8ª IGDS

Modelos más utilizados en el SDLC.

Luis Alejandro Gomez Santillán

UTXJ

8ª IGDS

Modelos Más Utilizados En El SDLC

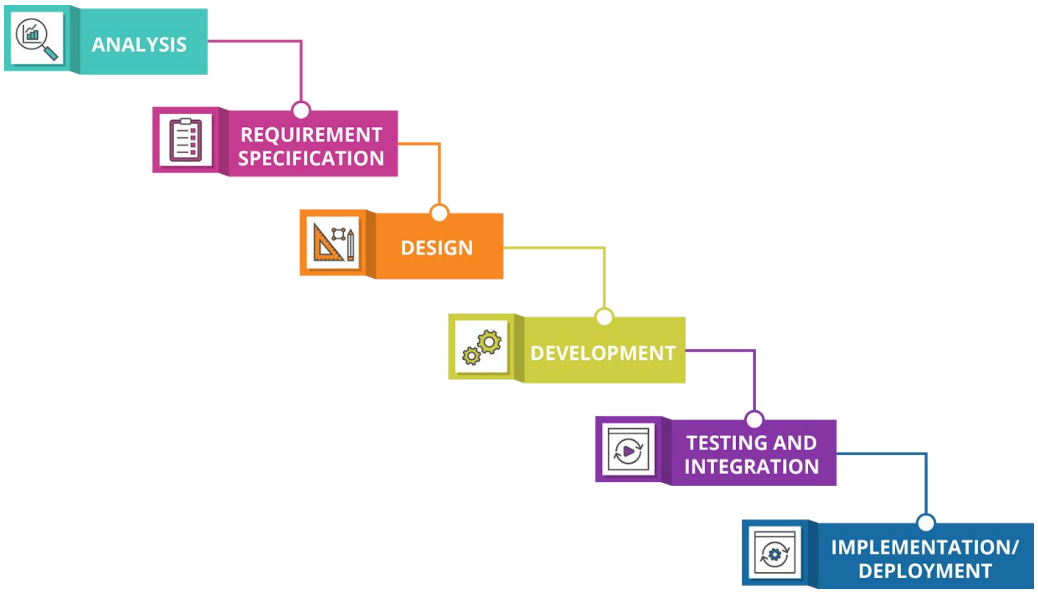
Existen varios modelos para agilizar el proceso de desarrollo. Cada uno tiene sus pros y sus contras, y es hasta el equipo de desarrollo a adoptar la más adecuada para el proyecto. A veces una combinación de los modelos pueden ser más adecuados.

1 Modelo En Cascada

El modelo muestra un proceso de cascada, donde los desarrolladores deben seguir estas fases a fin de:

1. Especificación de Requisitos (Análisis de requerimientos)
2. Diseñar
3. Aplicación (o codificación)
4. Integración
5. Pruebas (o de validación)
6. Implementación (o instalación)
7. Mantenimiento

En un modelo de cascada estricta, después de finalizar cada fase, se procede a la siguiente. Las revisiones pueden ocurrir antes de pasar a la siguiente fase que permite la posibilidad de cambios (que puede implicar un proceso formal de control de cambios). Sin embargo, se desaconseja volver a examinar y revisar cualquier fase anterior una vez que esté completo. Esta "rigidez" en un modelo en cascada pura ha sido una fuente de críticas por otras más "flexible" modelos.



2 Modelo De Espiral

La principal característica de un modelo en espiral es la gestión del riesgo en las etapas regulares en el ciclo de desarrollo. Un modelo de espiral de caracol a un número de iteración, los cuatro representantes, diagraman un cuadrante de las siguientes actividades respectivamente:

