UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO

FACULTAD DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN YTELECOMUNICACIONES

Nombre: Jorge Arturo Aliaga Valencia
Registro: 218166141

Materia: Sistemas De Información I
Semestre: 1-2024

Conceptos Generales

1. ¿Qué es un Ciclo de Vida?

- Un ciclo de vida en ingeniería de software es un conjunto de etapas por las que pasa un producto de software desde su concepción hasta su retirada del mercado. Estas etapas incluyen el desarrollo, la implementación, el mantenimiento y la retirada.

2. ¿Qué es un Proceso?

- Un proceso en ingeniería de software es un conjunto estructurado de actividades que guían la producción de un producto de software. Define quién hace qué, cuándo y cómo.

3. ¿Qué es un Producto de software?

- Un producto de software es cualquier programa o aplicación informática que se desarrolla para satisfacer las necesidades de los usuarios.

4) Actividades de los paradigmas de ciclos de vida:

- Cascada: Definición de requisitos, Diseño del sistema, Implementación y Pruebas, Integración y Pruebas de sistema, Operación y mantenimiento.
- Spiral: Identificación de objetivos, Análisis de riesgos, Desarrollo y validación, Planificación, Evaluación.
- Prototipos: Definición de requisitos, Desarrollo de prototipos, Evaluación del prototipo, Desarrollo del sistema.
- PUDS (Proceso Unificado de Desarrollo de Software): Inicio, Elaboración, Construcción, Transición.
- Scrum: Planificación de Sprint, Reuniones diarias, Revisión de Sprint, Retrospectiva de Sprint.
- DRA (Desarrollo Rápido de Aplicaciones): Planificación, Diseño del prototipo, Construcción del prototipo, Transición.
- Incremental: Planificación, Análisis, Diseño, Implementación, Pruebas, Evaluación.
- Desarrollo basado en componentes: Identificación de componentes, Análisis de requisitos, Selección de componentes, Integración y pruebas.
- Programación Extrema (XP): Planificación del juego, Diseño simple, Pruebas, Refactorización, Programación en pareja.
- Win Win: Identificación de objetivos, Negociación, Desarrollo, Implementación, Evaluación.
- Recursivo Paralelo: Planificación, Desarrollo, Pruebas, Integración.
- Framework: Definición de requisitos, Diseño del framework, Implementación de componentes, Integración.

5)Productos de Software

Aplicaciones de Escritorio:

Son programas diseñados para ser ejecutados en computadoras personales.

Ejemplos: Microsoft Word, Adobe Photoshop.

Aplicaciones Web:

Son programas accesibles a través de un navegador web.

Ejemplos: Google Docs, Facebook.

Aplicaciones Móviles:

Son programas diseñados para ser ejecutados en dispositivos móviles como smartphones y tablets.

Ejemplos: Instagram, WhatsApp.

Sistemas Operativos:

Software que controla el funcionamiento básico de una computadora y gestiona los recursos del sistema.

Ejemplos: Windows, macOS, Linux.

Sistemas de Gestión de Bases de Datos (DBMS):

Software diseñado para gestionar y manipular bases de datos.

Ejemplos: MySQL, Oracle Database.

Herramientas de Desarrollo:

Software utilizado por desarrolladores para crear, depurar y mantener otros programas.

Ejemplos: Visual Studio, IntelliJ IDEA.

Sistemas de Gestión Empresarial (ERP):

Software que integra y gestiona las principales funciones de una empresa como finanzas, recursos humanos, inventario, etc.

Ejemplos: SAP, Oracle ERP.

Sistemas de Control de Versiones:

Software que facilita la gestión de cambios en el código fuente de un programa.

Ejemplos: Git, Subversion.

Sistemas de Gestión de Contenidos (CMS):

Software utilizado para crear y gestionar contenido digital como sitios web y blogs.

Ejemplos: WordPress, Drupal.

Software Embebido:

Software diseñado para ejecutarse en dispositivos electrónicos integrados, como sistemas de control en automóviles, electrodomésticos, etc.

Ejemplos: Firmware de routers, sistemas de navegación GPS.

