PROCESOS DE SOFTWARE, MODELOS Y COMPARATIVA

CRISTHIAN URREGO SALAZAR

CENTRO DE ESTUDIOS TECNOLÓGICOS DEL NORTE DEL VALLE COTECNOVA

TECNOLOGÍA EN SISTEMAS DE INFORMACIÓN

CARTAGO VALLE

2017

PROCESOS DE SOFTWARE, MODELOS Y COMPARATIVA

CRISTHIAN URREGO SALAZAR

NOMBRE PROFEROS

DOCENTE DISEÑO DE SISTEMAS DE INFORMACIÓN

CENTRO DE ESTUDIOS TECNOLÓGICOS DEL NORTE DEL VALLE COTECNOVA

TECNOLOGÍA EN SISTEMAS DE INFORMACIÓN

CARTAGO VALLE

2017

**QUE ES UN PROCESO DE SOFTWARE**

La meta de la ingeniería de software es construir productos de software, o mejorar los existentes; en ingeniería de procesos, la meta es desarrollar o mejorar procesos.

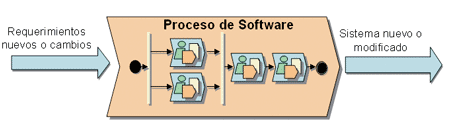
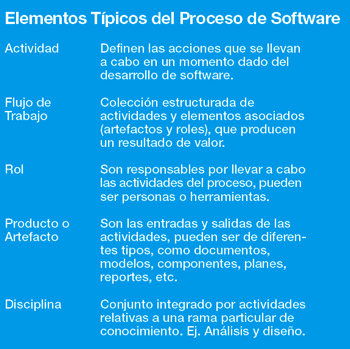
Un proceso de desarrollo de software es un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías, y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software.

Un proceso de software efectivo habilita a la organización a incrementar su productividad al desarrollar software:

* Permite estandarizar esfuerzos, promover reusó, repetición y consistencia entre proyectos.
* Provee la oportunidad de introducir mejores prácticas de la industria.
* Permite entender que las herramientas deben ser utilizadas para soportar un proceso.
* Establece la base para una mayor consistencia y mejoras futuras.

Un proceso de software mejora los esfuerzos de mantenimiento y soporte:

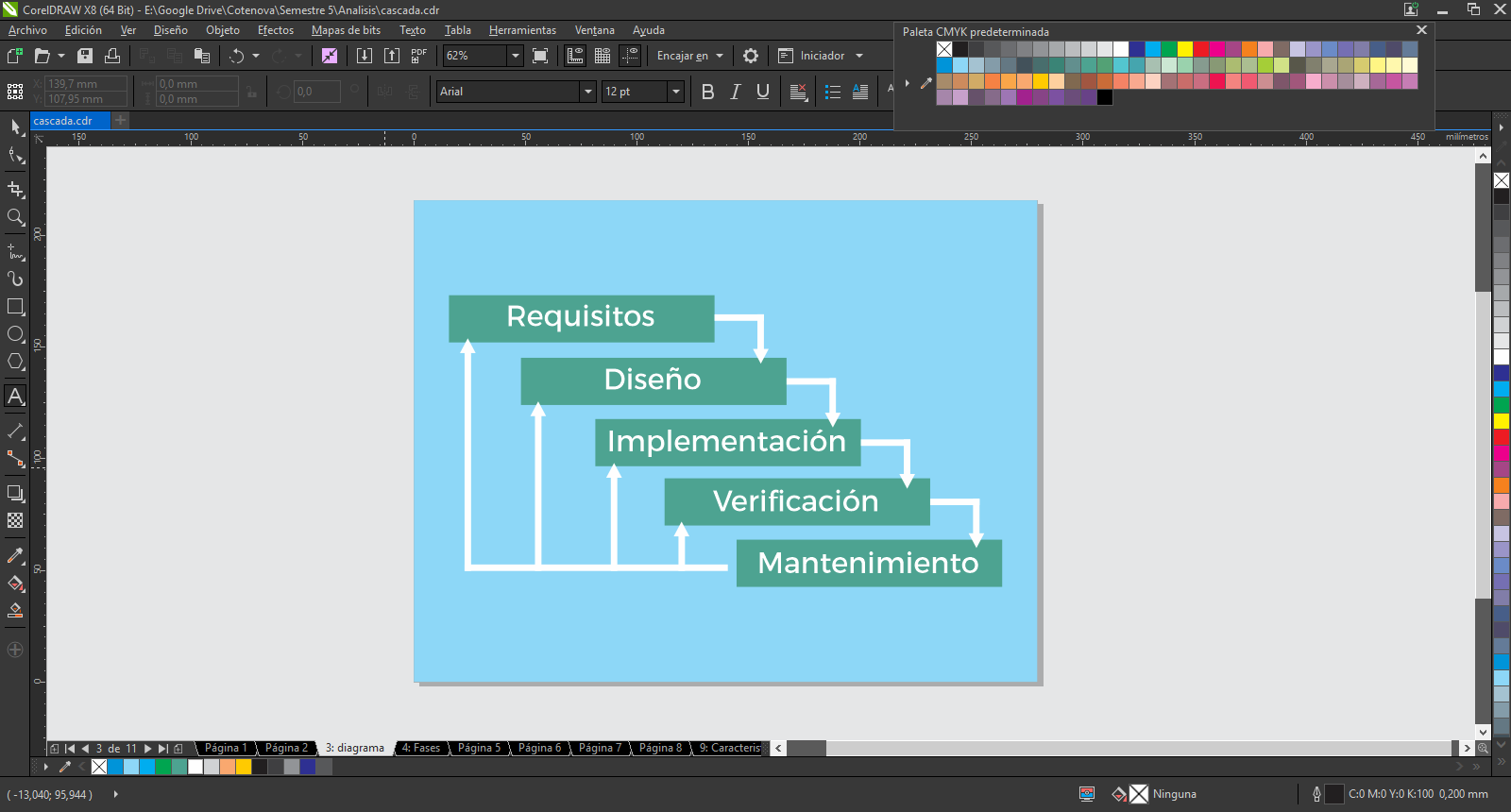
* Define cómo manejar los cambios y liberaciones a sistemas de software existentes.
* Define cómo lograr la transición del software a la operación, y cómo ejecutar los esfuerzos de operación y soporte.

#### MODELOS DE PROCESOS

**MODELO CASCADA**

Su visión es muy simple: el desarrollo de software se debe realizar siguiendo una secuencia de fases. Cada etapa tiene un conjunto de metas bien definidas y las actividades dentro de cada una contribuyen a la satisfacción de metas de esa fase o quizás a una subsecuencia de metas de la misma.



**FASES**

**Requisitos**

**Ingeniería y Análisis del Sistema:** Debido a que el software es siempre parte de un sistema mayor, el trabajo comienza estableciendo los requisitos de todos los elementos del sistema y luego asignando algún subconjunto de estos requisitos al software.

**Análisis de los requisitos del software:** el proceso de recopilación de los requisitos se centra e intensifica especialmente en el software. El ingeniero de software debe comprender el ámbito de la información del software así como la función, el rendimiento y las interfaces requeridas.

**Diseño**

El diseño del software se enfoca en cuatro atributos distintos del programa; la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del software con la calidad requerida antes de que comience la codificación.

**Implementación**

El diseño debe traducirse en una forma legible para la máquina. Si el diseño se realiza de una manera detallada, la codificación puede realizarse mecánicamente.

**Verificación**

Una vez que se ha generado el código comienza la prueba del programa. La prueba se centra en la lógica interna del software y en las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren.

**Mantenimiento**

El software sufrirá cambios después de que se entrega al cliente. Los cambios ocurrirán debidos a que se haya encontrado errores, a que el software deba adaptarse a cambios del entorno externo (sistema operativo o dispositivos periféricos) o a que el cliente requiera ampliaciones funcionales o del rendimiento.

**Características**

•Cada fase genera documentación para la siguiente. Esta documentación debe ser aprobada.

•Una fase no comienza hasta que la anterior ha terminado.

•Requiere disponer de unos requisitos completos y precisos al principio del desarrollo.

•Es una visión del proceso de desarrollo de software como una sucesión de etapas que producen productos intermedios.

