UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FISICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

APOYO A LA ADOPCIÓN DE GESTIÓN ÁGIL A TRAVÉS DE LA HERRAMIENTA PAINLESS TRACKING

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

ROBERTO ANDRÉS CARRASCO CARRASCO

PROFESOR GUÍA:
AGUSTÍN ANTONIO VILLENA MOYA

MIEMBROS DE LA COMISIÓN:
MARÍA CECILIA BASTARRICA PIÑEYRO
EDUARDO SALVADOR GODOY VEGA

SANTIAGO DE CHILE
OCTUBRE, 2008

RESUMEN DE LA MEMORIA
PARA OPTAR AL TITULO DE
INGENIERO CIVIL EN COMPUTACIÓN
POR: ROBERTO CARRASCO C.
FECHA: 27/10/2008
PROF. guía: Sr. AGUSTÍN VILLENA MOYA

"APOYO A LA ADOPCIÓN DE GESTIÓN ÁGIL A TRAVÉS DE LA HERRAMIENTA PAINLESS TRACKING"

Hoy en día los beneficios de las metodologías ágiles son tan evidentes que ha crecido el interés de muchas empresas lograr adoptar alguna. Pero la adopción de una metodología ágil es muy compleja y riesgosa para las empresas, esto se debe a los principios algo radicales en que este tipo de metodologías se sustenta. Por esto la comunidad ágil ha volcado sus esfuerzos en como hacer esto posible sin perder los principios que la sustentan.

Así es como aparecen modelos para facilitar la adopción de agilidad, que permiten que este proceso sea más controlado y efectivo. Cursos como CC61A y CC62V, de nuestro departamento, son ejemplos de como lograr con éxito la adopción, en especial, de la "gestión ágil" que es, en mayor parte, lo que hace diferente a estas metodologías, y es lo más complejo de adoptar.

Para apoyar la gestión ágil en los cursos es creada la herramienta Painless Tracking, que ayuda a los alumnos en el día a día de su gestión durante el proyecto que enfrentan. Pero, al ser introducida en una empresa real la herramienta no es de real ayuda, ya que los desafíos son distintos a los de los cursos, también son diferentes los conocimientos sobre gestión ágil.

Por ello, se elaboró un modelo de adopción de gestión ágil que permite guiar el proceso y medir el nivel de adopción de un equipo de desarrollo, y se construyó una nueva versión de la herramienta Painless Tracking que apoya el modelo propuesto, facilita el aprendizaje e incrementa el número de prácticas ágiles en el equipo. Con esto la herramienta avanza y se convierte en una plataforma de gestión y apoyo a la adopción de metodologías ágiles.

AGRADECIMIENTOS

A profesor Agustín Villena que mantuvo su apoyo y preocupación durante todo el desarrollo del trabajo, y siempre su prioridad fue que aprendiera, porque esto sería de utilidad para mi futuro.

A mis amigos Ignacio Ortega y Marijn Vriens, que siempre dieron tiempo para discutir conmigo sobre como era mejor enseñar agilidad, sin estos consejos, el resultado no hubiese sido una herramienta más simple.

A los que me aman, mis padres, mi hermano, mis amigos, mi pastor y mi iglesia, que me han acompañado y apoyado durante mi educación.

A mi esposa Cecilia que con su amor elevó mis ánimos hasta hacer posible terminar.

A mi Dios y Señor, Jesús que nunca me deja y siempre hace cosas increibles conmigo.

Pero dedico el trabajo al amor y el esfuerzo de mis padres, Roberto y Ana, que espero honrar no sólo hoy sino también en el futuro.

Índice

Indice de Figuras

Indice de Tablas

1. INTRODUCCIÓN

Este trabajo se refiere a las nuevas funcionalidades de la herramienta Painless Tracking en su búsqueda por ser ayuda en la adopción de gestión ágil. Esta es una herramienta que permite, de manera indolora, gestionar el desarrollo de un producto de software, y este trabajo cuenta sobre las mejoras agregadas a ella para permitir una adecuada adopción de los principios y prácticas de gestión ágil que ella contiene, y que se espera que un equipo de desarrollo logre manejar, y así el equipo también pueda tener un camino de adopción más claro. Este capítulo entrega un acercamiento a la problemática de la adopción de metodologías ágiles y como se enfrenta a través del trabajo realizado.

1.1. El Interés en Adoptar una Metodología Ágil

La industria del software ha mostrado evidencias de que el cliente no obtiene los resultados deseados de los desarrollos que compra. La mayor parte de los proyectos de desarrollo de software tienen problemas de sobrecosto y sobretiempo, y aunque el cliente pague o espere para así tener un software completo, la funcionalidad que finalmente recibe no es tan útil [2].

En respuesta a estos problemas, desde finales de los ochenta comienzan a aparecer metodologías de desarrollo [3][4][5][6][7][8][9] que, yendo contra las corrientes tradicionales de desarrollo de software se proponen como alternativa a la forma actual de enfrentar el desarrollo de un proyecto de software, estas se autodenominaron, metodologías ágiles. Entre las más destacadas suenan los nombres de las metodologías Extreme Programming (XP) y Scrum.

Desde el año 2001 que estas metodologías comenzaron a llamarse “ágiles” y dan a conocer sus principios a través del “Manifiesto Ágil” [11]. Así los cultores de este movimiento inician de manera organizada un camino para dar a conocer a la industria del software sus ideas sobre cómo el desarrollo del software debe ser enfrentado.

Estos métodos ágiles nacen de la experiencia de profesionales del área en proyectos reales, quienes han tenido que lidiar con los desafíos y limitaciones de los métodos tradicionales proyecto tras proyecto. El desarrollo ágil a través de reglas livianas facilita la mejora del funcionamiento y la rápida absorción de tecnología (siempre útil en el área) manteniendo el foco en la rápida entrega de valor de negocio, con esto es posible reducir significativamente el riesgo

aún cuando las metodologías permiten que el cliente decida en todo momento qué hacer (cambiando el rumbo si así lo quiere). Esto es posible gracias a que los equipos mantienen alineada la entrega de software con las necesidades de negocio del cliente, y así siempre puedan fácilmente adaptarse a los cambios en los requerimientos.

Los beneficios de la agilidad (por ejemplo, la entrega temprana de valor) sólo son posibles a través de un proceso que permita **"la continua planificación y entrega de información"** ("planning and feedback"), con esto es posible asegurar el compromiso de los clientes y maximizar la entrega de valor durante todo el proceso de desarrollo. Por esto es claro que la forma primaria en que miden el estado de un proyecto es simplemente a través del **"trabajo efectivamente realizado"**, esto les entrega información sobre el pasado y presente del proyecto y en base a estos resultados enfrentar el futuro del proyecto.

En definitiva lo que se busca a través de la agilidad es un software que sí esté en la dirección a las necesidades del negocio y las del cliente. El siguiente diagrama muestra en resumen las ventajas más apreciadas de un desarrollo ágil en contraposición con un desarrollo tradicional.

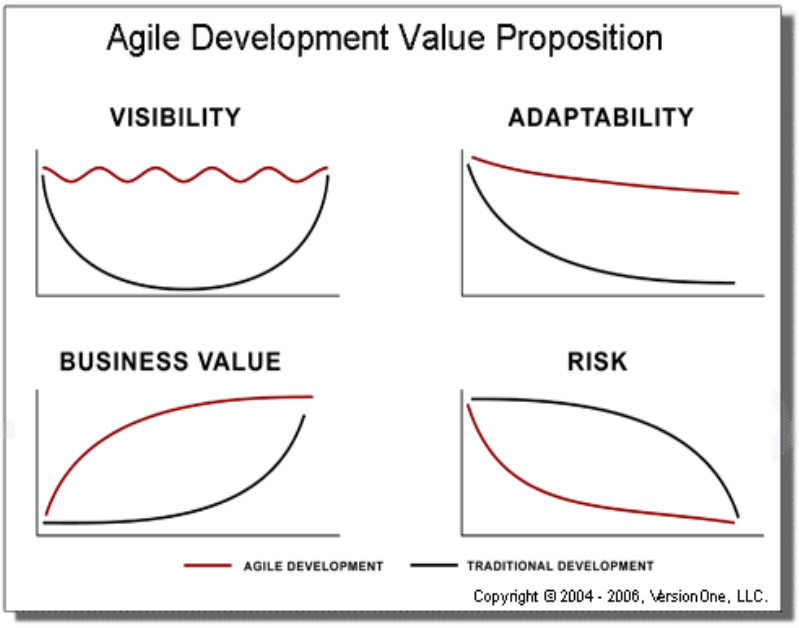


Figura 1: Beneficios Propuestos por el Desarrollo Ágil

1.1.1. Los beneficios reales de la mirada ágil

Después de presentadas todas estas nuevas ideas, algunas empresas y equipos comenzaron a modificar su forma de organización para introducir alguna de estas metodologías y así alcanzar estos beneficios. Los años han pasado y la evidencia empírica nos sorprende con los casos exitosos que han sido desarrollados ágilmente [12][13][14][15][16][17][18], exhibiendo los beneficios que pueden ser apreciados al abrazar prácticas ágiles en distintos frentes como:

- **Retorno de la inversión:** es mucho más temprana, por lo que el riesgo se disminuye pronto,
- **Puesta del producto en el mercado:** ocurre en un tiempo menor y es posible obtener mucho antes utilidades de los productos en distintas fases de su desarrollo,
- **Calidad del producto:** aumenta, tanto en lo técnico como en la percepción del usuario, por lo que el producto es más valioso. El software cada vez es menos defectuoso y existe menos código gris implementado.
- **Relación entre Cliente y Desarrolladores:** es mucho más cercana y de confianza, además, los equipos de desarrollo son vistos mucho más alineados al negocio y por esto responden mucho mejor a las necesidades de los clientes.
- **Moral del Equipo:** los desarrolladores se sienten mucho más satisfechos con el trabajo que realizan y en como se pueden potenciar con el resto del equipo, ya que cada uno siente que cumple con un rol importante, por lo que es común ver en los equipos una actitud de proactividad.

Todo ello ha llevado a las empresas a mirar a las metodologías ágiles como algo necesario de introducir, es tanto así que ya no se preguntan si es bueno o no adoptar una metodología ágil, más bien se están preguntando cómo hacerlo.

En el siguiente cuadro responde el por qué una organización debiese adoptar prácticas ágiles, en la práctica cómo la agilidad lo enfrenta y dónde impacta el beneficio:

Beneficio	En la práctica	Impacto
Fácil comunicación y Feedback	Iteraciones cortas, programación de pares y espacio de trabajo informativo.	Interno
Visibilidad del Proyecto (Estado)	Reuniones de diarias, información de avance y estado	Interno
Calidad del código	TDD, iteraciones cortas	Interno
Los requerimientos pueden cambiar	Plan que se adapta, refactorización	Interno
Desarrollador Satisfecho	Programación de Pares, compartir el conocimiento	Interno
Cliente Satisfecho	Entregas continuas y rápidas	Externo
Administración Satisfecha	Reuniones diarias, información de avance y estado, tracking	Externo
Mejor Colaboración del Cliente	Iteraciones cortas	Externo

Tabla 1: Beneficios de la Agilidad

1.1.2. Problemas de la adopción de una metodología ágil.

Muchos de los problemas que actualmente se observan en los equipos que están adoptando alguna metodología ágil, ya Ken Beck analizaba anticipadamente en su libro [3], donde explica Extreme Programming. En la experiencia del autor, su metodología es simple en el detalle, ya que tiene conceptos simples y prácticas que son rápidas de entender y que son lógicas, pero ejecutarla es algo complejo. Plantea varios desafíos que las organizaciones deben enfrentar. La problemática de la adopción que expresa puede ser resumida bajo las siguientes premisas:

- La forma de medir el valor cambia, por lo tanto, es necesario modificar la forma en que este lo gestionamos.
- La capacidad para aprender que las personas tienen es lo que limita la velocidad al adoptar.
- La colaboración de las personas no es fácil de obtener pero necesaria, y no sólo involucra al equipo de desarrollo sino que a toda la organización.

En apoyo a esto, la problemática de la adopción, el profesor Villena[2] explica y compara en extenso los desafíos de la adopción con la actual forma de realizar ingeniería de software; es más, estos desafíos los clasifica en 2: extrínsecos e intrínsecos. Los primeros explican la problemática desde el punto de vista de cómo la agilidad se presenta como una alternativa y revisión a las metodologías de software tradicionales, este tipo de desafíos ha sido presentado anteriormente y puede ser resumido con la Figura 1. Los desafíos intrínsecos son los que nacen de la forma en que está construida una metodología ágil particular, es decir, sus principios y prácticas.

Todo esto permite deducir que la problemática de la adopción nace de tres puntos que todo

equipo tendrá que enfrentar:

- La nueva forma de ver el valor.
- La revisión que hace de las metodologías tradicionales y sus propuestas para mejorar los problemas de estas últimas.
- La importancia de las personas por sobre los procesos durante el desarrollo de software [19]. ya que los cambios se deben realizar en las personas.

1.1.3. La respuesta del Movimiento Ágil.

Como adoptar una metodología ágil no es un camino simple, el movimiento ágil ha comenzado a concentrarse en buscar formas que permitan a diferentes organizaciones adoptar metodologías ágiles.

Ahmed Sidky y James Arthur, siguiendo esta línea, desarrollaron un método para adoptar prácticas ágiles en una organización: el "Agile Adoption Framework" [1]. Para este se definen pasos a seguir y métricas que permiten ver el grado de asimilación de los principios y prácticas en la organización. De esta investigación se puede demostrar que el proceso de adopción no es un proceso simple debido a que las metodologías ágiles por su fuerte crítica a los métodos tradicionales implica en algunos casos un cambio de mentalidad total que necesita de modelo que permita de manera menos riesgosa guiar la adopción y así hacerla exitosa.

También nuestro Departamento de Ciencias de la Computación de la Universidad de Chile se ha interesado en el tema de la adopción de metodologías ágiles y como un paso concreto es que desde el año 2005, el curso "CC61A: Proyecto de Software" cambió su enfoque de desarrollo tradicional por uno basado en metodologías ágiles donde utilizan algunas de las prácticas de la metodología Extreme Programming [21]. Este cambio logró mejorar notoriamente el resultado final de los desarrollos entregados a clientes y las satisfacción de estos últimos aumento también, tanto así que un 94,7% de los productos desarrollados entran en producción. Sin duda el nuevo enfoque favorece el desarrollo al estar más cercano al cliente. Este nuevo enfoque está fuertemente inspirado en la experiencia obtenida del curso "CC62V: Taller de metodologías ágiles de desarrollo de software" que se imparte desde el año 2002, el cual nos presenta en forma más detallada los aprendizajes que pueden ser obtenidos durante la enseñanza de una metodología ágil, estas experiencias serán vistas en más detalle adelante.

1.2. Motivación

Durante el desarrollo de estos cursos se ha utilizado e introducido una herramienta para apoyar la gestión ágil, esta herramienta implementada en Excel permite a los alumnos junto con el cliente definir cuáles serán los alcances del proyecto(el conjunto de funcionalidades de este) las que son divididas en iteraciones. Además de detallar el plan como antes se explicó, los alumnos durante el desarrollo pueden hacer tracking del avance en las tareas que ejecutan. Con todo esto se pueden ver informes de avance que permiten al equipo revisar sus rendimientos, y al cliente ver el avance en las funcionalidades elegidas por él. Todo esto también permite que el cliente pueda en todo momento repriorizar, agregar nuevas funcionalidades, modificar el plan (por ejemplo, borrando algo que no quiere), y otras características que permiten a los equipos realizar la gestión de sus proyectos de una forma simple y barata en tiempo de utilización pero aún así obtener un alto valor con la información que entrega.

Aunque hoy existen varias herramientas (entre las más destacadas están "Version One" [26] y "Mingle" [27]que permiten apoyar la gestión ágil, como la Painless Tracking, su enfoque no está en si quienes la utilizan conocen y aplican correctamente los principios y prácticas ágiles que rigen su diseño. Tanto es así que se tiende a pensar que la sola utilización de alguna de estas herramientas te permite o permitirá ser ágil. Esto está muy lejos de ser real, ya que el solo uso de una herramienta no asegura ni dice nada de la agilidad de un grupo. En el caso de la Painless Tracking que ha sido utilizada en varios cursos y ha sido enseñada a los alumnos de los cursos antes nombrados no da claridad de qué tan bien los alumnos han logrado aprender su correcto uso, esto de manera empírica se puede mostrar en que la entrega del valor al cliente siempre se esta produciendo cuando se está más cerca del termino del curso, esto es un indicio claro de que no se ha entendido claramente el significado de la gestión ágil y como la Painless Tracking lo apoya.

Así es como la razón de este trabajo de título es apoyar el proceso de adopción de gestión ágil para casos internos y externos, esto es cursos como CC61A y CC62V, y empresas de desarrollo de software, a través de la incorporación de nuevas funcionalidades y métricas a la Painless Tracking que permitan dar visibilidad del aprendizaje adquirido por todos los miembros de un equipo de desarrollo de todos los principios y prácticas ágiles que contiene la herramienta, y así convertir a la herramienta Painless Tracking en un verdadero complemento para el apoyo de adopción de metodologías ágiles.

1.3. Objetivos

1.3.1. Objetivo General

Convertir la herramienta Painless Tracking en una herramienta que también permita potenciar la adopción de los principios y prácticas de Gestión Ágil sobre los que ha sido diseñada para que esta sea incorporada a la realidad de una empresa de desarrollo de software. Para esto se debe añadir nuevas funcionalidades a la herramienta y definir métricas y reportes que permitan conocer el nivel de adopción de los desarrolladores que la utilizan.

1.3.2. Objetivos Específicos

- Agregar nuevas funcionalidades a la herramienta Painless Tracking que permitan facilitar el aprendizaje de los principios y prácticas de Gestión Ágil contenidos en ella.
- Definir y desarrollar métricas y reportes que permitan conocer cuanto un desarrollador entiende y aplica los principios de Gestión Ágil reflejados en la planilla Painless Tracking usando la información que en la herramienta se registra.
- Diseñar herramientas que permitan potenciar la adopción de los principios y prácticas de Gestión Ágil en que se basa la Painless Tracking como: tutoriales, capacitaciones, interfaces wizards [R.10], entre otros.

1.4. Metodología Utilizada

El desarrollo del trabajo fue realizado considerando los siguientes pasos:

1. **Marco Teórico**

   Revisar trabajos publicados y memorias anteriores con temas relacionados. Se seleccionarán temas afines y se evaluará la reutilización de conceptos, diseños e implementaciones presentes en estos trabajos encontrados. En especial se revisarán los principios y prácticas ágiles desarrollados en la Painless Tracking. De esta investigación se diseñará un modelo que permita evaluar la agilidad de los equipos de desarrollo.

2. **Análisis y Evaluación de la herramienta Painless Tracking Excel**

   La herramienta basada en Excel será puesta a prueba en el contexto de una empresa real

de software. Con ello se determinarán cuáles son los problemas reales de la herramienta que impiden su introducción en la industria del software.

3. **Evaluación de la Agilidad Inicial de un Equipo**

   Con el modelo de evaluación será evaluado un equipo real y así involucrarlos directamente en el proceso de adopción de gestión ágil mostrando la realidad de su equipo.

4. **Determinación de expectativas del equipo y cómo pueden ser incorporadas a la herramientas**

   Como resultado de la evaluación se podrá determinar la agilidad que el equipo desea obtener, con ese camino definido se determinarán las funcionalidades que deben ser añadidas a la herramienta para apoyar la adopción.

5. **Implementación iterativa de las nuevas funcionalidades**

6. **Evaluación de la adopción de los principios y prácticas ágiles contenidos en la Painless Tracking a través de las métricas diseñadas y medir como las nuevas herramientas y funcionalidades favorecen la adopción en proyectos reales.**

2. ANTECEDENTES

La investigación realizada permite dar cuenta de los desafíos que existen al adoptar una metodología ágil. Esta sección se refiere a esto y a cómo ha sido enfrentado tanto en el mundo y en el departamento de Ciencias de la Computación de la Universidad de Chile.

2.1. Gestión ágil

La gestión ágil [9][23] es de los temas más apreciados y el motor de toda la maquinaria ágil. Este nace de la experiencia de varios desarrolladores de software que por lo sujeta a los procesos que es una metodología tradicional salieron en respuesta a esto con nuevas formas de gestionar el cliente, los requerimientos, la planificación y la producción durante un proyecto de software.

En todo proyecto de software se pueden identificar las siguientes variables: Tiempo, Recursos, Alcance y Calidad, donde Tiempo corresponde al plazo de entrega o que se espera terminado, Recursos que son personas, dinero, etc., Alcance que define el conjunto de funcionalidades que deben ser desarrolladas y Calidad con la que se describen que tan eficaz debe ser el software ante diferentes escenarios, restricciones y usos. Pero, en la práctica, las variables tiempo, recursos y alcance generalmente son constantes, las variables tiempo y recursos son estimadas y el alcance es fijo, por lo que el impacto sobre la calidad del software es algo notorio y al sacrificar esto es la única forma de lograr términos de proyecto. Por lo que el conflicto entre desarrollador y cliente no es extraño, ya que uno defiende la calidad del producto y el otro un producto más valioso.

A esta forma de gestionar el desarrollo basado en que dado que tengo que lograr el alcance definido con las restricciones estimadas tiempo y recursos, a esto se le llama “Gestión Guiado por el Plan”, esto es posible dado que el alcance es algo fijo que en teoría no va cambiará.

Pero la gestión ágil se basa en el modelo de “Gestión Guiada por el Valor”, donde se fijan los recursos y tiempo, y se estima el alcance del proyecto permitiendo así no sacrificar la calidad y cambiar la forma en que se mide el éxito en base a la cantidad de funcionalidad entregada realmente al cliente y la rapidez con que esta se entrega.

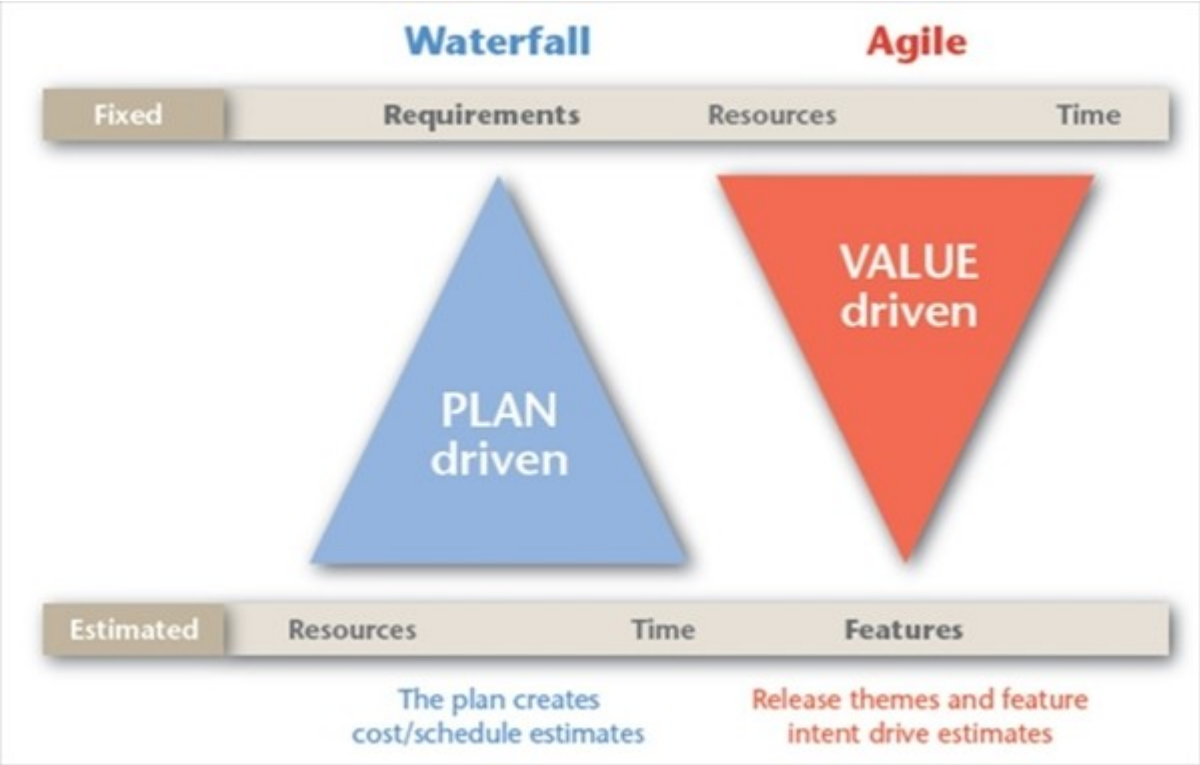


Figura 2: Planificación Guiada por el Valor

En esta mirada, el cambio de funcionalidades durante el desarrollo es permitido, aquí el riesgo es algo que debe gestionarse de manera adecuada para disminuir pronto la incertidumbre (esto es algo que el modelo tradicional también existe, pero esta se busca eliminar con todo el proceso detallado de toma de requerimientos y diseño). La siguiente figura muestra la diferencia que existe cuando los requerimientos se tienen claros desde el comienzo versus cuando estos no lo están y como estos debiesen gestionarse.

