Ingeniería en Sistemas Computacionales

Diseño de Software

Enfoques en el desarrollo del software

Marco Ricardo Cordero Hernández

Zapopan, Jal., 01 de septiembre de 2022

Modelos predictivos

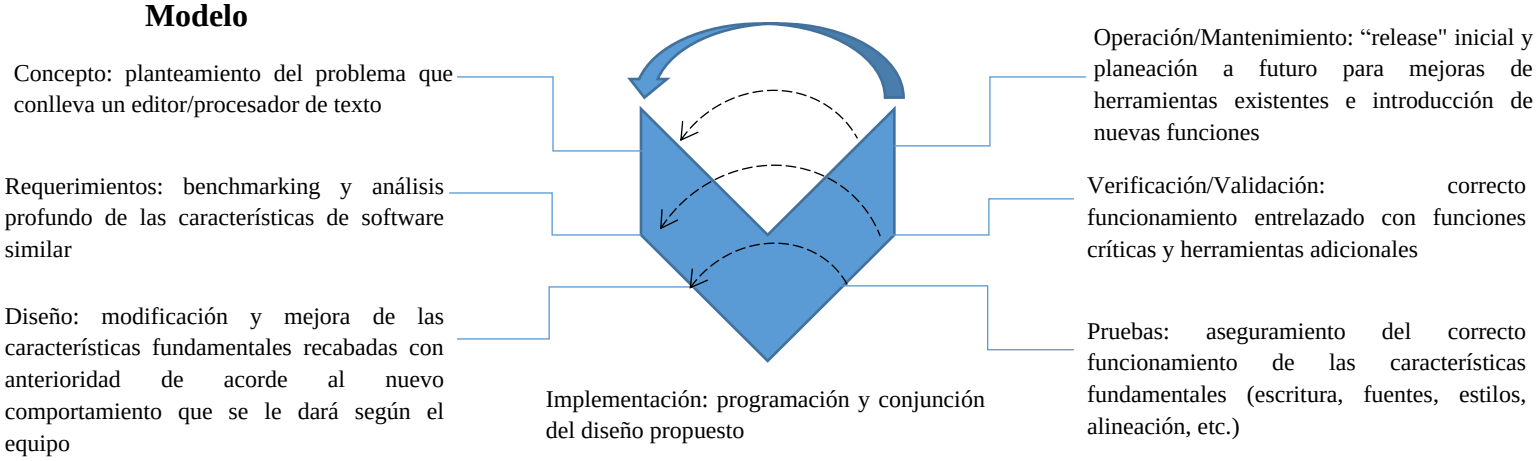
Se propone el siguiente enunciado:

> Un cliente espera que tú y tu equipo de cinco desarrolladores intrépidos escriban una aplicación de procesamiento de texto tan poderosa como *Microsoft Word* en los próximos tres meses.

¿Qué modelo predictivo usarías?
R: Modelo en V

¿Cuál sería la razón para usar tal modelo?
R: Poniendo en contraste otras características de modelos, como la necesidad de cuantiosos integrantes para *sashimi* o la necesidad de requerimientos definidos de *cascada*, el *modelo en V* toma en cuenta el delicado recurso del tiempo de manera que se combina con la característica incertidumbre que siempre viene acompañada con la falta de especificidad por parte del cliente. A pesar de ser una de las sumamente conocidas insignias de la ofimática global, *Word* no es específico en cuanto a su funcionamiento, después de todo, no es un software libre, por lo cual es que se debe partir puramente desde el núcleo y esencia de su construcción, partiendo desde las estructuras XML hasta el procesamiento de texto con algoritmos como *Knuth-Morris-Prat*. Hacer un bloc de notas libre de formato es sencillo, las codificaciones y extensiones también lo son, incluso pudiendo ser hecho en un par de días, no obstante, el formateo amigable al usuario siempre es lo más complejo. Al no contar con un equipo global como posiblemente lo hace *Microsoft*, la flexibilidad que ofrece el modelo en V resulta idónea para entregar un producto superior al MVP, haciendo posible un alcance superior a lo que un modelo "waterfall" podría entregar.

Modelo



Requerimientos de cada etapa

Concepto: propuesta abstracta del producto final; idealmente, todo el equipo estaría involucrado.

Requerimientos: investigación y obtención de funciones puntuales y críticas para la operación del software a desarrollar; serían necesarias entre 2 a 3 personas para esta etapa.

Diseño: esquema y diagramas de flujos para definir el comportamiento de la aplicación; de ser posible, comenzar a la par de la etapa anterior, contando con 3 personas para una función completa.

Implementación: código y algoritmos necesarios; todo el equipo debería estar presente al mismo tiempo.

Pruebas: de ser posible, a la par que nuevas funciones son implementadas, un integrante o un par de ellos debería estar probándolas de manera que generen resultados por si solas.

