Holística Organizacional.

En Holística Organizacional, dado su ADN corporativo y la naturaleza de los clientes bases, que en su mayoría pertenecen al sector público, se ha decido trabajar una metodología mixta que comprende dos de los marcos de trabajo más usados en el mercado:

- PMI: Project Management Institute.
- Scrum: Marco de trabajo ágil.

En nuestro relacionamiento externo, se trabajará con PMI para el seguimiento y control de las actividades con el cliente, tanto de planificación como de cierre de proyectos, pero en la etapa de ejecución de PMI, utilizaremos Scrum al interior con el objetivo de tener una metodología de desarrollo de software madura y que genere los resultados esperados de una forma continua.

Marco Metodológico PMI para cliente

El marco metodológico que utilizaremos en Holística Organizacional será PMI, pero no es competencia de este documento explicar el funcionamiento y aplicación metodológica de los procedimientos que competen al área de Gerencia de Proyectos de la compañía, ya que para ello existe el documento: **Procedimiento de Planificación de Proyectos** y **Plantilla de Proyectos.**

Sin embargo, en este documento dejamos claridad de cómo se alinean las dos metodologías.

Las etapas del PMI son las siguientes:

1. Inicio
2. Planificación
3. Ejecución
4. Supervisión y control
5. Cierre

Dado las etapas mencionada anteriormente, es en la tercera, **Ejecución,** la cual se desarrollará bajo la metodología ágil **SCRUM** bajo los lineamientos propios de esta y es de total competencia de este documento explicar de forma granular los componentes y el desarrollo de cada uno de ellos.

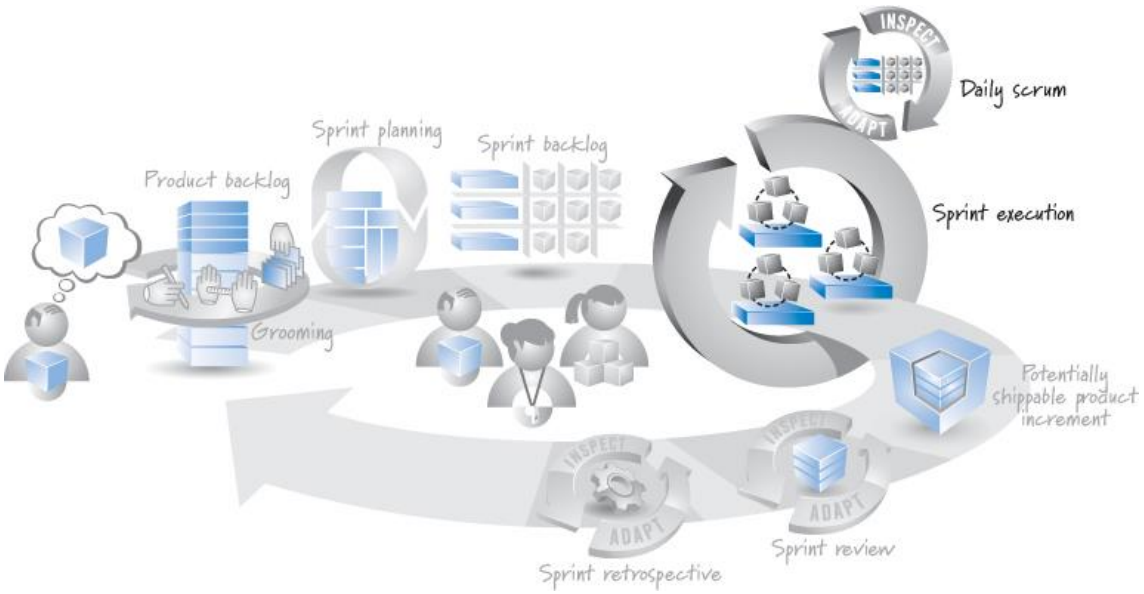
Cabe resaltar que el seguimiento de los proyectos PMI se realiza a través del porcentaje de cumplimiento de las actividades establecidas en el cronograma en Project y en SCRUM se hace a través del ***Burndown,*** el cual hace referencia a las Historias de Usuario culminadas en los Sprints establecidos. En este punto debe existir una alta comunicación y un establecimiento de las reuniones que se deben realizar entre el PM (Project Manager) y el Scrum Master (líder de la célula de desarrollo) para determinar los avances de cara al cliente.

