

Modelos de Ciclos de Vida Ágiles



MC Cristina Zerpa, PMP



Agenda

- Importancia de la elección del CV
- Los clásicos
- Los patentados
- Los ágiles
- Cómo elegir
- Comentarios y Conclusiones



Importancia de la elección del CV

- **Decisión estratégica**
 - Factor crítico para la satisfacción
 - Factor crítico para el costo
 - Impacta en el desarrollo de las personas
 - Intangibles: Madurez y Capital Intelectual



Decisión estratégica



- Debe realizarse al comienzo
- Es un input del plan de incorporación de recursos
- Muy relacionado con el equipo de trabajo



Factor crítico para la satisfacción

El objetivo es lograr máxima satisfacción y el óptimo aprovechamiento de recursos en un ***contexto estratégico***, contemplando los intereses corporativos



Factor crítico para el costo

- El valor de las TICs para las organizaciones (entregas: qué, cómo y cuándo)
- Cada método tiene requerimientos diferentes de:
 - Competencias y cantidad de personas
 - Ubicación física
 - Intensidad y frecuencia de comunicaciones interpersonales
- Riesgos trasladados a los precios

Impacta en el desarrollo de las personas



- Cuánto podemos/queremos invertir en desarrollarlo?
- Curva de aprendizaje – meses, años
- Aspectos culturales
- Posibilidades reales – dificultad de revertir

Intangibles: Madurez y Capital Intelectual



Lo tenemos claro?

Cuál es el capital intelectual disponible?



-
- Los clásicos
-
-
-
-



Ciclo de vida

- Modelo de Proceso -> Ciclo de Vida del Software
- El Proceso del Software es el conjunto de herramientas, métodos y practicas que usamos para producir software
- Un proceso está bajo control estadístico si su performance futura es predecible dentro de limites estadísticos



Los clásicos

- Escribir y arreglar (Do & Fix)
- Cascada
- Ciclo en V
- Evolutivos
- Prototipado
- Espiral de Bohem

Evolución de los Modelos de Proceso



- Escribir y arreglar
- Clásico o en Cascada
- Espiral (Boehm)
- Prototipado



Escribir y Arreglar

- Primer Modelo -

- Codificar
- Arreglar

- **Problemas -**

- después de varios arreglos el código se vuelve difícil de mantener (s/diseño)
- en general no se adapta a los requerimientos (s/fase de requerimientos)
- resulta muy caro (ausencia de planificación)

MODELO DE CASCADA o CLÁSICO



- Reconoce etapas sucesivas y loops entre ellas -
 - factibilidad
 - análisis y especificación de requerimientos
 - diseño y especificación
 - codificación y test de los módulos
 - test de integración
 - implementación
 - explotación y mantenimiento



■ Dificultades -

- limita los loops a las etapas inmediatamente anteriores
- su plan se basa en linearidad
- es rígido al exigir que se termine totalmente una fase para continuar
- no reconoce la necesidad de interactuar con el usuario

