Instructions

When Good Enough Software is Best

Me hizo acordar a que la arquitectura en general debe ser good enough.

Software suficiente. Relación tiempo, costo, calidad. Ya no se produce software perfecto.

Tarde nunca es mejor

No es necesario hacer un software perfecto, sino que sea lo suficientemente bueno para el usuario final

es complejo educar a los clientes, cómo tener paciencia para educarlos para que sepan lo que quieren

el concepto de "suficientemente bueno"

Se busca el balance entre costo y calidad

Me parecio interesante como la cantidad de errores es un parámetro más, no lo habia pensado

When Good Enough Software is Best

Los usuario hoy ven al software como commodity.No siempre es mejor priorizar calidad sobre tiempo.Las exigencias son irreales y se deben negociar, se deben presentar las consecuencias de los trade off

Una negociación clara y directa puede determinar el éxito del proyecto ante los ojos del cliente.

Me copó lo de los 3 enfoques, y cómo a cada parte le importa más uno u otro (además de los trade offs)

la idea de que entregar software con errores puede ser la mejor decisión posible en ciertos contextos

Siempre es un trade-off de varias partes con lo que cada una considera como software de calidad

Que "lo mejor" no siempre es lo mejor. A menudo lo bueno es muchísimo mejor que "lo mejor".

Las altas espectativas

Habla de 3 variables: costo tiempo y alcance, y dice que solo se puede tener 2 de estas, teniendo en el medio la calidad. Tambien dice que sacar un producto tarde aunque sea mejor que otros no es lo

When Good Enough Software is Best

Hay que priorizar a los clientes inteligentes

semejanza con la realidad con lo clientes y mismo con los jefes de uno mismo.

Sigue vigente todo lo que plantea (o la mayoría)

cuando cumple los requerimientos planteados dentro del tiempo y costo acordado

No ser perfeccionistas y entender que los parametros de un proyecto son imposibles de cumplir en un 100% sin realizar tradeoffs.

Los trade-offs entre los 5 atributos que propone, y la ruptura con la noción de sacar las cosas sin errores. La idea de que el tiempo es incluso mas importante que la correctitud.

Tiene que ser lo suficientemente bueno para el cliente, no el mejor. Tiene que cambiar la cultura alrededor del desarrollo de software en este sentido.

En la lectura está bueno el ejemplo de que hay que priorizar algunas cosas según la necesidad del momento. Me refiero específicamente al ejemplo del producto A, B y C

When Good Enough Software is Best

Es el cliente quien decide el balance justo de los parametros

Es el cliente quien decide cual es el balance apropiado de parámetros.

La triada imposible: rápido, barato y perfecto

Entregar un software “suficientemente bueno” a tiempo y dentro del presupuesto es mejor que uno perfecto, caro y tardío. Lo importante es negociar las prioridades.

El software tiene que ser suficientemente bueno y cumplir con los requerimientos de cualquier forma

Que no se pueda optimizar al maximo calidad, costo, tiempo, funcionalidad y rrhh a la vez me llamo la atencion, si bien es algo logico si me pongo a pensar, nunca lo habia pensado

just 2 of 3

distintas perspectivas de lo que importa

When Good Enough Software is Best

negociación de calidad

Antiguo

La tecnologia evoluciono a pasos agigantados, pero podria leer este articulo hoy y sigue aplicando

Es una aproximacion a lo que hoy en día llamamos MVP, se prioriza cumplir con lo justo y necesario para que sea utilizable el sistema

Proyectos Extensos en el que no se puede tener todo controlado al 100%. Y si el proyecto tiene que ser lanzado lo antes posible

Se siente bastante actual el articulo a pesar de ser antiguo

Es el cliente quien decide el balance adecuado entre los parametros

Un software es bueno cuando cumple con lo que se pide, se minimizan los errores que se pueden detectar y que tenga la posibilidad de cambio.

When Good Enough Software is Best

Costos, calidad, rapidez, 3 parámetros dificil de conseguir

elegi 2 de 3

tarde nunca es mejor

Que el desarrollador tiene que establecer un balance entre las variables de costo, tiempo, recursos, calidad del producto