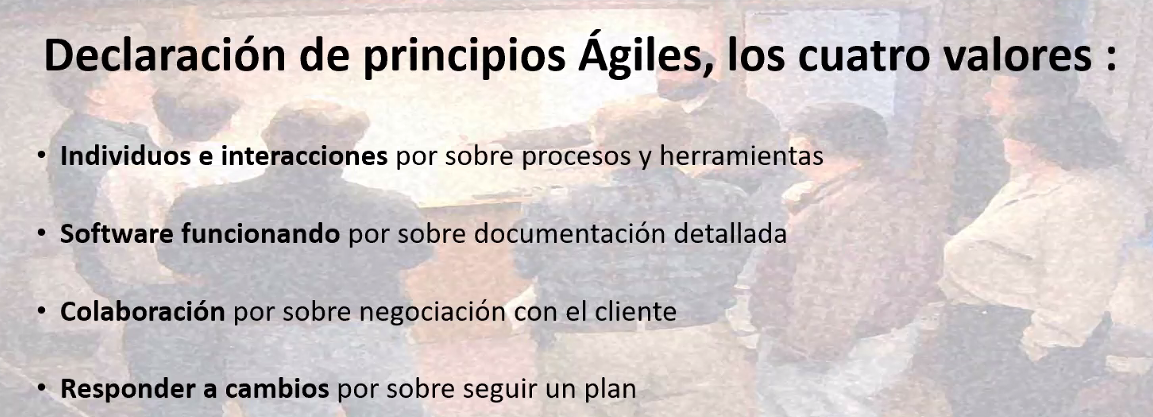
**Clase 04/08/20**

¿Diferencia entre un proceso definido y uno empírico?

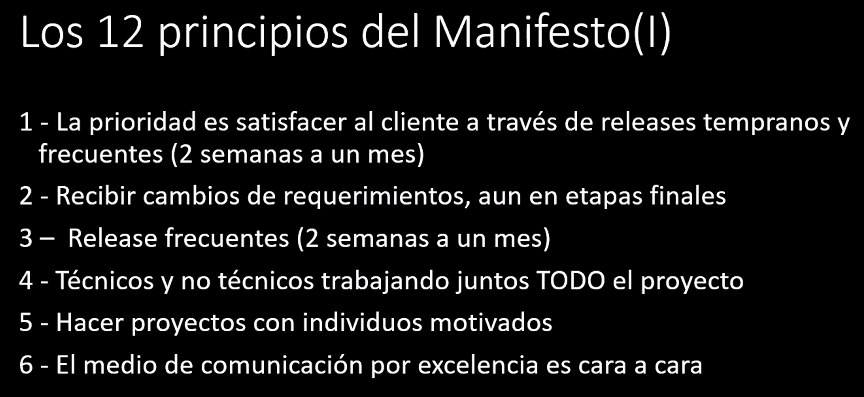
Un proceso definido es por ejemplo el PUD, donde uno tiene definidas actividades, como se realizan, quienes la realizan, entradas, salidas, etc. Está definido el paso a paso de que se debe hacer, como, quien…

Un proceso empírico, en vez de tomar esta plantilla, se construye como equipo viendo que es lo que nos conviene, que nos sirve. Existen momentos de retroalimentación y cambio del proceso para siempre ir mejorándolo.

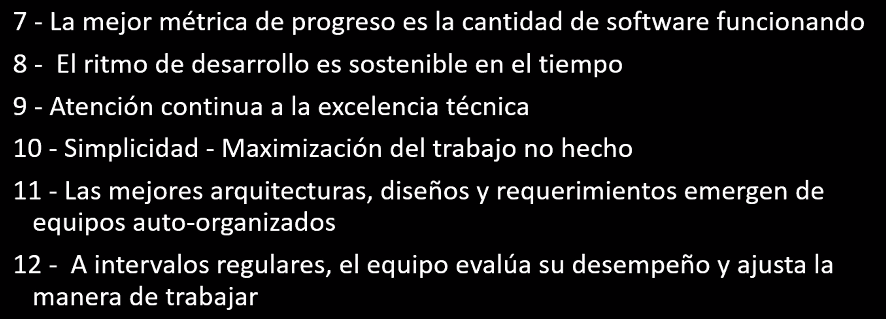


Solamente enmarcan como deberíamos estar estructurados mentalmente para hacer desarrollo de sw ágil.