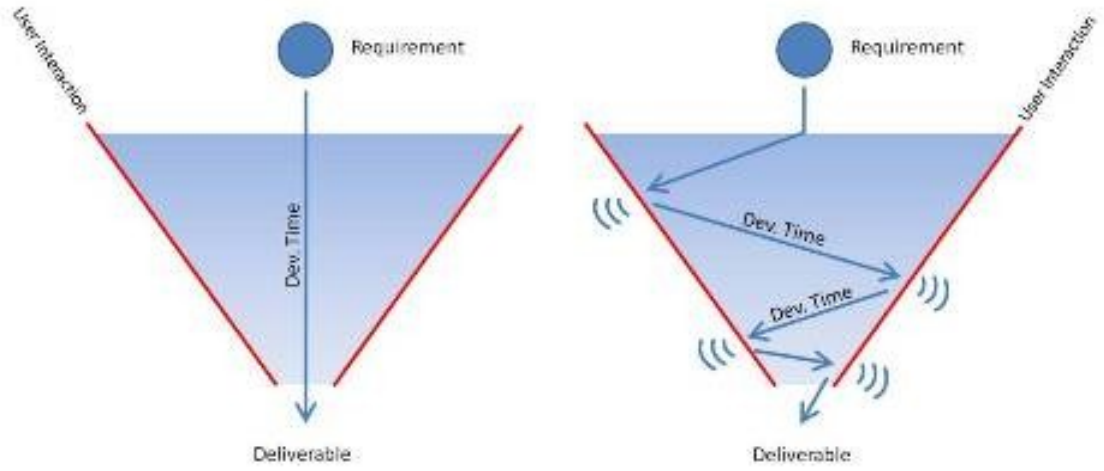


Figura 3: El Cambio y su Gestión

Esta figura muestra los elementos principales de la gestión ágil:

- La interacción continua con el cliente y usuario(las líneas diagonales muestran esta interacción)
- Los tiempo de desarrollo y entrega deben ser más cortos(cada rebote es una entrega o tiempo de desarrollo) y también más **rápidos.**
- La vista siempre debe estar puesta en la entrega (la salida de la bola es el entregable)
- Los requerimientos se definen y redefinen en cualquier momento (la bola no cae recta sino que puede rebotar).
- El tiempo de desarrollo es fijo (el sector azul es el tiempo de desarrollo).

En términos ágiles, sólo se habla de 3 elementos [24] Iteraciones, Historias de Usuario y Velocidad de desarrollo (en lo siguiente, IUV, iterations, user stories and velocity). Los tres elementos de la IUV son la forma de enfrentar los proyectos de manera ágil. Estas funcionan de la siguiente manera, el cliente define funcionalidades según el valor de negocio las que se llaman historias de usuario, el equipo de desarrollo enfrenta la producción de estas a través de la entrega continua de funcionalidades en tiempos cortos y fijos, a los que llaman iteraciones y así estos últimos sólo se concentran en lograr y entender un conjunto reducido de funcionalidades, así el cambio de una funcionalidad por otra o la redefinición no afecta tanto al desarrollo, como el tiempo es fijo el equipo puede poner su mirada también en cuan rápido logran desarrollar funcionalidades y que estas sean entregadas efectivamente al cliente, buscando que cada vez la

entrega sea más rápida que la anterior, dado que el valor esta en entregar funcionalidades. La velocidad les permite comparar rendimientos y en lo siguiente mejorarlos.

Pero la IUV es incompleta, ya que lo verdaderamente valioso para el cliente son los productos de software que forman un conjunto de funcionalidades, lo que es llamado "Entregable", la mirada del equipo debe estar puesta en el producto de software que se va a entregar y en cuantas funcionalidades de este se logran. Esto agrega a la visión estratégica de la gestión ágil (donde se busca reducir la incertidumbre a través de las entregas rápidas en tiempo fijo) la visión productiva donde el valor esta puesto en cuanto y que tan rápido se está logrando la entrega efectiva de funcionalidad.

Ken Beck aclara esto último diciendo: "Son más valiosas la entregas frecuentes con menos funcionalidad, que las entregas menos frecuentes de más funcionalidad" [24].

La existencia de Entregables e Iteraciones se basa en [9]:

1. El cliente no se anticipe o aplace la toma decisiones desinformadas (por eso elige un conjunto de funcionalidades).
2. Validar más temprano que es lo valioso para el cliente y así este pueda cambiar el rumbo si es necesario.

Tomando todo esto la gestión ágil puede ser finalmente resumida como un conjunto de reglas para el cliente y para el desarrollador [2]:

	Cliente	Desarrollador
Desea maximizar	**Valor recibido** por cada semana de desarrollo	**Calidad del trabajo** realizado
Puede definir	**Qué será implementado** , y en **qué prioridad** , según las **necesidades de su negocio**	**Cuánto** se **estima que demorará** una tarea (idealmente)
Puede cambiar	**Funcionalidades solicitadas** por otras no implementadas de costo equivalente (canjear)	Sus **estimaciones** en base a nuevos descubrimientos

Figura 4: Derechos y deberes de clientes y desarrolladores para gestión ágil[2][28]

En resumen, la gestión ágil se basa en la gestionar el alcance de un proyecto, esta estrategia utiliza las entregas rápidas y pequeñas como un medio para disminuir la incertidumbre y aumentar en cada paso el valor que el cliente percibe, sin sacrificar la calidad del producto desarrollado y siempre haciendo un uso más eficaz de los recursos del cliente. Para esto se definen funcionalidades como historias de usuario, las que son agrupadas en Entregables que tienen un plazo fijo de entrega. Además, está preocupada de la producción de software del equipo que es gestionada a través de la definición de iteraciones que permiten al equipo de manera temprana evaluar su conocimiento sobre el problema de negocio que enfrentan y recibir prontamente retroalimentación del cliente de la correctitud de sus avances, y también midiendo la velocidad con que ellos entregan software, para así minimizar el uso de recurso y maximizando el valor entregado al cliente.

2.2. "Agile Adoption Framework": Un modelo de Adopción de Metodologías Ágiles

Recién a mediados del año 2007, Ahmed Sidky, desarrolló un modelo para guiar la adopción de prácticas ágiles en una organización, este modelo ha sido llamado "Agile Adoption Framework" [19], este es el primer modelo de adopción ágil que se conoce y aparece como respuesta precisamente de la falta de estos. Del modelo se destacan las siguientes cualidades:

- una evaluación que permite discriminar entre organizaciones preparadas para la adopción y las que no,
- un framework para guiar la adopción según el potencial propio de la organización,
- un modelo para medir y clasificar en niveles la agilidad.

Una las frases más importantes de Sidky y que pueden indicar en resumen la dirección de su trabajo es:

> *"las organizaciones se vuelven más ágiles cuando adoptan más prácticas ágiles, por lo que medir el nivel de agilidad de una organización esta directamente relacionado con las prácticas que han logrado adoptar" [R.19]*

Por esto, hacer más ágil a una organización implica enseñar más prácticas ágiles a ella.

Además, este modelo puede servir de complemento a organizaciones que además de ser ágiles desean obtener una certificación CMM-I, ya que hasta el momento, ya que esta propuesta permite hacer de la adopción de agilidad y de la agilidad misma un proceso medible y controlable, ya que para estar en línea con CMM-I necesita de estas cualidades. Esto es porque CMM-I ha sido diseñada para medir la madurez de los procesos y sus capacidades, pero no la agilidad [19].

Esta guía ha sido reconocida por la comunidad ágil como una buena forma de incorporar a una organización al ambiente ágil, aunque reconocen que la adopción de los principios y prácticas ágiles sigue siendo una pregunta difícil de responder y que su modelo es sólo un primer acercamiento para responderla.

La propuesta completa de Sidky se compone de dos elementos:

- SAMI: es la forma en que se mide la agilidad y el potencial de agilidad de una organización, independiente de la organización y la metodología ágil deseada por adoptar. Además propone niveles (similares a los de CMM-I) de agilidad que permiten a las organizaciones conocer su grado de madurez o manejo de principios y prácticas ágiles.
- 4-Stage process: esta diseñado para asegurar que un proceso de adopción es guiado de manera efectiva. Se compone de 4 pasos, los que son: Identification of Discontinuing Factors, Project Level Assessment, Organizational Readiness Assessment y Reconcilliation.

2.2.1. SAMI: un modelo para medir agilidad

SAMI es importante para este trabajo por como permite determinar el nivel de agilidad de las organizaciones, este posee cuatro componentes que son:

- Niveles de agilidad: Colaborativo, Evolucionario, Efectivo, Adaptativo y Englobador. Para cada nivel define un objetivo que se alcanza al tener ese nivel así es posible entender el impacto que tendría la adopción en el nivel por alcanzar (ver tabla 2).
- Principios Agiles: estos son Abrazar el cambio para Entregar al cliente valor, Planificación y entrega de software frecuentemente, Humano-céntrico, Excelencia Técnica y Colaboración del cliente. Estos principios son las características que debería tener una organización para ser llamada ágil. Estas se basan en el manifiesto ágil [11].
- Conceptos y Prácticas Ágiles: las actividades concretas que deben existir para que sean consistentes con los principios ágiles. Esto para asegurar la agilidad.
- Indicadores: son las preguntas que el examinador utiliza para determinar el nivel de agilidad de una organización después de una evaluación.

Niveles de Agilidad	Nombre	Objetivo
Nivel 1	Colaborativo	Facilitando la comunicación y colaboración
Nivel 2	Evolucionario	Entrega de software temprana y continua
Nivel 3	Efectivo	Produciendo software de calidad
Nivel 4	Adaptativo	Respondiendo al cambio a través de diferentes niveles de feedback
Nivel 5	Englobador(Ambientado)	Estableciendo un ambiente que permita mantener la agilidad

Tabla 2: Niveles de Agilidad

La Figura 5 muestra como reúne estos 4 componentes para construir la matriz que permite a SAMI clasificar e identificar el nivel de agilidad de la organización.

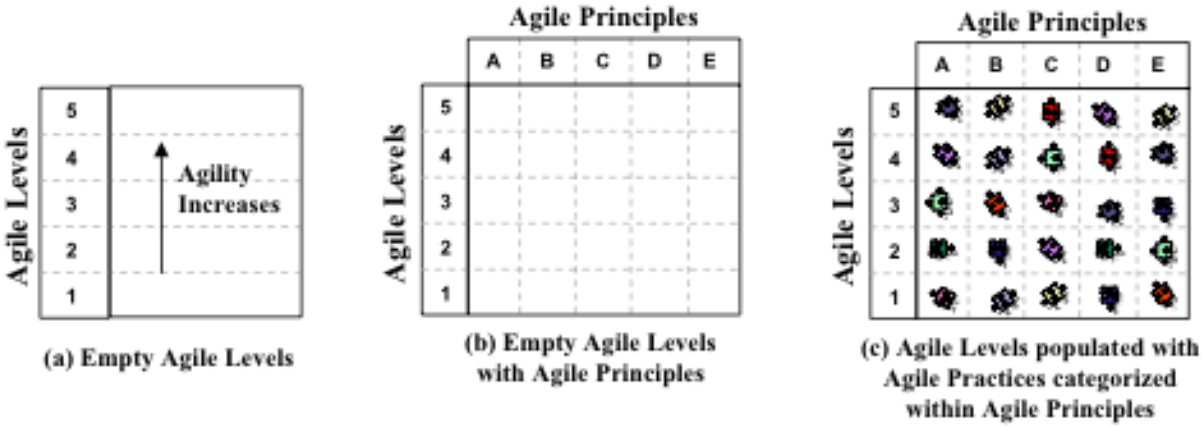


Figura 5: Partes de SAMI

La matriz que entrega SAMI nos muestra que la forma de aumentar la agilidad es agregando más prácticas al equipo, con esto en mente e identificados el potencial de la organización es posible determinar un camino de adopción basado en que prácticas están ausentes o débiles y cuales pueden ser adquiridas, también nos permite determinar que tan factible es adoptar una u otra práctica de acuerdo a lo que la organización permite.

	Agile Principles				
	Embrace Change to Deliver Customer Value	*Plan and Deliver Software Frequently*	*Human-centric*	*Technical Excellence*	*Customer Collaboration*
Level 5 **Encompassing** *Establishing a vibrant environment to sustain agility*	Low process ceremony	Agile project estimation	Ideal agile physical setup	Test driven development Paired programming No/minimal number of level -1 or 1b people on team	Frequent face-to-face interaction between developers & users (collocated)
Level 4 **Adaptive** *Responding to change through multiple levels of feedback*	Client driven iterations Continuous customer satisfaction feedback	Smaller and more frequent releases (*4-8 weeks*) Adaptive planning		Daily progress tracking meetings Agile documentation User stories	CRACK Customer immediately accessible Customer contract revolves around commitment of collaboration
Level 3: **Effective** *Developing high quality, working software in an efficient an effective manner*		Risk driven iterations Plan features not tasks. Maintain a list of all features and their status (backlog)	Self organizing teams Frequent face-to-face communication	Continuous integration Continuous improvement (refactoring) Unit tests 30% of level 2 and level 3 people	
Level 2: **Evolutionary** *Delivering software early and continuously*	Evolutionary requirements	Continuous delivery Planning at different levels		Software configuration management Tracking iteration progress No big design up front (BDUF)	Customer contract reflective of evolutionary development
Level 1: **Collaborative** *Enhancing communication and collaboration*	Reflect and tune process	Collaborative planning	Collaborative teams Empowered and motivated teams	Coding standards Knowledge sharing tools Task volunteering	Customer commitment to work with developing team

Figura 6: Matriz de SAMI

La figura 6 nos muestra: las prácticas que fueron seleccionadas, como estas se agrupan y el nivel de complejidad que se presume de estas de acuerdo al nivel en que se encuentran clasificadas.

Esta clasificación es útil para entregar un objetivo claro a las organizaciones que adoptan de cuales serán los beneficios que obtendrán y el porqué se seleccionan unas prácticas por sobre otras de acuerdo al mayor beneficio que puede obtener la organización, además permite evaluar cuales son las debilidades de una organización que busca adoptar agilidad.

2.2.2. 4-Stage Process: un proceso para guiar la adopción

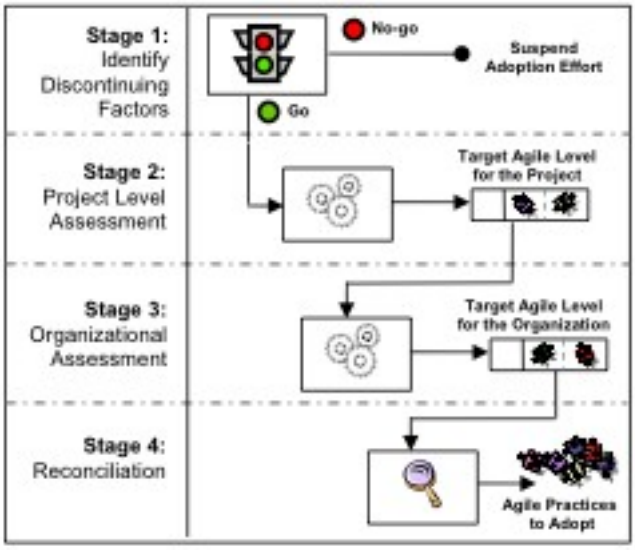


Figura 7: Diagrama de 4-Stage Process

La parte principal del "Agile Adoption Framework" es el "4-Stage Process" (proceso de 4 fases), que propone un procedimiento para guiar la adopción en una organización. Estos cuatro pasos se agrupan en dos objetivos:

1. Determinar si una organización está lista para avanzar en agilidad. Esto se logra en la fase 1.
2. La identificación de las prácticas ágiles que tienen más posibilidad de ser incorporadas en la organización. Esto se logra con las fases 2, 3 y 4.

En lo siguiente se describen las fases de esta propuesta:

Fase 1: Identificación de Factores de Discontinuidad

En esta fase se descubren la presencia de cualquier tipo de problemas o posibles topes que no permitan el éxito en el proceso de adopción. Por ejemplo, la disposición de las personas y organización, capacidad de aprendizaje, etc. Se culmina esta fase con la conclusión de si se continúa o no con el proceso de adopción.

Fase 2: Evaluación a Nivel de Proyecto

Como el proceso de adopción ha comenzado, en esta fase se mide el nivel agilidad de un proyecto en particular, para lograr esto se utiliza SAMI.

Fase 3: Evaluación de Preparación de la Organización

La adopción no sólo involucra a un proyecto en particular sino a la organización en la que este se desarrollará, para que en un proyecto se puedan adoptar algunas prácticas es necesario que la organización también adopte algunas que sean complementarias y así el proceso de adopción sea exitoso, para esto se utiliza SAMI para determinar el nivel de agilidad de la organización para luego evaluar que tan preparada esta se encuentra para enfrentar el proyecto.

Fase 4: Reconciliación

Tomando los resultados de las fases 2 y 3 se determinan cuales serán las prácticas que si será posible adoptar.

En resumen, el "Agile Adoption Framework" logra guiar el proceso de adopción midiendo cuales son las capacidades de la organización mientras se realiza el proceso, pero no entrega guías de como se deben enseñar prácticas, aunque si es muy valiosa la forma en que cada práctica es clasificada para que la forma de introducir prácticas sea desde las más "fáciles" a las más "difíciles".

2.3. La experiencia de los cursos CC62V y CC61A

En los siguientes subcapítulos se invita a conocer la experiencia de los cursos CC62V y CC61A que imparte el DCC. Por una parte CC62V presenta un modelo probado de adopción de XP que ha sido exitoso, y por otra parte CC61A se incluye por ser el caso más demostrativo de como la adopción de algunas prácticas ágiles, en especial las de gestión, permite el desarrollo exitoso de proyectos y que como la utilización la herramienta Painless Tracking ha afectado estos éxitos.

2.3.1. CC62V: Taller de Metodologías Ágiles de Desarrollo de Software

Desde el año 2002, que el DCC se incorporó a la enseñanza de metodologías ágiles, con el curso de CC62V "Taller de metodologías ágiles de desarrollo de software" [2], cuyo objetivo es evaluar en la práctica (proyectos reales) Extreme Programming. En este curso los alumnos desarrollan un software **sólo** durante las horas de clase (no se permitían inversión en horas extras), el cuál al final produce un software validado, funcional y útil para los clientes. Por esto la opinión de los clientes y de los alumnos ha sido que "XP funciona"[2].

En el contexto de adoptar una metodología ágil, la adopción de XP es complicada, por sus prácticas y principios que al ser puestas al límite complican a los equipos que intentan llevarlas a cabo y también por la dependencia de estas con condiciones iniciales de las organizaciones donde se desea aplicar, pero el modelo instruccional del curso logra la adopción de varios principios y prácticas de manera clara y sostenida.

En lo siguiente se describe el curso CC62V.

2.3.1.1. Descripción del curso

El curso busca que los alumnos comprendan en la ***práctica*** lo que es una metodología de desarrollo ágil, en particular Extreme Programming, a través de la entrega de conocimiento sobre los valores, principios y prácticas que componen la metodología ágil, la aplicación de esto a través del uso de las prácticas que la metodología ágil propone, todo esto buscando que los alumnos puedan discernir el valor relativo que tiene cada práctica y permita la posterior aplicación de estas en su vida laboral.

Como aplicación directa del principio constructivista "Aprender Haciendo" el curso se construye

como una metáfora de XP, y logra sus objetivos a través de los siguientes elementos y como estos los aplica:

- Aplicación de Valores:

  El profesor realiza charlas introductorias. Los alumnos profundizan dentro de los valores y prácticas exponiendo sobre aspectos de XP, este trabajo es mejorado con las intervenciones de alumnos y profesor. Además los alumnos se enfrentan semanalmente a la reflexión que comparten con sus compañeros, y finalmente, deben realizar un ensayo sobre lo aprendido y como esto puede ser aplicado.

- Aplicación de Prácticas:

  El aprendizaje de las prácticas es realizada durante las horas de taller en donde los alumnos aplican estas enfrentándose al desarrollo de un software y al trabajo en equipo desde la perspectiva ágil.

- Principio de dedicación de tiempo fijo:

  Sólo durante las horas de taller semanales los equipos pueden desarrollar, y no en la casa.

- Miniaturización de XP:

  Como el tiempo de desarrollo está restringido sólo a las horas de taller, el largo de las iteraciones también es corto y sólo duran algunas sesiones de taller, lo corto de las iteraciones dificulta el logro de un buen ritmo de desarrollo por lo que la definición de estas debe ser flexible, es decir, pueden existir pocas o varias iteraciones dependiendo de los equipos de desarrollo.

- Planificación del desarrollo:

  Luego de un taller que explica el “Planning Game” los alumnos comienzan utilizar esta práctica para la planificación del desarrollo.

- Organización del equipo:

  Cada equipo elige para liderar la gestión del equipo un coach, el cual es apoyado por la labor del tracker que debe mantener actualizadas las herramientas de comunicación grupal. Pero la responsabilidad de todas las labores del desarrollo es compartida por el equipo.

- Autenticidad de la experiencia a través de clientes reales con necesidades reales:

  Esto es lo que hace que el taller permita experimentar un verdadero desarrollo utilizando una metodología ágil, así los alumnos pueden enfrentarse a la realidad de la negociación de un proyecto con alcance variable. Esto es claro dado que los recursos y tiempos son fijos y lo que se busca es maximizar el valor dadas esas restricciones.

Cada uno de los elementos antes mencionados permite a los alumnos vivir una experiencia completa de lo que significa el desarrollo ágil.

2.3.1.2. Organización de un proyecto en torno a XP

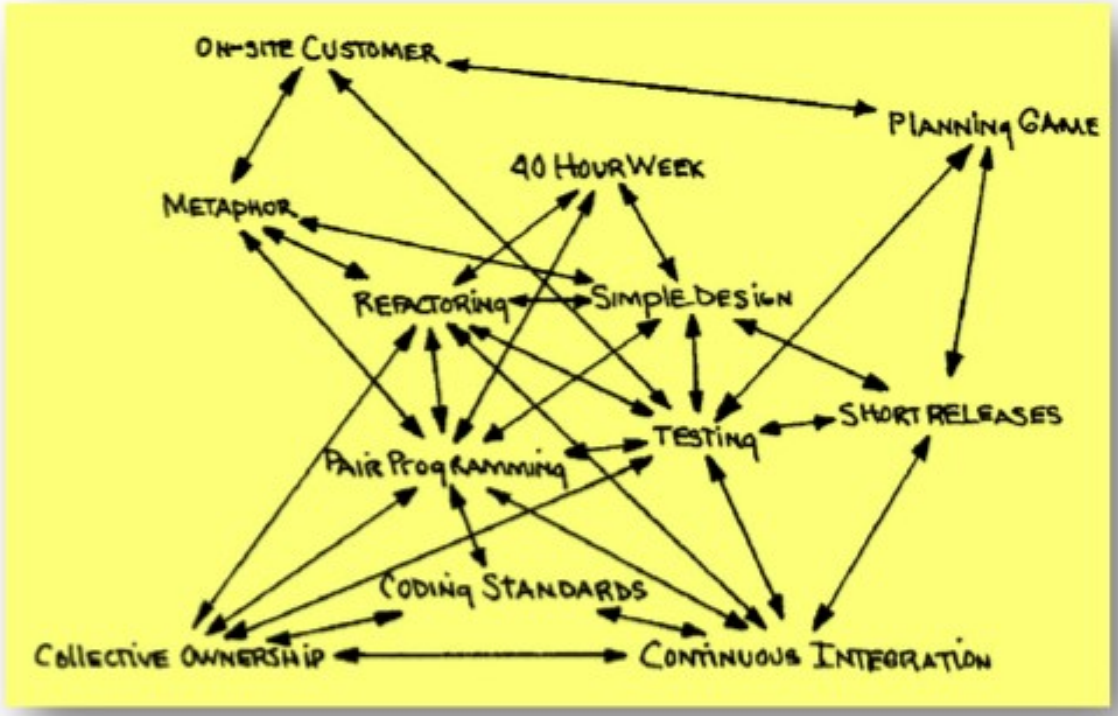


Figura 8: Principios y Prácticas de Extreme Programming

Extreme Programming propone diversos principios y prácticas para hacer ágil el desarrollo, estas permiten lograr las bondades de la agilidad descritas ya en el capítulo 1, pero entenderlas como todas juntas funcionan no es simple, basta ver la Figura 8, es por ello que estas deben ser puestas en contexto y clasificadas de modo de permitir a los alumnos comprender como estas funcionan en conjunto y logran armonía.

