Requerimientos del Software

Prof. Emiliano Molina

CICLO DE VIDA SISTEMA

ANÁLSIS

En esta etapa el analista se encarga de estudiar los procesos de una empresa o sistema a fin de dar con el problema. Para ello utiliza diferentes herramientas que le permiten recolectar información y así poder encontrar los problemas. Esto no solo implica ver desde afuera la empresa, el analista debe hablar con las personas implicadas en los mismos (gerentes, empleados, etc) asi como ver documentación. A la gente no les simpatiza mucho que una persona analice y busque fallas, por lo que el analista hace un trabajo poco agradable.

¿Qué Problema/s tiene el cliente por lo que necesita un nuevo sistema?

Recolección de información

CUESTIONARIOS: Nos sirve para analizar varios aspectos de un proceso u obtener la opinión de un grupo grande de personas

ENTREVISTA: Nos sirve para analizar aspectos o personas de forma individual. Cuando necesitamos información específica sobre procesos suele ser la mejor herramienta.

REVSIÓN DE REGISTROS: Nos sirve para analizar aspectos relacionados con documentación de la empresa. Como pueden ser protocolos, manuales, datos de empleados, etc.

OBSERVACIÓN: Esta técnica es muy útil porque hay cosas que no se ven en los papeles, un analista que es buen observador puede darse cuenta de muchas cosas que sirven para estudiar el sistema

Fin del Análisis

- La etapa de análisis termina cuando entregamos de forma específica al los desarrolladores de sistemas los requisitos del sistema, esto se traduce en las nuevas funciones que debe tener el sistema. Pero a la hora de redactar los requisitos del nuevo sistema debemos tener en cuenta algunos factores.

Cómo debería ser una especificación de requisitos

- Completa: describe todas las necesidades relevantes.
- Consistente: carece de conflictos entre requisitos
- Correcta: todo es pertinente y no contiene errores
- Modificable: facilidad para efectuar cambios de forma sencilla, completa y consistente
- Verificable: existencia de un proceso acotado que determine si el sistema final satisface el requisito
- Trazable: el origen del requisito está marcado de forma clara; y se puede seguir el impacto del requisito a lo largo del SDLC
- Clara y no ambigua: una única interpretación

Redactar un requisito.

- Si bien la redacción de requisitos es importante, ustedes en la mayoría de los casos estén haciendo todo el trabajo solo, por ello es importante que sepan determinar los requerimientos del software. Es decir no siempre redactarán un documento con una serie de requisitos para el equipo de desarrollo porque serán ustedes mismos quienes hagan todo el trabajo. Pero saber estos conceptos le ayudarán a tener en claro en sus mentes qué debo saber (información) para saber qué voy a programar.

- ¿Cómo redactar un requisito? Siguiendo esta estructura:

Usuario	**El operador del *Call Center*…**
Acción	… deberá ser capaz de ver…
Objeto	… detalles de cada compañía contactada…
Cuantificador	… en menos de dos segundos.

- Ejemplo: El sistema debe ser capaz de importar ficheros ABC. El proceso debe ser amigable y rápido para el usuario

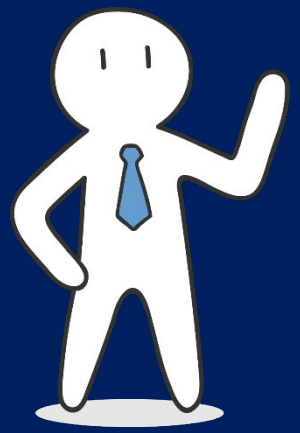
¿Está bien esta redacción? No. Por varios motivos, primero no debemos dar especificaciones de diseño (importar Ficheros ABC), segundo, no debemos usar abreviaturas(ABC), tercero no se usa un cuantificador (proceso amigable y rápido).

- Ejemplo: En mi opinión, ningún cliente debería poder nunca enviar órdenes al equipo de empaquetado. Ya lo hicimos así en un proyecto hace tres años y el resultado fue nefasto.

¿Está bien esta redacción? No. Por varios motivos, primero no debemos dar opiniones personales de un proceso (En mi opinión), segundo no es claro porque usa varias sentencias negativas (ningún cliente debería poder nunca)

DISEÑO

Vitruvio, romano crítico de arquitectura, afirmaba que los edificios bien diseñados eran aquellos que tenían resistencia, funcionalidad y belleza. Lo mismo se aplica al buen software. Resistencia: un programa no debe tener ningún error que impida su funcionamiento. Funcionalidad: un programa debe se apropiado para los fines que persigue. Belleza: la experiencia de usar el programa debe ser placentera.