Framework SCRUM interno Holística Organizacional

SCRUM es un enfoque ágil que es utilizado para el desarrollo de productos y servicios. La manera de iniciar con este enfoque es que se inicia creando un ***Product Backlog***, lo cual es una lista priorizada de características y otra de capacidades requeridas para desarrollar exitosamente un producto.

Guiados siempre por el ***Product Backlog,*** siempre se tendrá el trabajo más importante con la prioridad más alta como primer ítem. Cualquier trabajo o requerimiento, que no se haya completado y que no genere valor, se moverá dentro del ***Product Backlog,*** a las posiciones más baja, tomando así la menor prioridad.

La siguiente figura muestra el marco de trabajo de SCRUM:



En Holística Organizacional, trabajaremos un enfoque ágil personalizado acorde a nuestro ADN corporativo.

La base fundamental para la creación del ***Producto Backlog***, es el levantamiento de los requerimientos a través de ***Diagnóstico Técnico y Funcional*** que se realiza en Holística.

El ***Diagnóstico*** consiste en una visita de campo donde previamente se han identificado los StakeHolders, los procesos, procedimientos y enfoque tecnológico para determinar los requerimientos funcionales y no funcionales de la solución. En este documento se traza la hoja de ruta y los objetivos estratégicos del proyecto y lo que se espera lograr acorde a la visión que tiene el cliente. Este documento es el alineamiento entre las expectativas del cliente y la realidad. A su vez, este análisis lo transformamos en ***Historias de Usuario*** que son el componente principal del ***Product Backlog.***

Partiendo de la base explicada anteriormente, a continuación detallamos cada uno de los componentes que hacen parte del enfoque ágil de SCRUM.

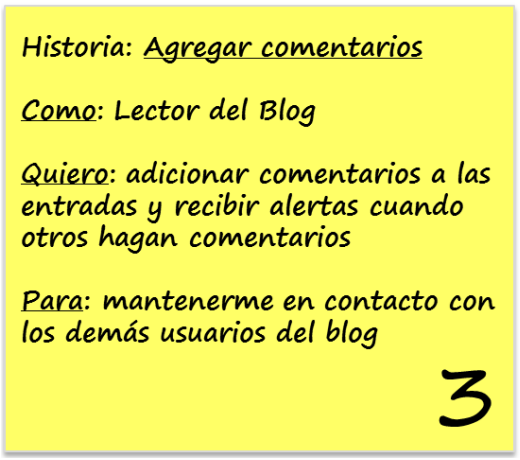
Levantamiento de Requerimientos

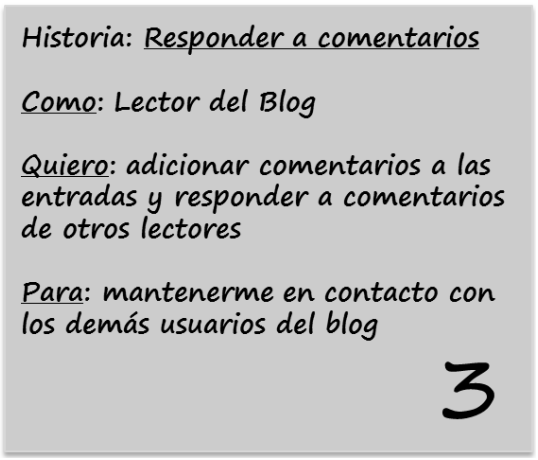
Definición de Historia de Usuario Creación de una Historia de Usuario

Las historias de usuario son un formato para expresar el valor deseado por el negocio. Las historias de usuario son desarrolladas en una forma que ambos, tanto el personal técnico como de negocio puedan entender. Adicionalmente pueden ser escritas en varios niveles de granularidad y son fácil de refinar progresivamente.