El cliente y el problema de negocio, la tecnología y un equipo de desarrollo son los ámbitos a los que un ingeniero de software se enfrenta en cada proyecto de software, en todo momento el intenta armonizar estos elementos que se encuentran en constante cambio y aún con ello lograr un buen producto de software. Es posible organizar entonces estos aspectos en mecanismos de retroalimentación que permiten mantener sincronizados estos ámbitos durante el proyecto y no perder con esto el principio de tiempo fijo y alcance variable y así lograr un ritmo de trabajo que permita entregas continuas y retroalimentación. La figura 9, nos presenta la relación entre los aspectos de un proyecto y como XP los organiza.

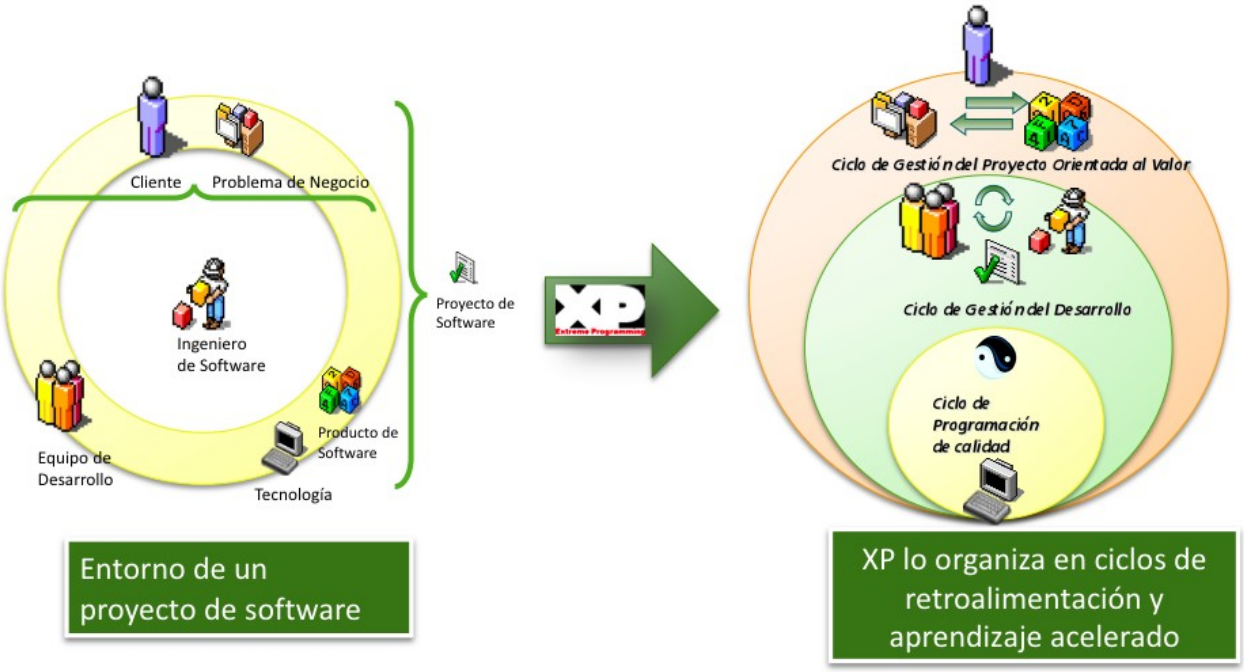


Figura 9: Ciclos de Sincronización de XP

Estos ciclos de sincronización dan al alumno la focalización necesaria de las prácticas de acuerdo al ámbito en que se mueve a diferencia de la vista en la figura 8. Los ciclos identifican y separan 3 grandes áreas de todo proyecto de software:

- La planificación y estrategia del proyecto, donde el cliente es parte fundamental del proceso, esto muestra claramente como el cliente es involucrado dentro del desarrollo y no sólo tomado en cuenta a la hora de definir requerimientos como en una metodología tradicional.
- El trabajo en equipo y producción de software, es el motor que a través de la planificación y comunicación mantiene controlada la incertidumbre entregando al cliente información

sobre avance, adelantos y atrasos, y se preocupa de que el ritmo de desarrollo sea máximo manteniendo al equipo informado e involucrado en el negocio al que el proyecto esta orientado.

- La calidad del software, existe una serie de prácticas que permiten tanto mantener el ritmo de desarrollo como asegurar que la calidad del software sea máxima.

La siguiente figura nos muestra como las prácticas de XP funcionan en conjunto:

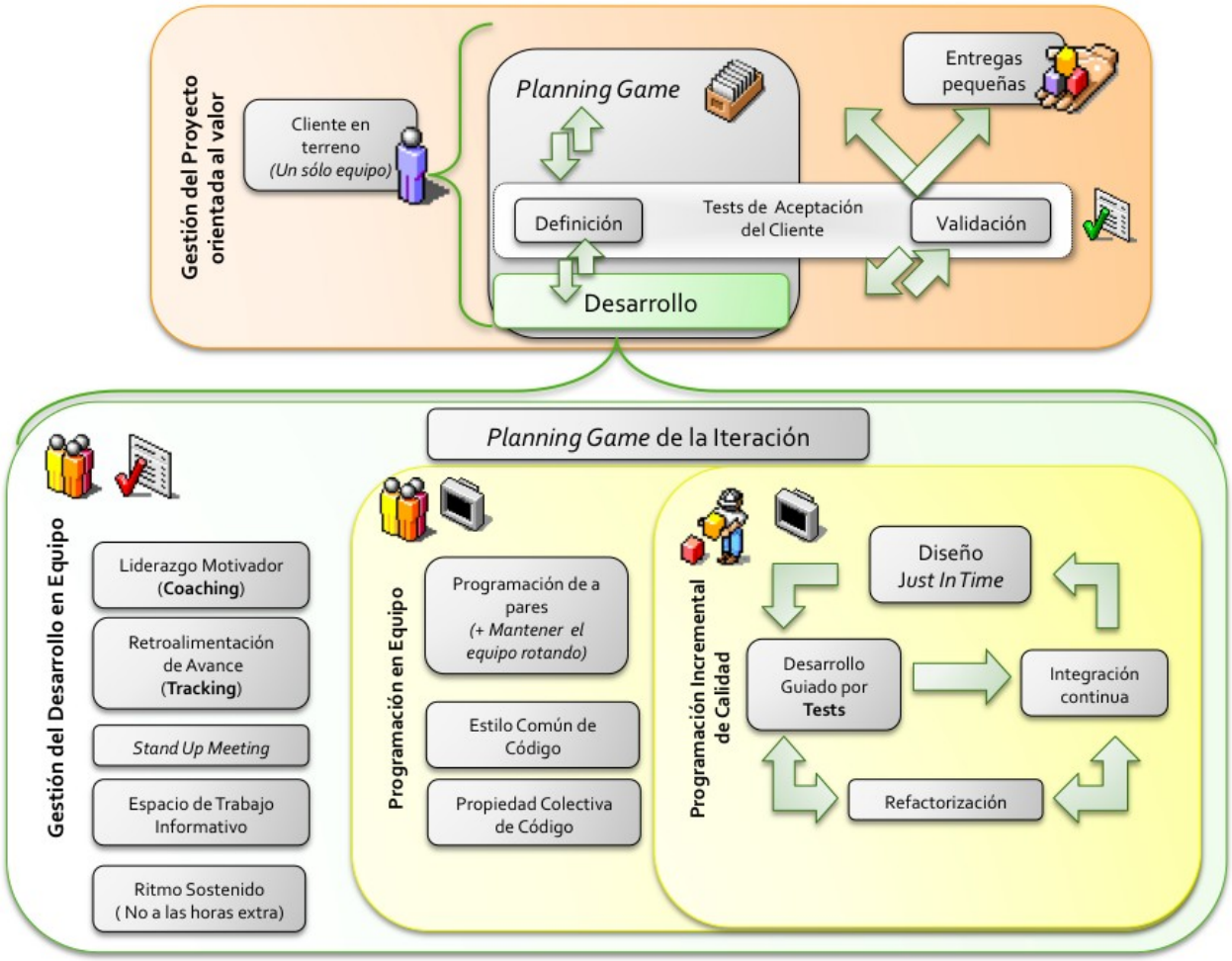


Figura 10: Armonización de Prácticas de XP

Resaltan de esta forma de trabajo 4 actores:

- El cliente quien define y prioriza que es lo que se debe desarrollar
- El Coach que lidera al equipo para asegurar buena integración del cliente con el equipo,

de monitorear el avance y mantener el ritmo de desarrollo.

- El Tracker quien mantiene registrado todo el avance y hechos sucedidos en el proyecto, y con esto entregar información útil para la toma de decisiones y dar visibilidad del avance.
- El equipo quienes buscan satisfacer las necesidades del cliente pero sin transar la calidad que el software debe tener.

La aplicación de estos principios y el como estos están ordenados permite a los alumnos del curso focalizarse y utilizar las prácticas correspondientes al ciclo al que estén enfrentando y así lograr una correcta apreciación y entendimiento de estas.

2.3.1.3. Rol del Profesor

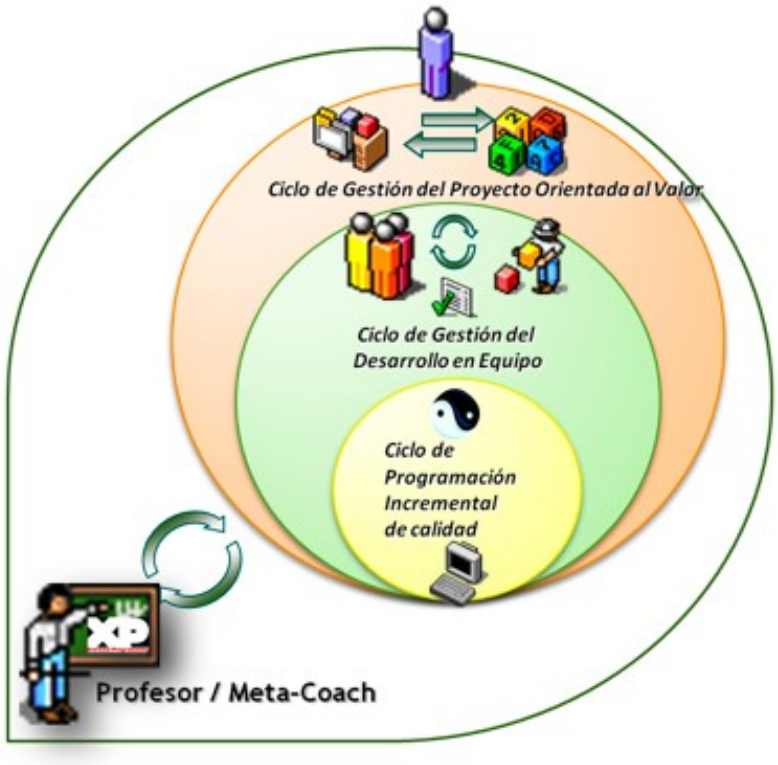


Figura 11: El "Meta-Coach"

El rol del docente es fundamental en este proceso de enseñanza práctica de metodologías ágiles ya que con la tensión que la práctica (el desarrollo de un software en poco tiempo) produce a los alumnos el profesor debe ser quien llama a reflexionar, a que los alumnos se den tiempo para

pensar en como mejorar. Por esto el profesor se pone en la posición de un llamado “Meta-coach” que se encarga de entregar todos los conocimientos básicos sobre una metodología ágil y luego procura promover una cultura de reflexión y aprendizaje sostenido. Las siguientes prácticas definen las labores realizadas por el docente:

- Reproducción fiel de XP
- Involucramiento temprano de los alumnos
- Corta introducción teórica
- Desarrollo en clases, investigación en la casa
- Retroalimentación temprana a los alumnos
- Aprendizaje cognitivo de nuevas destrezas

Durante el desarrollo entrega consejo sobre como aplicar correctamente los principios y prácticas, y está preocupado de velar que el ritmo del desarrollo sea el adecuado si ve problemas entonces interviene a través de llamar al equipo a reflexionar en los problemas y encontrar tempranamente soluciones a estos.

2.3.1.4. Resultados

El curso tiene muy buena aceptación por parte de los alumnos que lo cursan, por esto sólo es necesario concentrarse en los resultados que tiene que ver más con que aprenden los alumnos más que con el gusto de ellos.

Figura 12: Rememoración Global de Prácticas

El siguiente gráfico muestra cuales son las prácticas que mejor los alumnos comprenden y/o practican durante el desarrollo del curso.

Por otra parte es conveniente analizar cuantas de estas prácticas realmente luego del curso son aplicadas en el ambiente laboral, el siguiente gráfico muestra aquello:

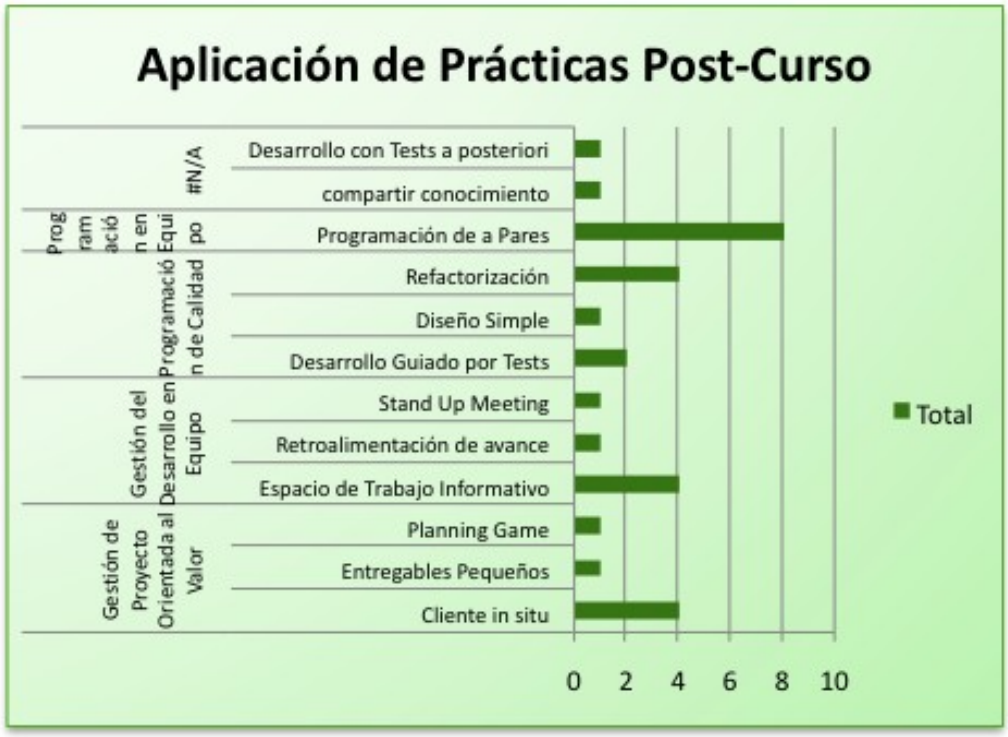


Figura 13: Aplicación de Prácticas Post-Curso

Al contrastar los gráficos 12 y 13, se aprecia que las prácticas Programación de a Pares, Refactorización, Espacio de Trabajo Informativo y Cliente in situ, tienen una mayor permanencia en el tiempo y es más posible que estas puedan ser aplicadas en el mundo laboral, no así el Desarrollo guiado por tests que por requerir mayor preparación el alcance de esta no trasciende del curso. Cabe señalar que la ausencia de prácticas en la línea de la gestión es preocupante y debiese el curso mejorar o ser apoyado esos aspectos, para que así el alumno pueda adquirir completamente la mecánica de una metodología ágil, esto es debido a que el tiempo es tan corto y la presión por producir es tan alta que la planificación y estrategia se dejan de lado, con esto el equipo se concentra en producir la mayor cantidad de software que puedan para agregar mayor valor al cliente, es destacable el hecho de que esto no es una obligación tomada con desagrado, todo lo contrario, el equipo está muy motivado a aumentar su velocidad de desarrollo y defiende que su ritmo no sea interrumpido para así lograr mayor valor para el cliente traducido en funcionalidades terminadas.

Otro estudio[22] realizado sobre 13 empresas de desarrollo de software en Grecia, el estudio muestra que prácticas como las metáforas, las 40 horas de trabajo por semana, cliente in situ y la

propiedad colectiva del código son las que más problemas presentan, y señala a la Programación de a Pares y el Desarrollo guiado por Tests como las prácticas que permiten a las empresas lograr el mayor éxito a la hora de introducir una metodología ágil, esto se puede explicar por que ellas promueven la comunicación y colaboración en el equipo, sobre en ellos donde no existe una cultura de equipo.

En vista de estos últimos resultados, el proceso de enseñanza del curso es adecuado para su transferencia al mundo empresarial y este puede entregar buenos resultados para la adopción de metodologías ágiles en las empresas debido a que promueve desde el inicio el trabajo en equipo y el aprendizaje en la práctica, con esto es posible romper con mayor rapidez las barreras culturales de desarrollo existentes en las empresas que impiden que una metodología ágil pueda ser adoptada.

2.3.2. CC61A: Proyecto de Software

Desde el año 1998 que los alumnos del DCC tienen la experiencia de desarrollar un proyecto de software con clientes y necesidades reales en un plazo de un semestre. Desde ese año que las experiencias obtenidas no eran exitosas hasta que en el 2005 se le comenzó a dar enfoque ágil que logra mejoras en lo efectivo y permite la repetibilidad de los éxitos. Esta modelo de gestión de proyecto [21] utiliza los siguientes elementos claves para lograr sus éxitos:

- Iteraciones mensuales (3 en total).
- Planificación adaptativa (de alcance variable como en agilidad).
- Uso de herramientas **simples** de planificación y gestión de riesgos.
- Equipos con estructura horizontal.
- Apoyo a los alumnos a través de tutores (ingenieros experimentados).
- Los proyectos a realizar son seleccionados.

Este modelo con simples elementos y con muy poca instrucción inicial ha logrado exitosos resultados, esto lo muestra la figura 14, donde en verde muestra los proyectos exitosos, en amarillo los con problemas pero igualmente terminados y los que fracasaron en negro.

Organizaciones Clientes según tamaño	Negocio	2005		2006		2007	
		Otoño	Primavera	Otoño	Primavera	Otoño	Primavera
Pequeña							
Acepta.com	Factura Electrónica	●		●			
Albagli, Zaliasnik Abogados	Bufete Legal	●					
Andinatech	Telefonía Móvil		●	●		●	
Centro Estudios Retail	Universidad				●		
Linuxcenter	OTEC	●					
METS	Call Center	●	●				
Novared	Desarrollo Sistemas					●	
Se Innova	Consultora ONG	●					
Tastets	Telefonía Móvil		●	●			
Tecnova	Desarrollo Sistemas					●	
Mediana y Grande							
Optimisa	Desarrollo Sistemas						●
Sixbell	Telecomunicaciones			●	●		
EntelPCS	Telefonía Móvil			●	●		●
Falabella	Retail						●

Figura 14: Desempeño de CC61A

No cabe duda que la inclusión de elementos ágiles en el curso es lo que permite en el corto tiempo que cuentan brindar una experiencia exitosa para los alumnos y las empresas.

A diferencia de lo que ocurre con el curso CC62V aquí los elementos de gestión si son mejor entendidos y esto se puede explicar por que el tiempo que tiene este curso es mayor (desarrollo de media jornada durante 3 meses) lo que permite a los alumnos reflexionar y analizar en mayor de detalle desde el punto de vista de la planificación y no sólo de lo productivo como ocurre con CC62V.

Pero, que la gestión sea algo clave en este curso no implica que desde el principio la apliquen correctamente, de hecho el curso es más apreciado a medida que avanza y alcanza su mayor importancia las prácticas de gestión desde el medio al final del desarrollo de este, de manera empírica, ha sido notorio que los equipos que logran más tempranamente comprender los principios de gestión ágil obtienen mayor éxito, esto es porque la medición actual de éxito en el curso evalúa la satisfacción del cliente por el software entregado pero no así su percepción de si el valor recibido realmente es el máximo que le permite sentirse totalmente feliz. Esta preocupación nace de la agilidad ya que en el mundo tradicional ya la entrega de software es necesaria para dar la sensación de satisfacción al cliente.

Por ello y para permitir que la satisfacción del cliente sea temprana es que se incorporan 2

herramientas para apoyar la gestión, una para la planificación (la Painless Tracking versión Excel) y una para la gestión de riesgos, estas son bastante simples y requieren de poca capacitación en como estas se usan.

Aunque las herramientas son simples de utilizar no lo es así lograr un correcto uso de ellas debido a que es necesario entender los principios de gestión ágil en los que estas se basan. Generalmente el buen uso de las herramientas, y en especial la de planificación, es logrado recién al final del curso por lo que la percepción de la importancia de las herramientas es muy tardía.

Con eso se pierden semanas de visibilidad que pueden llevar al cliente a verse poco involucrado y sobretodo menos satisfecho.

También cabe destacar aquí el rol de los tutores que da a los alumnos tranquilidad y seguridad sobre sus avances y transmite confianza y consejos que permitan llevar a mejor puerto todos los esfuerzos realizados en el proyecto. Con ellos los alumnos logran mejor compresión sobre que tanto de lo realizado es realmente avance ya que su rol más alejado del desarrollo permite dar una mejor percepción del avance que la que tienen los alumnos.

El curso en estos momentos para permitir que la gestión ágil sea entendida desde los inicios, se han agregado, a partir de la experiencia de CC62V, que los alumnos y cliente participen en un juego práctico de gestión ágil llamado “Extreme Hour”[29], en donde, además de conocerse, aprenden los roles que deberán interpretar durante el proyecto.

3. La herramienta Painless Tracking Excel

3.1. Painless Software Schedules

Un artículo del año 2000 escrito por Joel Spolsky [20] comenta sobre lo poco satisfactorio que es para los desarrolladores de software el hecho de crear y mantener un plan de trabajo, las razones pueden ser variadas pero lo común es que no se hace porque es realmente doloroso(largo, complejo y poco valioso) hacerlo y dos el plan nunca se sigue, por lo tanto, no aporta valor.

