

planificación ágil

¿por qué planificar?

porque necesitamos:

- asegurarnos de que siempre estamos trabajando en lo más importante que tenemos que hacer
- coordinarnos con otras personas
- entender las consecuencias de que ocurran eventos inesperados

errores típicos

calendario optimista

expectativas no realistas, debido a especificaciones ambiguas

confusión de estimaciones con objetivos :

tiene que estar en tres meses

omisión de estimación de tareas necesarias: pruebas, documentación,
gestión de la configuración...



no tengo tiempo

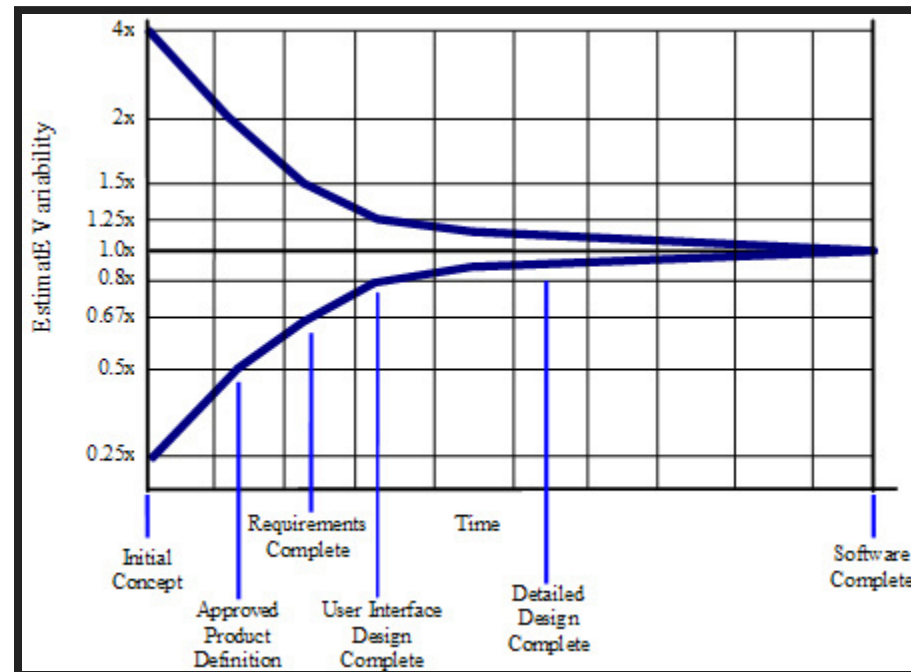
mejor dicho: tengo mucho que hacer

y entonces:

- puedo priorizar y no hacer algunas cosas
- puedo reducir el alcance de algunas de las cosas que tengo que hacer
- puedo pedir a otros que hagan algunas cosas

referencia: [Rework](#): *no time is no excuse*, David Heinemeier Hanson, Jason Fried

cono de incertidumbre



estimación

nuestras estimaciones son horribles

“creemos que podemos adivinar cuánto tiempo nos va a llevar algo, pero lo cierto es que no tenemos ni idea. No somos capaces de predecir qué va a pasar”

“si no podemos ser precisos estimando unas pocas horas, ¿cómo pretendemos estimar con precisión la duración de un proyecto software?”

¿y cómo lo hacemos?

dividiendo las cosas grandes en cosas más pequeñas

refinando el plan a lo largo del proyecto

planificando sólo con el **detalle necesario** para entregar el siguiente incremento de producto y estimando el resto en trozos más grandes