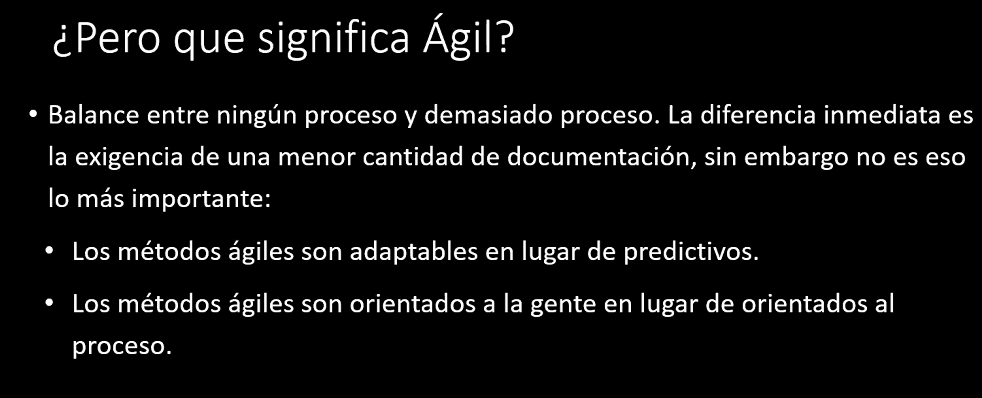
* El foco, justamente, no son procesos definidos, planillas, etc. Si no en asegurarnos que la gente que va a trabajar tenga los conocimientos para poder encarar los proyectos, necesitamos gente con conocimiento a gran nivel, y también es necesario que la gente se comunique constantemente.
* No voy a tener todo documentado, no es necesario tener tanto documento ya que el objetivo es tener un sw que funcione. Rápidamente hay que tener sw funcional para poder mostrarle al cliente y poder validar si es lo que desea.
* No necesito la documentación, para sobre esa documentación estar negociando con el cliente. Que esta fuera y que está dentro de la ERS y ese tipo de discusiones no deberían ser necesarias. Es por esto que tratamos de ir sacando sw funcional para poder ir mostrándole al cliente y que vea si es lo que quiere o no.
* No definir un plan y apegarnos estrictamente a eso. Si no que se hace un roadmap que marca el camino a seguir, pero acepta el cambio, ya que nos va a salir más barato cambiar durante la marcha que modificarlo en el futuro porque se traba todo.



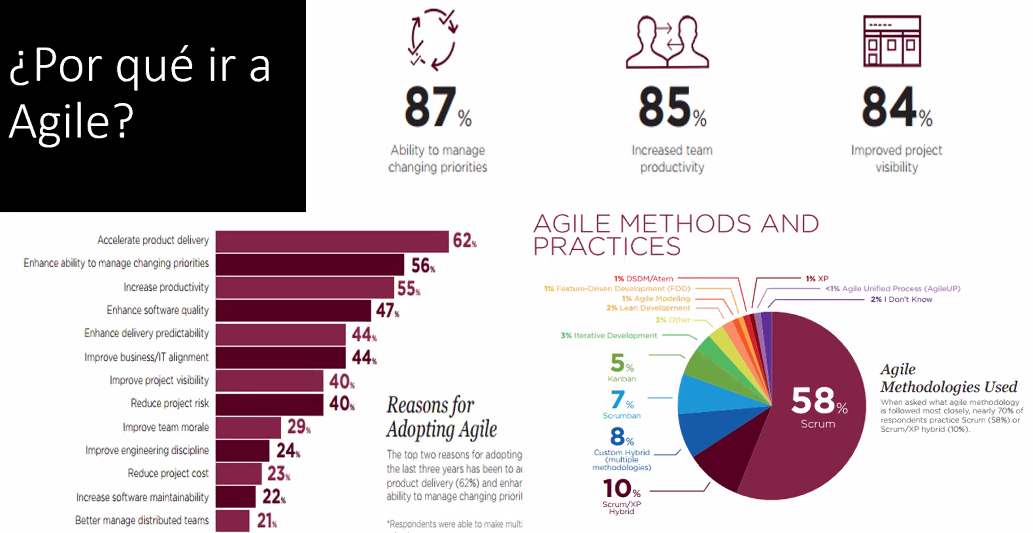
1. Quiero dar sw funcionando lo antes posible, esto lo intentamos hacer entre las 2 semanas y 1 mes.
2. Se acepta el cambio en todo momento. Cualquier tipo de requerimiento, funcional o no funcional.
3. Entregar lo antes posible. Cada dos semanas tengo una versión factible de ponerse en producción, no es que la vamos a poner en producción, pero tiene que estar a ese nivel.
4. Todos tenemos el mismo objetivo
5. Fundamental en el agilísimo que la gente tenga ganas de hacer su trabajo porque es más productiva.
6. Siempre es lo mejor.



