INSTITUTO TECNOLÓGICO
de Pabellón de Arteaga
TEC

Tareas Tercera Unidad

Materia:

Ingeniería de Software

Docente:

Eduardo Flores Gallegos

Alumno:

José Alejandro Chávez Rendón

Carrera:

Ingeniería en Tecnologías de la Información y Comunicaciones

Índice: No. Página

INSTITUTO TECNOLÓGICO
de Pabellón de Arteaga

TEC

Tarea 1

Reseña: "Las leyes en el desarrollo de Software"

Materia:

Ingeniería de Software

Docente:

Eduardo Flores Gallegos

Alumno:

José Alejandro Chávez Rendón

Carrera:

Ingeniería en Tecnologías de la Información y Comunicaciones

12/04/2019

Las leyes en el desarrollo de Software

Ficha:

Sommer, T. (18 de diciembre de 2017). *Famous Laws Of Software Development.* Antwerp, Bélgica: Ghost Foundation. Recuperado el 10 de abril de 2019, de: https://www.timsommer.be/famous-laws-of-software-development/.

Tim Sommer es un desarrollador web y de Windows Insider con más de 9 años de experiencia en el tema .NET y con un gran enfoque en el desarrollo de HTML5 y Java Script. Lo que aborda en su artículo en inglés “Famous Laws Of Software Development” son las leyes más utilizadas y famosas sobre el Desarrollo de Software, establecidas por científicos y académicos ilustrados en el tema, es una recopilación de dichas leyes, la interpretación del autor y sus pensamientos, además el articulo incluye la explicación de las leyes de forma entendible y aún así más detallada.

El objetivo del autor en este artículo es que conozcamos esas leyes, principios y reglas de las que mucha gente habla en internet, esto debido a que muchas veces tratamos de simular que entendemos una plática sobre Desarrollo de Software, en la cual se toca el tema de las leyes o se nombra a algún autor de las mismas sin que nosotros entendamos de que se habla , para eso el autor da a conocer una colección de las leyes de Desarrollo de Software más utilizadas así como su propia interpretación de cada una de ellas, nos dice con las palabras originales en

qué consisten e incluso las explica con palabras más entendibles. Todo esto nos deja un artículo extenso con bastante información interesante, el autor, consciente de eso, incluye un índice para dirigirnos a la parte que sea más de nuestro interés. Luego de una breve introducción en la que el autor nos hace entrar en el contexto de lo que tratara el artículo, tenemos el índice con cada tema y cada ley que lo conforma. Lo siguiente es la primera ley, una muy famosa que no solo es aplicable al Desarrollo de Software que nos deja como conclusión que una computadora hace lo que el usuario escribe y no lo que el usuario quiere. Seguimos con la ley de Brook que establece que si un proyecto está retrasado, el hecho de añadir personas al mismo lo hará ir peor. La siguiente es la ley de Hofstadter que también puede aplicarse a otros temas dado que habla de cómo es complicado estimar el tiempo que se tomara para llevar a cabo tareas en el desarrollo de software. Toda esta información se complementa con dibujos y diagramas que le añaden un toque extra al escrito. La final el autor nos deja su conclusión dejando claro que el objetivo se cumplió, ahora tenemos una lista de las leyes más usadas y famosas en el desarrollo de software, cuando estemos hablando sobre el tema, podremos entender sobre el asunto y compartir nuestra opinión.

Para mí es un articulo muy interesante, aun sí el lector no se relaciona con desarrollar software, es información útil y sí se está trabajando un proyecto de software es una lectura casi obligada, deja muy claras las leyes y nos da una nueva idea de lo que hay que hacer cuando se programa algo, un artículo muy recomendado.

Chávez Rendón José Alejandro.

Tarea 2

Reseña: "50 años de Ingeniería de Software"

Materia:

Ingeniería de Software

Docente:

Eduardo Flores Gallegos

Alumno:

José Alejandro Chávez Rendón

Carrera:

Ingeniería en Tecnologías de la Información y Comunicaciones

17/04/2019

50 años de Ingeniería de Software

Ficha:

Oktaba, H. (27 Octubre de 2018). *50 Años de la Ingeniería de Software Problemas, Logros, Tendencias y Retos.* San Luis Potosí, México: SG Software Guru. Recuperado el 15 de Abril de 2019, de: https://sg.com.mx/revista/58/50-anos-de-la-ingenieria-de-software-problemas-logros-tendencias-y-retos.