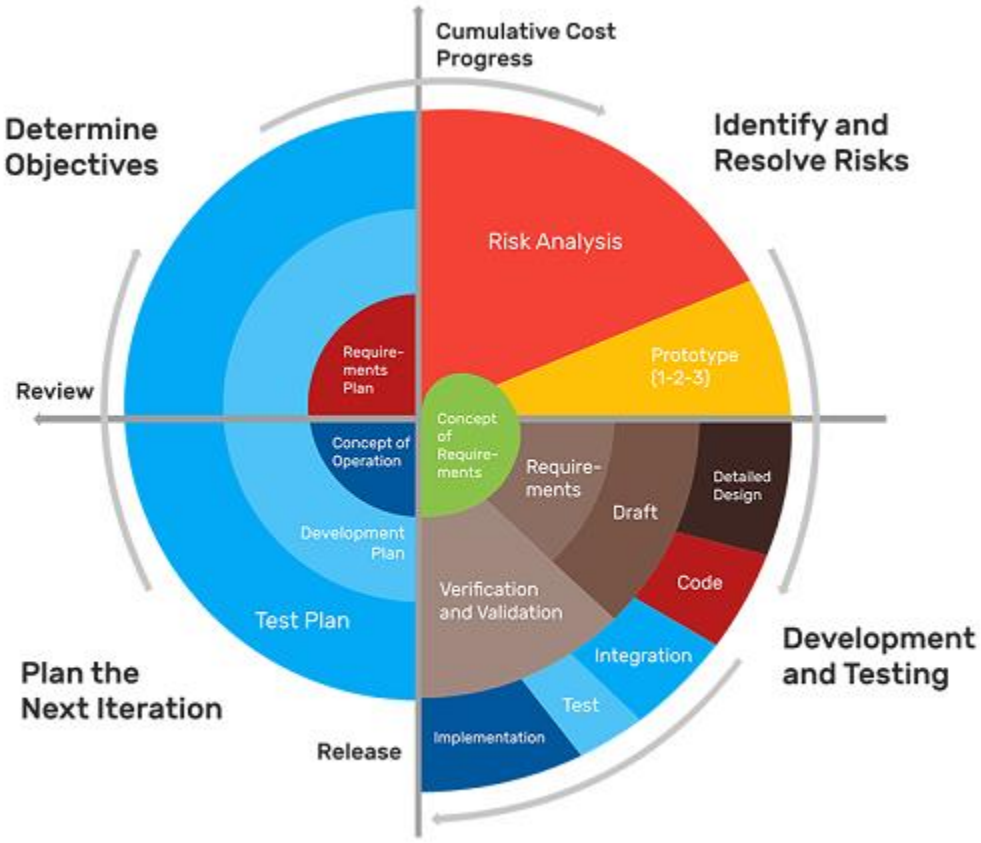
1.- formular planes a identificar blancos de software seleccionado para implementar el programa, aclarar las restricciones proyecto de desarrollo

2.- Análisis de riesgos: un análisis y evaluación de los programas seleccionados, para estudiar la manera de identificar y eliminar el riesgo

3.- La ejecución del proyecto: la aplicación de desarrollo de software y de verificación

4.- Evaluación de los clientes: la evaluación de la labor de desarrollo, la propuesta de enmiendas, los planes de formular el siguiente paso.

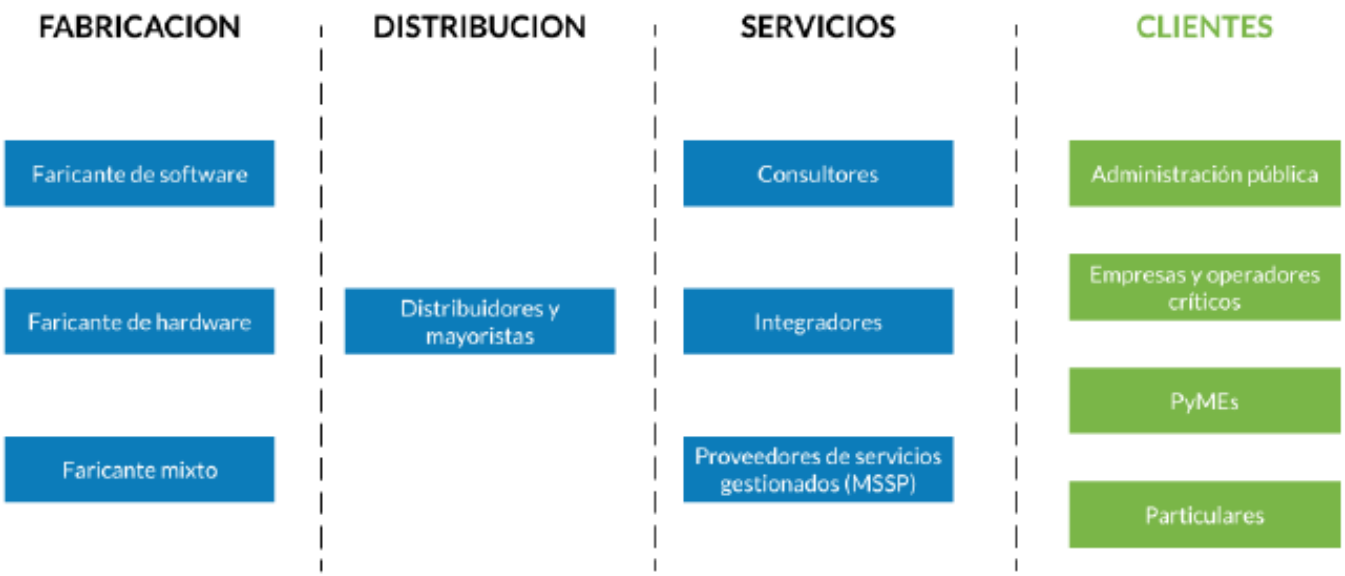
Riesgo modelo basado en espiral, haciendo hincapié en las condiciones de las opciones y limitaciones a fin de apoyar la reutilización de software, la calidad del software puede ayudar como un objetivo especial de la integración en el desarrollo de productos.



