Estimación de póquer

From Scrum Manager BoK

Es una práctica ágil, para conducir las reuniones en las que se estima el esfuerzo y la duración de tareas. James Grenning ideó este juego de planificación para evitar discusiones dilatadas que no terminan de dar conclusiones concretas.

El modelo inicial de Grenning consta de 8 cartas, con los siguientes valores: ½, 1, 2, 3, 5, 6, 7 e infinito. Las mismas fueron desarrolladas para dar soporte a las estimaciones de versión en eXtreme Programming.

El funcionamiento es muy simple: cada participante dispone de un juego de cartas, y en la estimación de cada tarea, todos vuelven boca arriba la combinación que suma el esfuerzo estimado.

Cuando se considera que éste es mayor de x horas ideales (el tamaño máximo considerado por el equipo para una tarea), se levanta la carta "infinito". Las tareas que exceden el tamaño máximo deben descomponerse en subtareas de menor tamaño

Cada equipo u organización puede utilizar un juego de cartas con las numeraciones adecuadas a la unidad de esfuerzo con la que trabajan, y el tamaño máximo de tarea que se va a estimar.

Variante: sucesión de Fibonacci

Basado en el hecho de que al aumentar el tamaño de las tareas aumenta también el margen de error, surgió una variante que consiste en emplear números de la sucesión de Fibonacci para realizar las estimaciones, de forma que:

- El juego de cartas está compuesto por números en sucesión de Fibonacci.
- La estimación no se realiza levantando varias cartas para componer la cifra exacta, sino poniendo boca arriba la carta con la cifra más aproximada a la estimación.

Así, si por ejemplo una persona cree que el tamaño adecuado de una tarea es 6, se ve obligado a reconsiderar y, o bien aceptar que parte de la incertidumbre apreciada no es tal y levantar la carta de 5, o bien aceptar una estimación más conservadora y levantar el 8.

En particular, los números de esta variación del Planning Póquer son: 0, ½, 1, 2, 3, 5, 8, 13, 21, infinito.

Si se quiere emplear la planificación de póquer para estimar requisitos a nivel de producto o de versión (funcionalidades, temas) además de usarlo al nivel de tareas de sprint, se pueden añadir cartas al juego para

permitir estimaciones de mayor tamaño (34, 55, 89, 144, etc.)

Es frecuente emplear una carta con un símbolo de duda o interrogación para indicar que, por las razones que sean, no se puede precisar una estimación. También es posible incluir otra carta con alguna imagen alusiva, para indicar que se necesita un descanso.

Operativa

- Cada participante de la reunión tiene un juego de cartas.
- Para cada tarea (historia de usuario o funcionalidad, según sea el nivel de requisitos que se va a estimar) el cliente, moderador o propietario del producto expone la descripción empleando un tiempo máximo.
- Hay establecido otro tiempo para que el cliente o propietario del producto atienda a las posibles preguntas del equipo.
- Cada participarte selecciona la carta, o cartas que representan su estimación, y las separa del resto, boca abajo.
- Cuando todos han hecho su selección, se muestran boca arriba.
- Si la estimación resulta “infinito”, por sobrepasar el límite máximo establecido, la tarea debe dividirse en sub-tareas de menor tamaño.
- Si las estimaciones resultan muy dispares, quien asume la responsabilidad de gestionar la reunión, con su criterio de gestión, y basándose en las características del proyecto, equipo, reunión, nº de elementos pendientes de evaluar, puede optar por:
    - Preguntar a las personas de las estimaciones extremas: ¿Por qué crees que es necesario tanto tiempo?, y ¿por qué crees que es necesario tan poco tiempo? Tras escuchar las razones, repetir la estimación.
    - Dejar a un lado la estimación de esa tarea y retomar al final o en otro momento aquellas que hayan quedado pendientes.
    - Pedir al cliente o propietario del producto que descomponga la funcionalidad y valorar cada una de las funcionalidades resultantes.
    - Tomar la estimación menor, mayor, o la media.

Este protocolo de moderación, evita en la reunión los atascos de análisis circulares en ping-pong entre diversas opciones de implementación, hace participar a todos los asistentes, reduce el cuarto de hora o la media hora de tiempo de estimación de una funcionalidad, a escasos minutos, consigue alcanzar consensos sin discusiones, y además resulta divertido y dinamiza la reunión.

