PROCESO UNIFICADO

INGENIERÍA DE SOFTWARE I

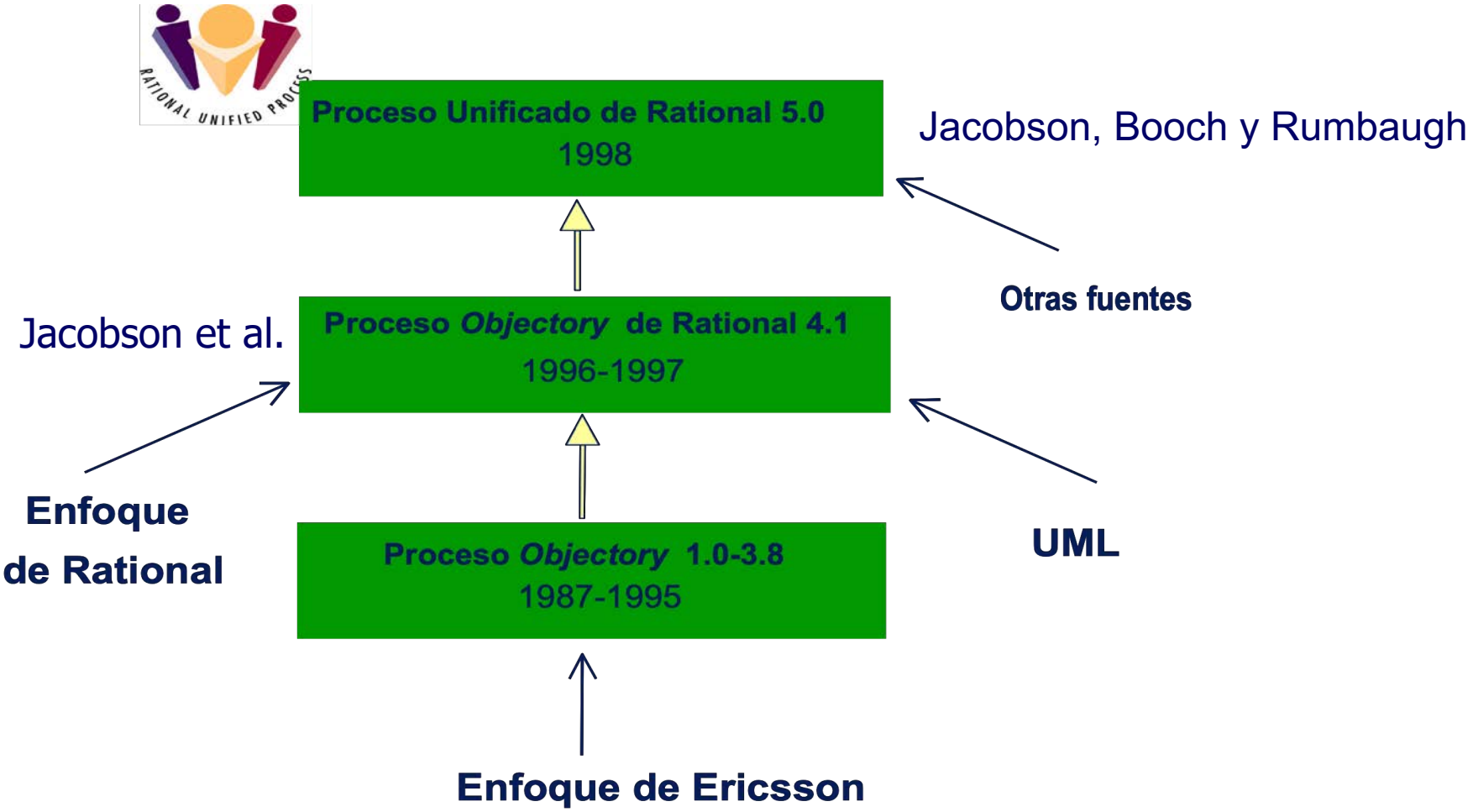
2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2020/2021

Francisco José García Peñalvo / fgarcia@usal.es
Alicia García Holgado / aliciagh@usal.es
Andrea Vázquez Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca

1. INTRODUCCIÓN

ORÍGENES


Proceso Unificado de Rational 5.0
1998
Jacobson, Booch y Rumbaugh
Otras fuentes
Jacobson et al.
Proceso Objectory de Rational 4.1
1996-1997
Enfoque
de Rational
UML
Proceso Objectory 1.0-3.8
1987-1995
Enfoque de Ericsson

EVOLUCIÓN

- En Febrero de 2003 IBM compró Rational y en 2006 creó una versión de RUP para procesos ágiles
  - Plataforma IBM Rational Method Composer V7.5.2
    - Herramienta de creación y publicación de métodos basada en Eclipse
    - Incluye una biblioteca de procesos
    - Ofrece una guía de mejores prácticas para el desarrollo de *software* (RUP y procesos ágiles)
- Open Unified Process fue donado en 2007 a la Fundación Eclipse
  - Eclipse Process Framework Project 1.5.2
  - http://www.eclipse.org/epf/

JUSTIFICACIÓN

- Diferentes proyectos tienen diferentes necesidades de proceso
- Diversos factores marcan las necesidades para un proceso más formal o más ágil
  - Tamaño del equipo
  - Localización geográfica
  - Complejidad de la arquitectura
  - Novedad de la tecnología
  - Cumplimiento de estándares
  - etc.
- No obstante, hay buenas prácticas en el desarrollo del *software* que benefician a cualquier proyecto
- La idea de Proceso Unificado es aportar un conjunto mínimo de prácticas que ayudan a los equipos de desarrollo a ser más eficientes con independencia del tipo de proyecto

DEFINICIÓN

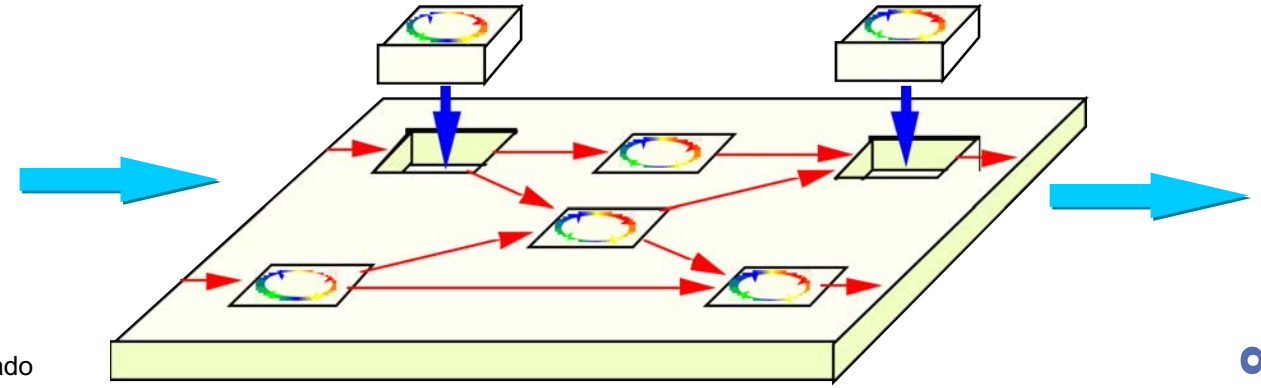
El Proceso Unificado es más que un simple proceso (Jacobson et al., 1999), es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas *software*, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos

CARACTERÍSTICAS

- Características generales
  - Está basado en componentes
  - Utiliza UML (Booch et al., 2005; OMG, 2017)
- Características principales (Jacobson et al., 1999)
  - Es un proceso conducido por casos de uso
  - Está centrado en la arquitectura
  - Es iterativo e incremental

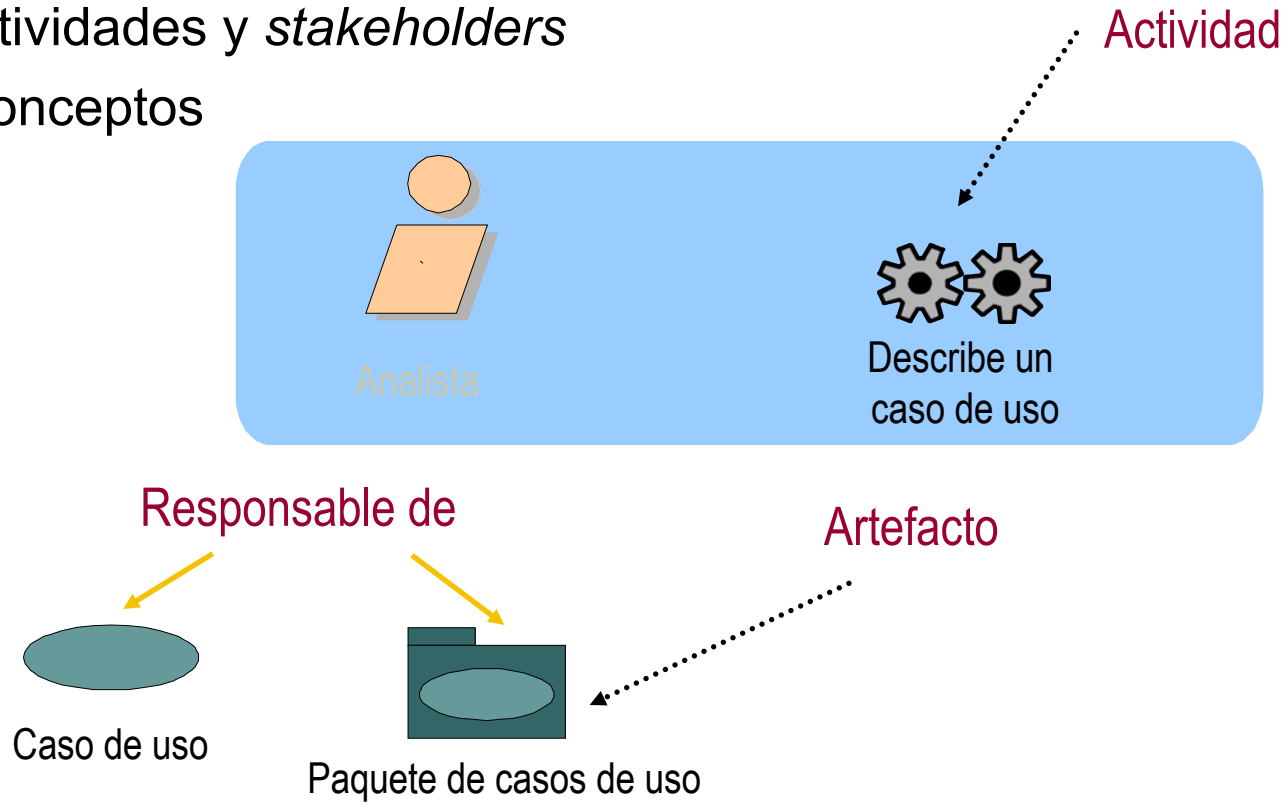
MARCO DE TRABAJO GENÉRICO

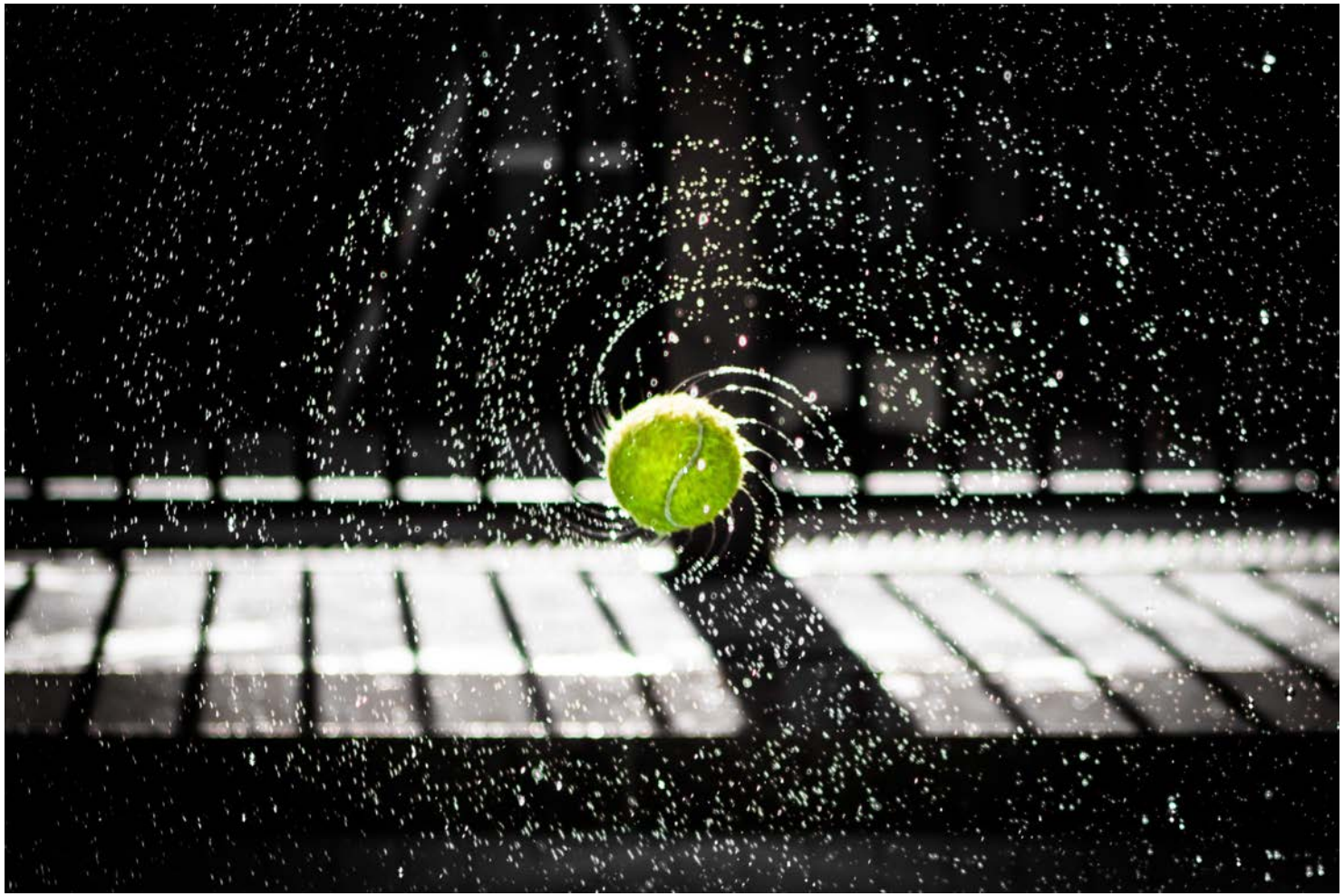
- No existe un proceso universal
- Puede extenderse y especializarse para una gran variedad de sistemas de *software*
  - Flexibilidad
  - Está basado en componentes
- Permite gran variedad de estrategias de ciclo de vida
  - Se pueden definir diferentes conjuntos de productos
  - Se pueden definir actividades y encargados de las mismas