Buscando cambiar esta percepción, Spolsky propone una forma indolora (en inglés, painless) y valiosa de hacer planes. Algunos de estos pasos son descritos en lo siguiente:

1. Usar simplemente Microsoft Excel, no es necesaria otra herramienta avanzada ya que el costo de aprender a usarla puede no ser necesario. Además, el equipo al usar otro tipo de herramienta debe adaptarse a cómo esta funciona.
2. Que sea simple. Sólo son necesarias algunas columnas que puedan ser memorizables.
3. Para cada funcionalidad defina las tareas que se realizarán.
4. Sólo el programador que escribirá el código puede anotar lo que hará y lo que hizo. Esto porque cualquier sistema que lo haga por las personas está condenado a la falla.
5. Elija un grano fino para estimar sus tareas, es aconsejable las horas y no días.
6. Mantener el registro (track) de la estimación original y la actual.
7. Actualizar diariamente la columna de estimación actual. Registrar diariamente no debería tomar más de 2 minutos.
8. Anotar también vacaciones, feriados, etc.
9. Agrega tiempo para arreglar problemas en el código. Con esto se asegura tiempo para arreglar problemas en el momento en que el código es escrito y en el momento en que es descubierto.
10. Poner el tiempo de integración en el plan.
11. Agregar un tiempo extra libre para cualquier imprevisto.
12. Nunca permitas que el cliente te obligue a reducir tus estimaciones.

13. Un plan es como bloques de madera en una caja. Si se tiene 6 meses para desarrollar pero tu plan dice 12, entonces estás retrasado, por lo tanto hay que eliminar funcionalidades para evitar el retraso.

En resumen, la idea de Spolsky permite mantener un plan y el estado del proyecto que permite tomar decisiones tempranas sobre el éxito del proyecto, y además hacer visibles los tiempos en que son ocupados durante el transcurso del proyecto. Cabe destacar la sugerencia de Spolsky de definir las tareas que las funcionalidades requieren para ser logradas.

3.2. La Herramienta Painless Tracking Excel

Tomando como inicio el artículo de Spolsky y buscando una herramienta de gestión ágil que permitiese dar visibilidad del avance de un proyecto, el profesor Villena construye la herramienta Painless Tracking, la que aprovecha las características de cálculo que ofrece la herramienta Microsoft Excel, para servir de apoyo a la gestión de los proyectos del curso CC61A. Esta fue introducida el año 2003 en CC61A y se transformó en la herramienta de gestión oficial del curso desde el 2005.

Los principales aspectos de la herramienta se enfocan en:

- Permitir que los desarrolladores construyan fácilmente las listas de las características solicitadas para el software, desmenuzándolas en tareas ad-hoc para después seguir el progreso de cada una de ellas.
- Apoyar en el desarrollo a usuarios múltiples, proyectos y más.
- Permitir la comparación de las estimaciones originales con el avance en tiempo real.
- Permitir el filtrado de tareas según sus características (responsable, avance, etc.)
- Incorporar las notas detalladas sobre cada característica y tarea.
- Manejar una historia completa de todos los cambios de la tarea.

Con la Painless Tracking los alumnos han podido conocer parte la gestión ágil y en especial los beneficios de ella como: dar al cliente visibilidad del estado del proyecto y así mantener o crear una relación de confianza con el cliente en el proyecto y el equipo de desarrollo, eliminar los atrasos en entregas, favorecer el trabajo en equipo ya que las tareas pueden ser vistas por todos y cualquiera puede tomar una y trabajar en ella siempre pensando el bien del proyecto, entre otros.

Pero aunque es simple de utilizar la fuerte carga en conceptos de gestión ágil en las que se basa no permite a los alumnos utilizarla con facilidad y así extraer todos los beneficios de ella.

3.3. Elemento de la Herramienta Painless Tracking Excel

La herramienta consta de tres áreas: planificación, tracking y reportes. La figura 15 presenta todo lo que hace la herramienta Painless Tracking, y se puede ver a la izquierda las áreas de planificación y tracking y a la derecha las de reportes. Algunos de estos elementos vistos en la herramienta se detallan en lo siguiente.

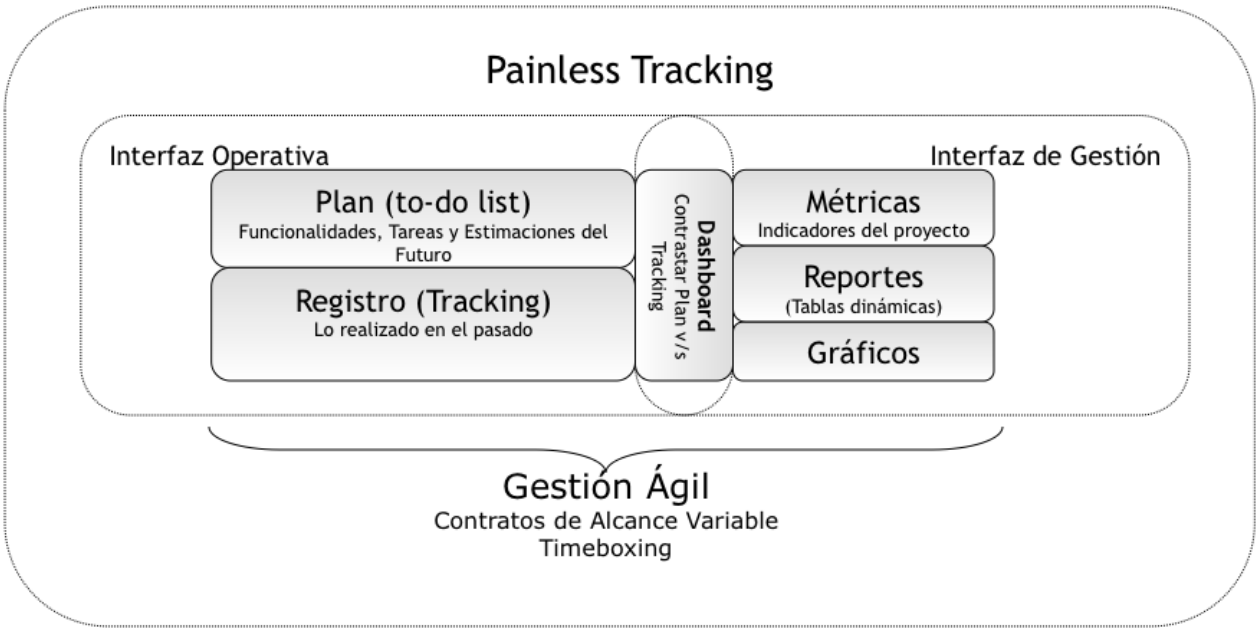


Figura 15: Painless Tracking Excel: Arquitectura Funcional

En el área de planificación, es donde se definen funcionalidades y tareas, se agrupan las tareas por iteración, se prioriza, se define el responsables y se estiman el tiempo que cada tarea tomará. La figura muestra un ejemplo de como esto es visto en la herramienta.

Iteración	Producto	Prioridad	ID Tarea	Orig. Est	LastUpdatedR	FechaUltimaAct	Ocup. Actual.	Rest. Est	Total Actual.
0	Hola mundo	1	hola.1	14	295	15-09-06	9	1	10
0	Hola mundo	1	hola.2	14	51	17-08-06	3	0	3
0	Hola mundo	1	hola.3	50	114	23-08-06	58,5	0	58,5
0	Hola mundo	1	hola.4	28	293	15-09-06	0	0	0
0	Hola mundo	1	hola.5	3,5	101	23-08-06	8	0	8
1	Entendimiento del problema	1	entend.1	6	41	16-08-06	6	0	6
1	Entendimiento del problema	1	entend.2	28	152	29-08-06	22,5	0	22,5
1	Entendimiento del problema	2	entend.4	4	34	16-08-06	0,5	0	0,5
1	Entendimiento del problema	1	entend.3	14	48	17-08-06	3	0	3
1	Entendimiento del problema	2	entend.5	5	122	24-08-06	3,5	0	3,5

Figura 16: Painless Tracking Excel: Área de Planificación

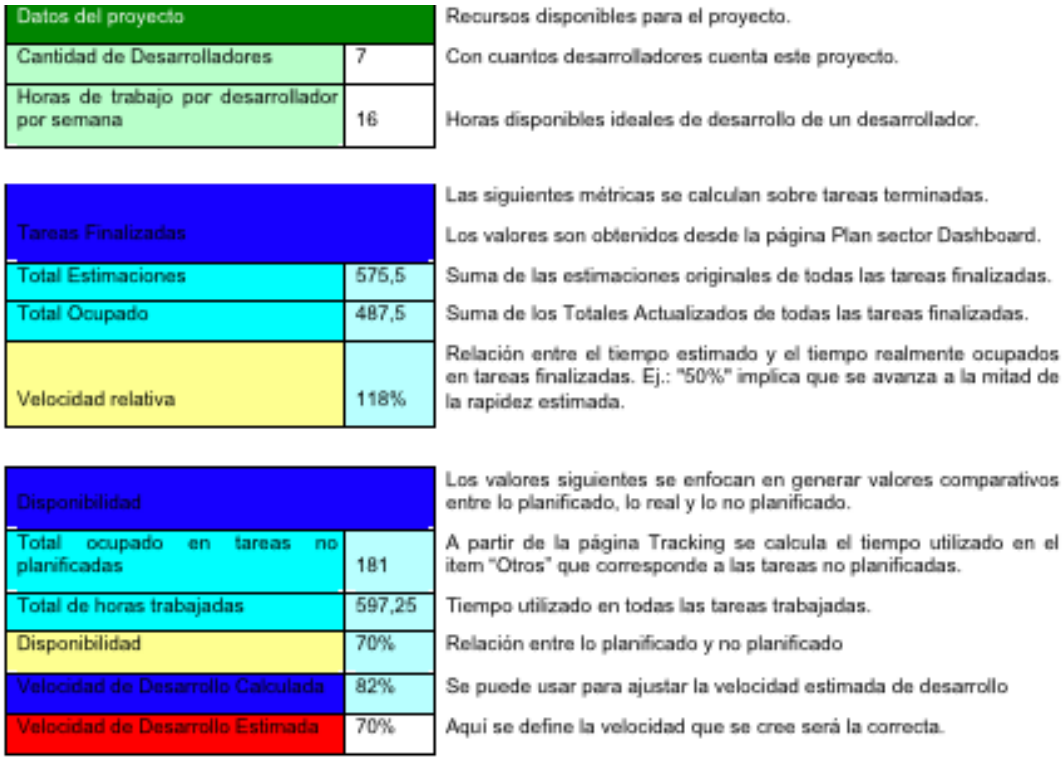
El área de tracking (ver Figura 17) es el verdadero corazón y elemento diferenciador de esta herramienta con las propuestas de Spolsky. En este lugar los desarrolladores registran el trabajo realizado y además estiman el trabajo restante, con esto sólo haciendo el tracking el plan se mantiene actualizado. Los campos necesarios para realizar el tracking son la iteración en la que se está trabajando, la fecha a la que corresponde el registro, identificar la tarea que se está realizando, el tiempo ocupado y restante, el desarrollador y comentarios de que es lo que se hizo y que falta por hacer.

Producto	Iteración	Fecha	ID Tarea	Ocupado	Restante Estimado	Desarrollador	Qué se hizo	Qué falta por hacer
Otros	0	15-08-06	otros	5	0	felipe	Feriado	
Otros	0	15-08-06	otros	7,5	0	carlos	Feriado	
Otros	0	15-08-06	otros	3	0	tguridi	Feriado	
Otros	0	15-08-06	otros	5	0	ftroncos	Feriado	
Hola mundo	0	14-08-06	hola.1	1,5	12,5	carlos	Llegamos, mi computador funciona	Tarjeta entrada
Hola mundo	0	14-08-06	hola.2	1	13	carlos	Instale turbogears, actualizacion de python en mi Mac	Falta instalar django y ver las conexiones al servidor de aplicaciones
Otros	0	14-08-06	otros	0,5	0	carlos	Tiempo de espera de llegada	
Hola mundo	0	14-08-	hola.3	1	83	carlos	Pruebas de Python,	Seguir estudiando el

Figura 17: Painless Tracking Excel: Área de Tracking

Y la última área es la de reportes, que permite visualizar el avance del proyecto, entrega métricas que indican el estado del equipo desarrollador y advierten de la posibilidad de retrasos. Esto indicaría al equipo qué debe comenzar a negociar, informa de las horas trabajadas por cada persona, y otros reportes que analizan en más detalle el avance del proyecto según funcionalidad,

iteración o proyecto. Esta área es implementada en base a las herramientas propias de análisis que provee Excel, como gráficos y tablas pivote. A modo de ejemplo, las siguientes imágenes muestran algunos de los reportes que es posible obtener:



Datos del proyecto		Recursos disponibles para el proyecto.
Cantidad de Desarrolladores	7	Con cuantos desarrolladores cuenta este proyecto.
Horas de trabajo por desarrollador por semana	16	Horas disponibles ideales de desarrollo de un desarrollador.

Tareas Finalizadas		Las siguientes métricas se calculan sobre tareas terminadas. Los valores son obtenidos desde la página Plan sector Dashboard.
Total Estimaciones	575,5	Suma de las estimaciones originales de todas las tareas finalizadas.
Total Ocupado	487,5	Suma de los Totales Actualizados de todas las tareas finalizadas.
Velocidad relativa	118%	Relación entre el tiempo estimado y el tiempo realmente ocupados en tareas finalizadas. Ej.: "50%" implica que se avanza a la mitad de la rapidez estimada.

Disponibilidad		Los valores siguientes se enfocan en generar valores comparativos entre lo planificado, lo real y lo no planificado.
Total ocupado en tareas no planificadas	181	A partir de la página Tracking se calcula el tiempo utilizado en el item "Otros" que corresponde a las tareas no planificadas.
Total de horas trabajadas	597,25	Tiempo utilizado en todas las tareas trabajadas.
Disponibilidad	70%	Relación entre lo planificado y no planificado
Velocidad de Desarrollo Calculada	82%	Se puede usar para ajustar la velocidad estimada de desarrollo
Velocidad de Desarrollo Estimada	70%	Aquí se define la velocidad que se cree será la correcta.

Figura 18: Painless Tracking Excel: Métricas

		Datos		
Iteración	Producto	Suma de Porcentaje Avance	Suma de Ocup. Actual.	Suma de Rest. Est.
1	Entendimiento del problema	100,0%	109,85	0
	Sistema para agregar nuevos tests	100,0%	186,4	0
	Tests reales	100,0%	9	0
	daemon que corra scripts	100,0%	49,5	0
	Planificacion	96,9%	15,5	0,5
Total 1		99,9%	370,25	0,5
2	Vista de tags	50,9%	14	13,5
	Correción de problemas varios	69,2%	4,5	2
	Funcionalidades interfaz	68,0%	51	24
	Funcionalidades demonio	83,7%	10,25	2
	Reportes	46,7%	22,75	26
	Documentos	3,3%	2,5	73,5
	Reuniones con Hernán	41,1%	9,75	14
	Reuniones grupo	100,0%	19,5	0
	Upgrade interfaz	100,0%	5,5	0
Total 2		39,1%	142,75	222,5
Total general		69,7%	513	223

Figura 19: Painless Tracking Excel: Reporte de Avance

4. Painless Tracking Excel en una Empresa

Este capítulo cuenta como afectó la vida de un proyecto de un equipo en un empresa real la herramienta Painless Tracking en su versión basada en Excel, y como se hace necesario el rediseño de esta para que la adopción de gestión ágil sea adecuada. Además, se visualizan los deficiencias que la herramienta exhibe al ser enfrentada a formas de gestión y desafios distintos a los del curso CC61A para los que la herramienta no estaba diseñada.

4.1. El equipo de I+D

Figura 20: Una reunión ampliada de planificación

Desde septiembre y hasta mediados de diciembre, el equipo de Investigación y Desarrollo(I+D) de la empresa Microsystem utilizó la herramienta basada en Excel. Este equipo durante ese tiempo estaba compuesto por 6 personas las que en conjunto tenían que enfrentar tanto el desarrollo de un proyecto de duración de 9 meses como el de otros proyectos más pequeños y/o mantención de algunos proyectos ya en producción.

Al hacer una vista general sobre como este equipo vive el día a día obtenemos las siguientes

características:

- Trabajan en múltiples proyectos.
- El cliente no está cerca.
- Deben responder a problemas rápidamente.
- El equipo es multidisciplinario.
- El equipo tiene conocimiento de metodologías ágiles y aplica algunas prácticas.
- Han utilizado durante algunas semanas la herramienta.
- No existen clientes que definan entregables.
- No existen tiempos parciales y fijos de entrega, sólo una fecha límite de desarrollo.

Es claro que las características de este equipo y lo que se pide de ellos es muy diferente a lo que ocurre en el curso CC61A, donde el cliente está cerca y comprometido, las entregas son iteraciones de plazo fijo, los alumnos sólo enfrentan un proyecto a la vez, etc.

4.2. Experiencia

La herramienta fue utilizada para gestionar el proyecto de duración más prolongada(9 meses). Al principio fue claro notar que la herramienta aportaba valor debido a que permite ver el avance en el desarrollo del equipo, priorizar la entrada de nuevas peticiones de trabajo (que podían ser para otros proyectos) y entregar información que podía ser utilizada por la Alta Gerencia.

Así durante los 3 primeros meses del proyecto la herramienta tuvo que ser enseñada y esperar la aceptación de la Alta Gerencia, la que con buenos ojos terminó mirando a la herramienta ya que permitía controlar mucha información sobre el avance y el plan futuro del proyecto, lo que implicó que la herramienta fuese introducida en otras áreas (aunque no con el mismo éxito).

De esta primera parte ellos han recomendado que la herramienta debe poseer elementos que permitan dar más visibilidad sobre:

- Los proyectos en que se trabaja, avance, que es lo que resta, fechas estimadas de entrega, responsables.
- Lo que cada equipo de desarrollo y desarrollador está realizando, cuanto tiempo han utilizado en cada proyecto.

Uno de los comentarios más interesantes de la gerencia fue sugerir el cambio de nombre para "planificación" por "estrategia" ya que el plan se tiende a ver como algo no modificable, lo que está muy lejos de los principios de gestión ágil.

Cuando la necesidad de la empresa empezó requerir más cambio y más gestión en los requerimientos, la herramienta empezó a colapsar, en los siguientes puntos se muestra aquello:

- Demasiados requerimientos no permiten un buen tracking. Es más, muchas veces los desarrolladores terminaban creando nuevamente las tareas sobre las que trabajaban olvidando que anteriormente las habían definido.
- Demasiados requerimientos no permiten claridad sobre en qué se está trabajando actualmente.
- La visibilidad de un plan se pierde al no mantener ordenada de esta forma la información.
- La alta gerencia y el equipo de desarrollo ve la misma información y es confusa para los primeros. Esto indujo a malas prácticas en el cliente, por ejemplo, a entender que las estimaciones eran plazos en que la funcionalidad estaría terminada.
- Los reportes de avance y métricas están más orientas al equipo de desarrollo que al cliente, en este caso la alta gerencia.
- La herramienta no permite la gestión de múltiples proyectos, y por esto muchas veces se pierde tiempo construyendo reportes que reflejen la información reunida de todos los proyectos.
- La disponibilidad del equipo no es debidamente manejada; en esto hay que ser riguroso ya que una mala definición de la disponibilidad impacta directamente en los plazos de entrega. Muchas veces se desestimó esto y los desarrolladores deberían haber trabajado 16 horas diarias para lograr las entregas. Este indicador necesita ser agregado a la herramienta para permitir coordinar los equipos de desarrollo con las actividades comerciales de la empresa,. Esta necesidad se ha hecho notoria porque este equipo al estar desarrollando un proyecto de larga duración (más de 5 meses) y sumamente importante, es complejo incluir proyectos pequeños dentro la coordinación del equipo.

La experiencia de los otros equipos no ha sido igualmente satisfactoria ya que se pudo apreciar una mala práctica muy tempranamente que es la sobreestimación. Lo sobrestimado por estos equipos ha sido muy notorio llegando a sobrestimar hasta un 300%. Esto ha producido

descoordinaciones y ha dado la sensación que los otros equipos no son tan rápidos o buenos como los que han sobrestimado. Esto hubiese sido posible de evitar si los equipos que sobreestiman actualizarán constante y correctamente el tracking ya que así se podría notar claramente lo sobreestimado.

También se pudo validar que ***es más importante conocer y empaparse de los principios, prácticas y metodologías que existen detrás de la herramienta que la herramienta misma.*** Al explicar cómo funciona la herramienta no dio tan buenos resultados como explicar la gestión ágil detrás de ella primeramente para luego explicar el funcionamiento. Después de una charla, al respecto, realizada se logró no sólo comprensión sobre la herramienta y las ventajas que traería utilizarla sino también lo bueno de incorporar gestión ágil en proyectos con alta incertidumbre y riesgo.

Luego de 3 meses de utilización de la herramienta y de una gestión controlada, la cantidad de cambio en los requerimientos, la falta de un cliente claro a quien se le entregase lo desarrollado y la falta de definición de entregables pequeños e iteraciones empezó a descontrolar la gestión del proyecto. El uso de la herramienta empezó a disminuir hasta que ya no se siguió utilizando y la visibilidad del proyecto comenzó a desaparecer. Esta tuvo que ser construida a partir de otra información o de lo que el equipo recordaba para luego ser presentada en reuniones de avance. Allí fue que entró un nuevo concepto a ordenar la cabeza del equipo, el "kanban", que será explicado en otro capítulo.

4.3. Análisis de la experiencia

Los problemas y deficiencias alcanzados durante la experiencia son explicados mayormente por:

- la ausencia de un cliente que priorizara y definiera entregas.
- la falta de educación en algunos principios de gestión ágil o el total desconocimiento.
- la ausencia y mal entendimiento de algunas prácticas.
- la carencia de tiempo para que el equipo pudiese analizar lo que ocurría.
- la herramienta no facilita u obliga la gestión ágil.

Otro de los problemas notorios es lo complejo que es hacer tracking, ya que a diferencia de un proyecto de CC61A, el plan es muy largo y detallado, por lo que encontrar realmente la tarea a la

que quiero registrar el avance quita tiempo y esta búsqueda elimina lo indoloro de la herramienta. Además la gestión del plan, por la cantidad de tareas detalladas, se hace muy compleja de mantener.

La herramienta Painless Tracking en un comienzo permitió dar orden al equipo y a la gestión del proyecto, pero más importante fue educar a la organización en el significado de la gestión ágil. Además fue notorio que la herramienta no permite una adecuada gestión cuando el plazo del proyecto es más largo, requiere de mucho tiempo y concentración el mantenerla, que como se explicó anteriormente, esto va en contra del principio indoloro de la herramienta. Esto implica en aportar mejoras a la herramienta que guíen y simplifiquen la gestión ágil para el equipo.

Otro elemento evidente que estuvo ausente es una persona que actúe de guía en el proceso de adopción, ya que la temprana reflexión y evaluación del impacto de las prácticas de gestión ágil hubiese permitido dar una solución.

Al analizar las partes de una herramienta de gestión del desarrollo, aparecen tres principales: la planificación, el seguimiento o tracking y los reportes. Estas partes pueden ser encontradas incluso en herramientas que siguen alguna metodología ágil(en especial, en las más importantes hoy [26][27]), pero en estas últimas se agregan elementos que hacen explícito las acciones que son necesarias en una metodología ágil, por ejemplo, un área para definir entregables, pruebas de aceptación, etc. Aunque la herramienta Painless Tracking basada en Excel posee las tres partes importantes si carece de elementos que permitan dar claridad de los principios de gestión ágil en que se basa, esto explica en parte el porque los usuarios que la han utilizado no logran sacar el mayor provecho y de porque es normal el abandonar su uso.

4.4. Resultado de la Evaluación

Aunque la herramienta en su versión Excel a demostrado ser de apoyo a los alumnos de CC61A, no lo es en un ambiente laboral, como el evaluado, esto por que se hace necesario que los equipo comprendan correctamente la gestión ágil y la apliquen, y en esa linea la herramienta basada en Excel no es solución, por ello se hace necesario:

- un modelo de adopción para los equipos de desarrollo.
- adecuar la herramienta para facilitar la gestión ágil.

En este sentido, es que se busca agregar un nuevo rol a la herramienta y es que permita apoyar el proceso de adopción, es decir, no sólo continuar siendo una herramienta indolora sino también explicitar los principios de gestión ágil para que permitan dar guía a quienes la utilicen.