Cascada

Análisis de requerimientos

Diseño del sistema

Diseño de programas

Codificación

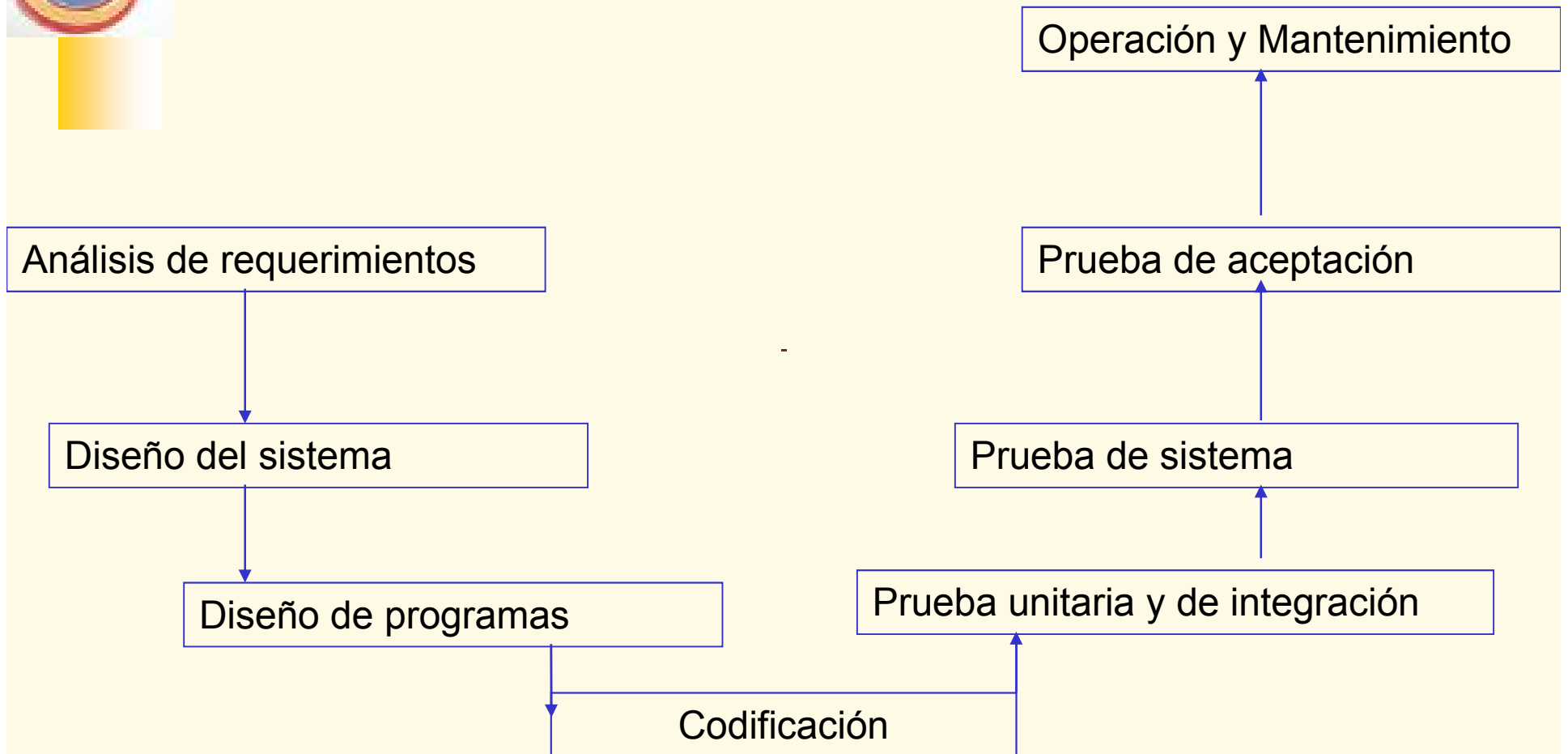
Prueba unitaria y de integración

Prueba de sistema

Prueba de aceptación

Operación y Mantenimiento

Modelo V





Modelo evolutivo

- Su estrategia es -
 - liberar
 - medir
 - ajustar

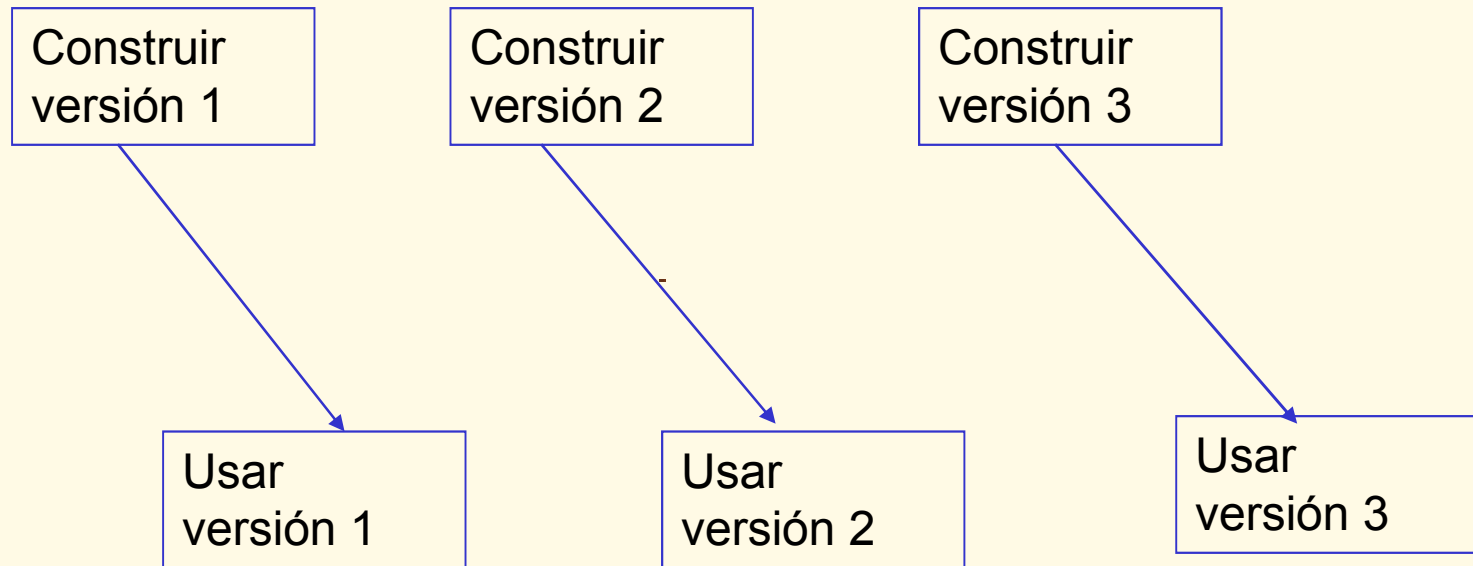
Desarrollo escalonado



desarrolladores

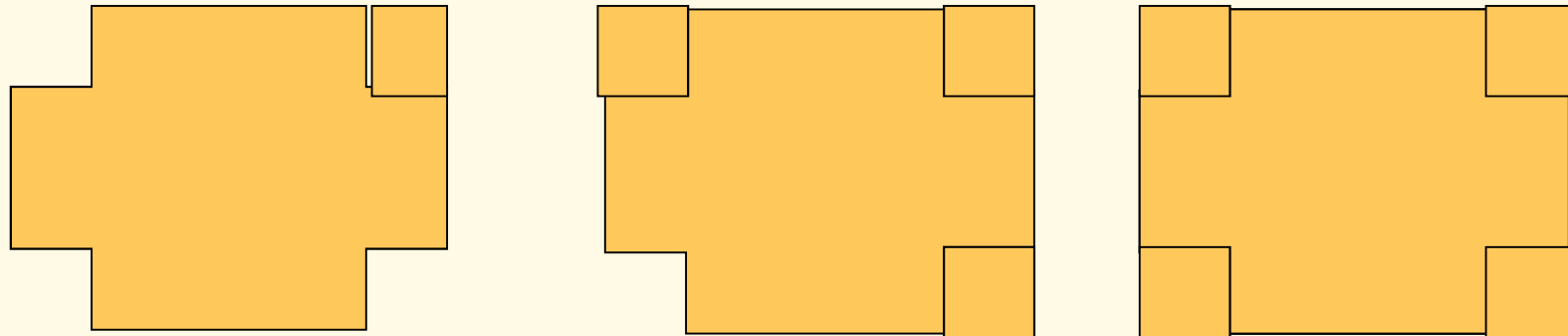
usuarios

Sistemas en desarrollo



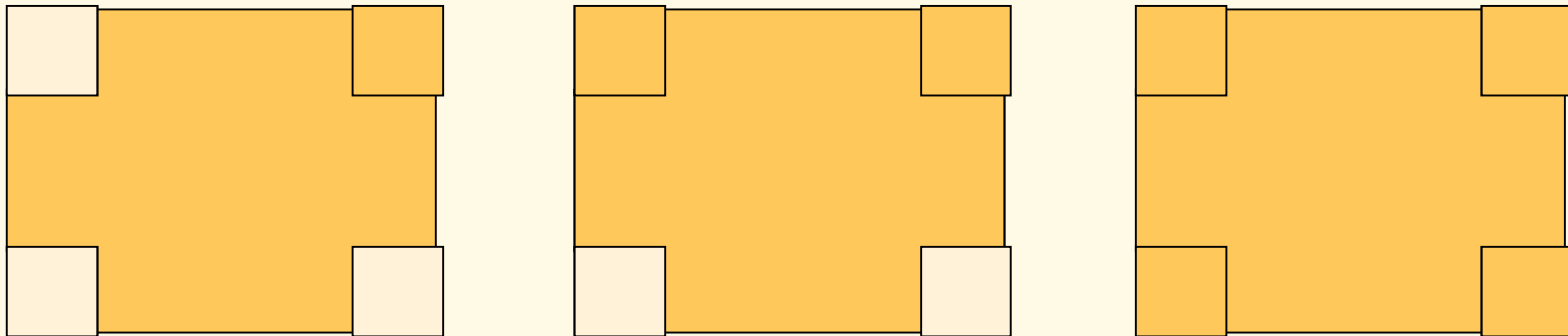


Incremental



c/entrega agrega funcionalidades

Iterativo



c/iteración -> 1 release (interno o externo)-> 1-6 sem.

Repito {Req., análisis, testing,}