FUNDAMENTOS

- Selecciona qué artefactos producir
- Define actividades y *stakeholders*
- Modela conceptos



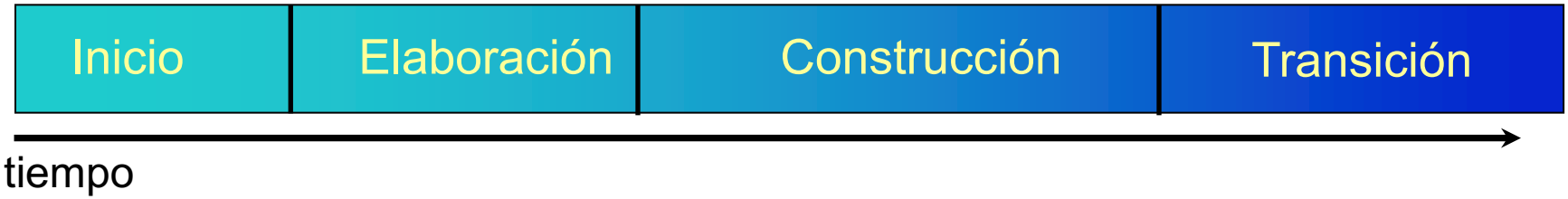


https://unsplash.com/search/spiral?photo=zcYRw547Dps

2. CICLO DE VIDA DEL PROCESO UNIFICADO

CICLO DE VIDA

- El Proceso Unificado se repite a lo largo de una serie de **ciclos de desarrollo** que constituyen la vida de un sistema
- Cada **ciclo de desarrollo** concluye con una **versión entregable** del producto
- Cada ciclo consta de cuatro **fases**
  - **Inicio**
    - Se define el alcance del proyecto y se desarrollan los casos de negocio
  - **Elaboración**
    - Se planifica el proyecto, se especifican en detalle la mayoría de los casos de uso y se diseña la arquitectura del sistema
  - **Construcción**
    - Se construye el producto
  - **Transición**
    - El producto se convierte en versión beta
    - Se corrigen problemas y se incorporan mejoras sugeridas en la revisión



CICLO DE VIDA

- Etapa de Ingeniería
  - Equipos pequeños, actividades poco predecibles (análisis, viabilidad, planificación)
  - Comprende las fases
    - Inicio
    - Elaboración
- Etapa de Producción
  - Equipos grandes, actividades predecibles, menos riesgos (programación, pruebas)
  - Comprende las fases
    - Construcción
    - Transición

CICLO DE VIDA

- Dentro de cada fase se puede, a su vez, descomponer el trabajo en **iteraciones** con sus incrementos resultantes
- Cada fase termina con un **hito**, cada uno de los cuales se caracteriza por la disponibilidad de un conjunto de componentes de *software*

HITOS

- Los hitos son puntos de control en los cuales los participantes en el proyecto revisan el progreso del proyecto
- Se pretende
  - Controlar el progreso del proyecto
  - Sincronizar las expectativas y la realidad
  - Tomar decisiones para continuar con la siguiente fase
  - Identificar los riesgos
  - Se evalúa la situación global del proyecto
- Se necesitan
  - Resultados tangibles para comparar con las expectativas
- Varios niveles
  - Hitos principales al final de cada fase
  - Hitos secundarios final de cada iteración

CICLO DE VIDA

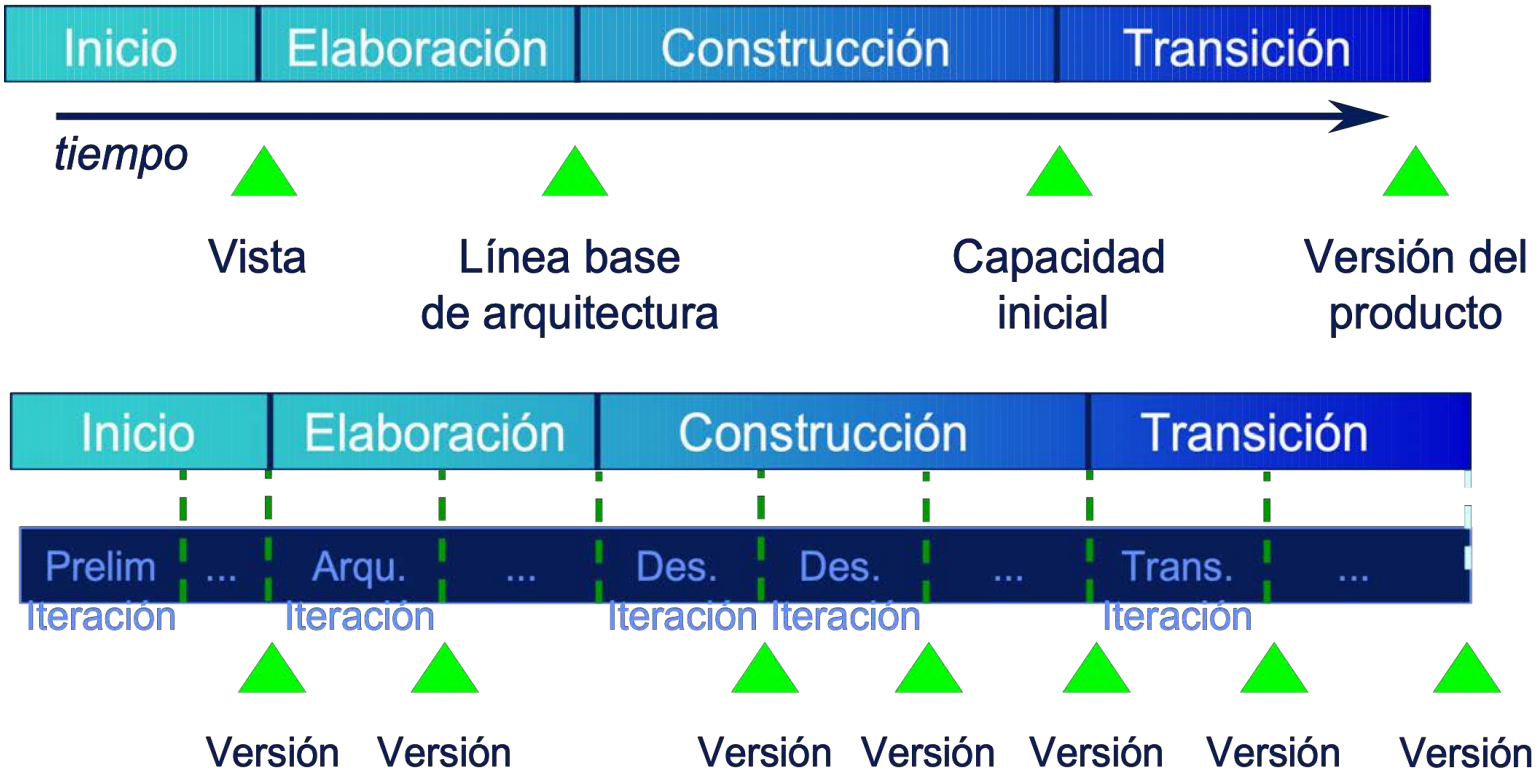
- Una **iteración** es una secuencia de actividades con un plan establecido y unos criterios de evaluación, cuyo resultado es una **versión ejecutable no orientada a la entrega** (hito secundario)
- Dentro de cada fase se puede, a su vez, descomponer el trabajo en iteraciones con sus incrementos resultantes
- Cada fase termina con un hito, cada uno de los cuales se caracteriza por la disponibilidad de un conjunto de componentes de *software*
- Las iteraciones discurren a lo largo de las disciplinas