La forma de representar una ***Historia de Usuario*** es simple, es una tarjeta o post-it, la cual tiene un tamaño pequeño y el objetivo principal es poder plasmar ideas simples y concretas, para no caer en demasiados tecnicismos y complicar la Historia a nivel funcional.

A continuación relacionamos dos ejemplos de ***Historia de Usuario:***





Como podemos ver en el ejemplo anterior, una ***Historia de Usuario*** es un requerimiento escrito de forma conversacional y simple pero que cumple una estructura base, la cual se debe cumplir.

La ***Historia de Usuario,*** comprende 3 factores fundamentales dentro de su escritura para su estructura:

- ***¿Cómo? o ¿Quién?:*** El cómo o quién, define el rol dentro de la organización y a su vez dentro del software que realiza la acción. En este caso es claramente definido que la acción es para el rol ***Lector.***
- ***El quiero o el ¿qué?:*** El quiero o el ¿qué?, define el requerimiento que necesita o desea el rol. En este caso el ***Rol Lector quiere o desea*** que el software le permita adicionar comentarios a las entradas y recibir alertas cuando otros hagan comentarios.
- ***¿Para qué?:*** El para qué define la justificación del quiero o el ¿Qué? En este caso el ***Rol Lector requiere adicionar comentarios...para mantenerse en contacto con los otros usuarios del blog.***

De esta forma y respetando la estructura podemos escribir las ***Historias de Usuario*** de forma fácil y entendible para los usuarios de negocio y los usuarios técnicos. Además cada una de ellas conforma el listado de características funcionales y no funcionales del software como ítem del faro guía que es el ***Product Backlog.***

Características de una buena Historia de Usuario

Para determinar si una ***Historia de Usuario*** es una buena ***Historia de Usuario,*** se ha determinado que debe cumplir con el **INVEST.**

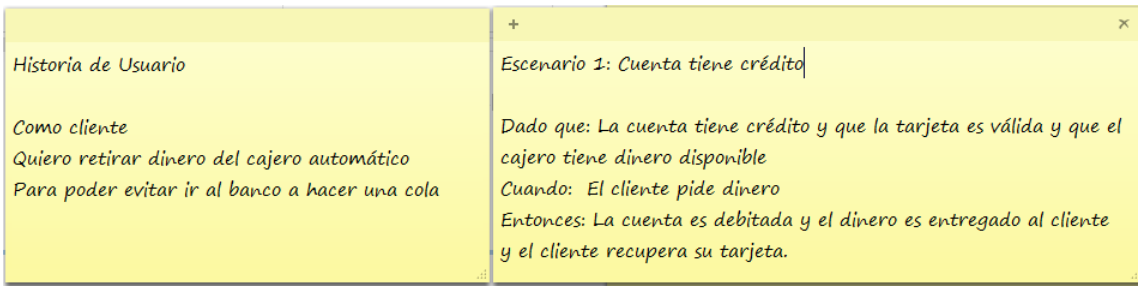
INVEST, traduce los siguientes aspectos que debe tener una buena ***Historia de Usuario:***

