Diapositiva 1

En este paper se hace una comparacion entre dos metodos de desarrollo de software:
El metodo catedral, donde el autor aclara que creia que el soft debe desarrollarse como una catedral, cuidadosamente elaborado por gente sumamente capaz y sin liberar versiones beta antes de tiempo.

Diapositiva 2

1. Todo buen trabajo de software comienza a partir de las necesidades personales del programador. (Todo buen trabajo empieza cuando uno tiene que rascarse su propia comezón)

Diapositiva 3

2. Los buenos programadores saben qué escribir. Los mejores, que reescribir (y reutilizar).

Diapositiva 4

3. "Contemple desecharlo; de todos modos tendrá que hacerlo." (Fred Brooks, The Mythical Man-Month, Capítulo 11

Diapositiva 5

4. Si tienes la actitud adecuada, encontrarás problemas interesantes

Diapositiva 6

5.La quinta leccion dice que “*Cuando se pierde el interés en un programa, el último deber es heredarlo a un sucesor competente”*.

Esta fue la actitud de Carl Harris. Con un programa abandonado, acordaron que los más lógico era que él asumiera el control del proyecto. De esta manera heredó Pop Client. Y junto con esto, lo cual nos lleva a su siguiente lección...

Diapositiva 7

6. … que dice “*Tratar a los usuarios como colaboradores es la forma más apropiada de mejorar el código, y la más efectiva de depurarlo”*.

… heredó su base de clientes. Estos usuarios con un estímulo diagnosticaran problemas, lo cual reduce notablemente el tiempo de depuración de un programa.

Diapositiva 8

7. La siguiente lección haciendo referencia al software dice que “*Libere rápido y a menudo, y escuche a sus clientes”.*

Linus, trataba a sus usuarios como colaboradores. Los repositorios mostraban una gran actividad. su logro fue llevarlo a un nivel de intensidad que estaba acorde con la complejidad de lo que estaba desarrollando.

Diapositiva 9

8. La siguiente lección “*Dada una base suficiente de desarrolladores asistentes y beta-testers, casi cualquier problema puede ser caracterizado rápidamente, y su solución ser obvia al menos para alguien”.*

Es decir, "con muchas miradas, todos los errores saltarán a la vista". Estimulando y recompensando a los usuarios maximizar las horas-hombre de desarrollo.

Diapositiva 10

9. Volviendo al tema del proyecto “*Las estructuras de datos inteligentes y el código burdo funcionan mucho mejor que en el caso inverso”.*

El proyecto debe evolucionar en algo que se entienda. Un código complejo hacer que el programa pierda sentido y desmotiva el corregir errores de lo que no se entiende.

Diapositiva 11

10. Si usted trata a sus analistas (beta-testers) como si fueran su recurso más valioso, ellos le responderán convirtiéndose en su recurso más valioso.

1. Liberaba rápido y a menudo
2. Ampliaba mi lista de analistas de versiones beta
3. Efectuaba anuncios espectaculares cada vez que liberaba una nueva versión, estimulando a la gente a participar.
4. Y escuchaba a mis analistas asistentes,
    consultándolos decisiones referentes al diseño
    tomándolos en cuenta cuando me mandaban sus mejoras

Diapositiva 12

11. Lo más grande, después de tener buenas ideas, es reconocer las buenas ideas de sus usuarios. Esto último es a veces lo mejor.

Gracias a que el autor estimaba mucho a sus beta testers (usuarios), estos siempre estaban dispuestos a proponer ideas y soluciones altamente creativas, que nunca hubieran imaginado.
Muchas de las funcionalidades que fueron agregando, habian sido propuestas por sus usuarios. Tambien porque ellos muchas veces conocen el producto mejor que sus creadores.

Diapositiva 13

12. Frecuentemente, las soluciones más innovadoras y espectaculares provienen de comprender que la concepción del problema era errónea.
Cuando usted se topa con un muro durante el desarrollo -cuando la encuentra difícil como para pensar mas allá de la corrección que sigue- es, a menudo, la hora de preguntarse no si usted realmente tiene la **respuesta correcta**, sino si se está planteando la **pregunta correcta**. Quizás el problema requiere ser replanteado.

Diapositiva 14

13. "La perfección (en diseño) se alcanza no cuando ya no hay nada que agregar, sino cuando ya no hay algo que quitar."
No hay que vacilar en desechar alguna característica superflua
si puede hacerlo sin pérdida de efectividad. Mantener solo lo esencial hace que un proyecto sea mucho mas facil de mantener y ayuda a se pueda incrementar mas facilmente su robustez.

Diapositiva 15

14 Toda herramienta es útil empleándose de la forma prevista, pero una *gran* herramienta es la que se presta a ser utilizada de la manera menos esperada.
Muy relacionada a la anterior, desarrollar herramientas robustas nos permite que puedan cumplir mas de una funcion en un futuro. Facilitando el agregado de funcionalidades en iteraciones posteriores.

Diapositiva 16

Cuando su lenguaje está lejos de un Turing completo, entonces el azúcar sintáctico puede ser su amigo.

El azúcar sintáctico se refiere a todas esas construcciones que no aportan nueva funcionalidad a un lenguaje, pero que permiten que sea más fácilmente utilizable por seres humanos.
Aca podemos visualizarlo con un Ejemplo en java donde se define el mayor
ej if..
pero Usando el operador condicional puede reescribir el ejemplo anterior en una sola línea siendo mas legible el codigo en el primer ejemplo

Diapositiva 17

Un sistema de seguridad es tan seguro como sus secretos. Cuídese de los secretos a medias

En este caso al autor se le sugiere encriptar las contraseñas de acceso al sistema y guardarlas en un archivo interno. Pero esto no proporciona ninguna protecion adicional, sino una seguridad a medias como dice el autor.
Un hacker solo tendría un paso más para acceder a dicha clave.

Diapositiva 18

Para resolver un problema interesante, comience por encontrar un problema que le resulte interesante.

nos preguntamos: ¿que nos puede resultar mas interesante que resolver un problema propio?
Que es precisamente lo que pasó con el autor y la transformacion de PopClient a Fetchmail

Diapositiva 19

Si el coordinador de desarrollo tiene un medio al menos tan bueno como lo es Internet, y sabe dirigir sin coerción, muchas cabezas serán, inevitablemente, mejor que una.
Teniendo un medio de comunicación con las partes del equipo y sabiendo dirigir utilizando el consenso y buen trato, varias personas desarrollaran de mejor manera que una.

Diapositiva 20

Gracias al sw libre y todo el proceso q se dio a partir de la creacion de linux nos dio una base solida para las metologias agiles que tenemos hoy en dia, donde el cliente juega un papel importante dentro del proyecto y todo es mas rapido
por otro lado el autor del paper es muy claro dando los fundamentos de por qué funciona linux de esa forma tan poco convencional.