CICLO DE VIDA

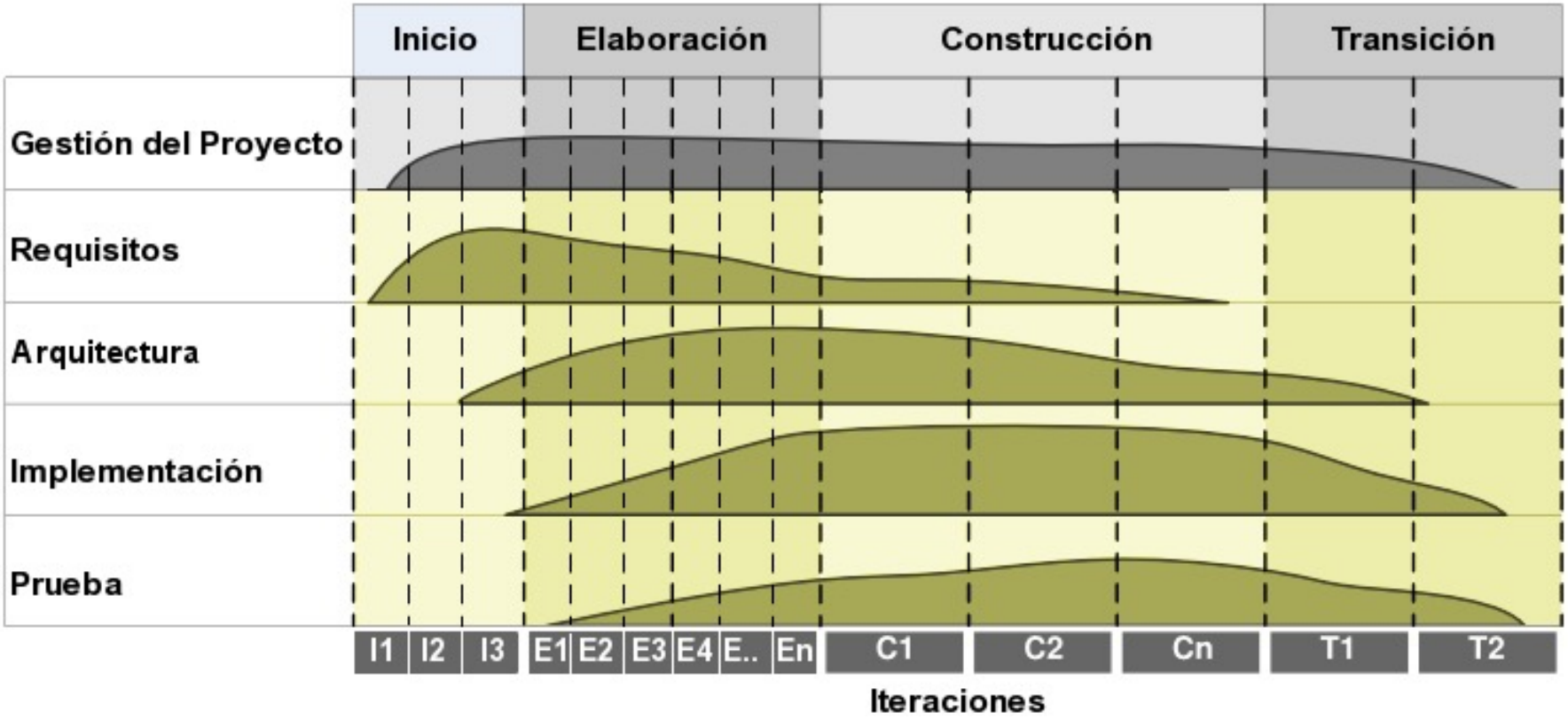
- Las disciplinas o flujos de trabajo organizan las actividades fundamentales de gestión y desarrollo del proyecto
  - **Disciplinas de desarrollo**
    - Requisitos, análisis, diseño, implementación, pruebas...
  - **Disciplinas de gestión o soporte**
    - Gestión de proyecto, gestión de configuraciones, entorno, evaluación...
- Al contrario de lo que ocurre con las fases, las distintas actividades del equipo de desarrollo se pueden solapar en el tiempo

CICLO DE VIDA

Cada ciclo concluye con una versión del producto para los clientes



CICLO DE VIDA



https://es.wikipedia.org/wiki/OpenUP#/media/File:Ciclo_de_Vida_OpenUP.png

https://visualhunt.com/f/photo/42946863 46/fa10e0e9c7/

3. EL PRODUCTO

PRODUCTO

- El producto que se obtiene es un **sistema de *software***
- El sistema lo componen todos los “artefactos” necesarios para representarlo de forma comprensible
- **Artefacto**
  - Término general para cualquier tipo de información creada, producida, cambiada o utilizada por los *stakeholders* en el desarrollo del sistema. Puede ser
    - De ingeniería
    - De gestión
- El artefacto más importante del Proceso Unificado es el **modelo**
- Un sistema posee una colección de modelos y las relaciones entre ellos

PRODUCTO

Un modelo es una abstracción semánticamente cerrada del sistema

Los modelos recogen diferentes perspectivas del sistema (perspectivas de todos los *stakeholders*)