Conferencista, profesora en la UNAM, pionera en el área de Ingeniería de Software en México, y autora de varios artículos muy interesantes sobre Ingeniería de Software; La Dra. Hanna Oktaba es una figura más que importante en la comunidad y su objetivo principal es generar y compartir conocimiento a través de la creación y promoción de estándares.

La Dra. Hanna se basa en el reporte de la reunión efectuada en 1968 para fundamentar su columna y comparar con la época actual lo que existía en ese entonces, y el resto, lo basa en la conferencia "The Future of Software Engineering", para hablar sobre los posibles problemas a los que se enfrenta la Ingeniería de Software.

La autora cumple su objetivo en la columna "50 Años de la Ingeniería de Software Problemas, Logros, Tendencias y Retos" escrita por ella misma sobre una conferencia en la que participó en San Luis Potosí, México. La columna aborda el 50 aniversario del nacimiento formal de la Ingeniería de Software, un evento importante que comenzó con la llamada Crisis de Software en 1968, aborda

también los problemas principales existentes en esa época; los cuales fueron identificados por los asistentes a la conferencia, los principales logros más importantes registrados hasta 2018 y nos deja también su propia visión o analogía sobre cada tema tratado.

Luego de una introducción en la que se menciona lo anterior a grandes rasgos y de dar el crédito a los primeros eventos de la Ingeniería de Software, la autora continúa su columna con los problemas y logros relacionados al proceso de desarrollo de software, problemas que según ella, aun hoy en día y a pesar del avance, siguen presentándose. Lo primero es la calidad del software, que si bien ha mejorado bastante, parece que para lograr la calidad deseada se busca probar y probar un software para hacer las correcciones necesarias en vez de tratar de prevenir los defectos del mismo, esto referente al producto, porque para el proceso se han introducido mas y mas técnicas de validación con el paso del tiempo. Seguimos con los costos, en 1968 se tomaba en cuenta el hardware, lo que complicaba estimar el valor del software, por ello se empezó a medir el mismo con diferentes técnicas, como la cantidad de líneas de código, que no es tan efectiva pues varía dependiendo del lenguaje con que se programe, y también una serie de parámetros que ahora conforman el estándar ISO/IEC 14113. Aún así, la autora afirma que el no poder tocar el software ni medirlo o pesarlo sigue provocando estragos al estimar costos, un problema notorio sobre todo con personas ajenas al desarrollo de software, a las que les parece que el costo para crear software y el tiempo invertido es exagerado.

El siguiente apartado en la columna es la gestión, un rubro problemático en el 68, debido a que a menudo es difícil cumplir con fechas, lo que tampoco ayuda a realizar una evaluación real del avance de los proyectos, afortunadamente esto se solventa con conocimientos prestados de otras disciplinas como la administración de proyectos. Por último se aborda la profesión como tal, en 1968 su problema principal era que apenas comenzaba, ahora ganado mucha importancia y desde años atrás a la fecha, muchas universidades han creado carreras relacionadas a la misma, pero aún falta mucho por hacer, según la doctora, aún hay que demostrar y crear conciencia sobre el impacto y la importancia de la Ingeniería de Software en la vida cotidiana, debido a que hoy en día está presente en muchos campos. Seguido, se habla sobre el futuro de la Ingeniería de Software, eso abarca los posibles problemas a los que se enfrenta y las modificaciones que se tendrían que hacer, todo esto por las nuevas tendencias, las nuevas tecnologías, las nuevas aplicaciones de la Ingeniería y los nuevos retos para los programadores, como la calidad y la seguridad del software, que como ya dio a conocer, no son temas fáciles de tratar, añadiendo a esto, un punto bastante importante hoy en día, reducir el consumo de energía, lográndolo haciendo que el software utilice lo menos posible de recursos de hardware. La columna termina con el lema de la Dra. Hanna: “La calidad de nuestras vidas depende de la calidad del software, y la calidad del software depende de la calidad de sus creadores y de las organizaciones que los respaldan.”

Para mí, es un artículo bastante interesante y muy informativo, ver hacía atrás y ver que se ha logrado referente al desarrollo de software nos pone a pensar sobre

que más se puede hacer y que necesitamos para lograrlo, independientemente de si el lector se relaciona de forma directa con software o no, la columna va recomendada para toda persona interesada en el tema. Nos deja de forma más entendible todo lo que implica hacer buen software así como todo lo que hoy en día tenemos a nuestro alcance para lograrlo.

Chávez Rendón José Alejandro.