- **I (Independent – Independiente):** Una buena historia, no debe depender de otra, ya que cuando hay un acoplamiento entre ellas el trabajo se empieza a retrasar, ya que el equipo de desarrollo no puede iniciar la historia hasta que no se haya terminado otra. Cuando esta situación se presenta dentro de un Sprint (Timebox – Tiempo de ejecución iterativo) se declara la historia como ***impedida.***
- **N (Negotiable – Negociable):** Las ***Historias de Usuario***, deben poder ser negociables, ya que no están escritas como si fuera un contrato; de hecho, debe ser posible que tanto el equipo técnico y de negocio puedan discutir y definir el nivel de detalle.
- **V (Valuable – Valiosa):** Las ***Historias de Usuario***, deben generar valor al cliente, al usuario o ambos. No debería ser posible tener historias dentro del ***Product Backlog*** que no le generen valor a nadie y de todas formas desarrollarlas, ya que se genera un esfuerzo que termina es desperdicio de tiempo y dinero. Se debe tener cuidado con las ***Historias de Usuario Técnicas***, ya que si bien son necesarias para el equipo de tecnología, para el ***Product Owner*** (Dueño del Producto), puede que no sea visible y no le genere valor.
- **E (Estimable):** Las ***Historias de Usuario,*** deben poder ser estimables por el equipo de desarrollo que las diseñará, construirá y probará. La estimación nos lleva a poder determinar el tamaño (Complejidad) y el esfuerzo que se debe dedicar para cumplir con el requerimiento expresado en esta. Entre más pequeña sea la historia es más fácil de estimar. Hay se aplica el concepto de ***Historias Épicas.*** Las ***Historias Épicas,*** son aquellas que son demasiado grandes y tienen altos componentes de software y de negocio, lo que hace difícil de estimar por el equipo de desarrollo de software. La recomendación con este tipo de ***Historias de Usuario*** es que se divida en múltiples ***Historias de Usuario,*** con el objetivo de reducir la complejidad y aumentar el entendimiento.
- **S (Size – Tamaño):** El tamaño de las historias debe ser pequeño, ya que no queremos que el ***Sprint*** sea consumido por una o dos historias que tengan un tamaño demasiado alto.
- **T (Testable - Probable):** Las ***Historias de Usuario,*** se deben poder probar. A partir de los criterios de aceptación de la historia, se debe determinar el estado de la prueba de forma binaria: “Aceptada” o “No aceptada”. Básicamente es decir si la historia cuando sea probada, cumple o no con los criterios de aceptación.

Definición y Creación de Criterios de Aceptación

Los C***riterios de Aceptación*** son el complemento de las ***Historias de Usuario*** para determinar si se está cumpliendo con el requerimiento o no. Al igual que para las ***Historias de Usuario*** hay una estructura que se debe llevar, lo mismo sucede con los ***Criterios de Aceptación.***

A continuación mostramos un ejemplo:



En la parte Izquierda de la imagen de ejemplo, tenemos la ***Historia de Usuario*** tal como la definimos en el segmento anterior y en la parte derecha tenemos los ***Criterios de Aceptación*** de la historia en cuestión.

Es importante que se analice la estructura de los criterios, por tal motivo se explica:

- ***Dado que: "***Dado que", hace referencia a la precondición que se debe cumplir para llevar a cabo la ***Historia de Usuario,*** en este caso la cuenta tiene crédito, la tarjeta es válida y el cajero tiene dinero disponible. Si no se cumple con estas pre-condiciones, no es posible llevar a cabo la ***Historia de Usuario.***
- ***Cuando:*** El "cuando" representa el tiempo y contexto del momento en que se realiza la acción. En este caso el cliente está pidiendo el dinero en el cajero.
- ***Entonces:*** El "entonces" representa la acción o las acciones que el software debe realizar a partir del contexto y las pre-condiciones establecidas. En este caso se debe debitar la cuenta, entregar el dinero y el cliente debe poder recuperar su tarjeta.

Definition of Done DoD

Conceptualmente **Definition of Done** es una lista de chequeo, dependiendo del tipo de trabajo que el equipo de desarrollo espera completar satisfactoriamente, antes de declararlo como libre.