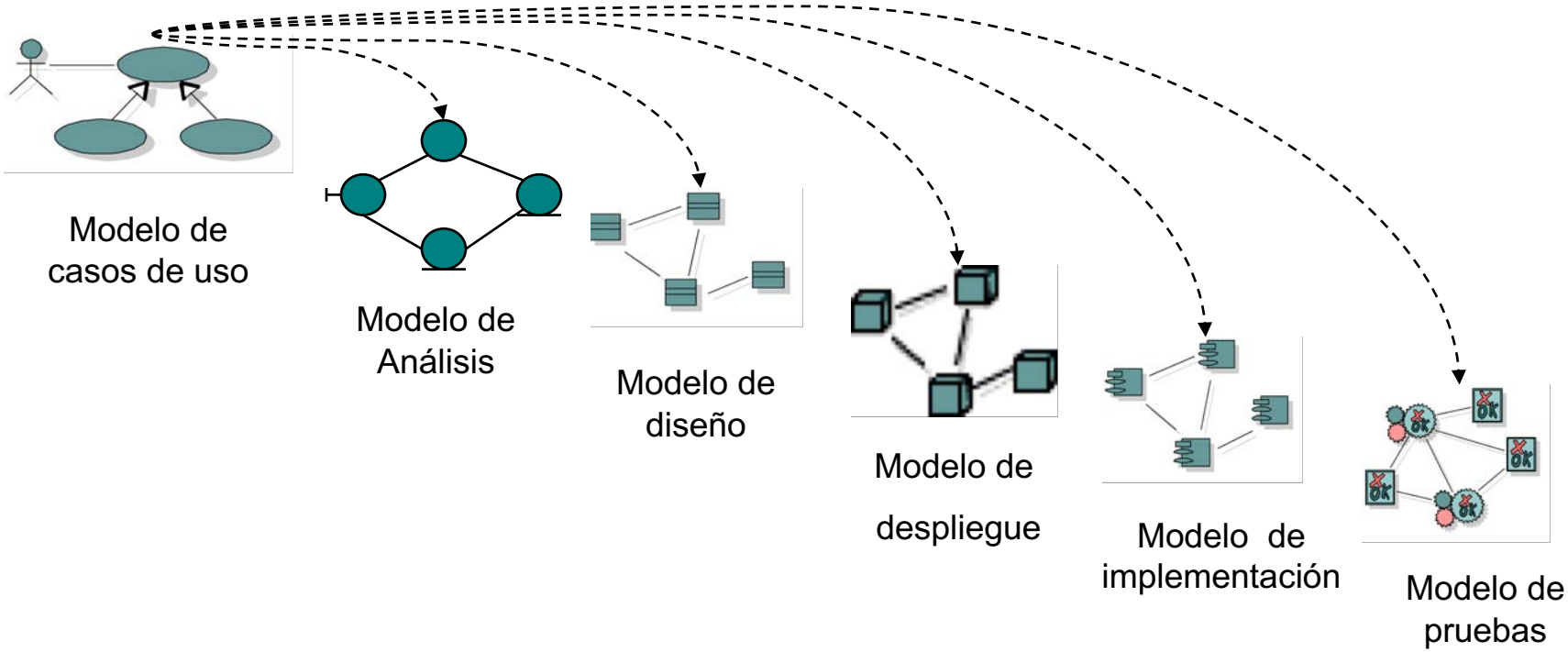
PRODUCTO

Modelos

- **Modelo de casos de uso**
  - Diagramas de casos de uso, secuencia, colaboración y actividad
- **Modelos de análisis y diseño**
  - Diagramas de clases, objetos, secuencia, colaboración y actividad
- **Modelo de despliegue**
  - Diagramas despliegue, secuencia y colaboración
- **Modelo de implementación**
  - Diagramas de componentes, secuencia y colaboración
- **Modelo de pruebas**
  - Todos los diagramas

PRODUCTO

Existen dependencias entre el modelo de casos de uso y los demás modelos



https://www.flickr.com/photos/kel0/7123959465

Photo credit: kel0 via Visual hunt / CC BY-NC-SA

4. EL PROCESO

PROCESO

- El proceso hace referencia a un contexto que sirve como plantilla que pueda reutilizarse para crear instancias de ella (proyectos)
- Las actividades relacionadas conforman **disciplinas** o **flujos de trabajo**
  - Su identificación parte de la identificación de los *stakeholders* y de los artefactos para cada tipo de *stakeholder*
  - Describen como fluye el proceso a través de los *stakeholders*

CARACTERÍSTICAS PRINCIPALES DEL PROCESO

Conducido por casos de uso

- Los casos de usos guían el desarrollo del sistema
- Como los casos de uso contienen las descripciones de las funciones, afectan a todas las fases y vistas

Centrado en la arquitectura

- La arquitectura se representa mediante vistas del modelo
- Se puede tomar como arquitectura de referencia el denominado modelo de arquitectura de 4+1 vistas propuesto por Philippe Kruchten (1995)

Iterativo e Incremental

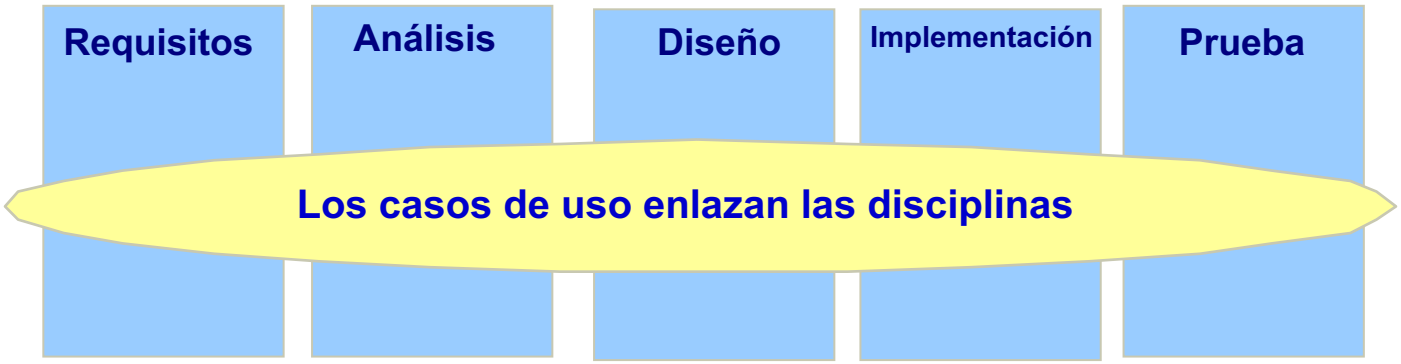
- En cada iteración se identifican y especifican los casos de uso relevantes, se crea un diseño basado en la arquitectura seleccionada, se implementa el diseño mediante componentes y se verifica que los componentes satisfacen los casos de uso
- Si una iteración cumple con sus objetivos se pasa a la siguiente
- En cada iteración se va desarrollando el sistema de forma incremental