Entregas incrementales -> 3-12 meses

Agrega funcionalidades y/o incrementa las existentes

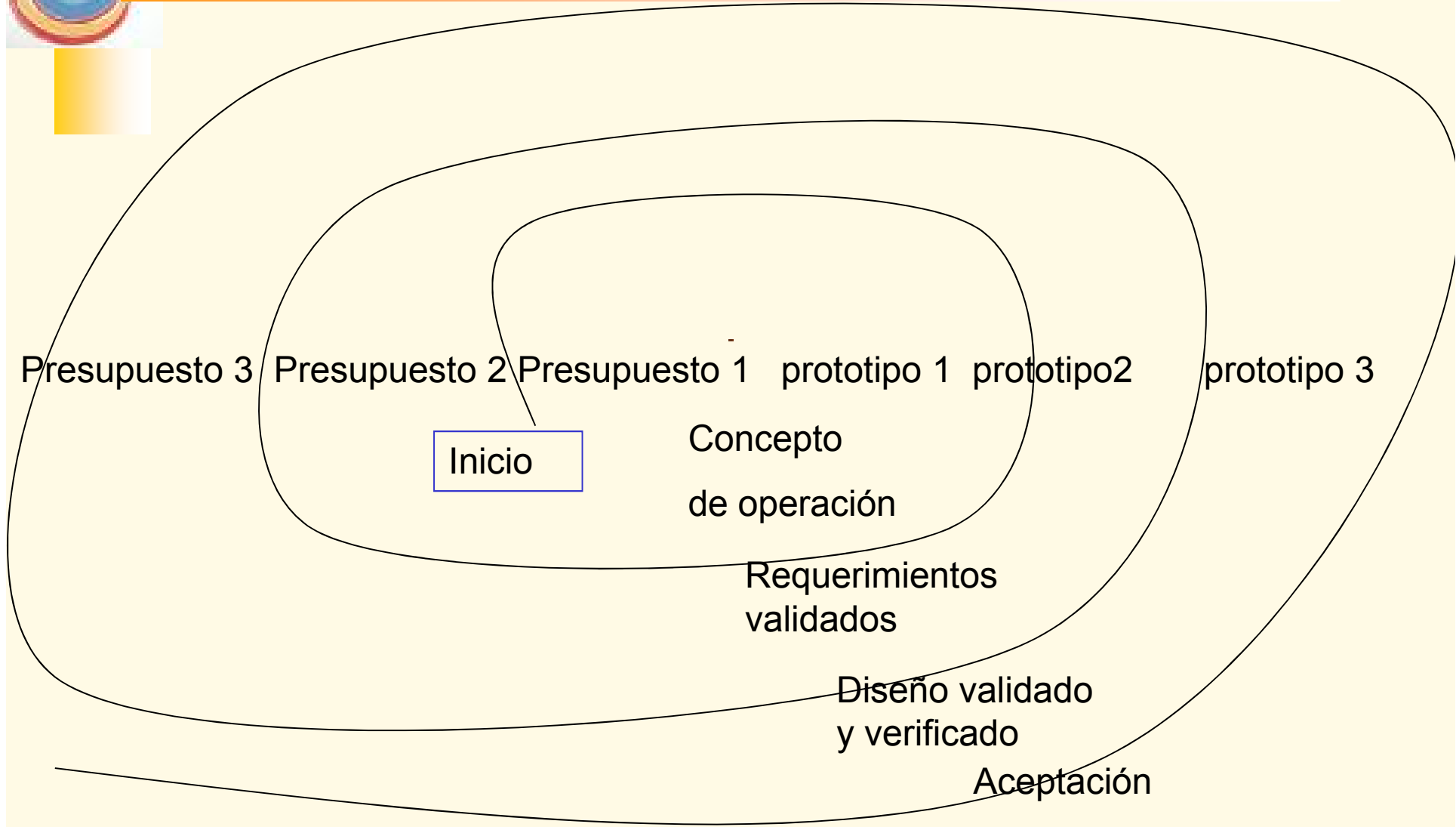


Modelo en espiral (BOEHM)

- Es un metamodelo porque se puede adecuar a varios modelos
- Su esencia es que el proceso de desarrollo es guiado por el riesgo
- Cada ciclo del espiral tiene 4 etapas -
 - objetivos
 - riesgo potencial
 - planificar el siguiente nivel
 - evaluar y revisar resultados de etapa cumplida



Espiral de Bohem





Espiral de Bohem

■ **Bucle 1**

- Inicio
- Presupuesto 1
- Alternativas
- Restricciones
- Análisis de riesgo 1
- Prototipo 1
- Concepto de operación
- Requerimientos
- Plan de ciclo de vida

