CHAPTER 11

Documentación

Aparte del código mismo, que es lo más importante, el desarrollo de un producto de software o de un servicio debe generar documentación. Esta documentación servirá a los futuros desarrolladores para extender o modificar el producto con mayor facilidad o a los usuarios para sacar el máximo partido de él.

Basta mencionar la palabra documentación para que un desarrollador mire hacia el cielo o huya despavorido. Es algo necesario que debe hacerse, pero que a pocos les gusta hacer. Muchos desarrolladores creen que es suficiente con un "*help*" incorporado en el mismo *software.*

La documentación orientada al usuario suele ser bastante mala porque muchas veces es producida por los mismos que construyeron el código (ingenieros, técnicos) y no por gente especializada en transmitir en forma efectiva un mensaje en forma escrita.

La documentación orientada a los futuros desarrolladores muchas veces es inexistente. Esto es especialmente así en organizaciones que trabajan con metodologías ágiles, quienes suelen fundamentar su poca preocupación por una mala comprensión del principio de "*working software over comprehensive documentation*". Ello no significa que no haya que escribir ninguna documentación, solo que el código es más importante. Esta idea aparece porque en los procesos de desarrollo clásicos (cascada) era muy común generar abundante y muy voluminosa documentación.

11.1 Los dos grandes tipos de documentación

Existen dos grandes categorías de documentación que suelen confundirse. El primer tipo está dirigido al usuario final del producto o servicio y toma

la forma de un manual. Como además del usuario típico están los usuarios especializados, puede haber, además del manual de usuario, un manual de instalación y un manual de operación. Estos manuales pueden tomar forma impresa o ser implementados como ayuda online.

El segundo tipo de documentación es más técnico y está orientado a los ingenieros y técnicos que tendrán que mantener el sistema, ya sea para corregir algún problema como para hacer cambios o extender las funcionalidades existentes.

Estos dos tipos de documentación también existen en productos que no son de *software*.

11.2 Costos vs Beneficios

Evidentemente, el esfuerzo de producir documentación tiene un costo. Es necesario contratar personas que se dediquen a este trabajo o pedirle a quienes están generando código que se desvíen hacia otras tareas. Sin embargo, el generar documentación puede traer costos mucho mayores.

Partamos con la documentación orientada al usuario. Dejemos de lado por ahora el caso en que simplemente es una obligación (parte del contrato). El entregar un producto con documentación muy pobre o nula va a impactar en el soporte una vez que esté en manos del usuario, que muchas veces va a contactar a la empresa simplemente por no ser capaz de realizar alguna función. Por otra parte, un usuario que se encuentra con una duda y es capaz de resolverla rápidamente gracias a la documentación será un usuario agradecido y feliz.

En el caso de la documentación técnica es incluso mucho más crítico. La ausencia de documentación hará mucho más costosa la mantención. Muchas veces requerirá que en el equipo participe alguien que haya estado involucrado antes en el desarrollo. A veces eso no es posible y simplemente hay que adivinar.

La documentación técnica es relevante no solo para el futuro (extensiones y mantención) sino que puede ser muy útil durante el proceso de desarrollo mismo. Por ejemplo, puede servir como insumo para las revisiones formales de código o para revisar alguna decisión de diseño.

11.3 Documentación Técnica

Es posible que hayas oído cosas como “código autodocumentado” o la mejor documentación es el código mismo. Hay algo de verdad en que el código fuente puede ser considerado también parte de la documentación técnica, sobre todo si ha sido escrito teniendo esta finalidad en mente. Sin embargo, el código no cuenta toda la historia, o al menos es extremadamente difícil

extraerla a partir de sólo el código. Entender, por ejemplo, la arquitectura de un producto revisando los cientos de miles o millones de lineas de código repartidos en miles de archivos es casi imposible. Más aún, aunque lográsemos entender la arquitectura a partir de la lectura del código, no podríamos reconstruir la racionalidad que hubo para llegar a ese diseño, lo que puede ser muy importante al momento de revisar el producto para nuevas versiones.

Lo que necesitamos entonces es documentación que nos permita tener distintas visiones o perspectivas del producto. Para entender mejor esta idea piensa en el siguiente ejemplo: estás interesada en visitar la ciudad de Paris y por supuesto lo primero que piensas es en tener mapas que te permitan llegar a los distintos lugares. Pero aun siendo el mapa muy importante, hay otras perspectivas que te ayudarán mucho en la visita, como por ejemplo un mapa turístico en que aparezcan las principales atracciones, una guía de la ciudad, una guía de los restaurantes, etc.

Hay muchas perspectivas que pueden ayudar a entender el producto desde una perspectiva técnica. Por ejemplo, el modelo de datos, las restricciones, los principios considerados, etc. (ver figura).

11.4 Características de la buena documentación

Una buena documentación debe ser precisa, completa, eliminar al máximo posibles ambigüedades y tener un acceso fácil. No hay nada más frustrante que tener que acceder a la documentación, buscar afanosamente lo que nos interesa para justo darnos cuenta de que esa materia no está incluida en ninguna parte. Peor aún es encontrar la información, pero que ella sea incorrecta o no coincida con la realidad del producto.

Este último problema (que la documentación no coincide con la realidad) se produce principalmente porque el código puede haber sido modificado y no se ha hecho lo mismo con la documentación. Por ello es necesario que la documentación se ponga también bajo un control de versiones, ya que es un artefacto vivo que también evoluciona en el tiempo.

Algunas buenas prácticas relacionadas con la documentación son las siguientes:

- No exagerar con el esfuerzo en documentación en lo que se refiere al código (no más de un 10% del tiempo de desarrollo)
- Debe ser puesta bajo control de versiones
- Considerar el código como una componente de la documentación técnica: comentarios adecuados, uso de variables que digan algo, etc.
- Documentos de diseño cortos y solo de lo más relevante (evitar poner en UML cada elemento del código)
- Documentar la racionalidad del diseño porque no será obvio en unos años
- Documentar las lecciones aprendidas

- Usar formatos estándar (de la organización) para la documentación

11.5 Documentación de Usuario

No se trata de escribir cualquier cosa sobre como se usa el producto. El manual de usuario es parte de la experiencia de usuario. Por lo tanto, va a influir en que el usuario nos recomiende o no en el futuro en la misma forma que el producto mismo. Un buen manual de usuario les transmite a los usuarios que nos preocupamos de ellos.

Algunos elementos de una buena documentación son las siguientes:

- KYC - *know your customer*
- Usar un lenguaje simple en lo posible (evitar jerga técnica, abreviaturas, etc)
- Preferir siempre lo más simple simple
- Usar mucho el apoyo visual
- Usar *screenshots* con anotaciones
- Foco en los problemas, no en los *features*
- Tabla de contenidos
- Índice y/o buscador
- Jerarquía (capítulos, secciones, etc)
- Cuidar el diseño (fonts, espacios en blanco, colores, etc)
- Obtener *feedback* con beta *testers*
- Incluir *links* o referencias a más información