PROCESO DIRIGIDO POR CASOS DE USO

Dirigen las actividades de desarrollo

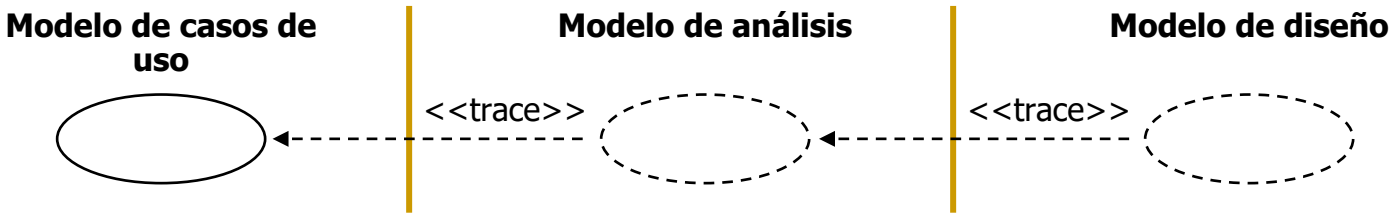
- Creación y validación de la arquitectura del sistema
- Definición de casos de prueba y procedimientos
- Planificación de iteraciones
- Creación de documentación de usuario
- Despliegue del sistema

Sincronizan el contenido de los diferentes modelos



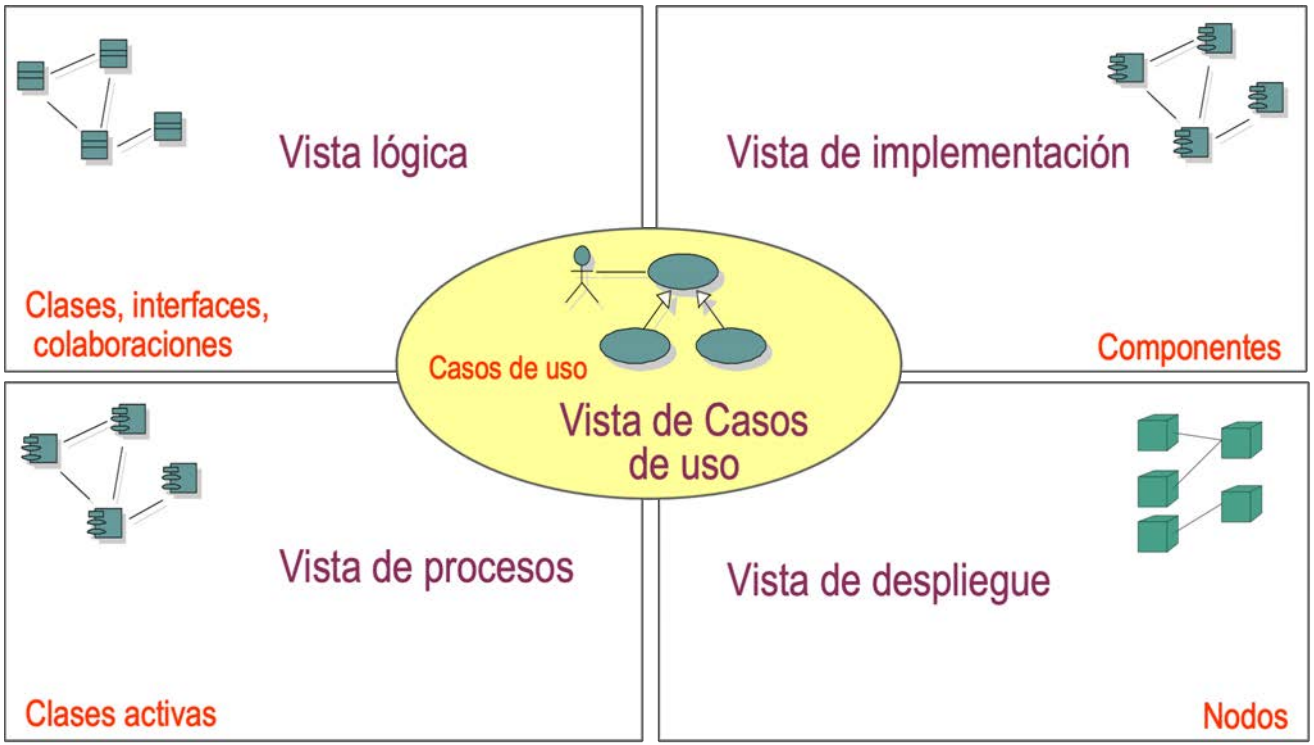
PROCESO DIRIGIDO POR CASOS DE USO

- Inicialmente los casos de uso se utilizan para la captura de requisitos funcionales
- Durante el análisis y el diseño se transforma el modelo de casos de uso mediante un modelo de análisis en una estructura de clasificadores y **realizaciones de casos de uso**
- En cada iteración, los casos de uso sirven de guía a través del conjunto completo de disciplinas