6)Características de los Productos de Software:

a) Funcionalidad: Debe realizar las tareas para las que fue diseñado de manera efectiva y eficiente.

b) Fiabilidad: Debe funcionar correctamente bajo condiciones normales y en situaciones excepcionales.

c) Usabilidad: Debe ser fácil de aprender y usar para los usuarios finales.

d) Eficiencia: Debe utilizar los recursos de manera eficiente, como memoria, CPU y almacenamiento.

e) Portabilidad: Debe poder ejecutarse en diferentes plataformas de hardware y software

f) Mantenibilidad: Debe ser fácil de modificar y mantener para corregir errores o agregar nuevas características.

g) Seguridad: Debe proteger los datos y la privacidad del usuario, así como también ser resistente a ataques maliciosos.

h) Escalabilidad: Debe poder adaptarse al crecimiento de la carga de trabajo y de usuarios sin perder rendimiento.

i) Interoperabilidad: Debe poder integrarse y funcionar con otros sistemas y aplicaciones de manera efectiva.

j) Documentación: Debe proporcionar documentación clara y completa para su instalación, configuración y uso.

Características y definición de paradigmas

a) Ciclo de Vida Clásico (Cascada):

- Características: Secuencial, fase posterior depende de la finalización de la fase anterior.
- Definición: Modelo lineal de desarrollo, donde cada fase debe completarse antes de pasar a la siguiente.

b) Spiral:

- Características: Basado en la gestión de riesgos, iterativo.
- Definición: Modelo de desarrollo iterativo que enfatiza la identificación y mitigación de riesgos.

c) Técnicas de Cuarta Generación:

- Características: Automatización del proceso de desarrollo, generación de código.
- Definición: Enfoque de desarrollo que se centra en la automatización de la creación de software.

d) Prototipos:

- Características: Iterativo, enfocado en la retroalimentación del cliente.
- Definición: Modelo donde se desarrollan prototipos para comprender los requisitos del cliente.

e) PUDS (Proceso Unificado de Desarrollo de Software):

- Características: Iterativo e incremental, centrado en la arquitectura.
- Definición: Marco de desarrollo que se centra en la gestión de requisitos, diseño y construcción del sistema.

f) Scrum (Manifiesto Ágil):

- Características: Iterativo, centrado en equipos autoorganizados y entregas rápidas.
- Definición: Metodología ágil que se centra en entregas incrementales y adaptabilidad a los cambios.

g) DRA (Desarrollo Rápido de Aplicaciones):

- Características: Enfoque iterativo, orientado a la rapidez de entrega.
- Definición: Modelo de desarrollo que se enfoca en la rápida entrega de aplicaciones mediante el uso de prototipos.

h) Incremental:

- Características: Desarrollo por etapas, entregas incrementales.
- Definición: Modelo donde el sistema se desarrolla en incrementos, agregando funcionalidad en cada iteración.

i) Desarrollo Basado en Componentes:

- Características: Reutilización de componentes, enfoque modular.
- Definición: Modelo que se centra en la reutilización de componentes de software existentes.

j) Programación Extrema (XP):

- Características: Enfoque colaborativo, pruebas continuas.
- Definición: Metodología ágil que se centra en la simplicidad, comunicación y retroalimentación constante.

k) Win Win:

- Características: Enfoque de negociación, satisfacción mutua.
- Definición: Modelo que se enfoca en la resolución de conflictos mediante la búsqueda de soluciones mutuamente beneficiosas.

l) Recursivo Paralelo:

- Características: Desarrollo simultáneo de múltiples componentes.
- Definición: Modelo donde se desarrollan diferentes partes del sistema de forma paralela.

m) Framework:

- Características: Estructura básica reutilizable, extensible.
- Definición: Modelo que proporciona una estructura básica para el desarrollo de software, permitiendo la extensibilidad y reutilización de componentes.

Ventajas y Desventajas

Ciclo de Vida Clásico (Cascada):

Ventajas:

- Estructura clara y fácil de entender.
- Requerimientos bien definidos desde el principio.
- Facilita la planificación y el seguimiento del proyecto.

Desventajas:

- Poca flexibilidad para cambios en los requisitos.
- Riesgo de sobrecostos y retrasos si los requisitos cambian.
- La retroalimentación del cliente se produce al final del ciclo, lo que puede llevar a desviaciones significativas respecto a las expectativas.

2) Spiral:

Ventajas:

- Enfoque en la gestión de riesgos, lo que permite identificar y mitigar problemas tempranamente.
- Adaptabilidad a cambios durante el desarrollo.
- Permite una mejor comprensión de los requisitos y del sistema en general.

Desventajas:

- Requiere un alto nivel de experiencia para gestionar los riesgos de manera efectiva.
- Puede resultar costoso y prolongar el tiempo de desarrollo debido a su naturaleza iterativa.
- La complejidad del modelo puede dificultar su implementación.