5. Painless Tracking como Apoyo a la Adopción de Gestión Ágil

5.1. Método de Adopción de Gestión Ágil

5.1.1. Descripción

En este capítulo se explica el método utilizado para apoyar la adopción de gestión ágil, teniendo como meta la adopción de prácticas de Extreme Programming. El método consiste en una corta serie de pasos que busca dar información para el seguimiento de la adopción y dar una medida que muestre el estado de avance de la adopción.

El método consta de 3 pasos que deben ser repetidos tantas veces como la organización lo requiera hasta lograr la adopción de gestión ágil. Los pasos son:

1. Evaluación Inicial de la Agilidad.
2. Definición de un camino de Adopción.
3. Monitoreo y Análisis del camino de agilidad.

Todos estos pasos deben ser acompañados por instrucción y guía que proveerá una persona encargada tanto de enseñar agilidad como de evaluarla.

5.1.2. Evaluación de Agilidad Inicial

En el inicio es preciso conocer cuál es el nivel de agilidad que la organización y las personas en particular poseen antes de comenzar cualquier proceso de adopción.

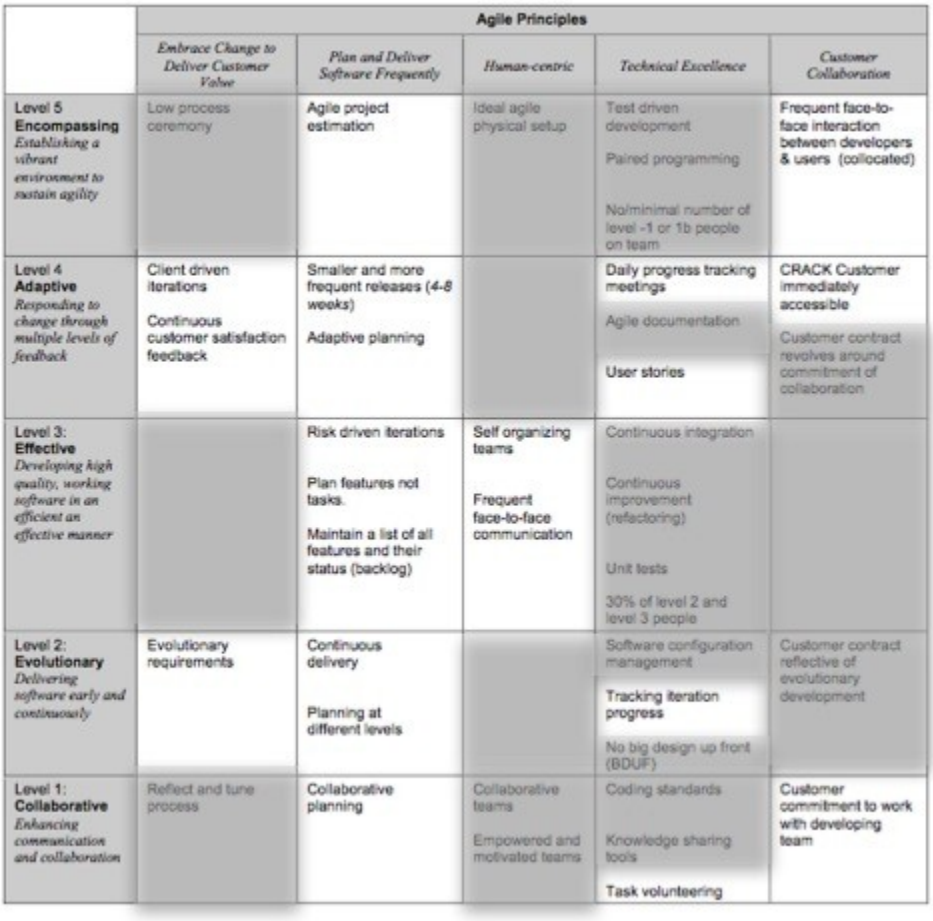
La primera pregunta al evaluar agilidad es: ¿qué evaluar?

En este caso, como sugiere Sidky[1] se ***evalúa la cantidad de prácticas que la organización utiliza*.**

Con esta premisa se describen los pasos de cómo está construido el modelo de evaluación.

Aprovechando la forma en que Sidky ordena por nivel de dificultad las prácticas ágiles basta primero con seleccionar cuáles de estas prácticas están más relacionadas con la gestión ágil. El

siguiente gráfico muestra esta elección, donde lo ennegrecido son las prácticas que no serán utilizadas:



	Agile Principles				
	Embrace Change to Deliver Customer Value	*Plan and Deliver Software Frequently*	*Human-centric*	*Technical Excellence*	*Customer Collaboration*
Level 5 Encompassing *Establishing a vibrant environment to sustain agility*	Low process ceremony	Agile project estimation	Ideal agile physical setup	Test driven development Paired programming No/minimal number of level -1 or 1b people on team	Frequent face-to-face interaction between developers & users (collocated)
Level 4 Adaptive *Responding to change through multiple levels of feedback*	Client driven iterations Continuous customer satisfaction feedback	Smaller and more frequent releases (*4-8 weeks*) Adaptive planning		Daily progress tracking meetings Agile documentation User stories	CRACK Customer immediately accessible Customer contract revolves around commitment of collaboration
Level 3: Effective *Developing high quality, working software in an efficient an effective manner*		Risk driven iterations Plan features not tasks. Maintain a list of all features and their status (backlog)	Self organizing teams Frequent face-to-face communication	Continuous integration Continuous improvement (refactoring) Unit tests 30% of level 2 and level 3 people	
Level 2: Evolutionary *Delivering software early and continuously*	Evolutionary requirements	Continuous delivery Planning at different levels		Software configuration management Tracking iteration progress No big design up front (BDUF)	Customer contract reflective of evolutionary development
Level 1: Collaborative *Enhancing communication and collaboration*	Reflect and tune process	Collaborative planning	Collaborative teams Empowered and motivated teams	Coding standards Knowledge sharing tools Task volunteering	Customer commitment to work with developing team

Figura 21: Paso 1. Selección de Prácticas

Luego de realizada la selección de prácticas, se obtiene una lista de prácticas generales ordenadas por dificultad, donde el orden considera los problemas que una organización normalmente presenta cuando intenta adoptar una metodología ágil.

El segundo paso, es transformar las prácticas generales en práctica de una metodología ágil particular. La metodología ágil seleccionada es Extreme Programming, y se utiliza la forma en que son organizadas las prácticas en el curso CC62V[2]. En el curso, las prácticas son organizadas en ciclos de sincronización, donde los de interés para la gestión ágil son:

- Ciclo de Gestión del Proyecto Orientado al Valor
- Ciclo de Gestión del Desarrollo en Equipo

Cabe destacar que, la gestión ágil principalmente considera sólo las prácticas del primer ciclo, pero en la práctica se ha apreciado que sin las prácticas del segundo ciclo el primero no funciona

La explicación del punto anterior está relacionada con que el desarrollo de software ágil se mueve en dos planos, el estratégico(primer ciclo) y productivo (segundo ciclo), y no es posible separarlo totalmente ya que existe dependencia entre estos. El plano productivo depende de las decisiones del estratégico y el plano estratégico depende de la información y resultados que entrega el productivo. Si estos dos planos no trabajan en armonía la gestión ágil y la adopción no serán posibles. Como ejemplo, la falta de cliente, de priorización y definición de entregables durante el desarrollo en el equipo I+D es uno de los puntos que provocó fracaso en el proyecto, y demuestra el porque lo estratégico no puede estar separado de lo productivo.

En el gráfico 22, se muestran las prácticas seleccionadas que se extraen de los ciclos:

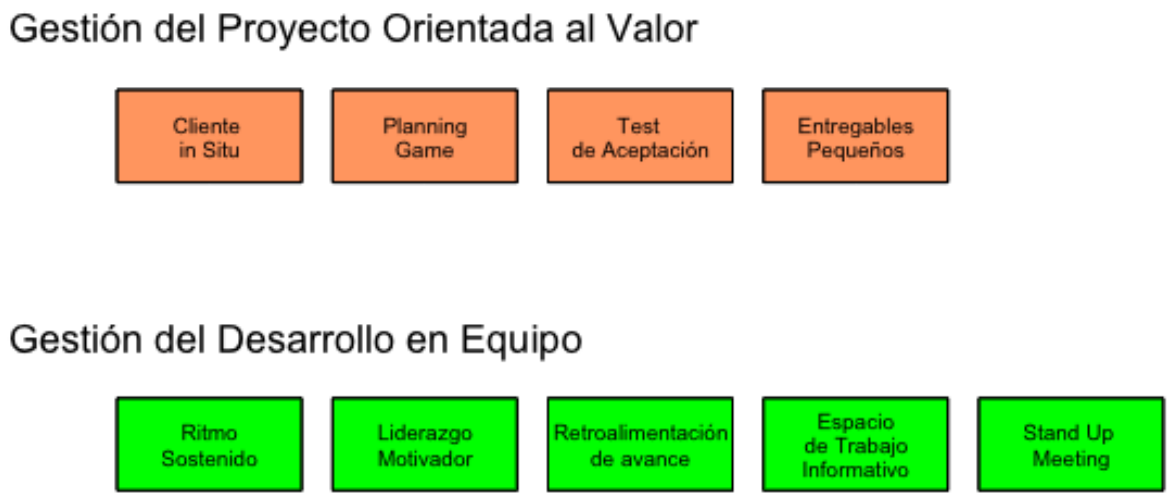


Figura 22: Paso 2a. Prácticas de XP

Las prácticas de Extreme Programming son más generales que las que se obtienen de SAMI (figura 21), esto por que SAMI se construye en base a varios principios y prácticas de varias metodologías, en el caso de Extreme Programming, la adopción de una práctica implica un esfuerzo completo que una organización no siempre estará dispuesta a realizar, en cambio Sidky divide algunas prácticas en pasos según el grado de dificultad. Tomando en consideración estos niveles, se proponen las siguientes características o prácticas menos absolutas, que en general, se busca en todo equipo ágil en relación con las prácticas de Extreme Programming:

Cliente in Situ:

- Cliente in Situ
- Reuniones diarias con el cliente
- Cliente inmediatamente Accesible
- Cliente dirige iteraciones
- Cliente comprometido con el desarrollo
- Cliente define entregables
- Reuniones semanales con cliente
- Es visible el avance de una feature
- Estado actual de todas las features
- Es visible el avance de un entregable
- Es visible el avance del proyecto

Planning Game

- Estimación de features
- Planificar entregable
- Historias de Usuario
- Minimal Marketable Feature (MMF)
- Administración de Riesgos
- Manejo de Desperfectos
- Planificación de Desperfectos
- Planificación por Features
- Requerimientos cambian
- Plan adaptable
- Requerimientos pueden cambiar
- Planificación colaborativa

- Estimación de Tareas

Test de Aceptación:

- Test de Aceptación

Entregables Pequeños:

- Feedback continuo del usuario
- Entregables pequeños
- Entregas continuas

Ritmo Sostenido:

- Planificación por iteración
- Velocidad de Desarrollo

Liderazgo Motivador:

- Reuniones de análisis de equipo
- Tareas elegidas voluntariamente
- Comunicación Cara a Cara
- Trabajo en equipo
- Equipo autoorganizado

Retroalimentación de Avance:

- Tracking diario

Stand Up Meeting:

- Reuniones diarias de grupo
- Reuniones de equipo cortas

Varias prácticas que se presentan en cliente in situ también podrían ser clasificadas en retroalimentación de avance, pero por su relevancia para el cliente(sin ellas el cliente no se involucrará) es que han sido clasificadas de esa forma.

Con las características definidas, en el paso 3 se definen los niveles en los que será evaluada la agilidad, estos niveles están relacionados con el camino necesario para lograr la adopción de

gestión ágil. Estos 3 niveles son seleccionados de SAMI. Los niveles son los siguientes:

- Nivel 1. Colaborativo: en este nivel el objetivo es facilitar la comunicación y colaboración entre las personas de un equipo, y entre el equipo y el cliente.
- Nivel 2. Efectivo: cuánto y cómo se produce software es lo fundamental en la gestión ágil, por esto en este nivel se busca que el equipo y cliente concentren el trabajo en lograr más y mejor software.
- Nivel 3. Englobador: este nivel identifica el estado en el que el equipo permite que la agilidad se mantenga y sea facilitada, donde también sea posible la incorporación de nuevas características.

Con los 3 niveles definidos basta organizar las características en los diferentes niveles, las siguientes tablas muestran este resultado:

	Cliente in Situ	**Planning Game**	**Test de Aceptación**	**Entregables Pequeños**
Nivel 3	Reuniones diarias con el cliente Cliente inmediatamente accesible Estado actual de todas las funcionalidades	Manejo de desperfectos Planificación de Desperfectos Plan es adaptable		Feedback continuo de los usuarios
Nivel 2	Cliente dirige las iteraciones con equipo Cliente comprometido con el desarrollo Es visible el avance de una funcionalidad Es visible el avance de un proyecto	MMF Administración de Riesgos Requerimientos cambian Requerimientos pueden cambiar Planificación colaborativa		Entregas Continuas
Nivel 1	Reunión semanal con cliente Cliente define entregables Es visible el avance de un entregable	Estimación de Funcionalidades Planificar Entregable Planificación por Features Historias de Usuario Estimación de Tareas	Test de Aceptación	Entregables Pequeños

Tabla 3: Paso 3. Características de Gestión del Proyecto

	Ritmo Sostenido	**Liderazgo Motivador**	**Retroalimentación de Avance**	**Espacio de Trabajo Informativo**	**Stand Up Meeting**
Nivel 1	Velocidad de Desarrollo	Reuniones de Análisis de Equipo Comunicación Cara a Cara	Tracking diario		Reuniones diarias de equipo
Nivel 2	Planificación por Iteración	Trabajo en Equipo Tareas elegidas voluntariamente			Reuniones de equipo cortas
Nivel 3		Equipo Autoorganizado			

Tabla 4: Paso 3. Características de Gestión del Desarrollo

La segunda pregunta al evaluar agilidad es: ¿cómo evaluar?

A diferencia de la forma de evaluar que utiliza el Framework de Sidky, en base a largos cuestionarios, sólo es necesario para este modelo la declaración de si una práctica existe o no. Para esto se utilizan 2 formas. La primera, dirigida a evaluar personas individualmente, se listan sobre una hoja todas las características, así los evaluados podrán una marca sobre las características que ellos aprecien como existentes en su propia forma de trabajar. La segunda forma va dirigida a evaluar el equipo, para esto se ponen todas las características en tarjetas que el equipo clasificará en existentes y no existentes. Previo a estos pasos, y de ser necesario, se deben explicar cada una de las prácticas para que así haya un consenso sobre lo que estas significan.

Aunque esta forma de evaluar no da una medida dura ni objetiva de la agilidad de las personas, lo que realmente se busca es dar el momento de análisis y retrospectiva al equipo en que ellos en base a lo que entienden de agilidad y a lo que han practicado evalúen la existencia de las prácticas que dicen y/o deben tener. También es recomendada la presencia de una persona más experimentada en metodologías ágiles que apoye al equipo para que logre un aprendizaje más correcto y profundo del nivel real y futuro del equipo.

5.1.3. Definición de Camino de Adopción

Después de evaluado y conocido el nivel de agilidad, el equipo debe definir un "camino de adopción". Este es un plan que indica al equipo las prácticas ágiles que debe fortalecer, modificar o agregar.

Las tablas 3 y 4 nos indican varias características que poseen las prácticas, por lo tanto, en el plan lo que hay que definir es primero las prácticas a las que se quiere apuntar, y luego definir en qué características se va a trabajar de acuerdo al nivel deseado y las necesidades reales de la organización.

Luego de definido el camino, el equipo debiese definir actividades que apunten a educar sobre principios y entrenar sobre prácticas. Aquí nuevamente el rol de un guía es recomendado, ya que este puede sugerir actividades que faciliten el aprendizaje, aunque esto es sólo recomendado ya que la literatura y la Web tienen muchos recursos que dan recomendaciones sobre cómo lograr la adopción de las prácticas, por esto lo primero es que el equipo tenga bien definido sus metas de

adopción y así concentrarse en buscar ayuda específicamente para cumplirlas.

5.2. Desarrollo de Painless Tracking Web Inicial: Planificación y Tracking

En esta parte se explica cómo fue construida inicialmente la herramienta. Esta es muy similar a la herramienta implementada en Excel, esto porque es la que mejor se comporta en un escenario ágil. La herramienta que resulta es utilizada para evaluar más en detalle qué elementos nuevos debiese tener la herramienta final y que realmente apoye a la adopción, aunque esta ya en un inicio se le añaden características que ya van en esa línea. Se debe adelantar que la herramienta en este estado no hace aportes para la adopción, más bien es utilizada para encontrar estos elementos faltantes y ser una base de software para construir la herramienta final.

5.2.1. Tecnologías utilizadas

La primera discusión que dio comienzo a la implementación de la herramienta fue determinar qué tipo de tecnología utilizaría. En este sentido existía la herramienta basada en Excel y la idea de comenzar una nueva basada en Web. El siguiente cuadro muestra la evaluación:

	Excel	**Web**
Construcción de Reportes	Simple, flexible, rápida	Compleja, lenta, depende de quien los implemente.
Planificar sin errores	Interfaz no adecuada, es fácil cometer errores (borrar fórmulas, por ejemplo)	La interfaz puede ser diseñada para permitir sólo hacer planificación.
Tracking rápido ligero	La experiencia muestra que esto no es posible, esto porque la asociación tarea y tracking es compleja cuando el proyecto es muy grande, y causa desorden.	Se puede construir una interfaz que facilite y favorezca el tracking
Incorporación de elementos que apoyen la adopción	Aunque se pueden diseñar e implementar interfaces adecuadas, esta es una planilla de cálculo y no un ambiente de desarrollo, por lo tanto, no es adecuado.	El desarrollo en esta tecnología puede ser facilitado por las librerías que hoy existen.

Tabla 5: Evaluación entre Microsoft Excel y Web

El cuadro anterior no evalúa características operativas que en otros trabajos sobre la herramienta han sido encontrados, sino se concentra en definir la tecnología que permita lograr mayores beneficios en el área de aprendizaje.

Como conclusión se decidió utilizar la Web para implementar la interfaz operativa, pero manteniendo a Excel como interfaz de reportes. Esto es explicado en resumen, porque Excel es

una planilla de cálculo y no una herramienta que permita la construcción adecuada de un sistema de gestión, es buena para prototipar, pero no para ser la interfaz final.

Como la utilización de la Web fue elegida como camino, la evaluación de tecnologías se concentró en determinar qué utilizar para construir las interfaces, ya que existía trabajo previo construido en base a ciertas tecnologías que permitían continuar desde allí desarrollar una nueva herramienta.

Las tecnologías actuales de la herramienta sobre Web son:

- Lenguaje Python.
- Framework Pylons, para construir los servicios Web.
- Librería SQLAlchemy, que permite operar con distintas base de datos relacionales de una misma forma.

Parte importante y faltante para construir la herramienta y como antes se adelantó, es la elección de como será implementada la interfaz para los usuarios. Para lograr una adecuada interfaz es necesaria la incorporación de Javascript. Como referencia se tomaron en consideración 3 alternativas de librerías de desarrollo en Javascript:

- Prototype y Scriptaculous
- JQuery
- ExtJS

La evaluación de ellas fue realizada en la práctica. Para esto se construyó una tabla que permitiese la administración de las funcionalidades y tareas de un proyecto y que debía contener al menos los siguientes elementos:

- Drag&Drop para priorizar tareas y funcionalidades.
- Obtención de datos vía AJAX desde un servicio Web.
- Edición sobre la misma tabla.

Aunque con las primeras dos alternativas si se logró la implementación de los elementos antes expuestos, la construcción no fue sencilla y menos reutilizable, al contrario, ExtJS sin ser más simple, permite lograr esas características y otras más, y al ser un Framework de desarrollo sobre Javascript permiter reutilización. Entre sus elementos destacados estan los elementos gráficos

como tablas, árboles, ventanas, menúes, etc. que pueden ser utilizados para distintos fines.

5.2.2. Arquitectura de la Herramienta

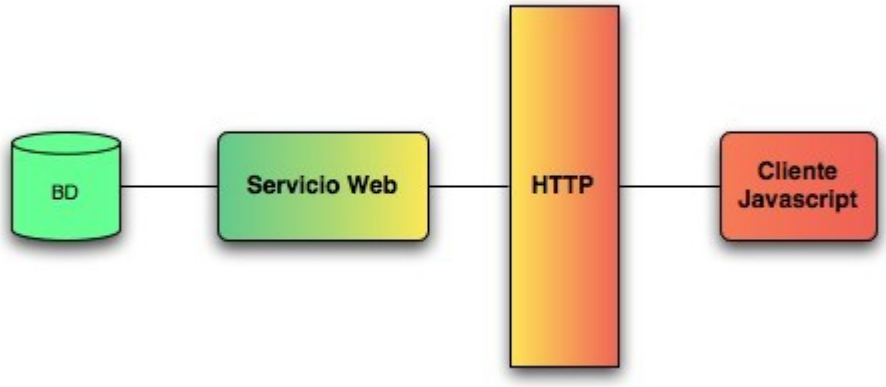


Figura 23: Vista Simple de la Arquitectura

La figura 23 describe las partes básicas de la herramienta. Eesta utiliza el protocolo HTTP para comunicarse, esta comunicación es realizada entre el servicio web y el cliente javascript, el cliente en ningún momento se comunica directo con la base de datos sino que el servicio web define un protocolo y representaciones de la información que maneja la base de datos. En ese sentido todas las validaciones y la correctitud de los datos son realizadas en el servicio y no en el cliente.

5.2.3. Descripción de la Herramienta

El servicio Web aunque es muy importante para la herramienta no es detallado en esta sección, esto porque el verdadero impacto en los usuarios se ve cuando estos entran en contacto con la interfaz gráfica, en esta sección se detalla lo desarrollado para el cliente Javascript.

El producto final de esta parte, es una herramienta similar a la Excel que junta el área de planificación con la de tracking, esto es así para lograr vencer el problema más común en la antigua herramienta que era lo complejo de hacer tracking, ahora manteniendo sin perder de vista el plan es posible revisar y realizar el tracking de una tarea. La figura 24 muestra cómo es el cliente, la sección superior es la de planificación y la sección inferior es la de tracking. La sección de planificación permite la priorización de tareas y funcionalidades arrastrando un elemento a la posición deseada (lo más prioritario es lo que esta más arriba en el árbol). Permite visualizar el

estado de cada elemento su estimación, el tiempo trabajado y el tiempo restante, además permite editar cualquier elemento en el mismo árbol simplemente haciendo doble click en la propiedad a editar.

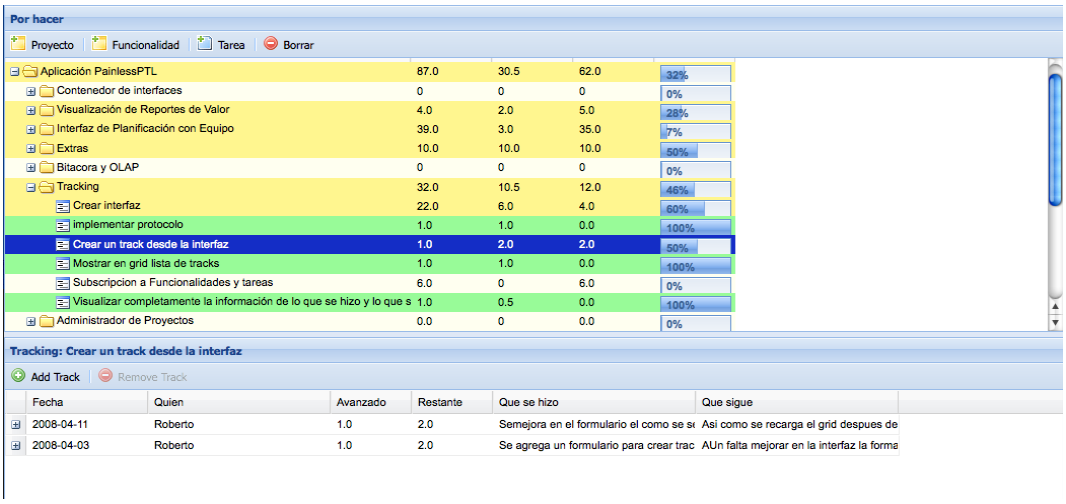


Figura 24: Vista Completa de la Herramienta

La creación de funcionalidades y tareas se realiza presionando los botones de la barra superior, los que presentan una ventana con el formulario a llenar para lograr la creación del elemento.