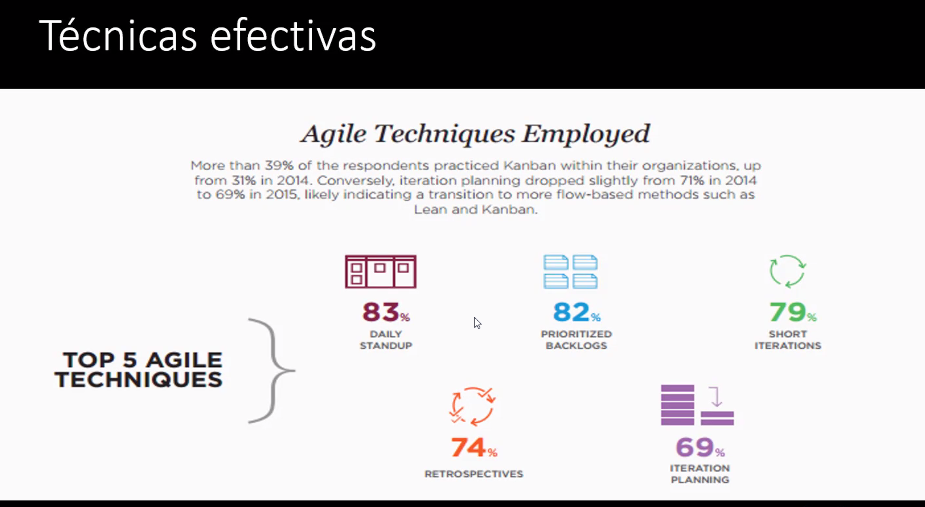
1. Una forma de saber si el proyecto va bien o mal, es ver la cantidad de features que tengo funcionando.
2. Hay que mantener un ritmo de desarrollo donde no se frene nunca.
3. Es fundamental para que no lleguemos al final, por ejemplo, sin ningún requerimiento no funcional hecho. Hay que ir respetando todos los lineamientos.
4. No me voy a poner hacer trabajo que no necesito hacer. Hago lo que necesito hacer para avanzar solamente. No me pongo a hacer grandes especificaciones tres meses antes. Hay que evitar los “ya que estamos le agregamos esto”. Esperar a que nos pidan hacerlo, no hacer cosas que no fueron explicitas. Hay que hacer lo más simple posible, que cumpla con el objetivo y lineamientos de calidad.
6. Como equipo, cada X tiempo (generalmente luego de un reléase) evaluamos el trabajo que hemos hecho, y proponemos mejoras.