■ **Bucle 2**

- Presupuesto 2
- Alternativas
- Restricciones
- Análisis de riesgo 2
- Prototipo 1
- Requerimientos
- Requerimientos validados
- Plan de desarrollo



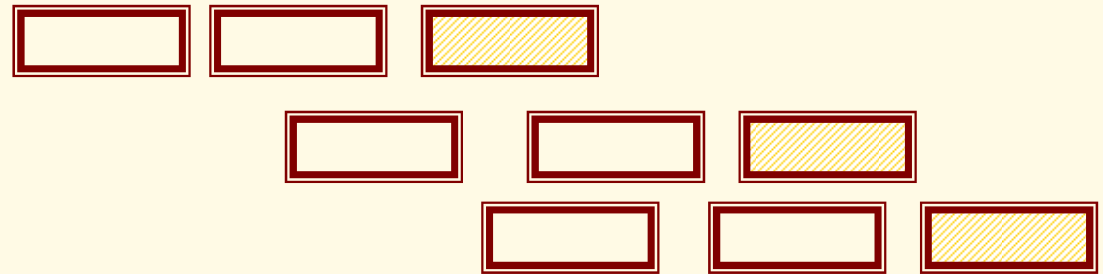
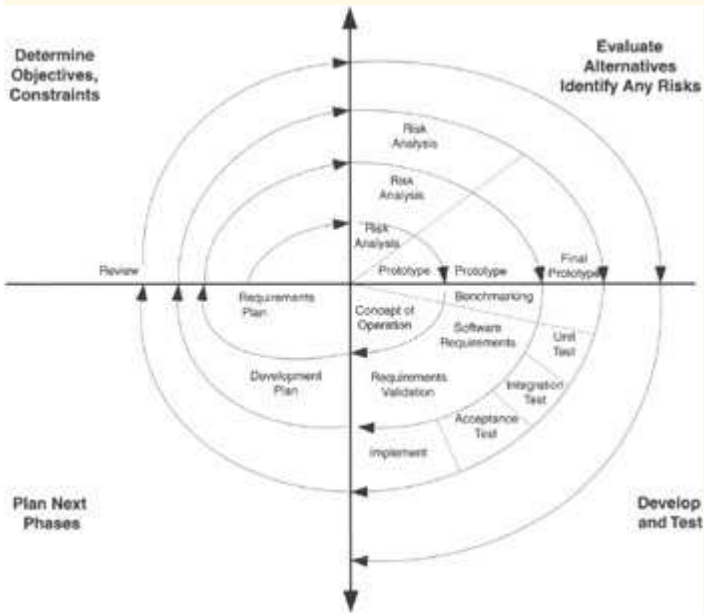
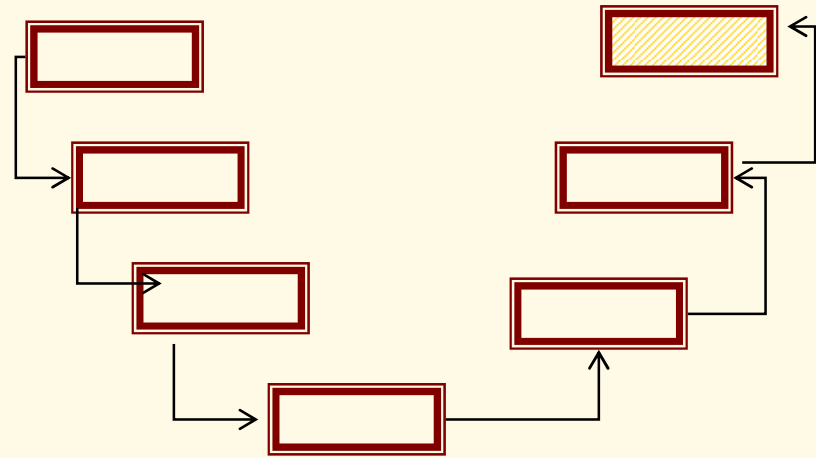
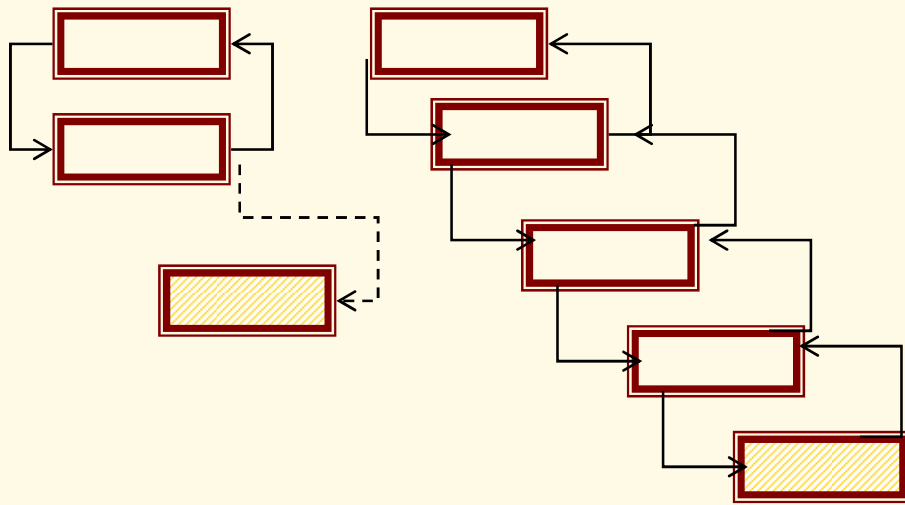
Espiral de Bohem