haciendo primero las cosas más importantes

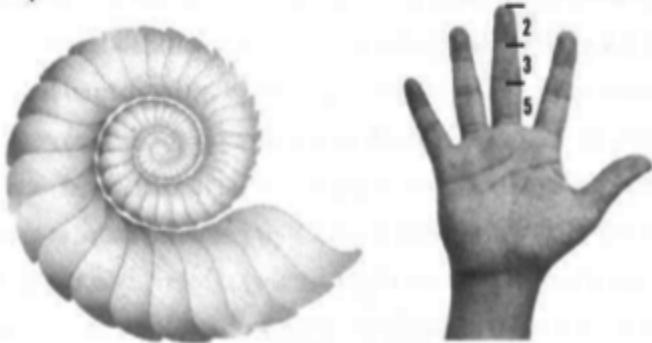
el tamaño importa...
relativamente



secuencia de fibonacci

Fibonacci Sequence: All Around Us

- The Fibonacci sequence is a pattern where the next number in the sequence is the sum of the previous two, e.g.,
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 . . .
- Ubiquitous in natural systems, so humans have millennia of experience with it



estimación relativa

los números de la secuencia de fibonacci están lo suficientemente separados como para distinguir diferencias

no es lo mismo la diferencia entre 5 y 6 que la diferencia entre 8 y 13...

si nuestras estimaciones se hacen en grupo, serán mejores

¿cómo estimar?

- sabemos que somos buenos comparando cosas
- tenemos una escala adecuada
- sabemos que estimando en grupo seremos más precisos

necesitamos **evitar sesgos** en el grupo:

- efecto carro
- efecto halo

el método delphi

método delphi

- admite un amplio conjunto de opiniones
- elimina sesgos
- utiliza opiniones de expertos y anónimas

pero...

lleva mucho tiempo

planning poker

- a cada miembro del equipo se le da un mazo de cartas con la secuencia de fibonacci
- se toma la historia a estimar y se expone en el grupo
- cada uno coge la carta que cree que representa el esfuerzo y la pone boca abajo sobre la mesa
- se da la vuelta a todas las cartas a la vez
- si se está a menos de dos cartas, se hace la media
- si se está a más de tres cartas, la más alta y la más baja discuten
- se vuelve a estimar hasta que converja el resultado

consejos prácticos

- equiparar el punto de esfuerzo a la hora de trabajo
- utilizar la experiencia pasada
- un par de malas estimaciones no implican un desastre
- no pasemos más de 2 horas estimando (ni en cualquier otra reunión)
- conocer la velocidad del equipo: puntos realizados / puntos planificados (se puede expresar en horas por punto)

ejercicio

como usuario **quiero** ser capaz de identificarme en el sistema **para** acceder a mi información personal

como gestor **quiero** poder importar automáticamente asientos de contabilidad desde el SIC del Gobierno de Cantabria **para** poder enviárselos al MINHAP desde mi aplicación **y** así cumplir con la normativa vigente

como programador **quiero** que cada vez que haga commit de un archivo XML al sistema de control de versiones este se valide previamente **para** evitar introducir y propagar errores en el servidor

estimar un proyecto

$$CT = \text{SUMA}(Fi) * Vd * C + ME$$

en donde:

- CT es el coste total en €
- Fi es el número de puntos de esfuerzo estimados por cada característica, historia o requisito y **SUMA ()** es un sumatorio
- Vd es la velocidad de desarrollo del adjudicatario en horas / punto
- C es el coste medio de la hora de trabajo del adjudicatario en € / hora
- ME es el margen de incertidumbre

y refinar

número de iteraciones

ejemplo

- proyecto de 250 puntos historia
- jornadas de 6 horas
- semanas de 5 días
- iteraciones de dos semanas
- velocidad de desarrollo de 1.5 horas / punto
- puntos por sprint: $6 \cdot 5 \cdot 2 / 1.5 = 40$

Nº de iteraciones: $250 / 40 = 6,25$ (7)

Duración: 14 semanas

duración de una iteración

respuesta rápida: de 2 a 4 semanas

depende de:

- frecuencia de demostración
- frecuencia de reajuste de objetivos
- tiempo que tarda una idea en transformarse en software

a menor tiempo de iteración, mayor sofisticación debe tener el equipo de desarrollo

¿duración variable?