¿Cómo lo solucionamos?

Principios generales del diseño

DESACOPLAMIENTO

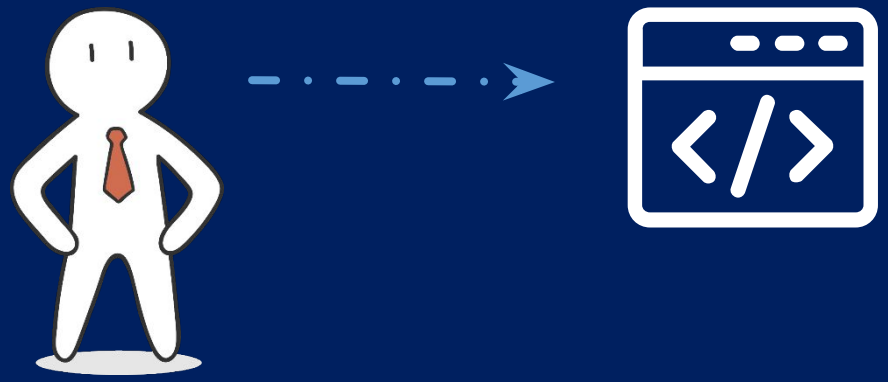
REUSABILIDAD

COHESIÓN

DESCOMPOSICIÓN

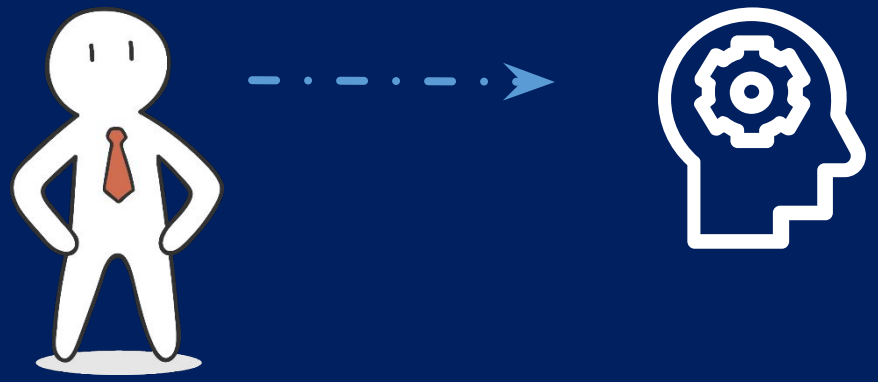
Desarrollo

En esta etapa se lleva a cabo el diseño del sistema, es decir, se codifica el programa. Pero no solo se realiza documentación sino que también se hace documentación del sistema, esto es, que los programadores deben explicar porqué ciertos procedimientos se codificaron de cierta forma o cómo. Esta documentación sirve para las etapas posteriores de prueba e implementación



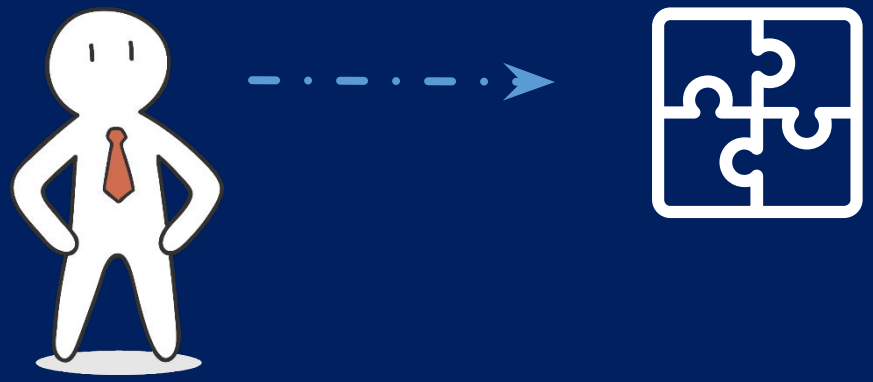
Prueba

En esta etapa se lleva a cabo una práctica conocida como “Testing”, en la que se prueba o testea el sistema terminado en un ambiente real de funcionamiento para poder ver el resultado y así poder encontrar posibles fallas para que el producto final sea sin fallas. En muchos casos se prueba con usuarios finales del sistema para que ellos mismos lo prueben.