■ Bucle 3

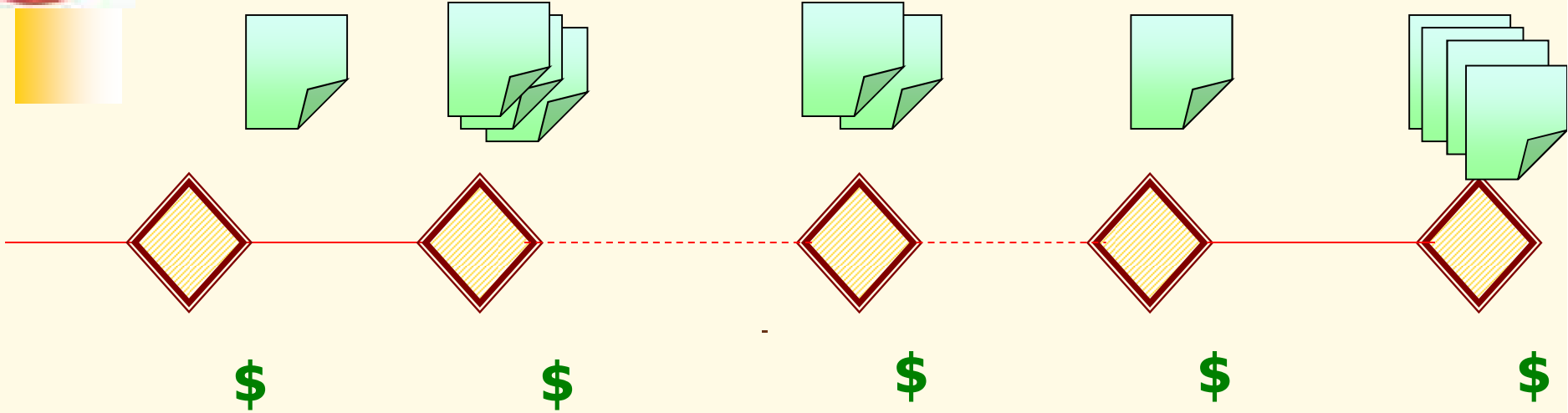
- Presupuesto 3
- Alternativas
- Restricciones
- Análisis de riesgo 3
- Prototipo 3
- Diseño
- Diseño validado y verificado
- Plan de prueba e integración

■ Bucle 4

- Presupuesto 4
- Alternativas
- Restricciones
- Análisis de riesgo 4
- Prototipo 4
- Diseño detallado
- Código
- Prueba unitaria, de sistema
- Prueba de aceptación
- Plan de implementación



Ritmo de entregas





-
-
- Los patentados
-
-



Los patentados

- **UML:** Unified Modeling Language
- **RUP:** Rational Unified Process



Casos de uso - JACOBSON

QUE ?

Análisis (ideal)

- modelo de requerimientos
 - casos de uso
 - interfases de usuario
 - dominio
- modelo de análisis
 - entidad
 - interfases de control

COMO ?

Construcción (real)

modelo de Diseño
modelo de
implementacion

TEST

unitario
integración



Rational Unified Process (RUP)



Rational Unified Process

1998

Rational Objectory Process

1996-1997

Objectory Process

1987-1995

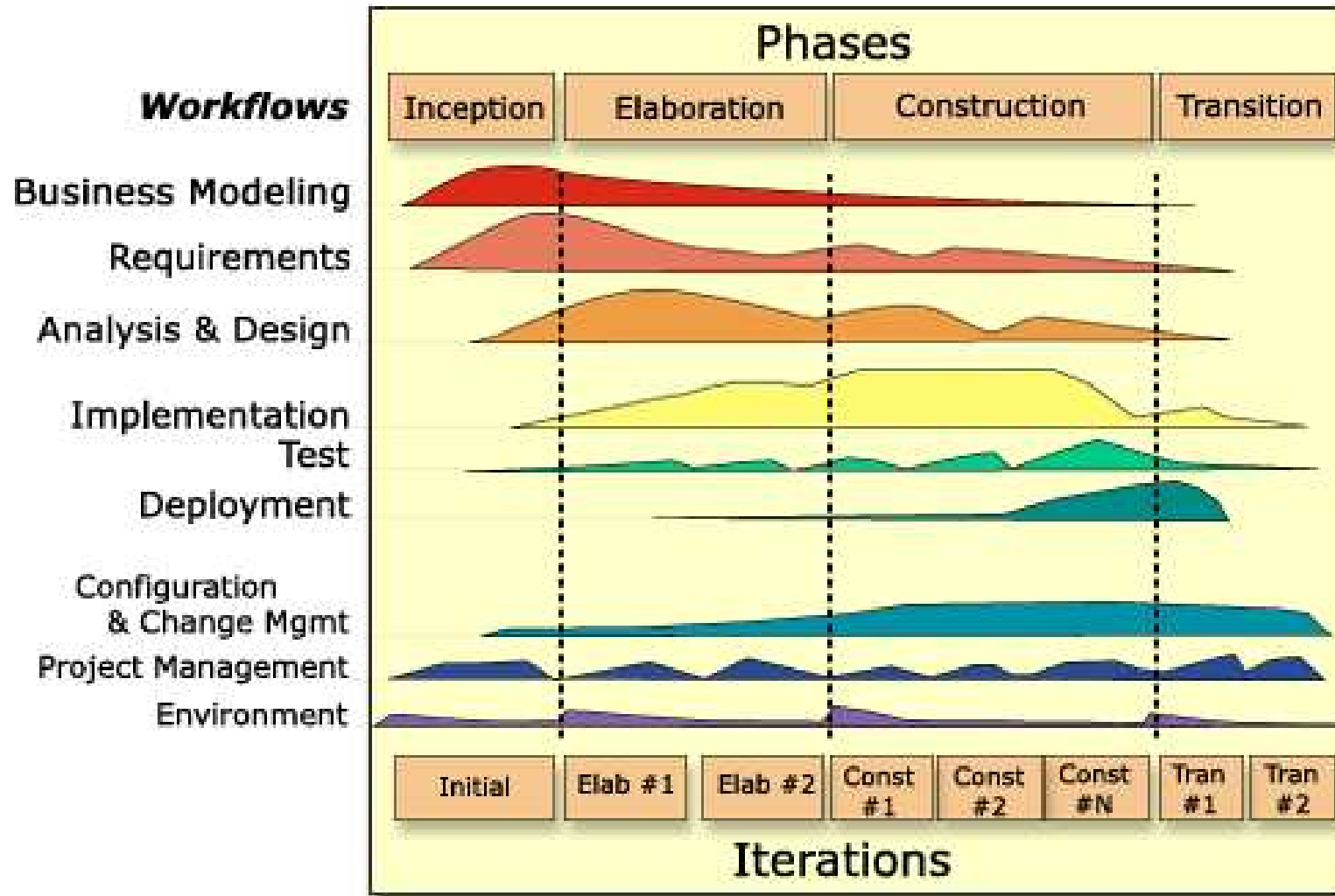
Enfoque Ericsson

- Pruebas funcionales
- Pruebas de desempeño
- Gestión de requisitos
- Gestión de cambios y configuración
- Ingeniería de Negocio
- Ingeniería de datos
- Diseño de interfaces