•Para que el proyecto tenga éxito deben desarrollarse todas las fases.

Si se cambia el orden de las fases, el producto final será de inferior calidad.

**Ventajas**

•Se tiene todo bien organizado y no se mezclan las fases.

•Ayuda a localizar errores en las primeras etapas del proyecto a un bajo costo.

•Ayuda a minimizar los gastos de la planificación porque permite realizarla sin problemas.

**Desventajas**

•Gran dependencia en los requerimientos iníciales

•Difícilmente un cliente va a establecer al principio todos los requerimientos necesarios, por lo que provoca un gran atraso trabajando en este modelo, ya que este es muy restrictivo y no permite movilizarse entre fases.

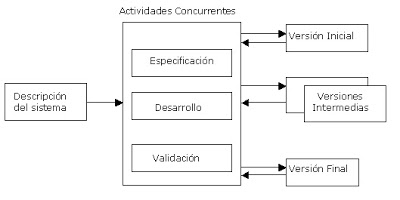
•El modelo genera pocos signos visibles de progreso hasta el final. Esto puede dar la impresión de un desarrollo lento, existe la incertidumbre de los clientes si sus proyectos serán entregados a tiempo.

•Inicio de la codificación muy tarde en el ciclo de vida del proyecto.

**MODELO EVOLUTIVO**

Los evolutivos son modelos iterativos, permiten desarrollar versiones cada vez más completas y complejas, hasta llegar al objetivo final deseado; incluso evolucionar más allá, durante la fase de operación. Los modelos “Iterativo Incremental” y “Espiral” (entre otros) son dos de los más conocidos y utilizados del tipo evolutivo.

La idea detrás de este modelo es el desarrollo de una implantación del sistema inicial, exponerla a los comentarios del usuario, refinarla en N versiones hasta que se desarrolle el sistema adecuado.Una ventaja de este modelo es que se obtiene una rápida realimentación del usuario, ya que las actividades de especificación, desarrollo y pruebas se ejecutan en cada iteración.



**Existen dos tipos de desarrollo evolutivo:**

·       Desarrollo Exploratorio: El objetivo de este enfoque es explorar con el usuario los requisitos hasta llegar a un sistema final. El desarrollo comienza con las partes que se tiene más claras. El sistema evoluciona conforme se añaden nuevas características propuestas por el usuario.

·       Enfoque utilizando prototipos: El objetivo es entender los requisitos del usuario y trabajar para mejorar la calidad de los requisitos. A diferencia del desarrollo exploratorio, se comienza por definir los requisitos que no están claros para el usuario y se utiliza un prototipo para experimentar con ellos. El prototipo ayuda a terminar de definir estos requisitos.

**VENTAJAS**

·       La especificación puede desarrollarse de forma creciente.

·       Los usuarios y desarrolladores logran un mejor entendimiento del sistema. Esto se refleja en una mejora de la calidad del software.

·       Es más efectivo que el modelo de cascada, ya que cumple con las necesidades inmediatas del cliente.

**DESVENTAJAS**

·       Proceso no Visible: Los administradores necesitan entregas para medir el progreso. Si el sistema se necesita desarrollar rápido, no es efectivo producir documentos que reflejen cada versión del sistema.

·       Sistemas pobremente estructurados: Los cambios continuos pueden ser perjudiciales para la estructura del software haciendo costoso el mantenimiento.

·       Se requieren técnicas y herramientas: Para el rápido desarrollo se necesitan herramientas que pueden ser incompatibles con otras o que poca gente sabe utilizar.

**METODOS FORMALES**

La denominación métodos formalesse usa para referirse a cualquier actividad relacionada con representaciones matemáticas del software, incluyendo la especificación formal de sistemas, análisis y demostración de la especificación, el desarrollo transformacional y la verificación de programas. Todas estas actividades dependen de una especificación formal del software.

Una especificación formal del software es una especificación expresada en un lenguaje cuyo vocabulario, sintaxis y semántica están formalmente definidos. Esta necesidad de unadefinición formal significa que los lenguajes de especificación deben basarse en conceptos matemáticos cuyas propiedades se comprendan bien. La rama de las matemáticas usada es la de matemática discreta, y los conceptos matemáticos provienen de la teoría de conjuntos, la lógica y el álgebra.