Categories: Scrum I | Glosario de términos

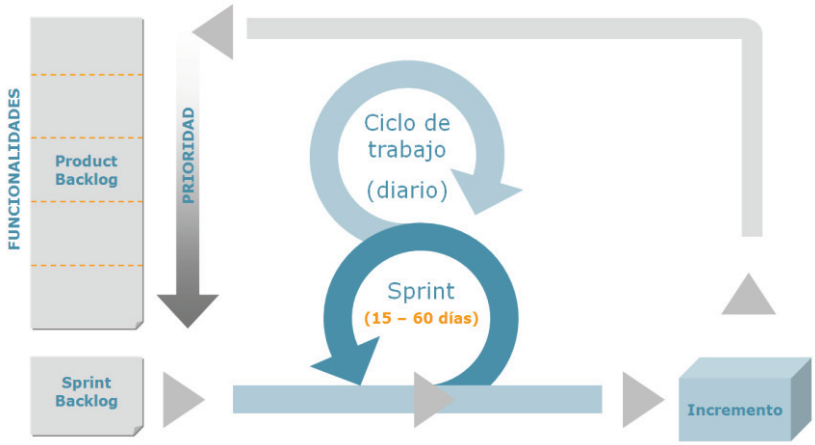
	INGENIERIA DEL SOFTWARE ESTRUCTURADA	INGENIERIA DEL SOFTWARE ORIENTADA A OBJETOS	INGENIERIA DEL SOFTWARE WEB
Descripción	También denominada “clásica” ó “Ingeniería del software orientada a funciones”. • Es la aplicada desde principios de los años 70. • Se basa en la idea de que un sistema software se compone de funciones que procesan datos. (Existen unas funciones generales y otras funciones específicas en las que se descomponen las primeras). • La Ingeniería del Software estructurada considera que el desarrollo de software ha de hacerse de forma descendente (top-down: desde una visión general cercana al usuario, hasta un nivel de abstracción mas detallado, cercano al programador), proponiendo la creación de modelos del sistema que representen de manera descendiente los siguientes aspectos: o Las funciones (también llamadas “procesos”) llevadas a cabo por el sistema. o Los flujos de datos de entrada, salida e internos de cada función del sistema. o La estructura de los datos procesados por las funciones del sistema.	• Se empieza a aplicar a finales de los años 80. • Se basa en la idea de que un sistema software se compone de objetos software que interactúan entre sí. • La funcionalidad de un sistema se reparte entre los objetos, asignando a cada objeto funciones específicas. • El objetivo final de la Ingeniería del Software Orientado a Objetos es construir software de la misma forma como se construye el hardware: mediante el ensamblaje de componentes. • Las ventajas de la orientación a objetos son: o Facilidad para la reutilización del software. o Simplificación del mantenimiento del software. o Mejora de la calidad del software.	También denominada “WEB ENGINEERING”. • Se trata de un tipo de ingeniería del software orientada a la naturaleza multidimensional de las aplicaciones web que implican, además de programación: o Definición de estructuras complejas de información (XML, DTD, ...). o Diseño de estructuras de navegación. o Gestión de contenidos. o Diseño gráfico. o Gestión de seguridad. o Gestión de diferentes perfiles de usuario.
Técnicas de Análisis	o Diagrama de Flujo de Datos. o Diagrama de Entidad Relación. o Diagrama de historia de la vida de una entidad. o Matriz de funciones-entidades.	o Diagrama de Casos de Uso. o Diagrama de Actividades. o Diagrama de Secuencia. o Diagrama de Colaboración. o Diagrama de Estados.	o Diagrama de Entidad Relación. o Diagrama de Slices (Entidad Relación extendido). o Diagrama de clases.
Técnicas de Diseño	o Diagrama de Estructura Modular (o “de Constantine”). o Diagrama de Módulos (o “de Jackson”). o Diagrama de Estructura Lógica de un programa (o “de Warnier”). o Diagrama de Árbol Programático (o “de Bertini”). o Diagrama de estructura de datos. o Diagrama de tablas (de una Base de Datos). o Diagrama de flujo de control (flowchart, organigrama, ordinograma). o Pseudocódigo.	o Patrones de diseño. o Diagrama de Componentes. o Diagrama de Despliegue. o	o Diagrama de navegación.
Técnicas de Programación	o Estructuras de control (secuencia, decisión (if), repetición (loop, for, while, repeat,..). o Subprogramas (Que permiten un tipo de programación estructurada conocida como programación modular (“un tipo de programación estructurada con una estructura conocida como subprograma”). o Lenguajes de programación estructurada: COBOL, FORTRAN, C, PASCAL, NATURAL, PL/SL(Oracle).	o Herencia: Objetos basados en otros objetos. o Polimorfismo: Funciones con el mismo nombre. o Agregación: Objetos que contienen objetos. o Lenguajes de programación orientada a objetos: JAVA, C++, C#, VISUAL BASIC, SMALLTALK, EIFFEL.	o Lenguajes de marcado (HTML, XML, ...). o Lenguajes de script (JavaScript, VBScript). o Programación CGI utilizando cualquier lenguaje (C, C++, Java, ..). o Lenguajes de programación: estructurados y orientados a objetos (XML, Perl, Java, C, C++, PL/SQL,...
Metodologías	• MERISE (Francia). • SSADM (Reino Unido). • MÉTRICA (España). • Otras...	• MÉTRICA 3 (España). • UNIFIED PROCESS (Rumbaugh, Booch y Jacobson). • FUSION (Hewlett-Packard). • OPEN (Henderson-Sellers). • Otras...	• OPEN (Henderson-Sellers). • HDM (Hypermedia Design Methodology). • OOHDM (Object Oriented HDM). • RMM (Relationship Management Methodology). • EORM (Enhanced Object Relationship Model). • Otras...