3) Técnicas de Cuarta Generación:

Ventajas:

- Automatización del proceso de desarrollo, lo que puede acelerar el tiempo de entrega.
- Mayor productividad gracias al uso de herramientas y generadores de código.
- Facilita el desarrollo de sistemas complejos en menos tiempo.

Desventajas:

- Requiere un alto nivel de experiencia en el uso de las herramientas y generadores de código.
- Puede resultar en una menor calidad del código generado.
- No es adecuado para proyectos que requieren un alto grado de personalización o control.

4) Prototipos:

Ventajas:

- Permite obtener rápidamente la retroalimentación del cliente.
- Facilita la identificación de requisitos y la validación de conceptos.
- Flexibilidad para realizar cambios durante el desarrollo.

Desventajas:

- Puede llevar a una arquitectura no sólida si no se gestiona adecuadamente.
- El cliente puede confundir un prototipo con el producto final.
- Puede resultar costoso si se construyen múltiples prototipos sin un plan claro.

5) PUDS (Proceso Unificado de Desarrollo de Software):

Ventajas:

- Enfoque iterativo e incremental que permite adaptarse a cambios en los requisitos.
- Centrado en la arquitectura del sistema, lo que facilita la gestión de la complejidad.
- Mayor visibilidad y control sobre el progreso del proyecto.

Desventajas:

- Requiere una planificación inicial detallada para definir las iteraciones.
- Puede resultar en una mayor complejidad si no se gestionan adecuadamente las dependencias entre los componentes.
- Requiere un equipo experimentado en la metodología para implementarla de manera efectiva.

6) Scrum (Manifiesto Ágil):

Ventajas:

- Entregas frecuentes y rápidas de funcionalidades.
- Mayor flexibilidad para adaptarse a cambios en los requisitos del cliente.
- Fomenta la colaboración y comunicación entre los miembros del equipo.

Desventajas:

- Requiere un compromiso constante por parte del equipo y del cliente.
- Puede resultar en una falta de documentación y planificación detallada.
- No es adecuado para proyectos con requisitos altamente definidos desde el principio.

7) DRA (Desarrollo Rápido de Aplicaciones):

Ventajas:

- Entrega rápida de prototipos y versiones preliminares del software.
- Mayor satisfacción del cliente al obtener resultados visibles en poco tiempo.

- Facilita la adaptación a cambios en los requisitos durante el desarrollo.

Desventajas:

- Requiere una fuerte implicación del cliente y del equipo de desarrollo.
- Puede resultar en una arquitectura no sólida si no se planifica adecuadamente.
- Menor énfasis en la documentación y en la calidad del código.

8) Incremental:

Ventajas:

- Entrega temprana de funcionalidades, lo que permite obtener retroalimentación del cliente rápidamente.
- Mayor flexibilidad para realizar cambios durante el desarrollo.
- Permite una gestión más eficiente de los recursos al distribuir el trabajo en iteraciones.

Desventajas:

- Requiere una planificación detallada para definir el alcance de cada iteración.
- Puede resultar en una mayor complejidad si no se gestionan adecuadamente las dependencias entre las funcionalidades.
- El cliente puede tener expectativas poco realistas sobre el alcance de cada iteración.

9) Desarrollo Basado en Componentes:

Ventajas:

- Reutilización de componentes, lo que reduce el tiempo y el costo de desarrollo.
- Mayor calidad y consistencia del software al utilizar componentes probados y bien establecidos.
- Facilita la adaptación a cambios en los requisitos mediante la sustitución o modificación de componentes.

Desventajas:

- Requiere una inversión inicial en la identificación y selección de componentes adecuados.
- Puede resultar en una mayor complejidad si no se gestionan adecuadamente las dependencias entre los componentes.
- La disponibilidad de componentes adecuados puede ser limitada en algunos casos.

10) Programación Extrema (XP):

Ventajas:

- Fomenta la comunicación y colaboración entre los miembros del equipo.
- Entregas frecuentes y rápidas de funcionalidades.

- Enfoque en la calidad del código mediante prácticas como pruebas continuas y refactorización.

Desventajas:

- Requiere un equipo altamente comprometido y autogestionado.
- Puede resultar en una menor documentación y planificación inicial.
- No es adecuado para proyectos con requisitos altamente definidos desde el principio.