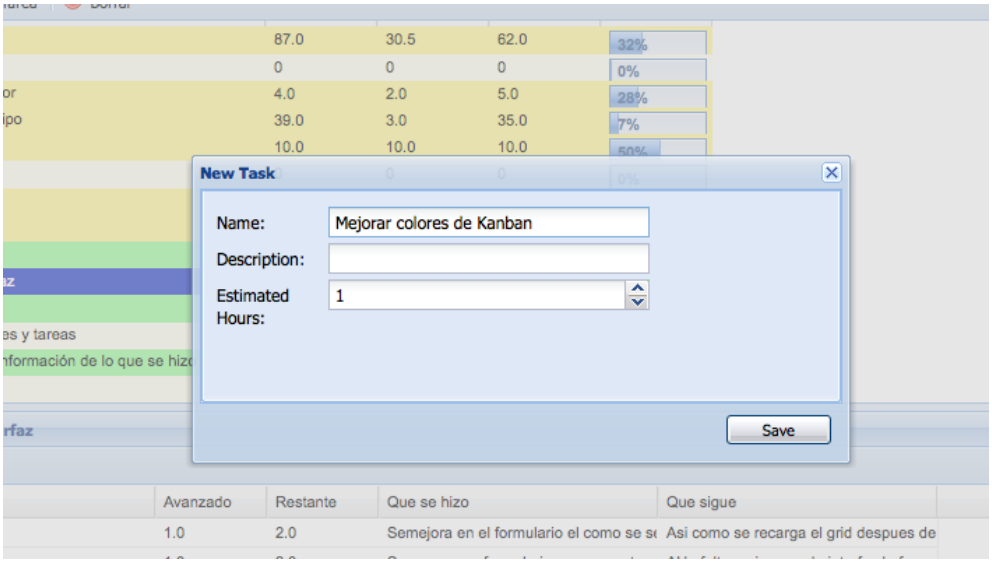


Figura 25: Creación de un Elemento de Planificación

El tracking es realizado sobre los elementos de planificación. Para esto se debe seleccionar del árbol alguno de los elementos y luego presionar el botón para crear un track, el que despliega el formulario que debe ser completado para agregar ese track. Su creación implica actualización del

plan, por ello inmediatamente después de creado los cambios en el plan se hacen visibles.

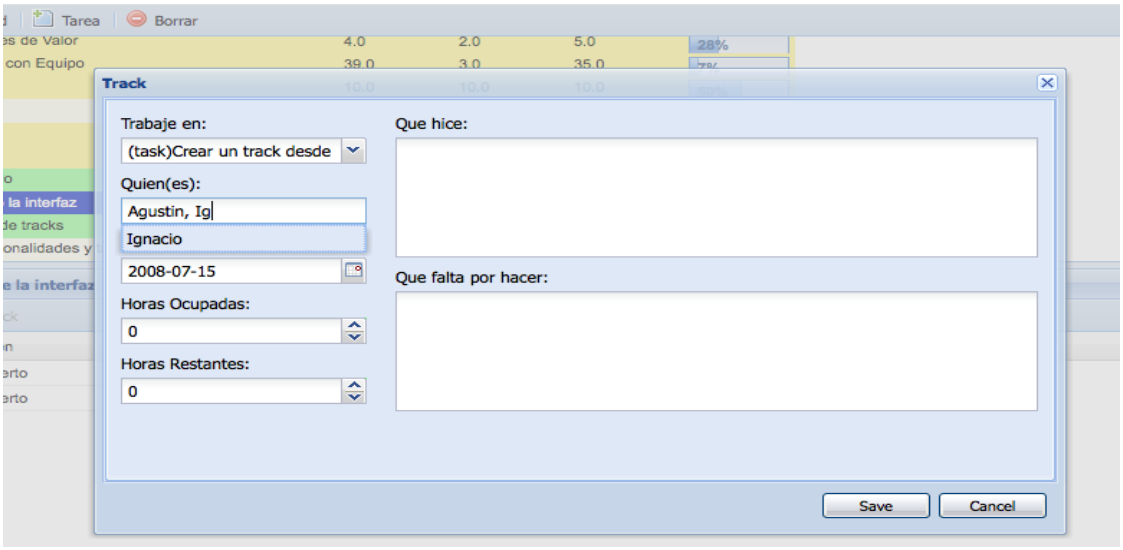


Figura 26: Tracking a un elemento de Planificación

La herramienta implementada facilita la planificación: al mantener organizadas las tareas por funcionalidades, agregar la visión del avance tanto en números como en colores, modifica la forma de priorizar en base a un orden absoluto y que la visión de árbol permite administrar naturalmente. El tracking además agrega la posibilidad de incluir más de un usuario en esa información. Además se incorporó la capacidad de registrar tracking de tareas no planificadas que tienen que ver tanto con el proyecto o con una funcionalidad o con nada, esto porque existen tareas que se realizan en la práctica que no siempre serán planificadas de antemano como problemas, reuniones, etc. y estas no necesariamente están relacionadas a funcionalidades o proyectos, pero igualmente se quieren registrar.

5.2.4. Evaluación de la Herramienta

En el momento en que la herramienta fue terminada, el equipo de Microsystem ya había dejado de utilizar la herramienta basada en Excel y estaba más concentrado en el plano productivo más que en el estratégico (donde la herramienta tiene mejor desempeño). Pero igualmente fue evaluada para determinar qué lleva a un equipo a dejar de utilizar la herramienta, para desarrollar mejoras que permitan fomentar el uso.

Si sólo se considera a la herramienta en el análisis de la gestión del equipo (esto es dejando de lado los conocimientos de gestión ágil del equipo) entonces, al igual que en la versión Excel, si el

plan es muy grande, es complicado saber en qué estaba, estoy o puedo trabajar. El color y la barra de avance son un aporte pero no son suficientes para permitir focalizar al equipo. Incluir filtros que permitan disminuir la cantidad de información desplegada en el árbol sería una forma de solucionar el problema. Otro elemento que puede ser mejorado es proveer un formulario que permita escribir de manera más correcta una funcionalidad, ya que este es un concepto complicado de entender pero existen varias sugerencias de formulario para guiar la definición de funcionalidades.

El equipo hace ver también la falta de apoyo que entrega la herramienta cuando se enfrentan a múltiples proyectos, aunque la herramienta excel y la web se pueden modificar para soportarlo, la simple modificación no apoyará gestión del equipo, por ello se requiere un diseño apropiado para el apoyo a la gestión de varios proyectos en forma ágil.

6. Apoyo a la Adopción Ágil en un Equipo

En el presente capítulo se documenta la experiencia obtenida de aplicar el modelo de adopción de gestión ágil sobre el equipo de I+D de la empresa Microsystem.

6.1. Modelo de Evaluación Inicial Aplicado

Siguiendo el modelo del capítulo anterior, siempre se debe comenzar evaluando al equipo para determinar el nivel de agilidad actual, para así definir un camino más concreto y dirigido a la obtención de prácticas.

Al equipo se le entregaron tarjetas con las características ágiles presentadas en las tablas 3 y 4, las que separaron en 3 grupos: No se realizan, Se realizan y no se quiere hacer. Este último grupo nace por la existencia de dos características adicionales que aunque no son negativas pueden ser realizadas en equipos que quieren hacer su proceso de adopción más paulatino y justamente en ese equipo no eran deseadas.

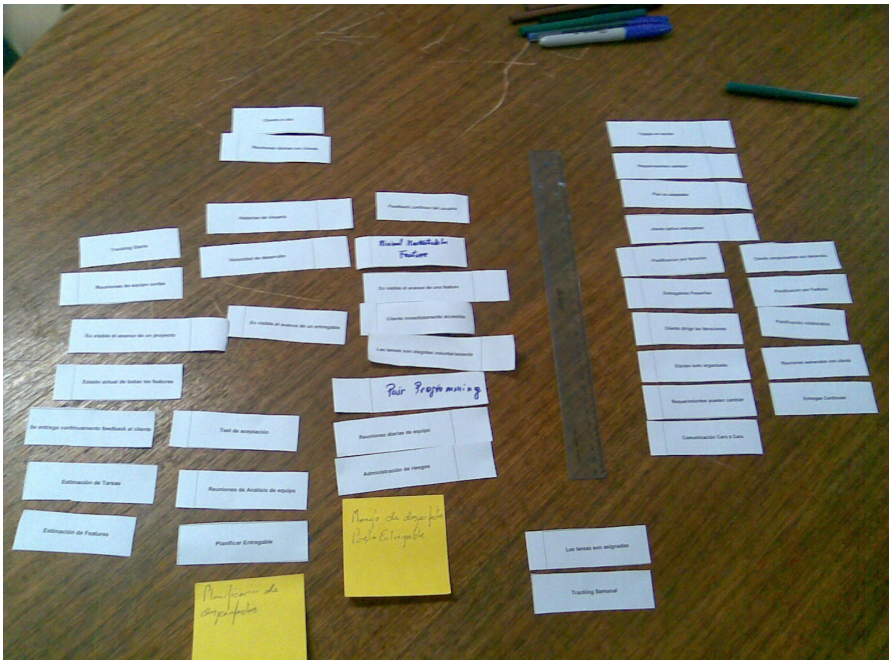


Figura 27: Vista Completa de Clasificación

En las figuras 27 y 28 se muestra este trabajo de clasificación realizado por el equipo. En este

proceso se agregaron 4 nuevas características: Minimal Marketable Feature, Pair Programming, Manejo de Desperfectos y Planificación de Desperfectos. Pair Programming es una práctica no relacionada con la gestión ágil pero el equipo cree que esta es una práctica que se debe recuperar.

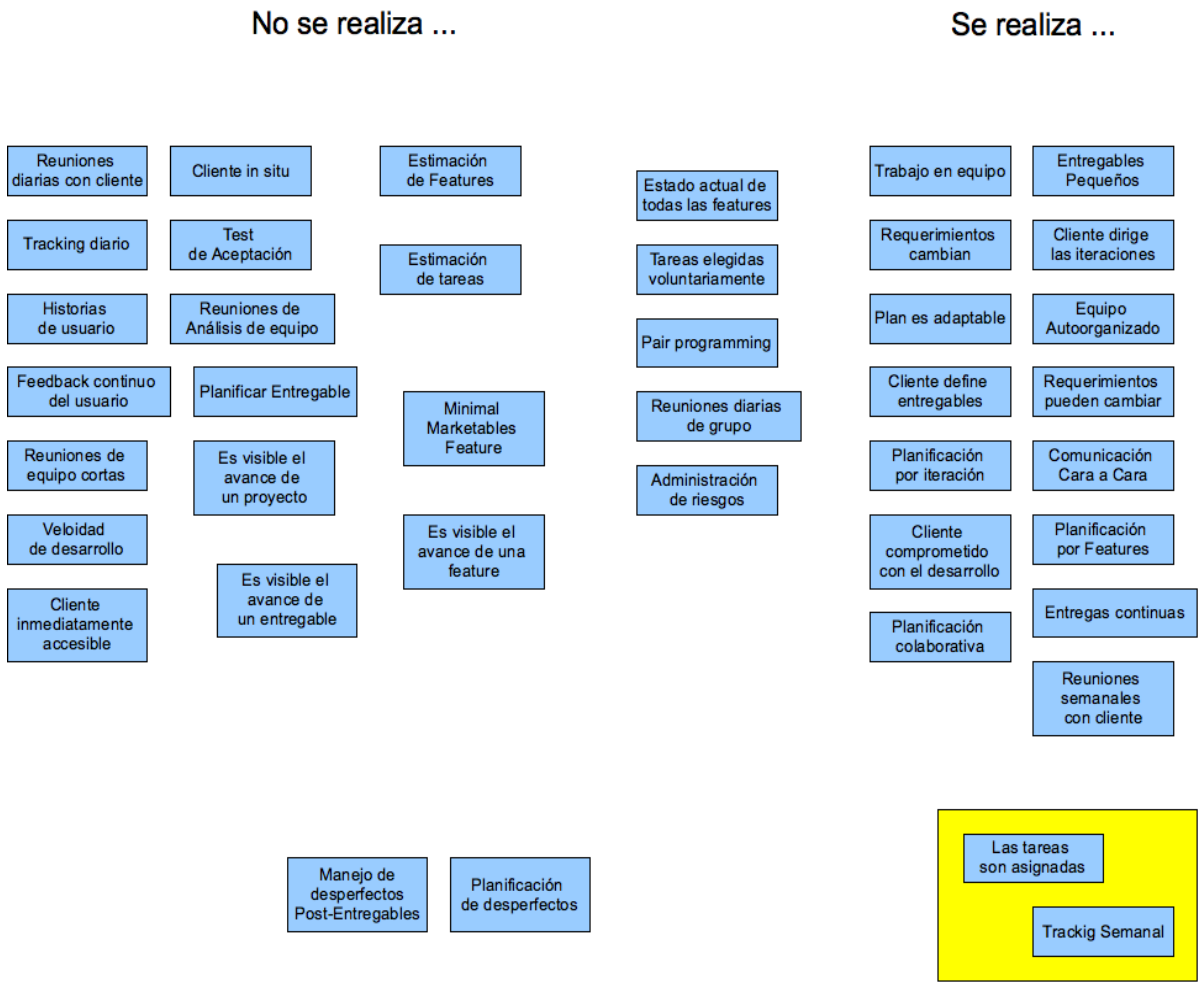


Figura 28: Vista Completa de Clasificación (digital)

El sistema de evaluación fue rápido y dio al equipo la oportunidad de reflexionar sobre sus actuales problemas y cómo podrían enfrentarlos.

6.2. Camino para adoptar agilidad deseada

6.2.1. Resultado y Recomendaciones Post-Evaluación Inicial

Las siguientes tablas muestran en resumen el nivel de agilidad del equipo por las características

que sí poseen y cuáles son las características que se seleccionaron para aumentar la agilidad del equipo. Las características que sí poseen están tachadas y las que esperan lograr están subrayadas.

	Cliente in Situ	Planning Game	Test de Aceptación	Entregables Pequeños
Nivel 1	~~Reunión semanal con cliente~~ ~~Cliente define entregables~~ *Es visible el avance de un entregable*	*Estimación de Funcionalidades* *Planificar Entregable* ~~Planificación por Features~~ Historias de Usuario Estimación de Tareas	*Test de Aceptación*	~~Entregables Pequeños~~
Nivel 2	~~Cliente dirige las iteraciones con equipo~~ ~~Cliente comprometido con el desarrollo~~ Es visible el avance de una funcionalidad Es visible el avance de un proyecto	MMF Administración de Riesgos ~~Requerimientos cambian~~ ~~Requerimientos pueden cambiar~~ ~~Planificación colaborativa~~		~~Entregas Continuas~~
Nivel 3	Reuniones diarias con el cliente Cliente inmediatamente accesible Estado actual de todas las funcionalidades	Manejo de desperfectos Planificación de Desperfectos ~~Plan es adaptable~~		Feedback continuo de los usuarios

Tabla 6: Evaluación aplicada sobre Gestión del Proyecto

	Ritmo Sostenido	Liderazgo Motivador	Retroalimentación de Avance	Espacio de Trabajo Informativo	Stand Up Meeting
Nivel 1	*Velocidad de Desarrollo*	*Reuniones de Análisis de Equipo* ~~Comunicación Cara a Cara~~	*Tracking diario*		*Reuniones diarias de equipo*
Nivel 2	~~Planificación por Iteración~~	~~Trabajo en Equipo~~ Tareas elegidas voluntariamente			Reuniones de equipo cortas
Nivel 3		~~Equipo Autoorganizado~~			

Tabla 7: Evaluación aplicada sobre Gestión del Desarrollo

Al equipo, como resultado de la evaluación, se le entregan las siguientes recomendaciones:

- **Deben ordenar la forma en que planifican, ya que es muy fácil tener caos**. Esto porque la falta de adecuada planificación ha hecho que el entregar sea algo poco claro y valioso.
- **Análisis semanal de equipo**. Este orientado a revisar su agilidad.
- Programación de Pares.

El objetivo de estas recomendaciones es: **"Mejora la forma de Gestionar el Desarrollo del Equipo"**. Esto es una aplicación directa de lo que Cockburn sugiere [24] como partida para un

equipo. La idea es que ordenen su trabajo orientado a producir el mayor valor en el corto plazo, y se les aleja del plano más estratégico para focalizar sus esfuerzos en generar entregables. Ahora la planificación debiese ser de un par de semanas y no de meses, y así la visibilidad del avance pueda ser notoria para el equipo. La figura 29 muestra gráficamente los objetivos del equipo.

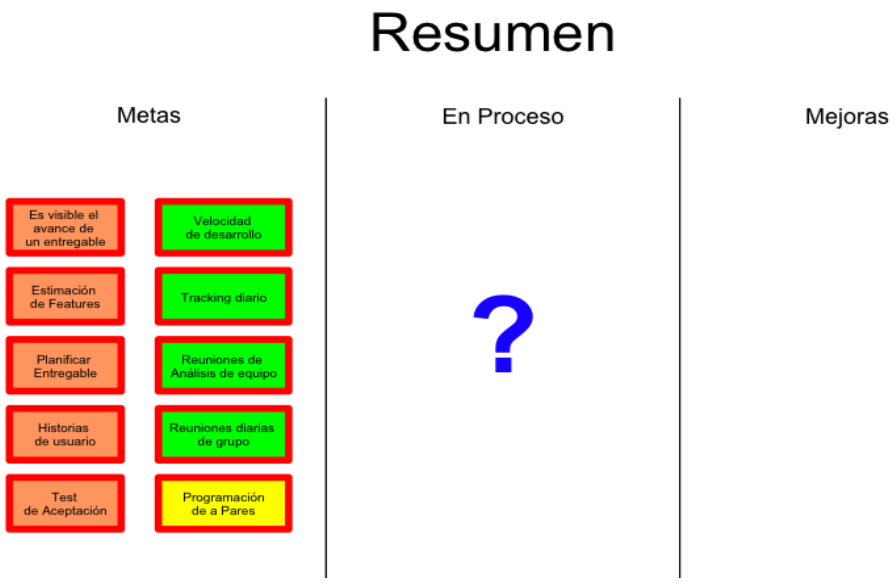


Figura 29: Kanban de Características por Lograr

6.2.2. Propuestas para la herramienta

Del análisis con el equipo se concluyeron los siguientes puntos que están relacionados con gestión ágil y las posibles mejoras que la herramienta debiese incorporar:

- la gestión ágil consta tanto de una parte estratégica como de una parte productiva, esta última es a la que se ven enfrentados diariamente los desarrolladores y debe ser apoyada de manera específica.
- la ausencia de entregables e iteraciones hace perder la sincronización del equipo con el problema de negocio a resolver, por esta ausencia es que entregar se volvió algo poco claro y frecuente.
- la definición de tareas de antemano es muy cara de administrar, las tareas son lo que más puede cambiar en un plan, el detalle al que se puede llegar en su definición en las horas de planificación inicial produce más incertidumbre, por lo tanto, es recomendable eliminarla

y sólo dejarla como una forma de organizar los elementos que debe tener la funcionalidad para ser completada. Otra cosa que se pudo concluir es que el tracking sobre la tarea no tiene sentido, al cliente no le interesa saber cuánto resta en la tarea sino cuanto resta en su funcionalidad. Por ello se sugiere la eliminación de tareas del plan.

6.3. Nuevo Diseño para la Herramienta

De la investigación realizada se confecciona una lista de principios generales de diseño para la herramienta, que buscan convertir a la herramienta en un soporte para la adopción de gestión ágil, estos son:

- Aumentar el involucramiento del cliente a través de la entrega de información valiosa y frecuente, así la información será vista por el cliente como una recompensa.
- Incentivar más principios que prácticas.
- Lo importante en un equipo es mantener y mejorar el ritmo.
- Que el cliente esté más involucrado no debe volverlo más técnico.
- El equipo se debe preguntar constantemente el cómo ser más ágil.
- La utilización de la herramienta debe ser indolora.
- La gestión ágil no sólo se mueve en el plano estratégico sino en el productivo, por lo tanto, el segundo plano debe ser también gestionado.

Considerando tanto la evaluación inicial del equipo, las sugerencias de este y la investigación realizada, la herramienta debiese contener al menos los siguientes elementos:

- Gestión del Proyecto según el principio de Planning Game.
- Gestión del Equipo según el principio de Planning Game de Iteración.
- Métricas que apoyen a la gestión del proyecto.
- Métricas que apoyen a la gestión del desarrollo.
- Métricas que apoyen la adopción.

Las versiones anteriores de la herramienta ya consideraban algunos de estos elementos, pero no en profundidad y tan explícitos, estos elementos van en línea con el cumplimiento de los

objetivos antes propuestos.

6.4. Modelo de Gestión Multiproyectos

La herramienta Painless Tracking nació como apoyo a la administración de sólo un proyecto a la vez, por lo que su introducción en un ambiente multiproyectos, como es la realidad de varias empresas de software chileno, en especial el equipo de I+D de Microsystem, permitió ver que no estaba preparada para el salto a la industria. El siguiente diagrama presenta la realidad actual que se tiene en algunas empresas relacionadas a la industria del software, y pudo ser vista en varias de las empresas y equipos con los que se tuvo contacto:

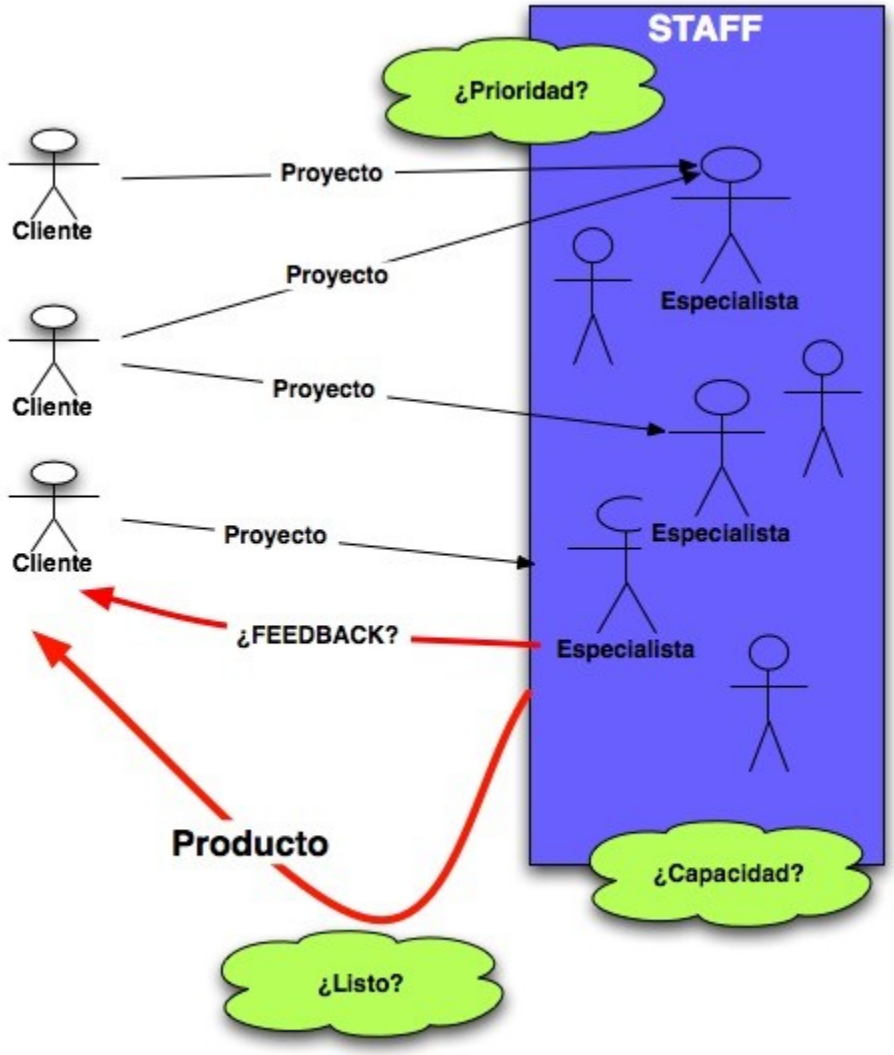


Figura 30: Gestión del Caos

El diagrama, llamado “Gestión del Caos”, muestra exageradamente la realidad actual, y en él resaltan los siguientes 5 puntos:

- **Prioridad: ¿cómo es controlada la prioridad?**, generalmente los proyectos entran al grupo de desarrollo, pero la prioridad no está definida, no se sabe qué proyecto entrega más valor y es común ver a personas intentando terminar varios proyectos al mismo tiempo. No existe un filtro para los proyectos y no se define claramente cuándo los proyectos debiesen estar entregados. Aquí aparece otra característica de este modelo los ***proyectos se desarrollan en paralelo.***
- **Capacidad: ¿En el tiempo comprometido el grupo podrá terminar los proyectos?**, raramente se hace una evaluación y nunca es calculada la capacidad para desarrollar que tiene el equipo, por lo que responder la pregunta es imposible.
- **Terminado: ¿El producto está terminado y listo para ser entregado?,** no siempre es realizada una detallada especificación de requisitos que implique conocer claramente el alcance de un proyecto.
- **Estado de Avance. ¿informo al cliente del avance del proyecto?,** se ha intentado introducir herramientas que faciliten esto pero el entregar información al cliente que diga cómo el proyecto avanza no es una práctica utilizada, además como no se sabe qué es lo que se entregará definir avance no tendría sentido.
- **Especialistas. ¿quién puede desarrollar el proyecto?**, aquí el grupo tiene gente preparada para realizar labores específicas de desarrollo, por lo que varios proyectos pueden estar asignados a la misma persona porque sólo él puede realizar una tarea aunque el grupo tenga otras personas disponibles.