Spiral Model Methodology and its Phases

3 Desarrollo Iterativo Y Creciente Iterativo

Prescribe la construcción de porciones inicialmente pequeño pero cada vez mayor de un proyecto de software para ayudar a todos los participantes a descubrir aspectos importantes principios antes que los problemas o suposiciones erróneas pueden llevar al desastre. Los procesos iterativos son preferidos por los desarrolladores comerciales, ya que permite un potencial de alcanzar los objetivos de diseño de un cliente que no sabe cómo definir lo que quieren.



EVALUANDO ERP.COM

4 Agile Development

Ágiles de desarrollo de software de desarrollo iterativo utiliza como base, pero los defensores de un pueblo más ligero y más centrada en el punto de vista que los enfoques tradicionales. Los procesos ágiles de votos utilizar, en lugar de planificación, como su principal mecanismo de control. La respuesta es impulsado por las pruebas regulares y las liberaciones de los programas en constante evolución. Hay muchas variaciones de los procesos ágiles. XP (Extreme Programming) En XP, las fases se lleven a cabo en muy pequeña (o "continuo") las medidas frente a la antigua, "lote" los procesos. La (intencionalmente incompleta) de primer paso a través de los pasos que podría tomar un día o una semana, en vez de los meses o años de cada paso completo en el modelo de cascada. En primer lugar, se escribe pruebas automatizadas, para proporcionar metas concretas para el desarrollo. Lo siguiente es la codificación (por un par de programadores), que se completa cuando todas las pruebas pasan, y los programadores no pueden pensar en más pruebas que se necesitan. Diseño y arquitectura emergen de refactorización, y viene en pos de codificación. El diseño es realizado por las mismas personas que hacen la codificación. (El último rasgo - la fusión de diseño y código - es común a todos los demás procesos ágiles.) La incompleta pero funcional sistema se instala, ni ha demostrado para subconjunto (algunos de) los usuarios (al menos uno de los cuales está en el equipo de desarrollo). En este punto, los profesionales de empezar de nuevo a escribir ensayos para la parte más importante del sistema. Rational Unified Process Scrum

5 Proceso de Mejora de Modelos de Capability Maturity Model Integration

Modelo de Madurez de la Capacidad de Integración (CMMI) es uno de los principales modelos y basado en las mejores prácticas. Evaluaciones independientes de las organizaciones de grado de lo bien que siguen sus procesos definidos, no en la calidad de los procesos o el software producido. CMMI ha sustituido a la CMM. ISO 9000 ISO 9000 describe las normas para un proceso formalmente organizados para la fabricación de un producto y los métodos de gestión y seguimiento de los progresos realizados. Aunque la norma fue creada originalmente para el sector manufacturero, las normas ISO 9000 se han aplicado al desarrollo de software. Como CMMI, la certificación ISO 9000 no garantiza la calidad del resultado final, solo que los procesos de negocio formalizado se han seguido. ISO 15504 ISO 15504, también conocida como Proceso de Determinación de Software Mejora de las Capacidades (SPICE), es un "marco para la evaluación de los procesos de software". Esta norma está destinada a establecer un modelo claro para la comparación de procesos. SPICE se utiliza mucho como CMMI. Que los modelos de los procesos para administrar, controlar, orientar y supervisar el desarrollo de software. Este modelo se utiliza para medir lo que una organización de desarrollo o el equipo de proyecto en realidad lo hace durante el desarrollo de software. Esta información es analizada para identificar las debilidades y la mejora de la unidad. También identifica los puntos fuertes que puede mantenerse o integrarse en una práctica común para esa organización o equipo. Los métodos formales Los métodos formales son métodos matemáticos para la solución de software (y hardware) los problemas en los requisitos, especificaciones y los niveles de diseño. Ejemplos de los métodos formales son el B-Método, redes de Petri, lo que prueba automática de teoremas, plantear y VDM. Varias anotaciones especificación formal están disponibles, tales como la notación Z. Más en general, la teoría de autómatas se puede utilizar para construir y validar el comportamiento de la aplicación mediante el diseño de un sistema de máquinas de estados finitos. Máquina de estados finitos (FSM), metodologías basadas en la especificación de software permiten ejecutable y por paso de la codificación convencional. Los métodos formales son más susceptibles de ser aplicadas en el software de aviónica, en particular cuando el software es indispensable para la seguridad. Normas de garantía de software de seguridad, tales como los métodos de la demanda DO178B formal al más alto nivel de categorización (Nivel A). Formalización de desarrollo de software se infiltra en él, en otros lugares, con la aplicación de Object Constraint Language especializaciones (y como Java Modeling Language) y, especialmente, con el modelo impulsado por la arquitectura permite la ejecución de diseños, si no las especificaciones.