UML



Dos Dimensiones



Dirigidos por Planes: Conceptos Importantes



- Mejora de Procesos
- Capacidad de los procesos
- Madurez Organizacional
- Grupos de Procesos
- Gestión de Riesgos
- Verificación
- Validación
- Arquitectura del sistema



■ Los ágiles





Los ágiles

- Manifiesto
- Principios
- Metodologías
- Cuándo conviene usarlos?
- Referencias



Ágil?

- Una palabra con gancho
- Un sistema en pre-producción casi desde el comienzo
- Entregas frecuentes – *pagos frecuentes!?!?*
- Liviano

Rápida respuesta a cambios



Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

<http://agilemanifesto.org> año 2001

Métodos Ágiles

Características (1)

- **12 puntos del Manifiesto**
 - Satisfacer al usuario con entregas tempranas y continuas
 - Bienvenidos los cambios que mejoran la competitividad
 - Entregas frecuentes (2 sem. a 2 meses)
 - Personas el negocio involucradas en la diaria
 - Personas altamente motivadas
 - Comunicación cara a cara

Métodos Ágiles

Características (2)

- El progreso está dado por el ***soft funcionando***
- Procesos ágiles promueven ***desarrollo sostenible*** – caminar junto a usuarios
- ***Excelencia técnica***, buen diseño
- ***Simplicidad*** (valorar el trabajo NO hecho)
- ***Equipos autogestionados*** para mejores productos técnicos
- ***Reflexionar regularmente en equipo*** para determinar mejoras



Métodos Ágiles – Énfasis

- ***Individuos e Interacciones*** más que Procesos y Herramientas
- ***Entrega de Funcionalidades*** más que Cumplir Actividades
- ***Colaboración con Cliente*** más que Negociación de Contrato
- ***Responder a Cambios*** más que Seguimiento de un Plan



Métodos más conocidos

- **eXtreme Programming (XP)**
- Adaptive Software Development (ASD)
- Crystal
- **SCRUM**
- RAD
- Feature Driven Development (FDD)



XP

- Kent Beck, Cunningham, ...
- El método más conocido y a menudo usado como sinónimo de ágil
- Historias informales en tarjetas
- Programación x pares
- Diseño simple
- Test temprano
- Integración permanente



ASD

- Jim Highsmith
- Desarrollo iterativo
- Planificación basada en features
- Revisiones de Grupos de usuarios
- Estilo colaborativo de gestión