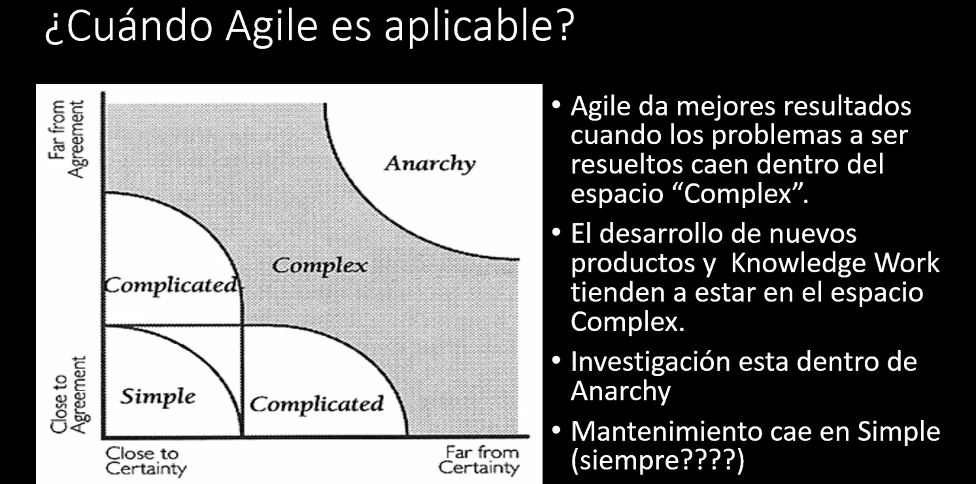
Se planifica, hay definiciones claras en los intervalos de tiempo donde hay que tener entregas. El proceso lo vamos a estar construyendo como equipo. Sin embargo, es un proceso muy liviano a otros que conocemos. Saca mucha documentación.

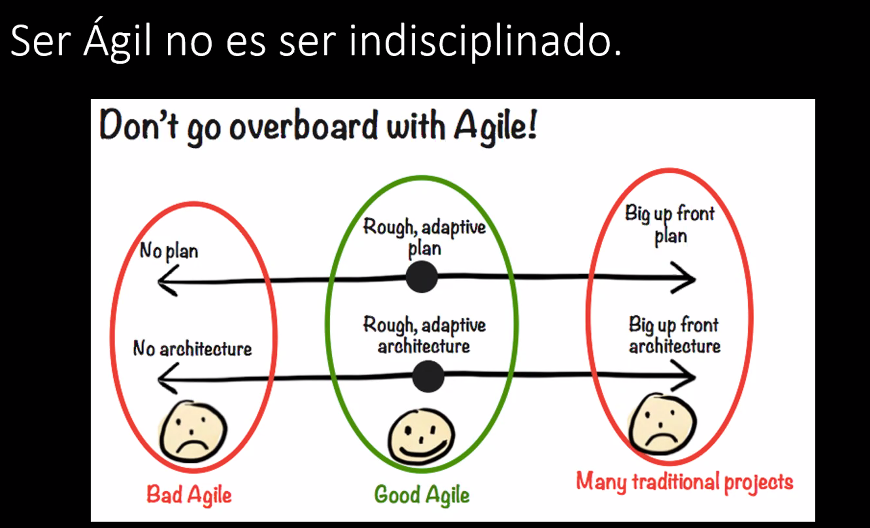


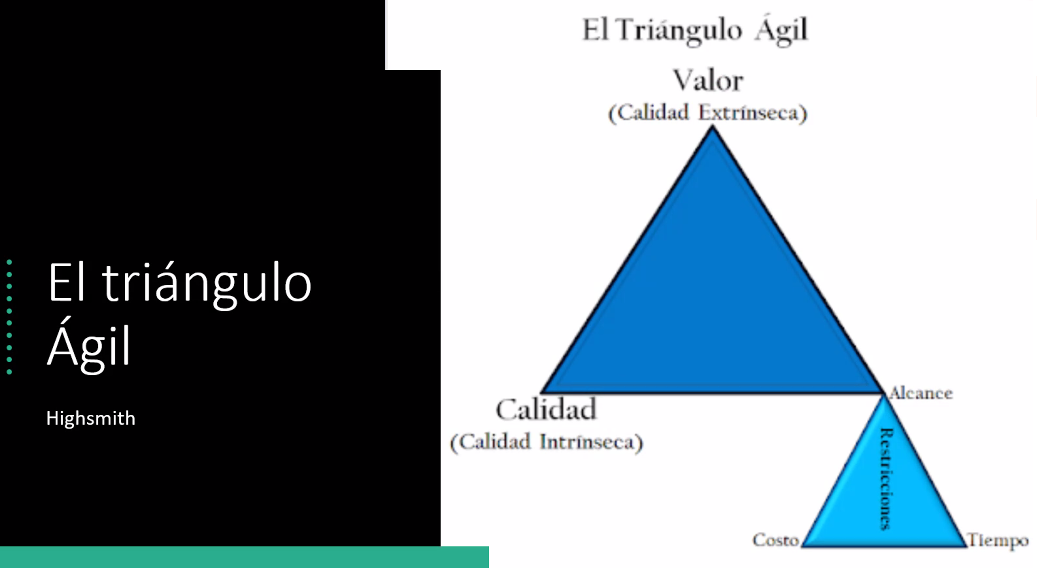
Estar entregando cada 2 semanas nos da mucha más visibilidad.