Como una forma de responder a este modelo de gestión se propone el siguiente modelo para permitir la adecuada gestión de varios proyectos, responder a la realidad conocida de la industria y permitir la incorporación de gestión ágil en un equipo de desarrollo, el siguiente diagrama explica esto:

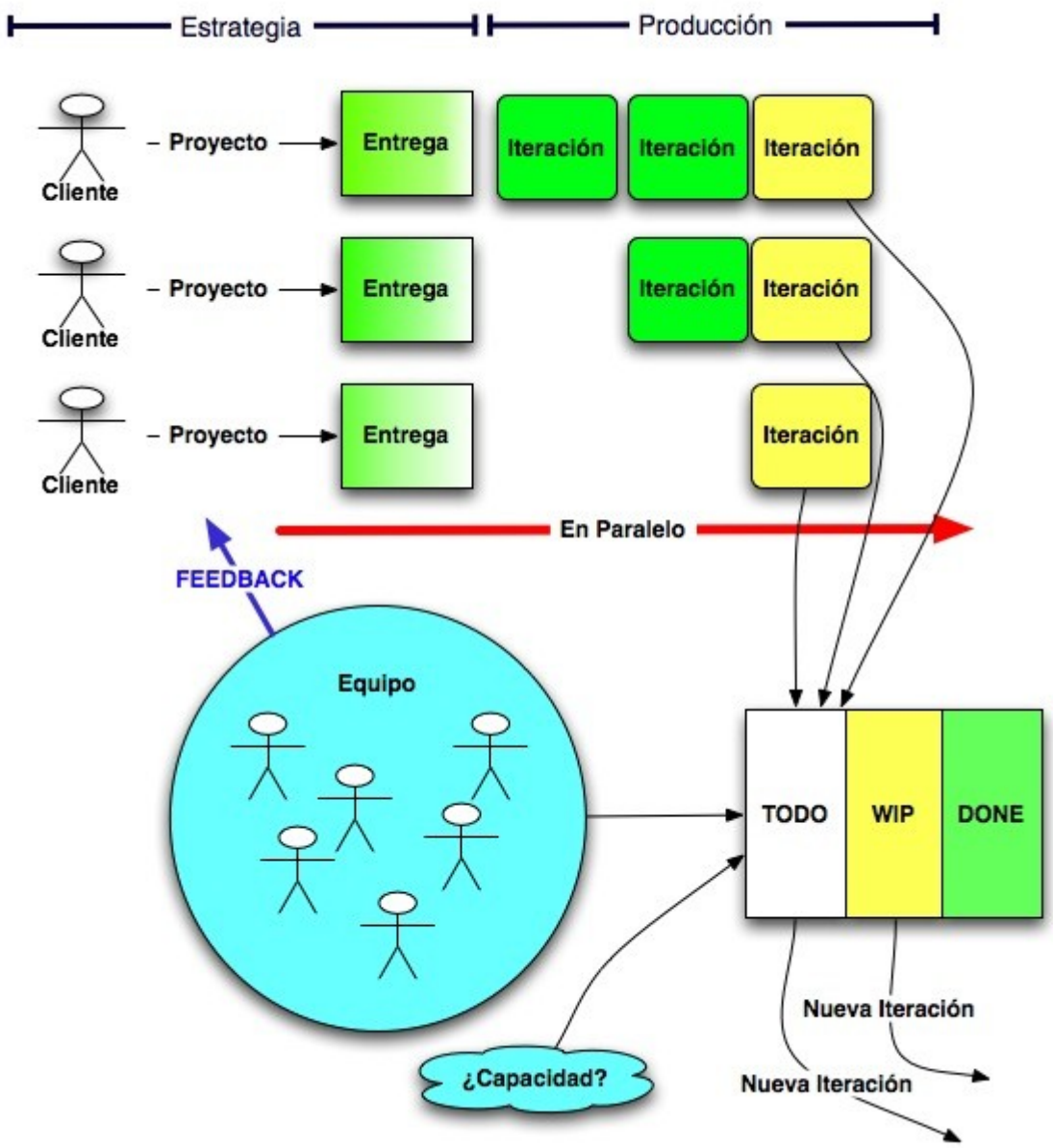


Figura 31: Gestión Ágil de Múltiples Proyectos

El modelo incorpora todos los elementos de la gestión ágil estudiados, y estos se sincronizan de la siguiente forma:

1. Definición de Entregables: el plano estratégico es el inicial en un proyecto de desarrollo, en él tanto cliente como equipo convierten una serie de funcionalidades para el sistema en conjuntos bien definidos llamados "Entregables" (Releases), con ellos es posible acotar el alcance.
2. Definición de Iteraciones: luego en el plano productivo, el equipo escoge según la

prioridad que el cliente puso a cada una de la funcionalidades del release, un conjunto más pequeño llamado iteración (Iteration), que les permita en un plazo de tiempo definido y más corto desarrollar guiado por el valor las funcionalidades del release, en este modelo la negociación y la información que se pueda entregar al cliente son claves, porque en el desarrollo de esta fase es posible determinar y precisar cuales funcionalidades son las que realmente el cliente necesita, con esto el alcance puede ser mejorado y el valor entregado siempre maximizado. Al término de la iteración las funcionalidades a medio terminar y las que no nunca se lograron empezar son devueltas y puestas para la siguiente iteración.

En este modelo la capacidad tiene tres formas de ser definida, estas son:

- acotándola, es decir, definiendo un número máximo de funcionalidades que realmente se cree que puede ser realizadas.
- estimándola, dado que la funcionalidades tienen un tiempo estimado de desarrollo y la iteración tienen un plazo de tiempo definido, la iteración define cual es la capacidad del equipo y las funcionalidades que en este se desarrollen deben estar dentro del tiempo de la iteración.
- calculándola, luego de realizada una iteración es posible determinar la capacidad real del equipo y este cálculo puede ser utilizado en la definición de las nuevas iteraciones.

La entrega de información de avance fluye por el sistema y el cliente puede saber en todo momento como avanza su proyecto. Además, permite que en el equipo las personas no necesariamente sean especialistas, buscando que el conocimiento sea compartido y así permitir el desarrollo de varios proyectos en simultáneo.

6.5. Desarrollo Painless Tracking Web Final

En lo siguiente se muestra la herramienta en su versión final y se explican los principios y prácticas, en detalle, que la herramienta intenta promover. En esta herramienta se pueden distinguir los dos planos del desarrollo ágil (el estratégico y el productivo), esta división no era evidente en la herramienta basada en Excel por esto en la nueva versión se hice explícito debido a la necesidad de dar mayor claridad de cómo la herramienta funciona y que sean más sencillos de aplicar los principios y prácticas de gestión ágil contenidos en ella.

6.5.1. Planning Game del Entregable

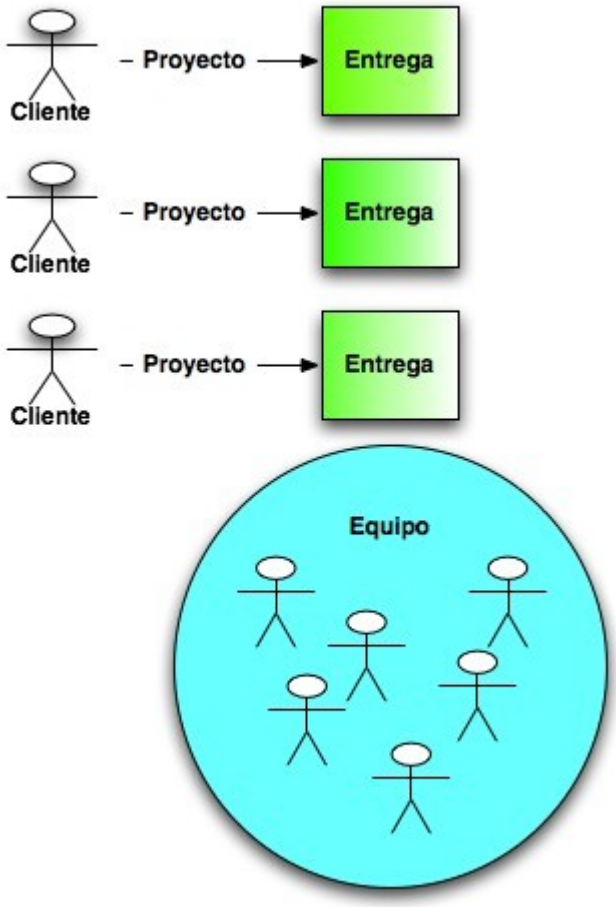


Figura 32: Release Planning para varios proyectos

La forma en que Extreme Programming negocia y define con el cliente el alcance de un proyecto es a través del planificación de entregables usando el Planning Game, en el modelo de múltiples proyectos antes presentado, el Planning Game debe ser aplicado como si se tratase de un solo proyecto. La siguiente figura muestra cómo Extreme Programming define el Planning Game de un entregable.

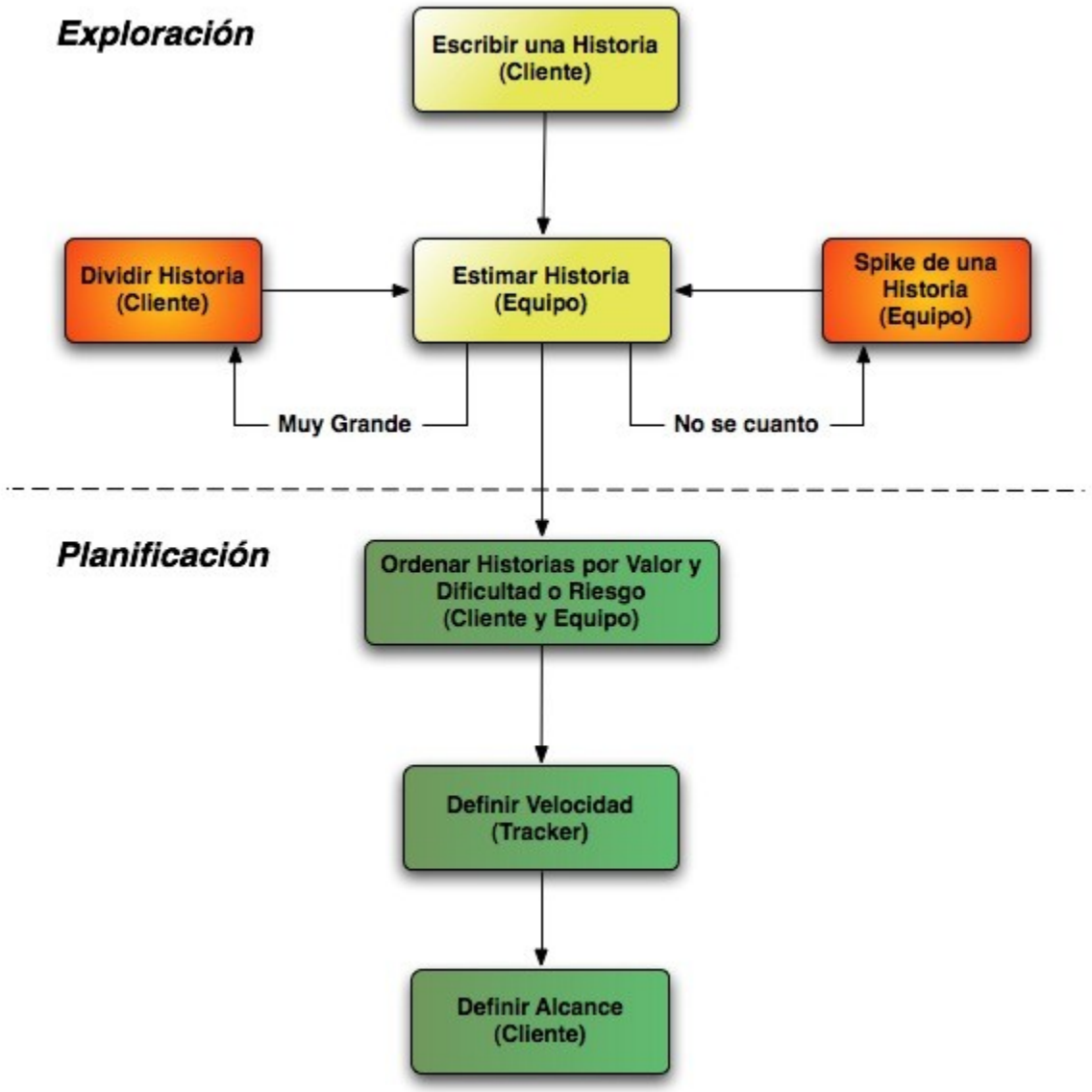


Figura 33: Extreme Programming. Release Planning Game

Siguiendo el Planning Game, el cliente web de la herramienta permite visualizar varios proyectos a la vez y los entregables definidos. La figura 34 muestra el árbol de planificación que realiza las mismas funcionalidades de la herramienta antes implementada; la diferencia con la anterior esta en la ausencia de la interfaz de tracking, la eliminación del elemento tarea y la incorporación del elemento entregable (Release).

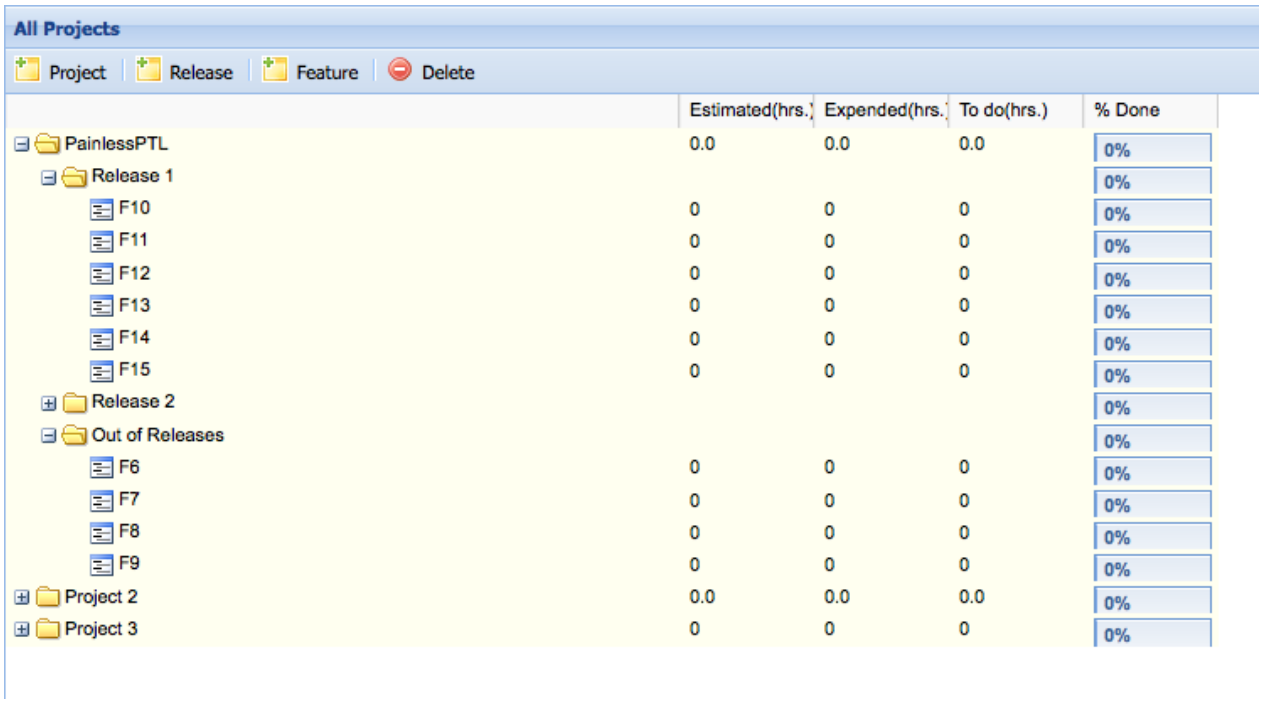


Figura 34: Planning View(Vista de Planificación)

Al definir un entregable se debe definir la velocidad que será utilizada durante el desarrollo de este y sólo podrán ser utilizadas las funcionalidades que tengan definida la estimación.

6.5.2. Planning Game de la Iteración

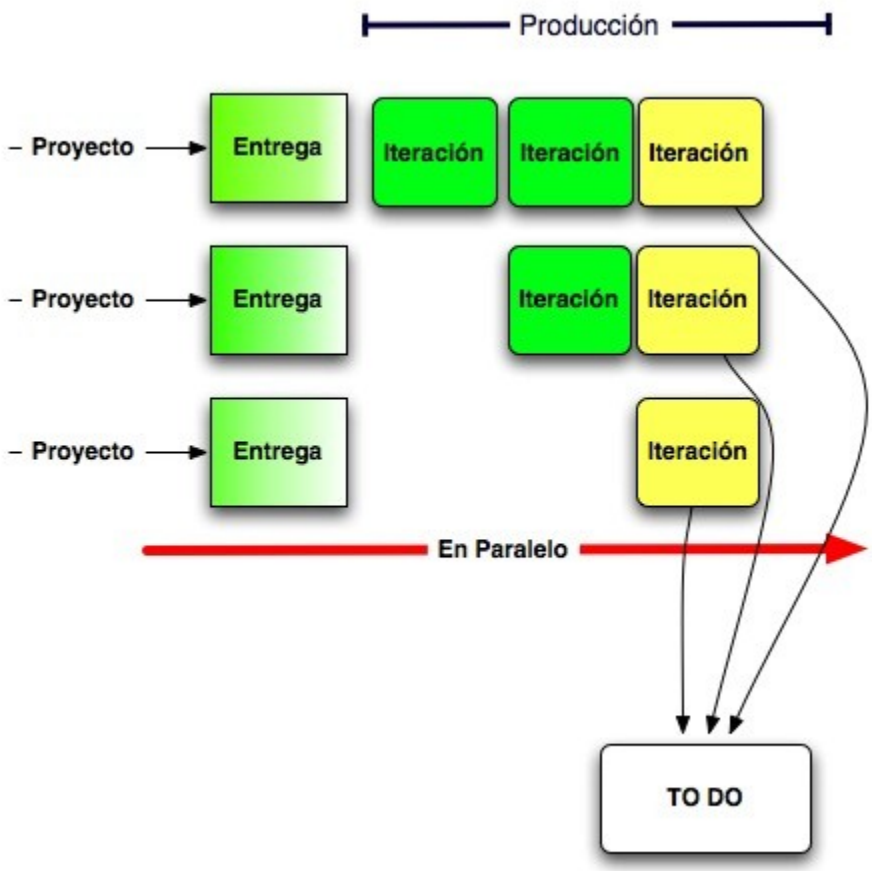


Figura 35: Iteration Planning para varios proyectos

Antes de pasar a la fase de producción el equipo utiliza la definición de iteraciones como una medida de disminuir la incertidumbre y procurar la entrega temprana de software al cliente para que este lo valide. Las iteraciones añaden al equipo un tiempo fijo para utilizar en el desarrollo de funcionalidades y obligan al cliente a priorizar con mayor detalle las funcionalidades a realizar si este desea modificar el plan actual, y permiten que el equipo pueda definir claramente cuál es su capacidad y luego demostrarla.

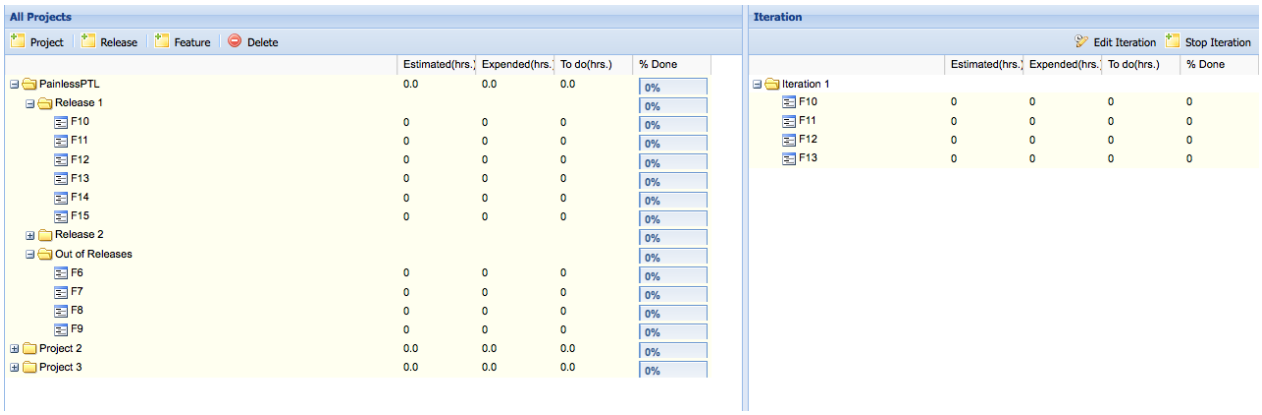


Figura 36: Iteration Planning View(Vista de Plan de Iteración)

En el cliente se implementa la fase de definición de la iteración incorporándola al árbol de planificación del release, como muestra la figura 36. La definición de una iteración requiere la definición de cuánto tiempo se ocupará en el desarrollo de este. Con esta definición es posible arrastrar funcionalidades del plan a la iteración y así definir una iteración para el equipo. Cada proyecto siempre tendrá una iteración activa, esta iteración termina cuando la detienen o cuando finaliza el tiempo para el que esta ha sido definida. La prioridad de las funcionalidades de la iteración está determinada por la prioridad en el plan general, no existe una prioridad especial en la iteración.