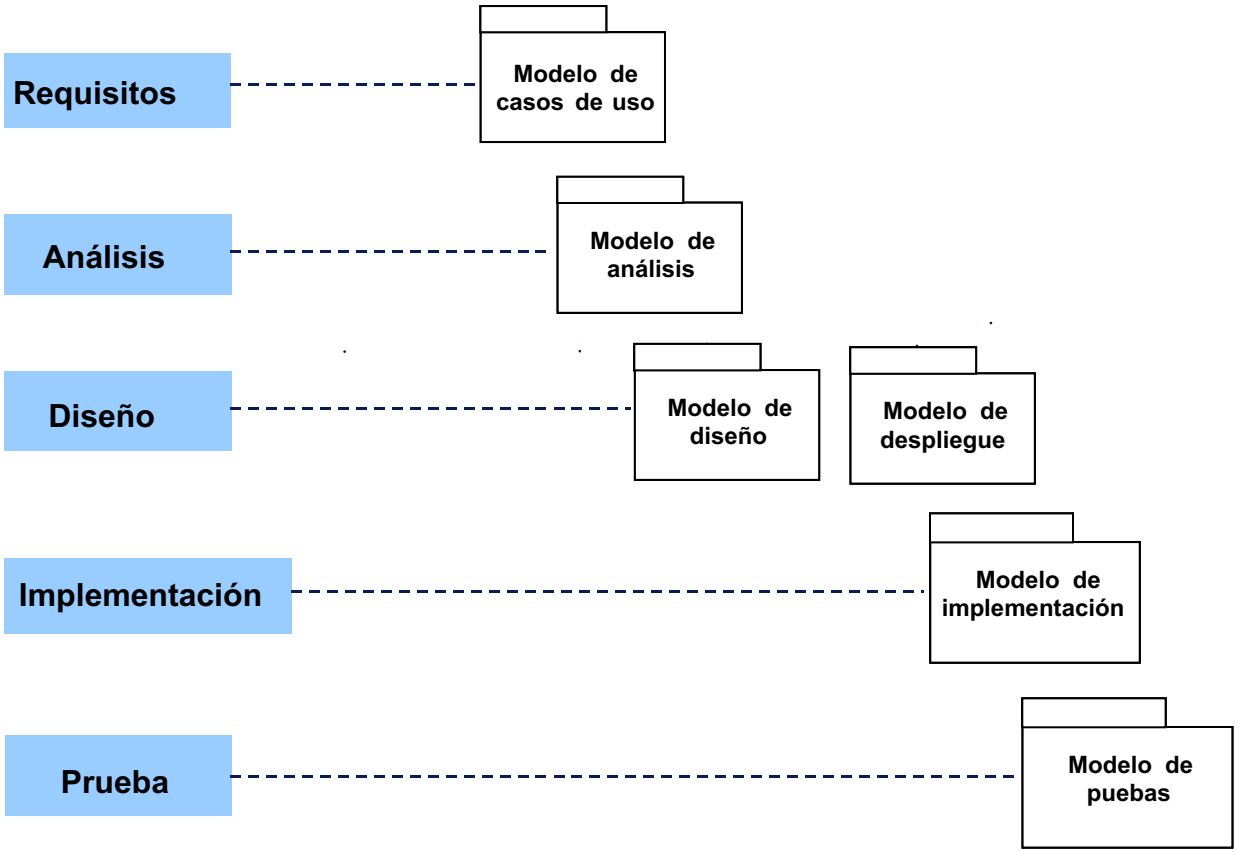
PROCESO CENTRADO EN LA ARQUITECTURA

- Se puede tomar como arquitectura de referencia el denominado modelo de arquitectura de 4+1 vistas, propuesto por Philippe Kruchten (1995)
  - Cada vista es una parte de un modelo



PROCESO CENTRADO EN LA ARQUITECTURA

Centrado en la arquitectura: diferentes vistas del sistema

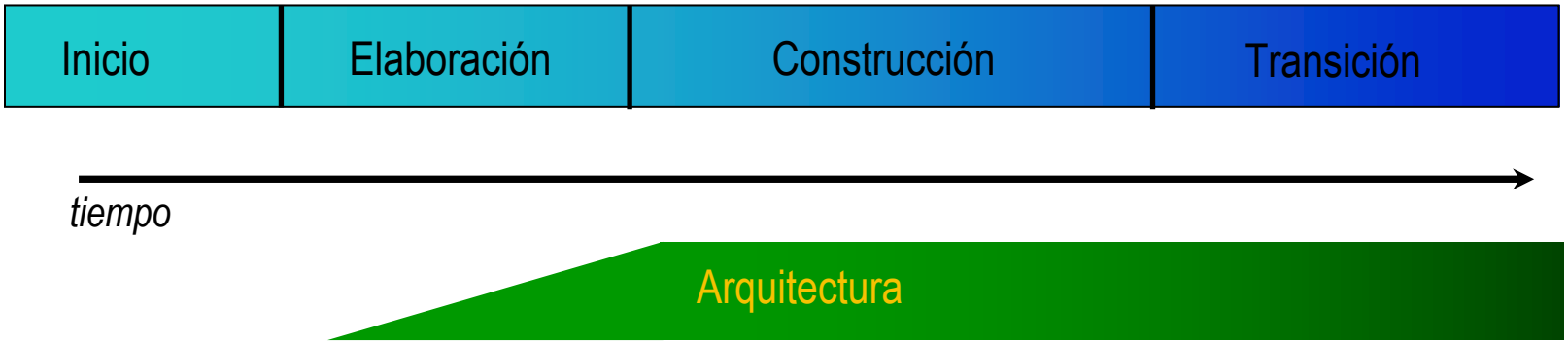


Relación entre los flujos de trabajo y los modelos que forman la arquitectura

PROCESO CENTRADO EN LA ARQUITECTURA

Centrado en la arquitectura: diferentes vistas del sistema

- Los **modelos** son los vehículos para visualizar, especificar, construir y documentar la arquitectura
- El Proceso Unificado prescribe los sucesivos refinamientos de una arquitectura ejecutable



PROCESO CENTRADO EN LA ARQUITECTURA

Diseño de la arquitectura

- Seleccionar escenarios: aspectos críticos y riesgos
- Identificar las clases principales y sus responsabilidades
- Distribuir el comportamiento en clases
- Estructurar en subsistemas, capas y definir interfaces
- Definir distribución y concurrencia
- Implementar prototipos de arquitectura
- Derivar casos de prueba a partir de los casos de uso
- Evaluar la arquitectura

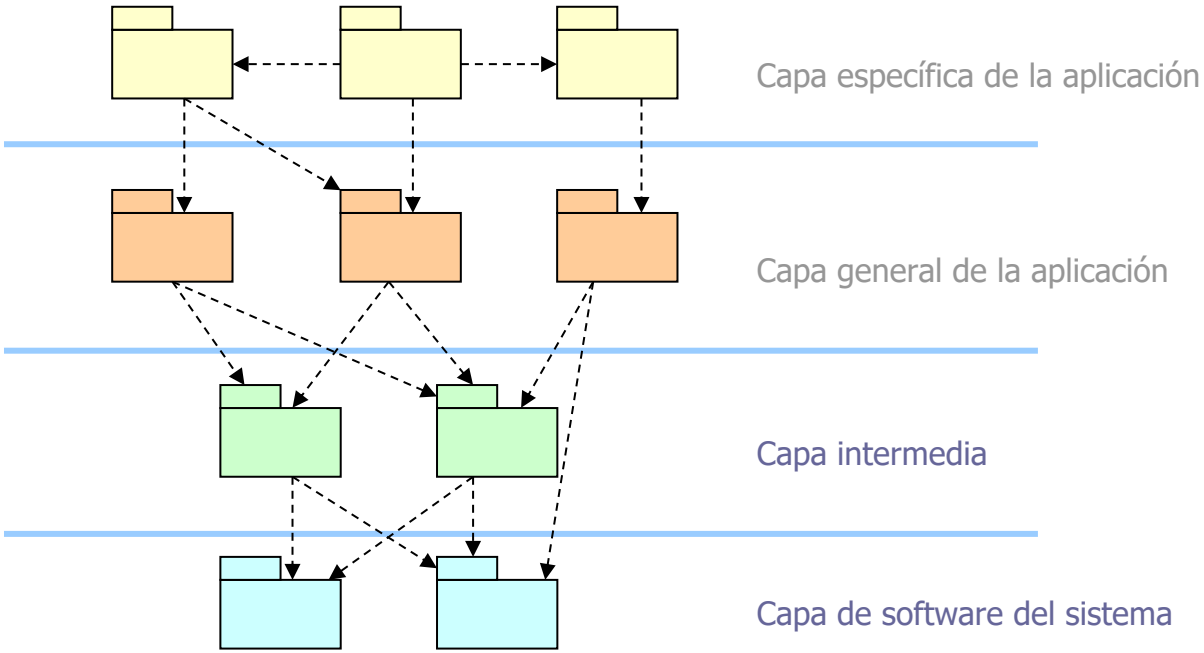
Iterar

La arquitectura se desarrolla mediante iteraciones (**en capas**)

- Comienza con una línea base de arquitectura (primera versión de los modelos)
- La línea base evoluciona hasta convertirse en un sistema estable

