1 Introducción

No es el ámbito de este proyecto el análisis de las ciencias de la computación, ni el desarrollo de tecnologías, ni temas de base.

Se ha querido considerar desde un punto de vista de un ingeniero industrial que ha terminado gestionando proyectos de desarrollo de software. Como en este ámbito profesional comercial del desarrollo de software hay un hueco para la visión de gestión que tiene un ingeniero industrial del mismo que tiene un encaje y añade valor al proceso. Al fin y al cabo la gestión de cualquier proyecto es el núcleo vertebral de lo que enseña esta escuela.

Los principales problemas del desarrollo comercial de software no difieren tanto de lo que podría parecer de la gestión de proyectos de cualquier industria. Analicemoslo Centrándonos en las bases más obvias de un proyecto:

- desafio tecnico
- gestion de tiempo/presupuesto
- adaptacion al cambio

El desafío técnico en la primera fase de un arte es la ebullición, cada artesano innova en su técnica dando lugar a los productos y resultados más dispares, la gente no sabe lo que compra, algunos adquieren fama y otros no. Nadie va a discutir la calidad que resulta de los mejores, rivalizando incluso con los procesos industriales más punteros.

Cuando los proyectos son más grandes el artesano empieza a ser un factor de riesgo, la mala elección de un arquitecto en una catedral sin estandarización de procesos de cálculo, uso de materiales puede malograr ingentes cantidades de recursos.

Quizá es en el último punto donde la industria del software se diferencia más de cualquier industria. los productos salen y son entregados, con mejor o peor resultado, incluso hay algunos que requieren mantenimiento. y se cambia para que los siguientes productos se adapten, sin embargo el proyecto de software nunca termina. Es extremadamente volatil, cadudo y cambiante.

En cada uno de estos puntos en la industria hay cantidad de métodos, herramientas, procesos, normativa que va encaminada a que haya la menor incertidumbre posible en el proyecto. Lo requiere el cliente y el profesional que ejecuta para poder obtener la satisfacción del mismo con el que progresar como profesional.

En el software hay técnicas o procesos que tienen más de 20 años, se han demostrado útiles y sin embargo no han penetrado en el software comercial en todos los puntos. El cliente sigue teniendo un desconocimiento total a

la hora de comprar software, hay pocas garantías y mucho dinero en juego. cuando compras un coche no sabes de coches, pero sabes qué preguntar, sabes qué mirar y tienes garantias de las grandes marcas. Hoy en día incluso las grandes consultoras tienen proyectos de software que a día de hoy son un fracaso.

Las principales son:

- como se va entregando valor de forma iterativa

  scrum, kanban, agile, extreme programing

- como se garantiza la expectativa del cliente con el producto

  Testing: TDD,BDD... diseño de casos de uso UML, UX, UI

- como se garantiza la posibilidad de cambio

  arquitectura de capas, hexagonal, DDD...

La creación de software ha vivido desde sus fases iniciales hasta hoy un proceso análogo al que hemos observado en cualquier industria. En una primera fase los artesanos, luego los gremios, estandarización producción en masa, optimización continua e innovacion.

el punto diferencial llega en la estandarización. En la cuestión de la gestión de los proyectos hay un despliege de medios en la industria que permiten al cliente comprar con garantias. En el software esto es complicado y este proceso está innmaduro. no son pocos los proyectos que no llegan a buen puerto, incluso los que disponen de un soporte financiero virtualmente ilimitado.

En el software, se ha hecho un esfuerzo constante en la estandarización de procesos para poder controlar la gestión de los proyectos

En mi experiencia profesional me he encontrado con los problemas de la asincronia que php no resuelve directamente, hay proyectos de terceros y ahora ya se ha desarrollado, pero era interesante ver como lo resuelve golang y porque tiene tanta fama y si compensa el coste de transicion del lenguaje. Y porque no node? O

Porque golang añade otra capa de que fuerza en tiempo de compilacion el estilo. Por ejemplo las interfaces porque tambien me he encontrado con muchos problemas de mantenimiento del codigo

El coste mas importante de un proyectos software siempre es el mismo: tiempo de lectura de codigo. Ademas es un coste que no añade valor. Por el que el cliente no va a pagar mas Es por esto que hay que reducirlo al máximo.

Este proyecto analiza desde el punto de vista de un manager cuáles son los puntos de ventaja que da este lenguaje de programacion en su dia a dia

Esto no solo se consigue con el lenguaje de programación, al fin y al cabo es una herramienta y como tal hay muchas pàrecidas, al final la elección de una herramienta puede dar una ventaja en algún aspecto en particular pero no va a ser nunca el factor diferencial de cara al buen desempeño del proyecto.

El diseño y el estudio previo del proyecto si

Arquitectura de capas: hablar en terminos del negocio que se quiere manejar y separarlo de su implementacion real. Soy un firme defensor que la parte del codigo por la que realmente paga un cliente deberia ser capaz de leerla el mismo. Debe hacerse el esfuerzo de que la parte por la que se paga de un software sea legible por gente ajena al mundo del desarrollo de software

Problemas con el REST y el beneficio del RCP