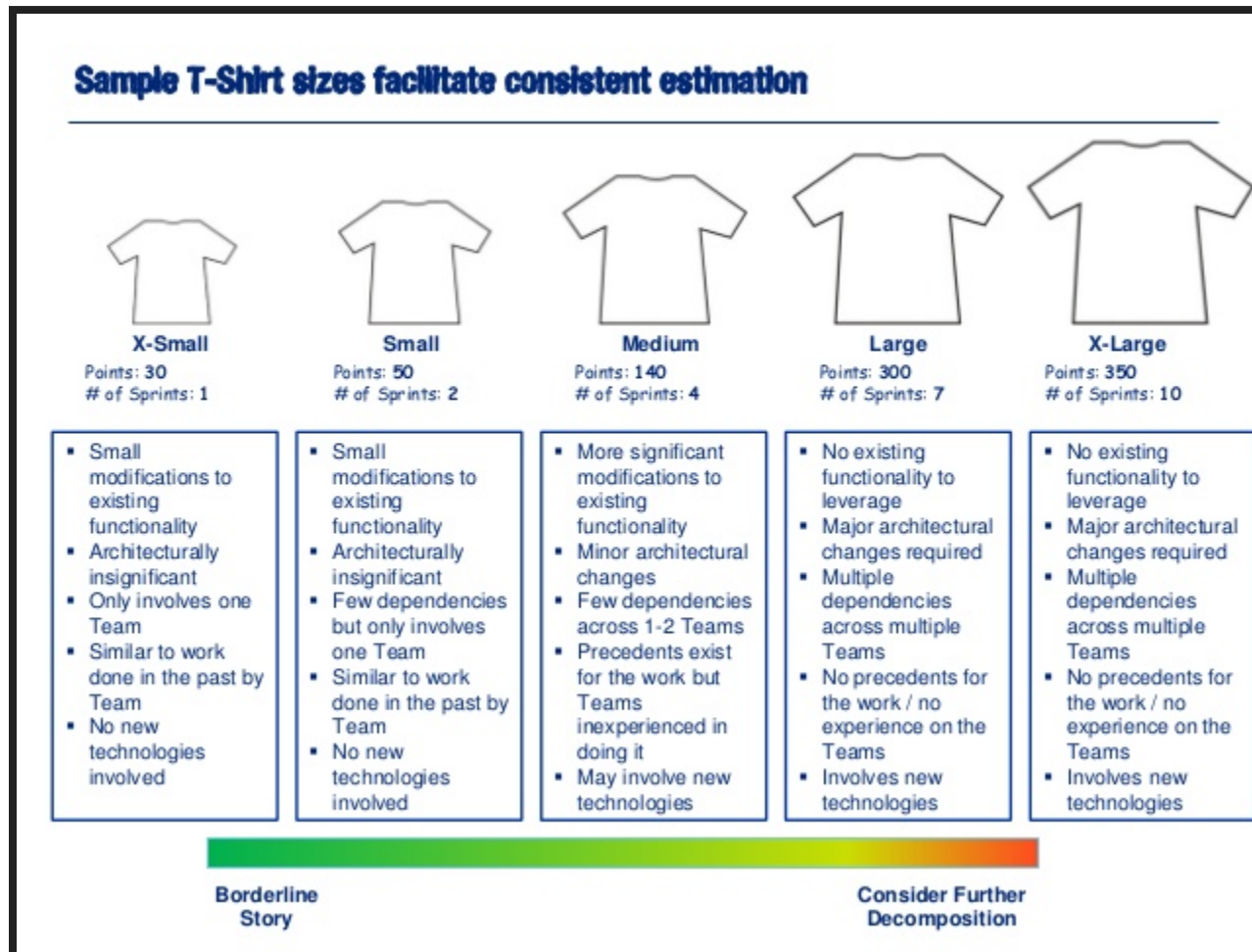
usar sprints con duración fija:

- facilita el cálculo y la exactitud de la velocidad de desarrollo
- sincroniza los calendarios
- ayuda a fijar el ritmo de trabajo

para equipos poco experimentados, variar la temporalidad de la iteración suele conducir a problemas

fuentes: [M00C Agilidad y Lean. Gestionando los proyectos y negocios del siglo XXI](#), Javier Garzás

estimación por tallas





[Acceso al repositorio del curso](#)

[Miguel Expósito Martín](#)