11) Win Win:

Ventajas:

- Enfoque en la resolución de conflictos y en la búsqueda de soluciones mutuamente beneficiosas.
- Mayor satisfacción del cliente al involucrarlo en el proceso de toma de decisiones.
- Mejora la comunicación y la colaboración entre el equipo de desarrollo y el cliente.

Desventajas:

- Requiere un alto nivel de habilidades de negociación y resolución de conflictos.
- Puede resultar en compromisos que no satisfacen por completo las necesidades de todas las partes involucradas.
- Puede ser difícil de implementar en entornos con culturas organizativas rígidas.

12) Recursivo Paralelo:

Ventajas:

- Desarrollo simultáneo de múltiples componentes, lo que acelera el tiempo de entrega.
- Permite una mayor flexibilidad para adaptarse a cambios en los requisitos durante el desarrollo.
- Facilita la identificación temprana de problemas de integración.

Desventajas:

- Requiere una coordinación cuidadosa entre los equipos de desarrollo.
- Puede resultar en una mayor complejidad si no se gestionan adecuadamente las dependencias entre los componentes.
- Mayor riesgo de conflictos y problemas de comunicación entre los equipos.

13) Framework:

Ventajas:

- Proporciona una estructura básica y herramientas reutilizables para el desarrollo de software.
- Facilita la extensibilidad y la reutilización de componentes.
- Promueve buenas prácticas de diseño y desarrollo de software.

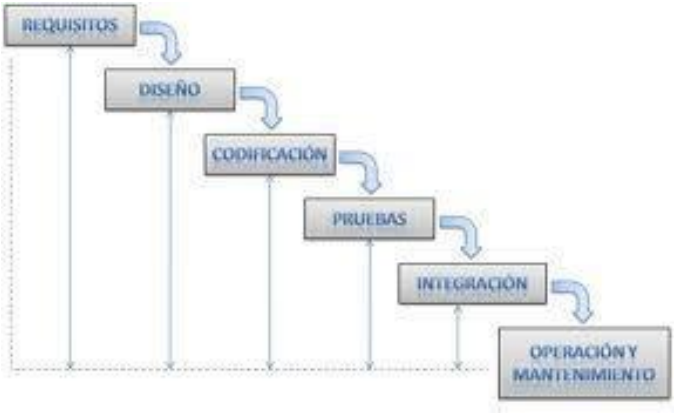
Desventajas:

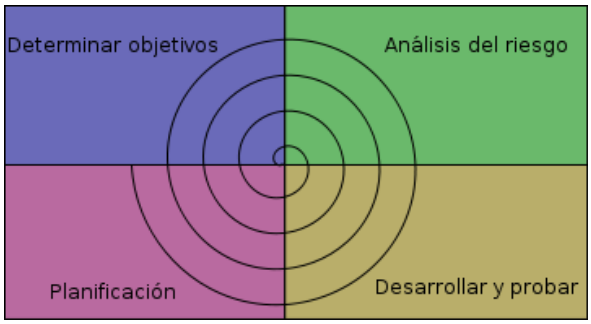
- Requiere una curva de aprendizaje para familiarizarse con el framework.
- Puede resultar en una dependencia excesiva del framework si no se utiliza correctamente.
- No es adecuado para todos los proyectos, especialmente aquellos que requieren una alta personalización o flexibilidad.