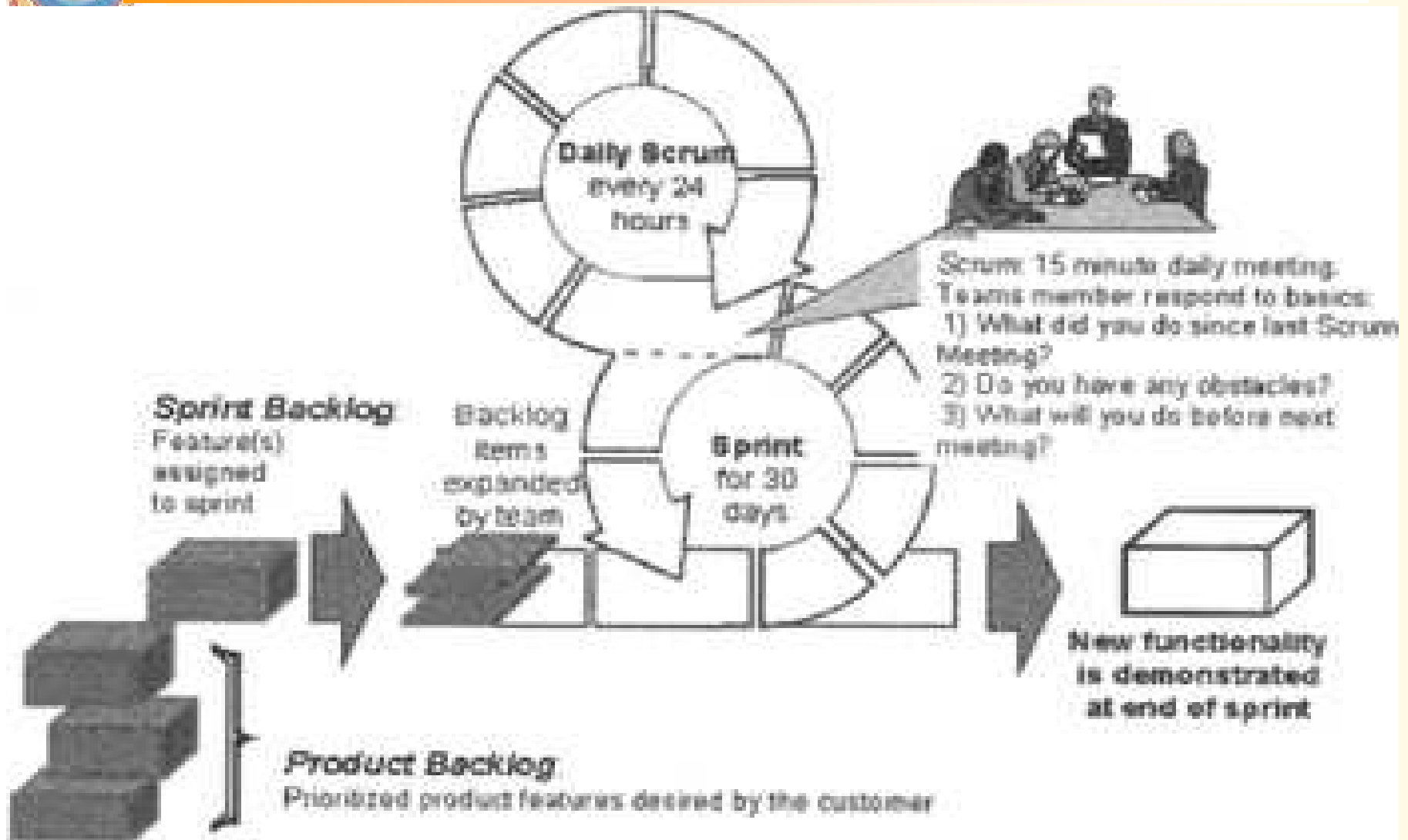
Crystal

- Alistar Cockburn
- Familia de métodos con diferente nivel de "ritual"
- Prácticas provenientes de plan-driven y de ágiles y de investigaciones de desarrollo organizacional y psicología



SCRUM

- Ken Schwaber
- Es más bien una técnica de gestión
- Se divide el proyecto en intervalos de 30 días de trabajo
- Se priorizan listas de requerimientos para ese lapso
- 15 minutos diarios para reuniones de coordinación (lo hecho, dificultades, lo que sigue)



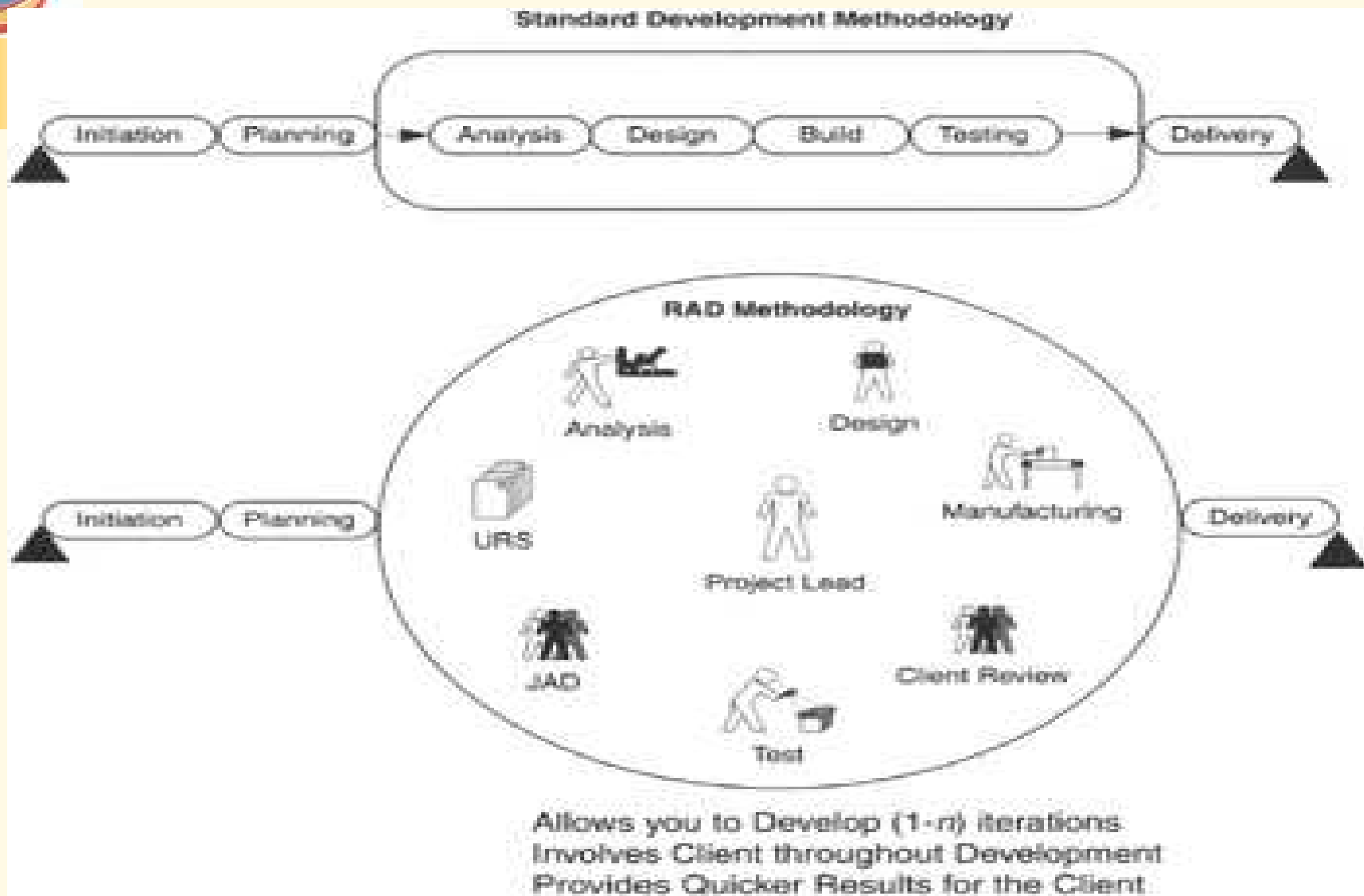


FDD

- Jeff DeLuca
- Método ultraliviano
- Se basa en arquitectura y lista los objetos que la componen
- Mantiene los roles de jefe de programación y de arquitectura
- Utiliza métodos de diseño OO como UML