A continuación listamos la lista de chequeo para los proyectos que Holística Organizacional, dentro de su core de negocio desarrolla a la fecha (Java, BPM Ophelia, BPM MetaStorm, PL/SQL), ya que esto puede evolucionar en el futuro:

Definition of Done		
1	Diseño revisado	
2	Código completado	1. Código refactorizado 2. Código en el formato estándar 3. Código comentado 4. Código inspeccionado 5. Código verificado
3	Documentación de usuario final actualizado	
4	Pruebas	1. Pruebas unitarias 2. Pruebas de integración 3. Pruebas funcionales
5	Cero defectos conocidos	
6	Pruebas aceptada	
7	Código versionado	

Definition of Ready

La Definición de Listo (DoR), está en manos del ***Product Owner***. Es él quien determina que la ***Historia De Usuario*** está lista para ser liberada del ***Product Backlog*** y ser incluida el ***Sprint Backlog.***

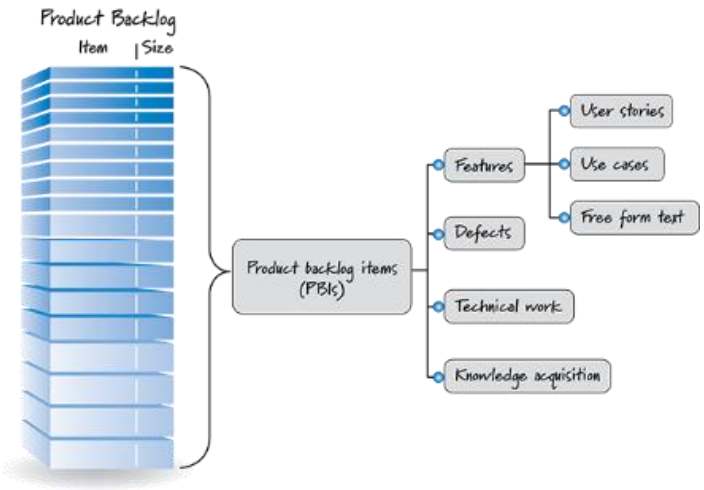
Los factores a considerar dentro de ***Holística Organizacional,*** para determinar que una ***Historia de Usuario*** está lista es:

- La tarea debe tener una estimación de complejidad (puntos de historia).
- El detalle funcional, detalle técnico y los casos de prueba de cada tarea deben ser claros para que sean entendibles por cualquier miembro del equipo.
- La tarea no debe tener bloqueos que impidan su ejecución.
- Las dependencias deben estar resueltas.
- La tarea debe poder validarse y verificarse dentro del Sprint.

Product Backlog

El ***Product Backlog,*** es una lista de funcionalidades deseadas y priorizadas para la construcción de un producto o servicio. Este, provee un entendimiento compartido y centralizado de lo que se va a construir y el orden en el que se debe construir. Es un artefacto altamente visible y es el corazón del framework de Scrum y es accesible por todos los participantes del proyecto.

El ***Product Backlog,*** está compuesto por ***Backlog Items*** (PBIs). La mayoría de PBIs son características, ítems de la funcionalidad que tendrán un valor tangible al usuario o al cliente.



Tal como lo muestra la imagen anterior, el PBIs, está compuesto de las características, que en el caso de Holística Organizacional, esto es la representación de Historias de Usuario. A su vez tenemos: defectos del software, trabajo técnico y la adquisición de conocimiento.

Tal como lo habíamos mencionados anteriormente, el ***Product Backlog***, es una lista priorizada de requerimientos, donde los que se encuentran en la parte superior de la pila, serán los más importantes y los que se encuentren en la base serán los de menos prioridad.

Sprint y Sprint Backlog

Un ***Sprint***, es una medida de tiempo que puede ir de 1 a 4 semanas. Dentro de este tiempo es lo que toma una iteración del ciclo de ***Scrum*** en completarse y en generar una pieza del producto de Software que estamos buscando construir.

Dentro del tiempo de un Sprint, se define el ***Sprint Backlog,*** el cual consiste en halar de la parte superior del ***Product Backlog*** las ***Historias de Usuario*** con mayor prioridad e iniciar la iteración para el desarrollo del producto o servicio.