	VENTAJAS	INCONVENIENTES	USO
CASCADA	Gestión simple - Fácil planificación y estimación - Posible revisión al final de cada fase	No hay producto hasta el final - No iteracciones ni paralelismos, poco real - Los errores aparecen tarde y son muy costosos	Sistemas muy claros desde el principio - Requisitos bien definidos - Poca innovación
PROTOTIPADO	Fácil identificación y validación de requisitos - Combina con otros modelos	Tentación de utilizar el prototipo, baja calidad - Prototipo demasiado bueno, mayor coste y esfuerzo - Desilución del usuario	Usuarios poco comunicativos o requisitos poco definidos - Entorno estable
INCREMENTAL	Menos planificación y estimación - Riesgo de implantación menor - Fácil detectar errores - Producto parcial disponible antes	Difícil evaluar plazo y coste final - Necesaria buena gestión de versiones y configuración - Dificil detectar unidades y servicios genéricos de todo el sistema - Peligro de sistema poco consistente	Compromiso del cliente - Sistemas pequeños y sencillos - Sistemas poco claros inicialmente
ITERATIVO	Software modular - Riesgo de implantación menor - Fácil detectar errores - Producto parcial disponible antes	Solo para subsistemas independientes y poco intregrados - Los errores en cada subsistema aparecen tarde	Sistemas fácilmente divisibles - Requisitos bien definidos
ESPIRAL	Permite iteracciones y vuelta atrás - Incorpora objetivos de calidad y análisis de alternativas	Mayor coste y tiempo derivado del análisis de alternativas - Compeljidad de la gestión, necesidad de un buen gestor experto	Sistemas con riesgos, entornos inestables o alta innovación

Visión general del proceso

El resultado final se construye de forma iterativa e incremental.

Al comenzar cada iteración ("sprint") se determina qué partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración.



Los componentes y conceptos empleados en Scrum son:

Las reuniones

- **Planificación del sprint**: Jornada de trabajo previa al inicio de cada sprint en la que se determina cuál es el trabajo y los objetivos que se deben cubrir con esa iteración.
  Esta reunión genera la "sprint backlog" o lista de tareas que se van a realizar, y en ella también se determina el "objetivo del sprint": lema que define la finalidad de negocio que se va a lograr.

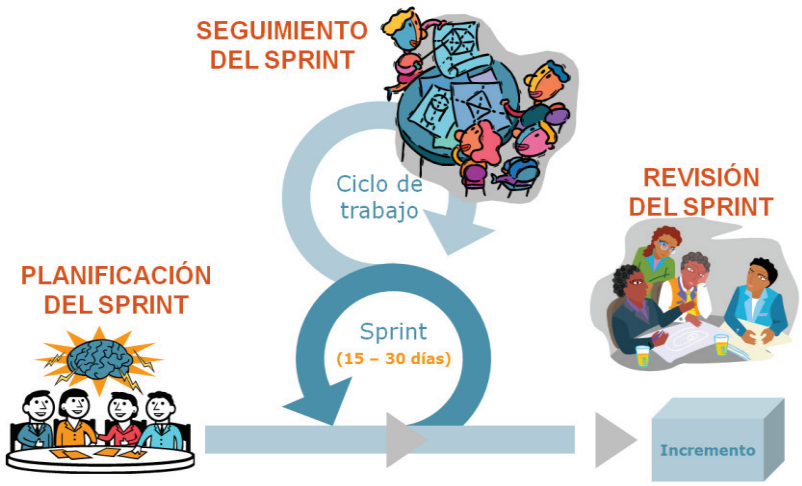
- **Seguimiento del sprint**: Breve reunión diaria para dar repaso al avance de cada tarea, y al trabajo previsto para la jornada.
  Sólo interviene el equipo, y cada miembro responde a tres preguntas:

  1.- Trabajo realizado desde la reunión anterior.

2.- Trabajo que se va a realizar hasta la próxima reunión de seguimiento.

3.- Impedimentos que se deben solventar para que pueda realizar el trabajo.

- **Revisión de sprint**: Análisis y revisión del incremento generado. Esta reunión no debe tomarse como un "acontecimiento especial", sino como la presentación normal de los resultados.



Los elementos

- **Product backlog**: Requisitos del sistema. Se parte de la visión del resultado que se desea obtener; y evoluciona durante el desarrollo.
  Es el inventario de características que el propietario del producto desea obtener, ordenado por orden de prioridad.

  Es un documento "vivo", en constante evolución.

  Es accesible a todas las personas que intervienen en el desarrollo.

  Todos pueden contribuir y aportar sugerencias.

  El responsable del product backlog es una única persona y se le denomina: propietario del producto.

- **Sprint Backlog**: Lista de los trabajos que realizará el equipo durante el sprint para generar el incremento previsto.
  El equipo asume el compromiso de la ejecución.

  Las tareas están asignadas a personas, y tienen estimados el tiempo y los recursos necesarios.

- **Incremento**: Resultado de cada sprint.
  Se trata de un resultado completamente terminado y en condiciones de ser usado.

Los roles o responsabilidades

El grado de funcionamiento de Scrum en la organización depende directamente de estas tres condiciones:

- Características del entorno (organización y proyecto) adecuadas para desarrollo ágil.
- Conocimiento de la metodología de trabajo en todas las personas de la organización y las implicadas del cliente.
- Asignación de responsabilidades:
  - Del producto.
  - Del desarrollo.
  - Del funcionamiento de Scrum

Responsabilidad del producto: El propietario del producto

En el proyecto hay una persona, y sólo una, conocedora del entorno de negocio del cliente y de la visión del producto. Representa a todos los interesados en el producto final y es el responsable del Product Backlog.

Se le suele denominar "propietario del producto" y es el responsable de obtener el resultado de mayor valor posible para los usuarios o clientes.

Es responsable de la financiación necesaria para el proyecto, de decidir cómo debe ser el resultado final, del lanzamiento y del retorno de la inversión.

En desarrollos internos puede ser el product manager, o responsable de marketing… quien asume este rol.

En desarrollos para clientes externos lo más aconsejable es que sea el responsable del proceso de adquisición del cliente.

Responsabilidad del desarrollo: El equipo

Todo el equipo de desarrollo, incluido el propietario del producto conoce la metodología Scrum, y son los auténticos responsables del resultado.

Es un equipo multidisciplinar que cubre todas las habilidades necesarias para generar el resultado.

Se auto-gestiona y auto-organiza, y dispone de atribuciones suficientes en la organización para tomar decisiones sobre cómo realizar su trabajo.

Responsabilidad del funcionamiento de Scrum (scrum manager)

La organización debe garantizar el funcionamiento de los procesos y metodologías que emplea, y en este aspecto Scrum no es una excepción.

En el modelo de Scrum definido por Jeff Sutherland, esta responsabilidad se garantiza integrando en el equipo una persona con el rol de ScrumMaster.

Considerando que las realidades de unas y otras empresas pueden ser muy diferentes, y que siempre que sea posible es mejor optar por adaptar las prácticas de trabajo a la empresa, y no al revés, en ocasiones puede resultar más aconsejable:

- Que en lugar de una persona con la función de "ScrumMaster", sean las personas y puestos más adecuados en cada organización los que reciban la formación adecuada y asuman las funciones correspondientes para cubrir esta responsabilidad.
- Que al compromiso de funcionamiento del proceso se sume también la dirección de la empresa, con el conocimiento de gestión y desarrollo ágil; y facilitando los recursos necesarios.