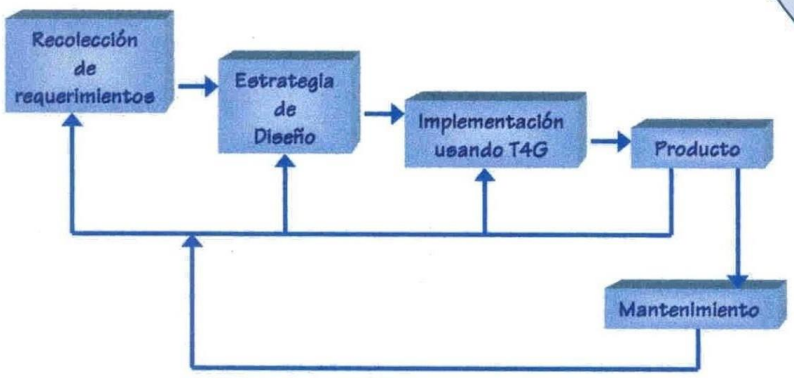
Ejemplo de Aplicación

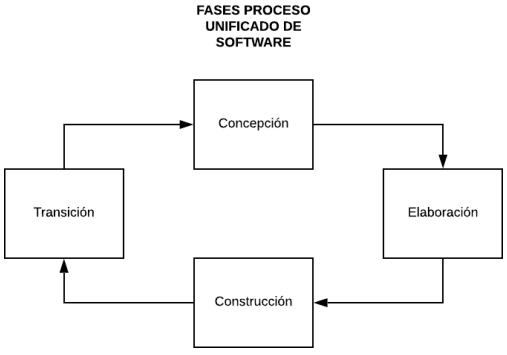
- Ciclo de Vida Clásico (Cascada): Desarrollo de sistemas de gestión empresarial. Tiempo de desarrollo: 6-12 meses.
- Spiral: Desarrollo de sistemas de control de tráfico aéreo. Tiempo de desarrollo: 1-2 años.
- Prototipos: Desarrollo de aplicaciones móviles. Tiempo de desarrollo: 3-6 meses.
- Scrum: Desarrollo de aplicaciones web. Tiempo de desarrollo: 3-9 meses.
- DRA (Desarrollo Rápido de Aplicaciones): Desarrollo de sitios web de comercio electrónico. Tiempo de desarrollo: 2-4 meses.
- Incremental: Desarrollo de sistemas de gestión de contenido. Tiempo de desarrollo: 6-12 meses.
- Desarrollo Basado en Componentes: Desarrollo de sistemas de gestión de inventario. Tiempo de desarrollo: 4-8 meses.
- Programación Extrema (XP): Desarrollo de aplicaciones de software para startups. Tiempo de desarrollo: 3-6 meses.
- Win Win: Desarrollo de sistemas de gestión de relaciones con los clientes. Tiempo de desarrollo: 6-12 meses.
- Recursivo Paralelo: Desarrollo de sistemas de análisis de datos en tiempo real. Tiempo de desarrollo: 9-18 meses.
- Framework: Desarrollo de sistemas de gestión de contenido empresarial. Tiempo de desarrollo: 6-12 meses.