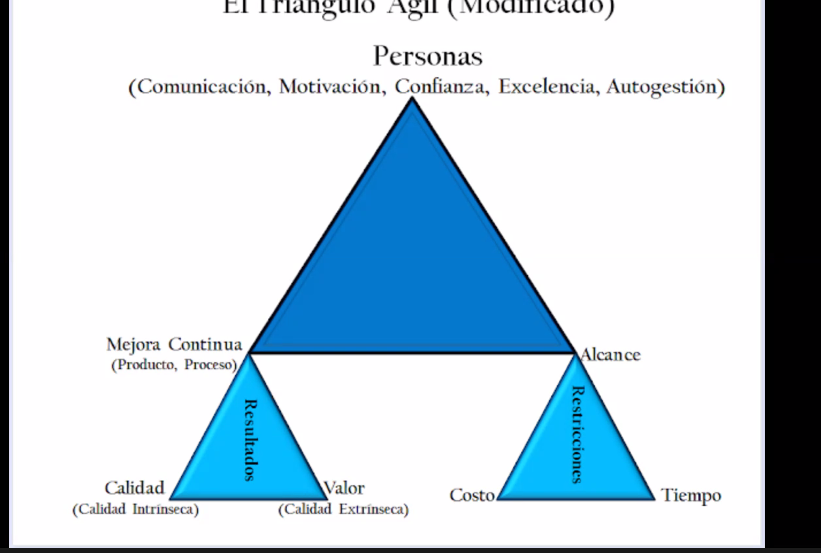


5 Cosas que se demostraron que son útiles y mejoraron mucho la productividad.









Actividad

A intervalos regulares…: Tiene que ver más con cambios de proceso, por lo tanto, sería más correcto que vaya en el valor 1. Además, tiene que ver en la forma que trabaja el equipo, y en función de eso cambiar el proceso.

**Clase 18/08/20**

En las estimaciones agiles vamos a estar comparando cosas, vamos a establecer un punto de inicio y vamos a comparar contra este. Esto solo va a tener vigencia en el proyecto de sw que estemos trabajando.

Dentro de las metodologías agiles surge una forma de estimar que es el story point. El story point va a tener en cuenta la complejidad, incertidumbre y esfuerzo. Este promedio ponderado de estas componentes nos va a dar el story point.

Dentro de este contexto podemos tener us que tengan un mismo valor asociado en cuanto a story points, pero si nos ponemos a analizar pueden ser diferentes, aunque tengan los mismos puntos.

El póker planning es un método en el cual el equipo de trabajo va a tomar un conjunto de funcionalidades que ya cumplen con una base de que sean estimables. Sobre este conjunto de funcionalidades o user stories, vamos a elegir la que tenga la menor incertidumbre, menor esfuerzo y menor complejidad. A esta us que la tenemos muy clara la vamos a tener como base de comparación, llamada user story canónica. Esto es porque todas las demás user stories que tengamos que estimar las vamos a comparar contra esta us canónica.

Se elige la sucesión de Fibonacci porque se ha comprobado que la complejidad del sw crece exponencialmente y la de esta serie también.