6.5.3. Fase de Producción de software

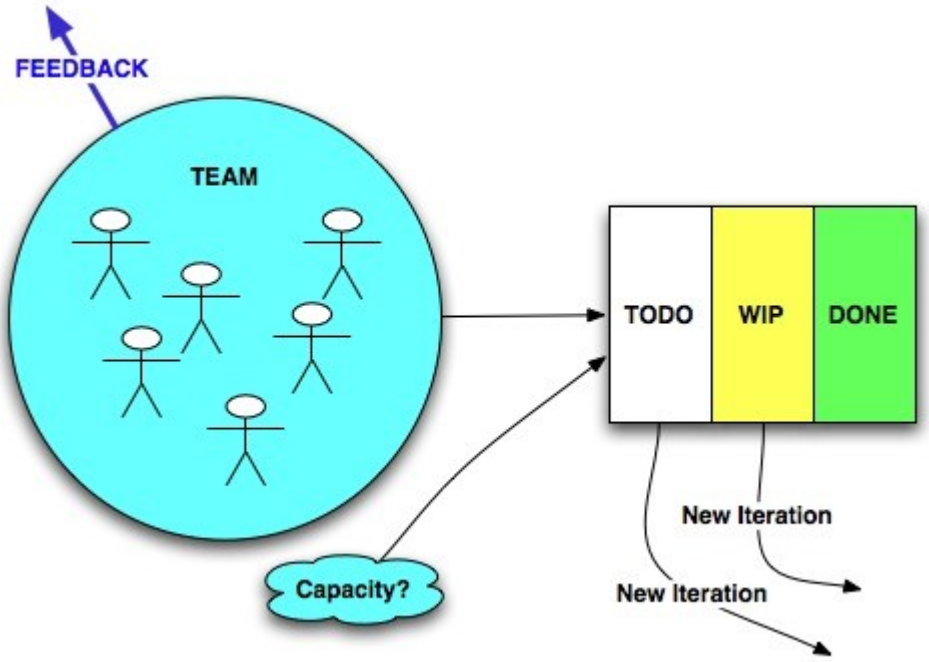


Figura 37: Fase Producción de Software

La fase de producción de software (Figura 37) es la que asegura 3 puntos claves:

- Facilitar la adquisición del ritmo de desarrollo
- Gestionar la capacidad
- Concentrar al equipo en la entrega de valor

Estos puntos se intentan lograr a través de la incorporación de 2 conceptos que han sido traducidos a herramientas, que son: el Kanban[9] y la Iceberg List[25].

6.5.3.1. El Kanban

La gestión de producción "Just in Time"[9] es caracterizada por gestionar el esfuerzo en el desarrollo con miras a la rápida y fluida entrega de productos. En este contexto aparecen herramientas o ayudas visuales que permiten a grandes equipos ordenar el trabajo y responder a la demanda evitando que el trabajo se aglomere y así desaparezcan cuellos de botella o la

incertidumbre de cuándo algo será terminado. Una de estas herramientas es el “Kanban” (del Japonés, tarjeta o señal), es un tablero donde se especifican las distintas fase de un proceso productivo y donde se ve el movimiento entre fases de tarjetas que identifican ordenes de trabajo. Este tablero es visible para todos y permite que todos sepan que estan haciendo, donde hay problemas y que puede continuar haciendo alguien cuando ha terminado su trabajo. A continuación y en términos generales, se explica como es aplicado al desarrollo de software[9]:

Se divide el trabajo en 3 estados (al menos).Estos son: lo que hay que hacer (To do), lo que se está haciendo (Doing) y lo que está listo (Done). Las tareas fluyen por la herramienta entre estados permitiendo de manera clara al equipo notar cuál es el avance del proyecto. Lo valioso para un equipo es todo lo que se encuentra en la columna de lo que está listo, ya que terminar es el objetivo.

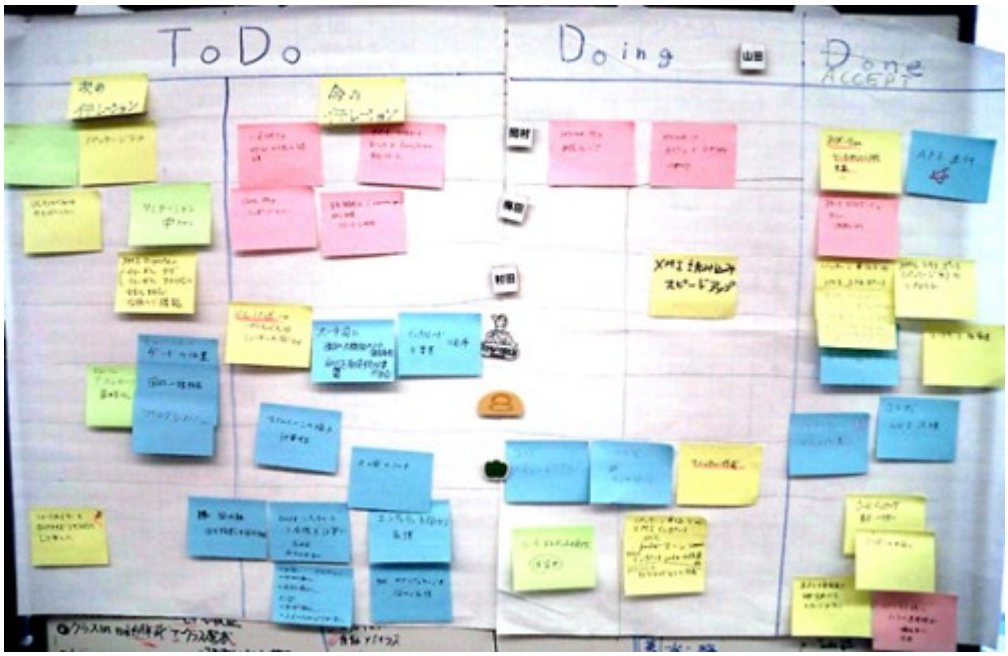


Figura 38: Kanban

En un ambiente donde la entrada de trabajo es continua y a veces sin filtro por el desconocimiento de la **capacidad** del equipo, este sistema permite visualizar y gestionar el avance del trabajo a realizar dado que la mira está en lo terminado, por ello el cliente o quien pida se verá obligado a incluir trabajo pensando en que este pueda ser terminado y no detenga el trabajo del equipo. En este sistema además el equipo sabe exactamente qué es lo que debe realizar en todo momento. Por ese lado no existen tiempos muertos por decidir que hacer luego de terminada una tarea, es decir, no se requiere de un jefe que diga al equipo qué hacer después el

Kanban muestra claramente cuáles son los pasos a seguir, así los jefes pueden ser cambiados por líderes que promuevan y faciliten el trabajo.

6.5.3.2. La Iceberg List

El concepto "Iceberg List" [25] nace por las reglas que deben aplicarse en la gestión de funcionalidades en analogía con un iceberg. En proyectos donde el tiempo es fijo la punta del iceberg o lo que está arriba del agua son las funcionalidades que el equipo realmente puede realizar dado el tiempo que resta. Así todo lo que está debajo del agua es lo que aún no puede ser realizado por el equipo dado los tiempos definidos. Cuando nuevas funcionalidades son agregadas a la lista de funcionalidades por realizar, por el peso que esa funcionalidad tenga naturalmente desplazará bajo el agua algunas funcionalidades y por lo tanto no podrán ser implementadas. Con esto el equipo y clientes pueden administrar de manera ordenada el cambio frecuente de requerimientos, y se obligan constantemente repriorizar y estimar cual es el valor a entregar.

6.5.3.3. Producción de Software en la Painless Tracking

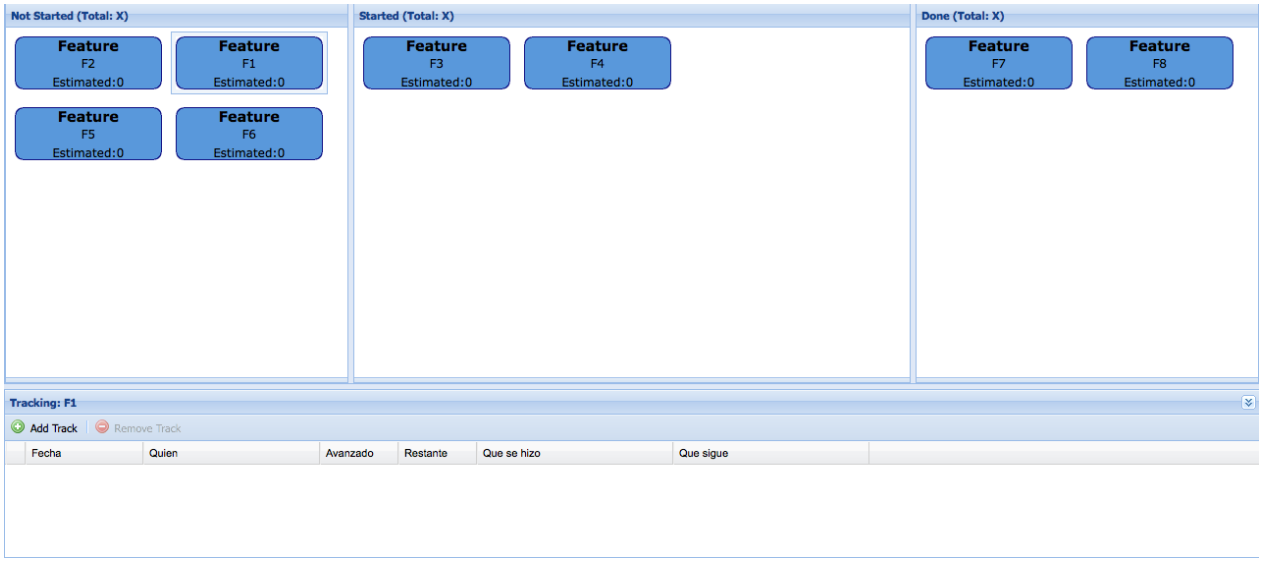


Figura 39: Development View (Vista de Desarrollo)

La herramienta considera los dos conceptos antes explicados, agrega el kanban para que el equipo administre las iteraciones y en esta incorpora el tracking, además el trabajo No iniciado (Not started) siempre mantiene en la lista las funcionalidades que sí pueden ser realizadas siguiendo los principios del iceberg list.

Con esta interfaz se recupera la capacidad de tracking indoloro que anteriormente ya tenía la herramienta basada en Excel y también se gana en mayor foco y visibilidad del avance actual del desarrollo. Con esto la administración diaria, la actualización del plan y el tracking se hacen simples y efectivos, sin perder de vista el entregable.

6.5.4. Métricas

La herramienta en su diseño ya tiene incorporado el apoyo de gestión ágil, pero para asegurar esto es que se proponen las siguientes métricas:

- Tiempo efectivo entre Entregables: esta consiste en el promedio de tiempo en que ocurre un entregable. Este tiempo puede ser clasificado por cliente, por proyecto o dar un número general. Con esto se busca apoyar al aumento de entregables.
- Número y Tiempo efectivo entre Iteraciones: el promedio de iteraciones efectuadas por entregable es un número que indica cuánto se está disminuyendo la incertidumbre y procurando entregar mayor valor más temprano, la madurez del equipo está relacionada directamente con la cantidad de iteraciones que logra hacer en el tiempo, o por la cantidad de entregables si estos son pequeños. También el promedio de tiempo a ocupar por iteración es una buena guía en la adopción equivalente a cómo es analizada en la métrica de tiempo para entregables.
- Número y Tiempo efectivo de Funcionalidades terminadas por día: el promedio de cuánto tarda una funcionalidad en estar terminada en la iteración por día y cuántas tardan en estar terminadas por día, para guiar la reorganización del equipo a diario y no permitir que el desarrollo se vea detenido. Esta métrica es similar a la velocidad pero vista desde otra perspectiva.
- Velocidad de Tracking: con esto se mide que cuántas horas por día efectivamente registra el usuario, lo ideal es 8 horas diarias, pero un usuario que no tiene bien adoptada la práctica, la velocidad puede ser mayor. En general, debiese esta velocidad bajar y estar siempre cercana a 8, casos menores que 8 implican que el usuario registra más seguido qué es lo que hace.

Conviene aclara que las dos primeras métricas pueden ser vistas como la velocidad de entrega de valor. Además, son incluidas las métricas propuestas cuando se usa un Kanban que son: Lead Time y Cicle Time[9]. Lead Time es el tiempo que trancurre desde que un cliente pide algo hasta

que este es entregado, en la herramienta, es el tiempo desde que se define un entregable(release) hasta que este es aceptado por el cliente. Cicle Time es el tiempo entre que una orden de trabajo es tomada por un equipo y cuando esta es entregada al cliente, en la herramienta es, el tiempo entre que el entregable se comienza a trabajar y se entrega al cliente. Estas dos métricas permiten visualizar cuan efectiva y rápida es la entrega al cliente. La diferencia entre estas dos métricas indica cuanto tiempo el equipo tarda en tomar una petición de trabajo, este tiempo muerto es indicador de una mala gestión de la capacidad.

Sin ser métricas se agregan los siguientes indicadores:

- Antigüedad de una Funcionalidad: esta medida permite mostrar cuánta funcionalidad planificada aún no se trabaja en ella.
- Número de horas trabajadas en funcionalidades no estimadas por proyecto.
- Número horas en Funcionalidades sobreestimadas y subestimadas por proyecto.
- Número de cambios realizados en el plan

6.6. Evaluación de la Herramienta

La simplicidad y la claridad son los aspectos más resaltados por los usuarios al utilizar la herramienta, aunque sí cabe destacar que esta claridad implica que también existan unos conocimientos mínimos sobre la metodología ágil Extreme Programming.

Los aspectos más destacados por los usuarios son:

- Aumenta la capacidad de autogestión, esto por la incorporación del Kanban.
- El tracking es más rápido y efectivo. Como este sólo se hace desde la interfaz de kanban los desarrolladores no tienen posibilidad de errar en qué es lo que han trabajado inclusive si el registro es hecho días después de realizado efectivamente el desarrollo.
- Planificación centrada en el valor. La separación entre lo que se quiere hacer de lo que efectivamente se hace permite que el cliente pueda ser involucrado directamente en el plan. Además, se recupera el concepto de entregable y se enfoca el desarrollo a este.

El nivel de agilidad actual del equipo no fue evaluada, por lo que no es posible contrastar estos comentarios con la evaluación, aunque si algunas de las metas de la figura 29 han sido alcanzadas, las que son:

- Es visible el avance de un entregable
- Velocidad de desarrollo
- Tracking diario
- Reuniones de análisis de equipo
- Reuniones diarias de grupo

7. CONCLUSIONES

7.1. Evaluación de la Herramienta

El trabajo realizado puede ser resumido en los siguientes pasos:

- Investigación sobre cómo se adoptan metodologías ágiles.
- Introducción y evaluación de la herramienta basada en Excel en una empresa.
- Contrucción de una nueva versión de la herramienta que solucionara en parte los problemas encontrados al evaluar inicialmente la herramienta.
- Definición y diseño de un modelo de adopción y evaluación de agilidad, basado en los modelos de Sidky y Villena.
- Mejorar la nueva versión de la herramienta siguiendo el modelo propuesto y nuevos aprendizajes adquiridos durante su utilización.

De todo este trabajo se concluyen los siguientes puntos:

- Al introducir una herramienta de gestión como la Painless Tracking en la realidad de una empresa,
- La razón fundamental de por qué una herramienta de gestión simple como la Painless Tracking no ha sido llevada a la industria es porque sus principios y prácticas son mucho más importantes que la herramienta.
- Aunque la adopción de gestión ágil sí puede ser apoyada por una herramienta pero sin descuidar que es más importante conocer y empaparse de los principios y prácticas de gestión ágil que contiene una herramienta, en especial la Painless Tracking.
- La figura de un consejero o guía en el proceso de adopción es algo que queda demostrado, esto es adelantado por el modelo de Sidky y las propuestas de los cursos CC62V y CC61A. Esto porque la experiencia del guía añade objetividad y tranquilidad al proceso de adopción que muchas veces puede ser un fracaso.
- Aunque la gestión ágil no busca hacer retrospectivas, un equipo ágil la necesita y es por esto que esa práctica no puede dejarse de lado y debe ser incorporada en un proceso de adopción, sin ella no se puede asegurar que el aprendizaje de agilidad en un equipo será

logrado. Con ella se logra favorecer el aprendizaje, la comunicación y la reflexión, y allí radica su importancia.

- La gestión ágil que no está centrada en los Entregables no es efectiva, tanto el cliente como el desarrollador deben enfocar sus esfuerzos a entregar el conjunto de funcionalidades que permitan producir valor.
- Un equipo en la correcta línea de adopción de gestión ágil debería destacarse por la cantidad y lo frecuente de sus entregables, la utilización de las iteraciones como un medio para disminuir incertidumbre y gestionar la capacidad, y la correcta definición de funcionalidades.
- Cuando las funcionalidades son correctamente definidas, estas aumentan de valor y simplifican el plan; esta es una de las razones de por qué las tareas desaparecieron del plan ya que estas no implican valor para el cliente y sólo pueden ser utilizadas como una forma de organizar el trabajo en pasos para el desarrollador y/o para ayudar al proceso de estimar. Notar que esto es sólo posible cuando el equipo entra en el proceso de mejorar sus funcionalidades hasta disminuirlas a elementos de valor indiscutible para el cliente.
- En un modelo de gestión ágil las clasificaciones por prioridad pierden sentido, sólo es necesario mantener ordenadas las funcionalidades y este orden debe ser la prioridad, ya que lo primero y prioritario es precisamente lo primero en una lista, esto porque el plan debe ser explícito y no implícito como ocurre cuando se manejan estas clasificaciones (en el caso de proyectos con muchas funcionalidades), clasificar las funcionalidades es común cuando se hace un gran diseño inicial que posee alta incertidumbre.
- Lo términos utilizados para definir los elementos en la herramienta deben ser claros, ya que la confusión que producen las palabras detiene un buen proceso de adopción.
- Otro aspecto importante notado durante el desarrollo del trabajo, es que un proyecto de software se desenvuelve en dos planos, el estratégico y el productivo, y estos deben ser tratados por separado para que los principios y prácticas de gestión ágil aparezcan de manera natural y los convierten en necesarios.
- Las prácticas más complejas de aprender son: la estimación de funcionalidades y la definición de funcionalidades, por lo que siempre se debe aconsejar obtenerlas ya en equipos más avanzados, esto en el contexto de una empresa.

- También en ese contexto, los equipos de desarrollo en la industria, perciben como más complejo el entender el aspecto de alcance variable que posee la gestión ágil. Pero este problema puede ser rápidamente abordado y solucionado dejando al equipo trabajar de la manera actual en que lo hacen pero agregando un sistema como el kanban para gestionar sus avances, luego de algunos días es claro que los términos valor y alcance variable cobran sentido para las personas y buscan una forma de mejorar su rendimiento, además pueden observar que en todo momento el trabajo realizado por ellos es voluntario y no impuesto, ya que eligen que hacer de la lista de cosas impuestas.

En resumen, el rol de la herramienta en el proceso de adopción es ayudar a las personas a realizar correctamente las prácticas, apoyar y estimular los principios, y monitorear los niveles de agilidad.

7.2. Logros

De la evaluación final se pueden destacar los siguientes logros:

- las nuevas funcionalidades, como el Kanban por ejemplo, apoyan la adopción de prácticas ágiles y permiten lograr prácticas más avanzadas en el equipo como la autogestión.
- aunque sólo algunas métricas pudieron ser apreciadas estas permiten evaluar que tan efectivo es el equipo, lo que también fortalece y anima al equipo a mantener sus prácticas adquiridas y obtener nuevas.

7.3. Trabajo Futuro

Aunque el trabajo experimentando fue extenso, el entender metodologías ágiles no es un tema rápido, del trabajo se puede estimar que la adopción de estas toma al menos 8 meses y teniendo reglas que permitan la rápida incorporación de las prácticas a el equipo de desarrollo, por ello se propone lo siguiente:

- Evaluar como la herramienta se comporta cuando se intenta adquirir prácticas de nivel 2 y 3 en un equipo y así determinar si es necesaria la modificación de esta a partir de los resultados.
- Investigar en profundidad los conceptos que apoyan que la gestión de múltiples proyectos puede ser posible y no es un invento reciente, de hecho esta se sustena en la teoría de

colas[9], la que no pudo ser desarrollada e investigada en extenso. Esta investigación debiese afectar el como la herramienta está actualmente construida y así mejorada.

- Determinar si la herramienta es útil y apoya la adopción en diferentes equipos de desarrollo, y/o que es necesario para que esto sea posible.
- Revisar si el modelo de evaluación propuesto es correcto cuando se evaluan equipos que ya tienen han alcanzado un nivel más maduro de agilidad, ya que este sólo fue evaluado en un equipo que estaba comenzando.
- Implementar reportes orientados al cliente que aprovechen las métricas desarrolladas.

8. Referencias y Bibliografía

[1] Sidky, Ahmed y Arthur, James. A Disciplined Approach to Adopting Agile Practices: The Agile Adoption Framework
http://arxiv.org/ftp/arxiv/papers/0704/0704.1294.pdf

[2] Villena, Agustín. Un Modelo Empírico de Enseñanza de las Metodologías Ágiles: el caso del curso CC62V-Taller de metodologías ágiles de desarrollo de software. Marzo 2008.

[3] Beck, Kent. Extreme Programming Explained
http://www.extremeprogramming.org/

[4] SCRUM Methodology
http://www.controlchaos.com/

[5] Crystal Methodology
http://alistair.cockburn.us/index.php/Crystal_methodologies_main_foyer

[6] Evolutionary Project Management (Evo)
http://www.spipartners.nl/data/Evo99.PDF

[7] Feature Driven Development (FDD)
http://www.featuredrivendevelopment.com

[8] Adaptive Software Development (ASD)
http://www.adaptivesd.com/

[9] Lean Development (LD) y Lean Software Development (LSD)
http://www.poppendieck.com

[10] Microsoft. Microsoft Hails 10 Years of Publisher
http://www.microsoft.com/presspass/press/2001/oct01/10-15TenYearsPublisherPR.mspx

[11] Manifesto for Agile Software Development
http://agilemanifesto.org/

[12] Parsons, David. An Analysis of Factors that Impact on the Success of Agile Software Development Using eXtreme Programming
http://www.massey.ac.nz/~hryu/DPHRRL.pdf

[13] Amber, Scott. Agile Adoption Rate Survey: March 2006

http://www.ambysoft.com/surveys/agileMarch2006.html#Discussion

[14] Uttam Narsu y Liz Barnett. Lessons Learned Practicing Agile Development

[15] Elssamadisy, Amr.Adopting Agile Development Practices: Using Patterns to Share our Experiences
http://www.infoq.com/articles/agile-practice-patterns

[16] L. Barnett. Agile Survey Results: Solid Experience And Real Results Agile Journal. 2006.

[17] L. Barnett and C. Schwaber. Adopting Agile Development Processes; Improve Time-To-Benefits For Software Projects Forrester Research, 2004.

[18] A. Law and R. Charron. Effects of agile practices on social factors, Proceedings of the 2005 workshop on Human and social factors of software engineering, ACM Press, St. Louis, Missouri, 2005.

[19] Sidky, Ahmed , A Structured Approach to Adopting Agile Practices: The Agile Adoption Framework, Phd Thesis, Virgina Tech 2007

[20] Spolsky, Joel. Painless Software Schedule. Marzo 2000.
http://www.joelonsoftware.com/articles/fog0000000245.html

[21] Agustín Villena, María Cecilia Bastarrica. "Un modelo ágil de inserción de alumnos de ingeniería de software en la industria. El caso del curso CC61A "Proyecto de Software"", IX Congreso Chileno de Educación Superior en Computación (CCESC), Iquique, Chile, Noviembre 2007.

[22] Panagiotis Sfetsos, Lefteris Angelis, Ioannis Stamelos. “Investigating The Extreme Programming System - An Empirical Study”.

[23] Kent, Beck y Cleal, Dave. “Optional Scope Contracts”.
www.xprogramming.com/ftp/Optional+scope+contracts.pdf.

[24] Cockburn, Alistair. “Are iterations hazardous to your project?” Humans and Technology, arc@acm.org, HaT TR 2005.04, Sept. 09, 2005 .
http://alistair.cockburn.us/index.php/Are_iterations_hazardous_to_your_project

[25] Cockburn, Alistair,”Earned-value and burn charts”,Humans and Technology HaT TR 2004.04, June 22, 2004
http://alistair.cockburn.us/index.php/Earned-value_and_burn_charts

[26] Version One. Herramienta de Gestión
http://www.versionone.com

[27] Mingle. Herramienta de Gestión
http://studios.thoughtworks.com/mingle-project-intelligence

[28] Beck, Ken. Fowler, Martin, “Planning Extreme Programming”, Julio 2000.

[29] Juego Explicativo de la planificación en Extreme Programming “Extreme Hour”. http://c2.com/xp/ExtremeHour.html