En la década de los 80, muchos investigadores de ingeniería del software propusieron que el uso de métodos formales de desarrollo era la mejor forma de mejorar la calidad del software. Argumentaban que el rigor y el análisis detallado, que son una parte esencial de los métodos formales, podrían dar lugar a programas con menos errores y más adecuados a las necesidades de los usuarios. Predijeron que, en el siglo XXI, una gran proporción del software estaría desarrollado usando métodos formales.

Claramente, esta predicción no se ha hecho realidad. Existen cuatro razones principales para esto:

*1. Una ingeniería del software exitosa.*El uso de otros métodos de ingeniería del software como los métodos estructurados, gestión de configuraciones y ocultación de la información en el diseño del software y procesos de desarrollo ha conducido a mejoras en la calidad del software. La gente que sugirió que la única forma de mejorar la calidad del software era usando métodos formales estaba claramente equivocada.

*2. Cambios en el mercado.*En la década de los 80, la calidad del software fue vista como un problema clave de la ingeniería del software. Sin embargo, desde entonces, la cuestión crítica para muchas clases de desarrollo del software no es la calidad, sino la oportunidad de mercado. El software debe desarrollarse rápidamente, y los clientes están dispuestos a aceptar software con algunos defectos si se les entrega rápidamente. Las técnicas para el desarrollo rápido del software no funcionan de forma efectiva con las especificaciones formales. Por supuesto, la calidad todavía es un factor importante, pero debe lograrse en el contexto de entrega rápida.

*3. Ámbito limitado de los métodos formales.*Los métodos formales no son muy apropiados para la especificación de interfaces de usuario e interacciones del usuario. El componente de interfaz de usuario se ha convertido en una parte cada vez mayor de la mayoría de los sistemas, de manera que realmente s6lo pueden usarse métodos formales cuando se desarrollan las otras partes del sistema.

*4. Escalabilidad limitada de los métodos formales.*Los métodos formales todavía no son muy escalables. La mayoría de los proyectos con éxito que han usado estas técnicas han estado relacionados con núcleos de sistemas críticos relativamente pequeños. A medida que los sistemas incrementan su tamaño, el tiempo y esfuerzo requerido para desarrollar una especificación formal crece de forma desproporcionada.

**VENTAJAS**

•Se comprende mejor el sistema.

•La comunicación con el cliente mejora ya que se dispone de una descripción clara y no ambigua de los requisitos del usuario.

•El sistema se describe de manera más precisa.

•El sistema se asegura matemáticamente que es correcto según las especificaciones.

•Mayor calidad software respecto al cumplimiento de las especificaciones.

•Mayor productividad

**DESVENTAJAS**

•El desarrollo de herramientas que apoyen la aplicación de métodos formales es complicado y los programas resultantes son incómodos para los usuarios.

•Los investigadores por lo general no conocen la realidad industrial.

•Es escasa la colaboración entre la industria y el mundo académico, que en ocasiones se muestra demasiado dogmático.

•Se considera que la aplicación de métodos formales encarece los productos y ralentiza su desarrollo.

**BIBLIOGRAFIA**

EduRed. (25 de Agosto de 2017). *EduRed*. Obtenido de https://www.ecured.cu/Modelo\_en\_cascada

Jortiz. (25 de Agosto de 2017). Obtenido de http://tema3isoftware.blogspot.com.co/p/modelo-de-metodos-formales.html

Ruvalcaba, M. (s.f.). *Procesos de Software*. Recuperado el 23 de 08 de 2017, de SG Buzz: https://sg.com.mx/revista/1/procesos-software#.WZ3alVFYGUk

Soloriio, M. (25 de Agosto de 2017). *metodologiaencascada* . Obtenido de blogspot: http://metodologiaencascada.blogspot.com.co/

Trejos, J. (25 de Agosto de 2017). Obtenido de http://jorgetrejos.blogspot.com.co/2010/08/modelo-evolutivo.html

Zulay, M. (25 de Agosto de 2017). Obtenido de http://modelosprocesosdesoftware.blogspot.com.co/p/modelo-en-cascada-o-lineal-secuencial.html