Una vez finalizado el sprint, se procede a tomar nuevamente de la parte superior del ***Product Backlog*** las historias que harán parte del nuevo ***Sprint Backlog.***

Para lograr definir, cuáles y cuantas historias puedo incluir dentro de un ***Sprint Backlog***, es necesario poder medir y estimar tanto la complejidad como el esfuerzo de cada Historia de Usuario. Para ello podemos contemplar dos alternativas:

- Tallas de Camisa: XS, S, M, L, XL.
- Serie Fibonacci: 1, 2, 3, 5, 8, 13.

En ambos escenarios el objetivo es poder determinar el peso de cada historia, teniendo en cuenta que el máximo esfuerzo a considerar no puede ser mayor a 5 días, ya que una historia superior a 5 días (Talla XL o un 13 en la serie Fibonacci) es una ***Historia de Usuario*** que se recomienda partir en más pequeñas.

Roles SCRUM dentro de Holística Organizacional SAS

Los Roles planteados por SCRUM son los siguientes:

- ***Product Owner:*** El ***Product Owner*** es el líder del proceso. Es la autoridad responsable para decidir cuales características y funcionalidades deben ser construidas y en qué orden debe hacerse. El ***Product Owner*** mantiene y comunica a todo el resto de participantes una visión clara de que es lo que el equipo de SCRUM está tratando de lograr. Es el responsable por el éxito general de la solución y a su vez es el responsable del Backlog. Este rol es cumplido por el "que sabe" del negocio por el lado del cliente. Por tal razón debe ser exigido más no puesto por Holística Organizacional.
- ***Team Member:*** Por definición los equipos Scrum son equipos Cross – Functional y están integrados por 7 máximo 9 personas. En caso de que el equipo requerido sea superior a este número se debe considerar el equipo por células de trabajo con responsabilidades definidas.
- ***Scrum Master:*** El ***Scrum Master*** es un rol de facilitador más no de JEFE. Sus responsabilidades son hacer que el equipo alcance el nivel de madurez de SCRUM, no hacer que el equipo se desenfoque, no permitir que el equipo se contamine con problemas del exterior y finalmente ser la persona que resuelve los impedimentos del equipo.

Por experiencia sabemos que los roles de Scrum se quedan cortos para la implementación de proyectos de alta complejidad, tal como los que maneja Holística, por lo cual, los roles propuestos para un equipo base serán los siguientes:

1. Arquitecto De Software.
2. Líder técnico que haga las veces de Scrum Master.
3. Un equipo de Desarrollo conformado de la siguiente manera con los siguientes roles:
    a. Ingeniero de infraestructura.
    b. Ingeniero de procesos.
    c. Diseñador gráfico.
    d. Equipo de desarrollo (Dependerá de la tecnología).
    e. Ingeniero de pruebas.

Talento Humano

Definición de la tecnología y tipo de proyecto

Selección de Humano: Roles, habilidades y responsabilidades

Pricing

Diseño de la Arquitectura

Definición de la Arquitectura
Patrones de Arquitectura
Documento de Arquitectura Vistas 4+1
Documento de decisiones de Arquitectura
Documento de Integraciones

Plan de pruebas

Definición del plan de pruebas
Pruebas funcionales
Pruebas Integrales
Pruebas de carga
Pruebas de estrés
Pruebas de seguridad
Control y seguimiento de Incidentes

Gestión de la Configuración

Tipo de proyecto
Definición de los artefactos por tipo de proyecto
Gobierno de la Configuración
Roles
Definición de promoción de software entre ambientes
Definición de Branches

Notas:

- Especificar la construcción del software
- Pruebas unitarias

- Estandarización de la codificación
- Trazabilidad entre la arquitectura y las pruebas
- Trazabilidad entre la arquitectura con las HU
- Trazabilidad entre la arquitectura y el desarrollo
- Metodología para el diagnóstico