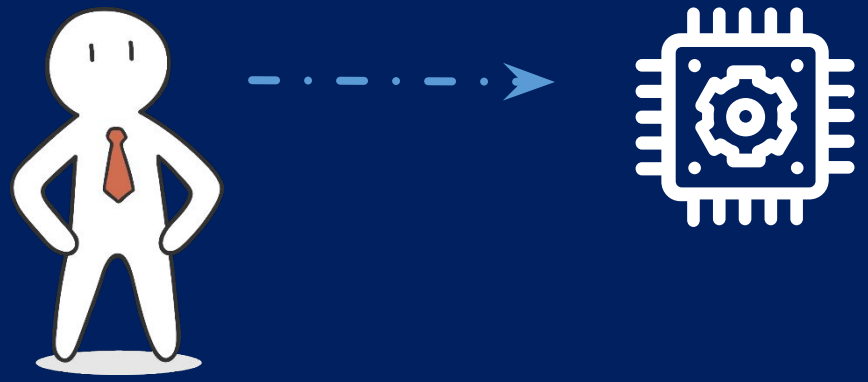
Implementación

En esta etapa instalamos el nuevo sistema funcionando en el entorno final, se capacita a los usuarios del mismo y se construyen todos los datos necesarios para su uso. Esta etapa es fundamental porque se capacita a los nuevos usuarios para el sistema.



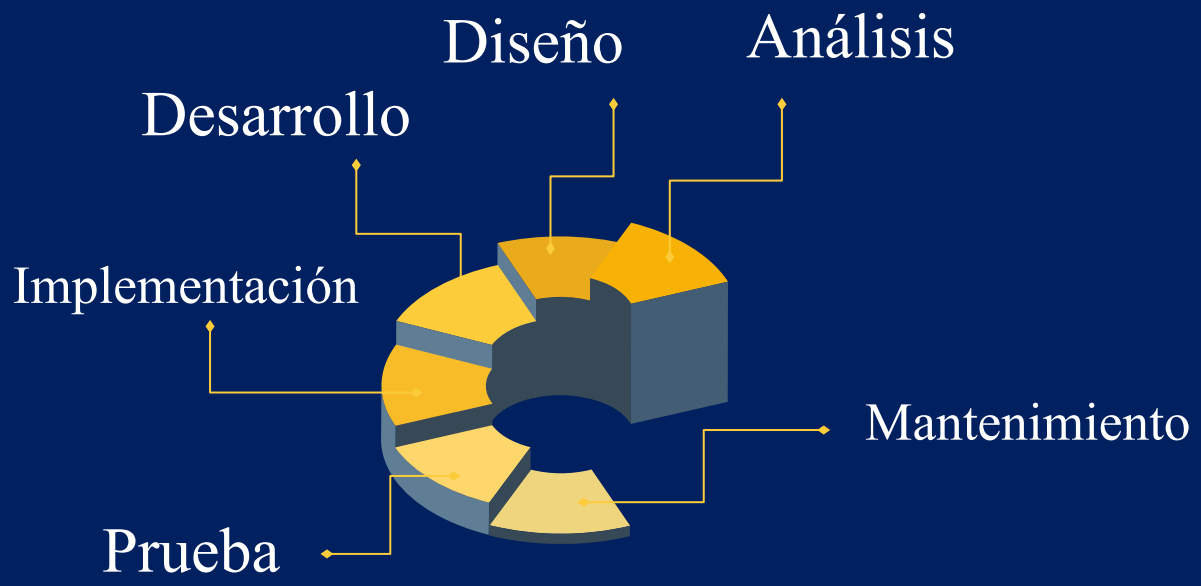
Mantenimiento

En esta etapa es cuando logramos descubrir las posibles mejoras al sistema al solucionar problemas. Evaluamos también el comportamiento del sistema. Esta etapa podría considerarse como una retroalimentación del sistema.



Metodologías de Análisis

- Estructuradas: Son metodologías que siguen una serie de etapas como las que hemos analizado de forma estructurada, es decir, una tras de otra no se puede realizar una etapa sin antes haber terminado la anterior



- Ágiles: las desventajas de las metodologías estructuradas es que no son flexibles a los cambios sobre la marcha, lo cual desemboca en pérdida de tiempo y dinero. Por ello las metodologías ágiles permiten lo contrario, son flexibles y tienen por objetivo la producción inmediata de software. Algunas ventajas:
  - Colaboración estrecha con el cliente
  - Predisposición y respuesta al cambio
  - Desarrollo incremental con entregas frecuentes
  - Comunicación verbal directa
  - Motivación, compromiso y responsabilidad del equipo por la autogestión
  - Simplicidad de procesos (sólo artefactos necesarios)
  - Evitar la burocracia innecesaria

Metodologías de Análisis

- En la próxima clase por zoom analizaremos una metodología ágil muy usada en la actualidad:

SCRUM