**Modelo en cascada**

Ventajas

1. Buena documentación en cada fase. Lo que ayuda a los involucrados en las próximas fases.
2. El modelo en cascada puede ser apropiado para proyectos estables con requisitos poco cambiantes en los que se puede producir un diseño correcto antes de comenzar con la implementación.
3. Se puede gestionar fácilmente, debido a que cada fase tiene entregables.

Desventajas

1. Como el cliente solo interviene al principio del proceso y al final, malos entendidos en los requerimientos puede hacer fracasar el sistema en el sentido que el producto no sea lo que el cliente solicito.
2. Mala administración de recursos, ya que al ser secuencial se debe esperar por ejemplo a la finalización de la documentación y el análisis para empezar la codificación.
3. Al ser secuencial, un error hallado en etapas tardías puede ser muy costoso repararlo.

**Modelo incremental**

Ventajas

1. Los clientes proporcionan a grandes rasgos los servicios que proporcionara el sistema y destaca los mas importantes. Esto hace que los servicios mas importantes se entreguen en etapas tempranas del proyecto y puedan ser sometidos mas tiempo a pruebas.
2. En cada incremento se tiene feedback del cliente, ya que en cada incremento se tiene se implementada funcionalidad parcial del sistema. Por lo tanto los clientes pueden obtener experiencia sobre los requerimientos posteriores.
3. Poco riesgo de fallo total del proyecto, ya que el cliente puede seguir la evolución del proyecto.

Desventajas.

1. Los incrementos deben ser pequeños y deben entregar funcionalidad al sistema.
2. Es difícil adaptar el tamaño de los incrementos a los requerimientos del cliente.
3. Dado que los requerimientos no se definen en detalle hasta que el incremento se implementa, se dificulta la asignación de recursos.

**Modelo DRA**

Ventajas

1. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite al equipo de desarrollo crear un "sistema completamente funcional" dentro de periodos cortos de tiempo.
2. Son útiles para desarrollar procesos que son comunes a todos los tipos de negocio.
3. Este enfoque permite que se desarrollen aplicaciones relativamente sencillas con un equipo reducido de personas.

Desventajas

1. El enfoque permite el desarrollo rápido de aplicaciones relativamente sencillas, pero es mas difícil de organizar cuando las aplicaciones son grandes.
2. No existe arquitectura explicita del sistema, lo que puede causar problemas cuando se necesitan cambios.
3. Puede ser difícil de implementar cuando los requerimientos son complejos.

**Modelo en espiral**

Ventajas

1. Al ser un modelo orientado a riegos, se hace análisis de estos en cada ciclo para reducirlos lo mas posible.
2. Planificación basada en los riegos encontrados en la etapa anterior, lo que permite poner atención en las partes mas criticas del sistema
3. El director del proyecto puede ajustar el numero de iteraciones para completar el software.

Desventajas

1. Puede ser difícil convencer a grandes clientes que es controlable.
2. Requiere una considerable habilidad para la evaluación de riesgos.
3. Pueden surgir problemas futuros en caso de no encontrar un riesgo a su debido tiempo.

**Modelos agiles**

Ventajas

1. Útil para procesos de desarrollo donde los requerimientos son cambiantes en periodos cortos de tiempo.
2. Están pensados en para entrega software funcional en forma rápida al cliente, quien puede descubrir nuevos requerimientos en iteraciones posteriores.
3. Participación activa por parte del cliente, quien tiene como tarea priorizar y descubrir nuevos requerimientos.

Desventajas

Difícil aplicarlos principios subyacentes

1. La participación del cliente muchas veces se complica por otras obligaciones que este tiene con otras cuestiones ajenas al proyecto.
2. Se debe tener en cuenta la selección de personal para este tipo de metodologías, ya que no todas las personas tienen la personalidad para seguir el ritmo.
3. No son adecuados para el desarrollo de sistemas grandes con un numero elevado de equipos ni para sistemas críticos donde se necesite un análisis detallado de los requerimientos

**Modelo por prototipos**

Ventajas

1. Es útil cuando no están bien claros los requerimientos del sistema.
2. Reducción de la incertidumbre y el riesgo, ya que se valida con el cliente lo que el sistema va a hacer mediante el prototipo.
3. El modelo de prototipos se puede implementar dentro del contexto de otro modelo.

Desventajas

1. El cliente puede no entender que lo que se muestra (Prototipo) es solo un desarrollo rápido para validar requerimientos y encontrar otros, pidiendo que se modifique este para llegar al producto final.
2. El prototipo es un sistema desechable, es decir no es parte de la solución final.
3. Al confundir el prototipo con el sistema final, el cliente puede pedir aplicarle cambios al prototipo, sin pensar en que los prototipos fueron creados con rapidez sin seguir ciertas normas de calidad de software.

**Iconix**

Ventajas

1. Es una metodología simple y precisa que favorece la participación de los usuarios finales y mantiene el desarrollo de los sistemas documentado. Es un punto intermedio entre RUP y otras metodologías agiles.
2. Presenta claramente las actividades de cada fase y los pasos que se deben seguir.
3. Los modelos utilizados para las especificaciones son de alto nivel de abstracción lo que permite que sean comprensibles.

Desventajas

1. No se puede implementar en sistemas de gran tamaño.
2. Necesita información rápida y puntual de los requisitos y las estimaciones
3. Gran parte la información esta en ingles, lo que requiere establecer bien su comprensión.