Scrum Manager designa por tanto, más que al rol, a la responsabilidad de funcionamiento del modelo. Puede ser a nivel de proyecto o a nivel de la organización; y en algunos casos resultará más apropiado un rol exclusivo (tipo ScrumMaster) y en otros, puede ser mejor que la responsabilidades de funcionamiento las asuman los responsables del departamento de calidad o procesos, o del área de gestión de proyectos…

Herramientas

Gráfico Burn-Up

Herramienta de gestión y seguimiento para el propietario del producto. Presenta de un vistazo las versiones de producto previstas, las funcionalidades de cada una, velocidad estimada, fechas probables para cada versión, margen de error previsto en las estimaciones, y avance real.

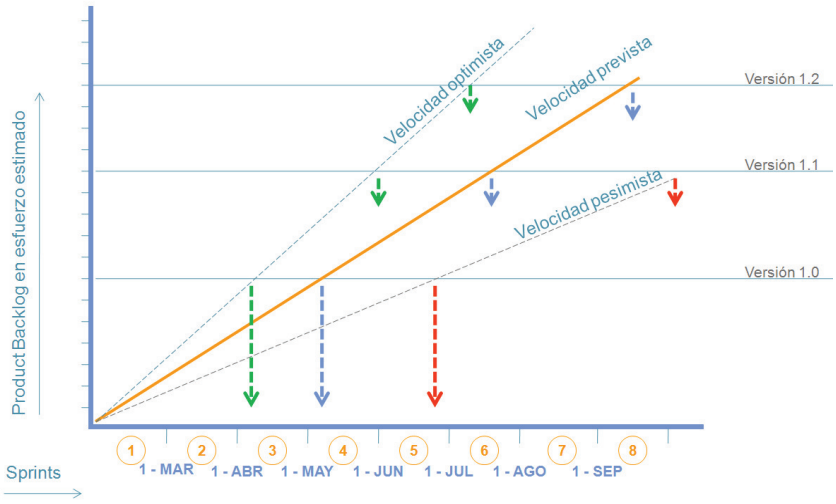


Gráfico Burn-Down

Herramienta del equipo para gestionar y seguir el trabajo de cada sprint.

Representación gráfica del avance del sprint.

Juegos y protocolos de decisión

Estimación de póker: Juego para agilizar y conducir la estimación de las tareas en la reunión de inicio del sprint.

Estimación a los chinos: Otro protocolo con formato de juego para realizar estimaciones en equipo.

Conceptos y métricas

Tiempo real o tiempo de trabajo.

Tiempo efectivo para realizar un trabajo. Se suele medir en horas o días.

Tiempo teórico o tiempo de tarea

Tiempo que sería necesario para realizar un trabajo en "condiciones ideales": si no se produjera ninguna interrupción, llamadas telefónicas, descansos, reuniones, etc.

Puntos de función o puntos de funcionalidad

Unidad de medida relativa para determinar la cantidad de trabajo necesaria para construir una funcionalidad o historia de usuario del product backlog.

Estimaciones

Cálculo del esfuerzo que se prevé necesario para desarrollar una funcionalidad.

Las estimaciones se pueden calcular en unidades relativas (puntos de función) o en unidades absolutas (tiempo teórico).

Velocidad absoluta

Cantidad de producto construido en un sprint. Se expresa en la misma unidad en la que se realizan las estimaciones (puntos de función, horas o días reales o teóricos).

Velocidad relativa

Cantidad de producto construido en una unidad de tiempo de trabajo.

P. ej.: puntos de función / semana de trabajo real; o horas teóricas / día de trabajo real…

Valores

Las prácticas de Scrum son una "carrocería" que permite trabajar con los principios ágiles, que son el motor del desarrollo. Son una ayuda para organizar a las personas y el flujo de trabajo; como lo pueden ser otras propuestas de formas de trabajo ágil: Cristal, DSDM, etc.

La carrocería sin motor, sin los valores que den sentido al desarrollo ágil, no funciona.

- Delegación de atribuciones (*empowerment*) al equipo que le permita auto-organizarse y tomar las decisiones sobre el desarrollo.
- Respeto entre las personas. Los miembros del equipo deben confiar entre ellos y respetar sus conocimientos y capacidades.
- Responsabilidad y auto-disciplina (no disciplina impuesta).
- Trabajo centrado en el desarrollo de lo comprometido
- Información, transparencia y visibilidad del desarrollo del proyecto