RAD





Ágiles: Conceptos Importantes

- **Abrazar los cambios;** son vistos como aliados no como enemigos, dar + valor a los clientes y usar creatividad
- **Ciclos rápidos/entregas frecuentes**
- **Diseño simple;** para la batalla no para la guerra
- **Refactoring;** reestructurar para dar + flexibilidad, simplicidad, mejorar comunicación, quitar duplicaciones
- **Programación x pares;** 2 programadores colaborando en 1 computadora en diseño, programación, testing
- **Retrospectiva;** revisiones post-iteración para verificar la efectividad de lo hecho, métodos y estimaciones
- **Conocimiento tácito;** promover conocimiento en la cabeza de los participantes, más que en documentos
- **Desarrollo dirigido por el test;** durante la codificación los usuarios y desarrolladores escriben los guiones de test de modo incremental



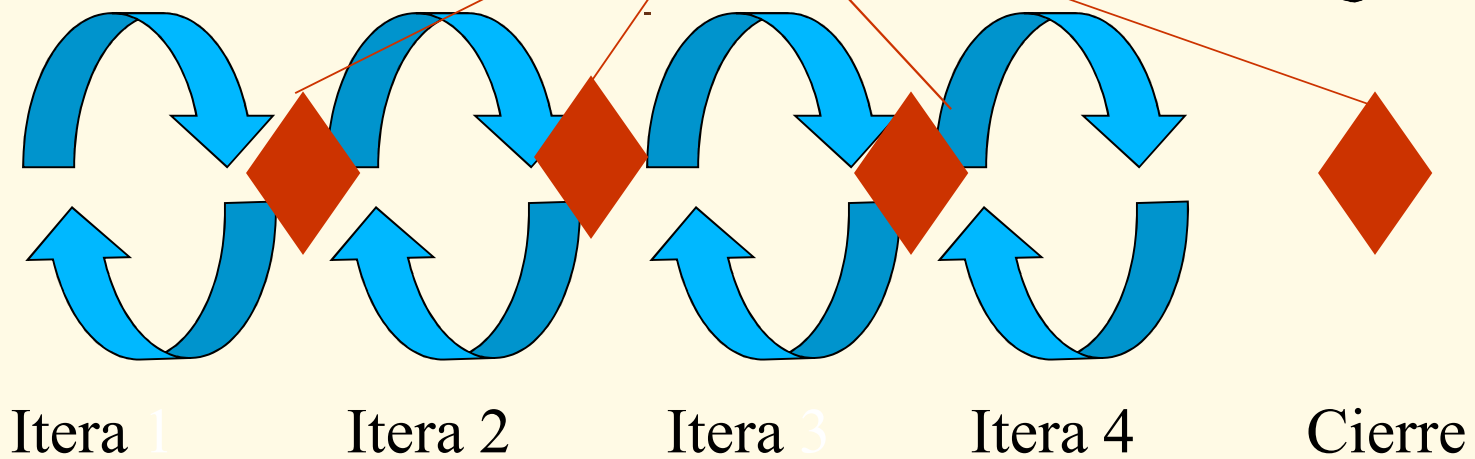
■ **Cómo elegir**



Cómo elegir



Aceptaciones



Plan
Inicial

Puesta en producción continua porque al final:



- Siempre subestimamos !
 - Hacemos sólo la mitad del trabajo ...
 - Entrenamiento del equipo
 - Levantamiento de bugs
 - Ajustes (Hardware | Architecture | Implementation | Performance | Administrability | etc) no quedan terminados totalmente
- Síndrome de “no es lo que el cliente quiere” ☺
- Los clientes se cansan de esperar los resultados (luego hacen de prestamistas)
- Necesitamos mantener focalizado al equipo
- Costo siempre creciente de la integración



Cómo elegir



- criterios y factores a considerar: tamaño, complejidad, criticidad, competencias del personal, cultura, riesgo, limitaciones, dependencias entre funcionalidades, estimación de costo y esfuerzo, alineación con la estrategia, planes de TI y otros intereses de los *stakeholders*.



Cuándo conviene usarlos?

- Qué elegir en los siguientes casos:
 - Soluciones de internet complejas
 - Application Servers, EAI, eCRM
 - Interfases con sistemas de back end u otros SI
 - Alta velocidad
 - Rápida entrada en el mercado
 - Gran oportunidad
 - Evolución de requerimientos
 - Competencia dura
 - Equipos distribuidos
 - Outsourcing, varios locales, etc
 - Proyecto de investigación
 - Ensayar viabilidad de combinar tecnologías emergentes



-
-
-
-
-
- Comentarios y Conclusiones



Conclusiones

Mi conclusión:

El secreto está en la mezcla

(un sabor para cada ocasión)

En la realidad no se usan los métodos puros

Priorizar el valor generado y resultados sustentables en el tiempo.



Ventajas de los ágiles

- Mejor espíritu de equipo
- Se mantiene el foco en los resultados
- No hay que pasar por meses de planificación para ver resultados
- Más chance de ***crear valor***!!!! Lo que al cliente realmente le sirve



Comentarios y Conclusiones

- Opiniones?
 - El cliente
 - Los costos
 - Pasaje a producción
 - Test: quien, cuándo, cuánto, ..
 - Overhead x integración





Preguntas

- Se puede utilizar en contratos de precio fijo?
- Efectivamente aumenta la productividad del equipo?
- Cómo planificar un proyecto iterativo?
- Cómo estimar?
- Cómo gestionar los costos?
- Riesgos y errores típicos de usar estos métodos
- Cómo adoptar estos métodos?



REFERENCIAS

- **Agile software development : principles, patterns, and practices**
Martin, Robert C.
- **Agile & iterative development : a manager's guide**
Larman, Craig
- **Effective project management : traditional, adaptive, extreme**
Wysocki, Robert K
- **Writing effective use cases**
Cockburn, Alistair
- **Agile software development**
Cockburn, Alistair



REFERENCIAS WEB

- **Tutorial UML**

- <http://sparxsystems.com.au/resources/uml2/tutorial/index.html>
- varios artículos sobre ágiles y links www.agilealliance.com
- Especializado en OO www.cetus-links.org
- Links de IS incluyendo iterativos www.bradapp.net
- Versión en inglés de links chinos a iterativos y ágiles www.iturls.com
- SCRUM www.jeffsutherland.com



Muchas Gracias!!!!!!

czarpa@gmail.com