Verificación: a la par de las pruebas, la verificación podría concretarse después de que las pruebas unitarias son correctas y completas; de seguir con la implementación, un miembro podría desprenderse de ella y corroborar el adecuado funcionamiento de toda la arquitectura.

Operación: se espera que para este punto, los desarrolladores hallan concluido con sus labores, de forma que los problemas que pudieran surgir en producción sean resueltos inmediatamente, comenzando así nuevamente un nuevo ciclo.

Modelos iterativos e incrementales

Se propone el siguiente enunciado:

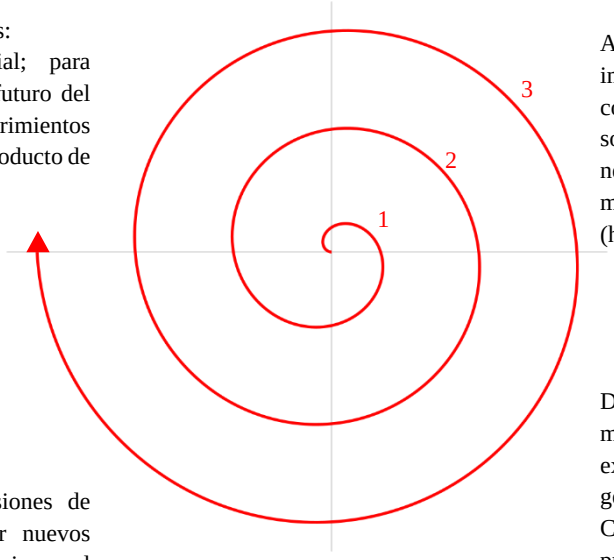
> Se planea implementar una aplicación médica (control de medicamentos) con 10 funcionalidades principales. Las especificaciones no están del todo claras. Sin embargo, el cliente desea comenzar con el proyecto lo antes posible e ir trabajando mientras se finaliza su desarrollo.

¿Qué tipo de enfoque para el desarrollo de software usarías?
R: Espiral (puro)

¿Conviene utilizar un modelo predictivo?
R: Según Stephens (2015), si no se conocen o no se entienden completamente los requerimientos del proyecto, lo más conveniente sería utilizar un enfoque iterativo. Particularmente para espiral, la permisividad que proporciona al momento de realizar los prototipos es la principal ventaja del mismo, involucrando tanto al equipo como al cliente, *algo que no está presente en los modelos predictivos*. Se pudieran realizar tantos prototipos como se requieran en cascadas incrementales, pero, si no se cuenta con un especialista en ciencias de la salud dentro del equipo, nunca se lograría lo que realmente quiere el cliente, el cual, de hecho, no estaría involucrado en ninguna etapa del proceso más que al inicio (más, probablemente, no en el pago final). La cuestión con espiral es que se asemeja a cascadas incrementales combinadas con retroalimentación, específicamente con la del cliente, por lo cual posiblemente podría sonar como algo viable un modelo predictivo, pero no hay que olvidar el objetivo final y el riesgo que puede conllevar el entregar un producto apresurado y construido ciegamente.

Diagrama

Objetivos/Alternativas/Restricciones: planeación de arquitectura inicial; para primeras iteraciones, visualizar el futuro del proyecto de manera que los requerimientos próximos sean compatibles con el producto de la iteración actual



Análisis de riesgo: tomar en cuenta las implicaciones tanto legales como éticas que conlleva desarrollar una herramienta como la solicitada sin el conocimiento necesario/adecuado; construir las bases de la mejora sobre lo que se encuentra actualmente (hablando de la iteración)

Desarrollo/Pruebas: proceder con la mayor medida de cautela o recurrir a datasets externas con el fin de entrenar a los modelos generados o corroborar resultados arrojados. Contactar al cliente con el fin de mostrar los progresos y comenzar a recolectar nueva información

Siguiente iteración: establecer sesiones de revisión con el cliente y recabar nuevos requerimientos con el fin de mejorar el producto de la siguiente iteración

Requerimientos de cada etapa

Objetivos/Restricciones: Analizar detalladamente la poca información brindada por el cliente, al menos para la primera iteración; Diseñar la siguiente fase del proyecto tomando en cuenta los avances anteriores con el fin de hacerlos compatibles con los que se harían a continuación.

Análisis de riesgo: Contactar con sectores externos y/o profesionales en la materia de forma que no se esté desarrollando un producto a ciegas.

Desarrollo/Pruebas: Seguir con el contacto anterior, buscando una validación profesional, y de ser posible, una involucración directa en el desarrollo del producto.

Siguiente iteración: Tomar en cuenta el “feedback” proporcionado por agentes externos y por el mismo cliente, existiendo algún tipo de documento estilo bitácora en donde se detallen las potenciales áreas de mejora y las mejores como requerimientos nuevos.

Referencias bibliográficas

Stephens, R. (2015). *Beggining Software Engineering*. Wrox Press.