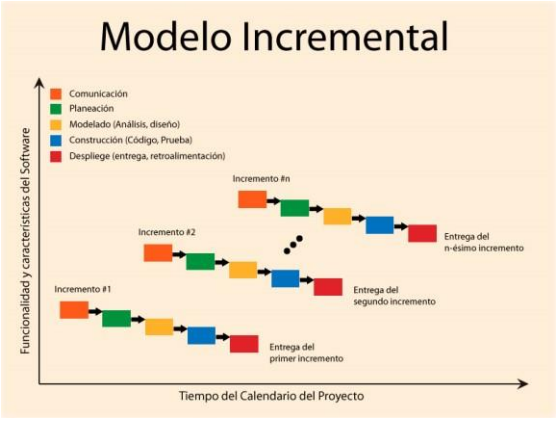
Diseño Grafico

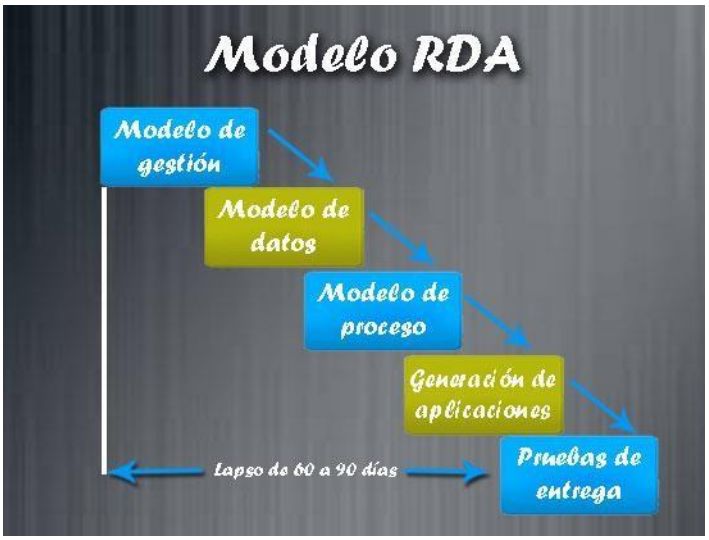


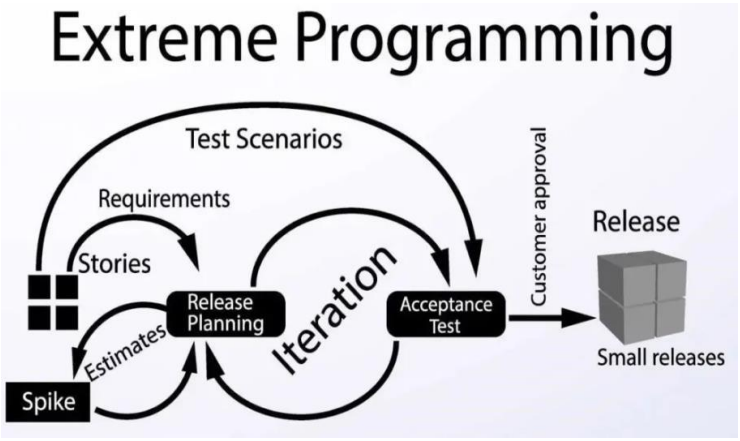




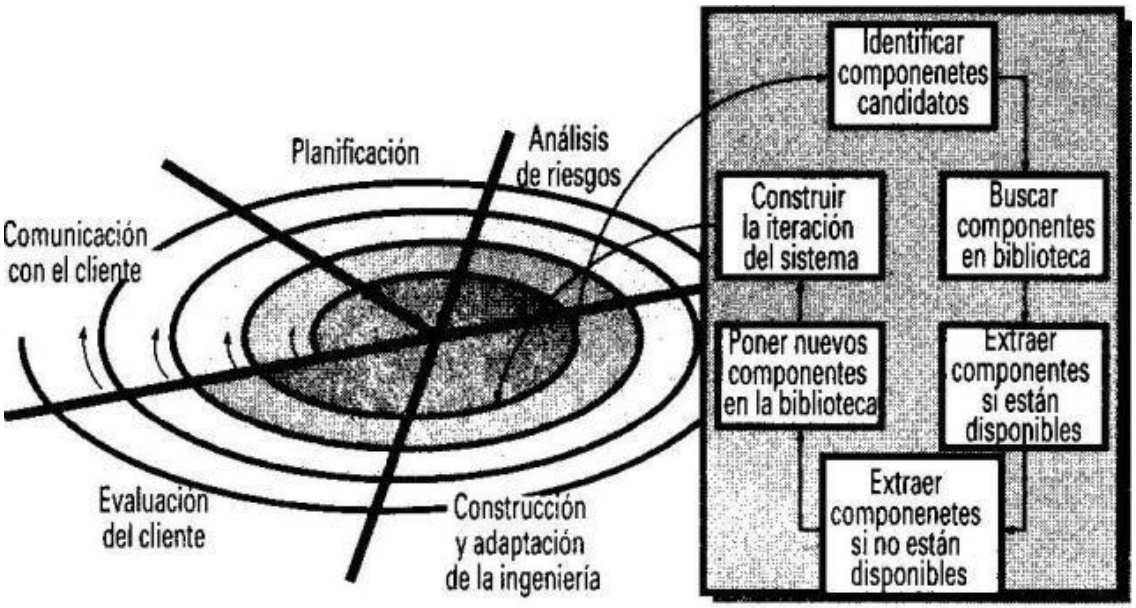



Modelo Incremental
Comunicación
Planeación
Modelado (Análisis, diseño)
Construcción (Código, Prueba)
Despliege (entrega, retroalimentación)
Funcionalidad y características del Software
Incremento #n
Incremento #2
Incremento #1
Entrega del n-ésimo incremento
Entrega del segundo incremento
Entrega del primer incremento
Tiempo del Calendario del Proyecto


Modelo RDA
Modelo de gestión
Modelo de datos
Modelo de proceso
Generación de aplicaciones
Pruebas de entrega
Lapso de 60 a 90 días


Extreme Programming
Test Scenarios
Requirements
Stories
Release Planning
Iteration
Acceptance Test
Customer approval
Release
Small releases
Estimates
Spike


Modelo Espiral WINWIN
3a. Reunir las condicines de victoria
3b. Establecer los objetivos, restricciones y alternativas del siguiente nivel
2. Identificar las condiciones de victoria de los directivos
4. Evaluar las alternativas del producto y del proceso y resolución de los riesgos
1. Identificar el siguiente nivel para los directivos
7.Revisión y comentarios
5. Definir el siguiente nivel del producto y del proceso, incluyendo
6.validar las


Planificación
Análisis de riesgos
Comunicación con el cliente
Evaluación del cliente
Construcción y adaptación de la ingeniería
Identificar componenetes candidatos
Construir la iteración del sistema
Buscar componentes en biblioteca
Poner nuevos componentes en la biblioteca
Extraer componentes si están disponibles
Extraer componenetes si no están disponibles