PROCESO CENTRADO EN LA ARQUITECTURA

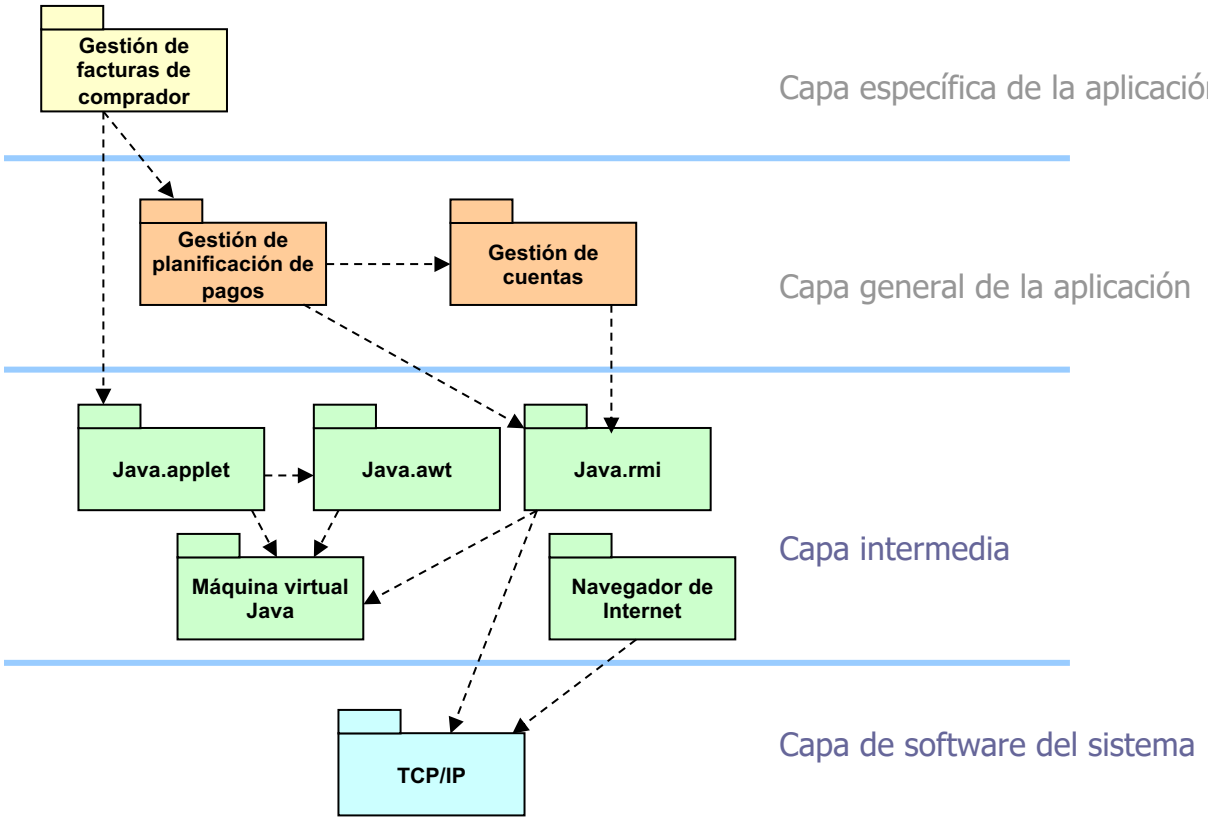
Diseño de la arquitectura



Patrón de capas de la arquitectura del sistema

PROCESO CENTRADO EN LA ARQUITECTURA

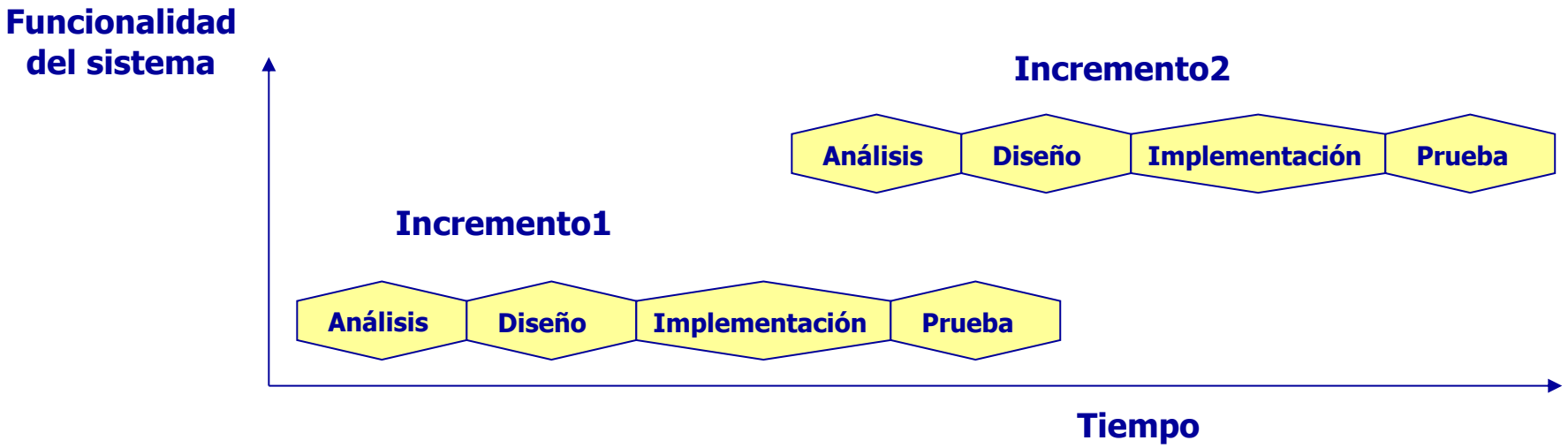
Diseño de la arquitectura



PROCESO ITERATIVO E INCREMENTAL

La característica fundamental del Proceso Unificado es ser un **proceso iterativo**

- Se basa en la ampliación y el refinamiento del sistema
- Una serie de desarrollos cortos (mini proyectos de 2 a 6 semanas, cada iteración reproduce el ciclo de vida a menor escala)
- No solo se mejora sino que el sistema también crece: proceso iterativo e incremental



PROCESO ITERATIVO E INCREMENTAL

- El resultado de cada iteración es un sistema ejecutable (aunque sea incompleto y no esté listo para su instalación)
- Un sistema instalable requiere varias iteraciones
- Evolución de prototipos ejecutables
- Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes
- Concepto de *time-boxing*
  - Cada iteración debe tener una duración fija (normalmente se mide en semanas)
    - En lugar de retrasar el final de una iteración se recomienda eliminar algunos de los requisitos (se dejan para la siguiente iteración)
- La realimentación del usuario es fundamental en este proceso
- El progreso es visible

PROCESO ITERATIVO E INCREMENTAL

Fases

- Es preciso diferenciar temporalmente las fases del ciclo de vida
- La división temporal necesita puntos de control

Puntos de control o hitos

- Separan las etapas, las fases, las iteraciones

Disciplinas o Flujos de trabajo

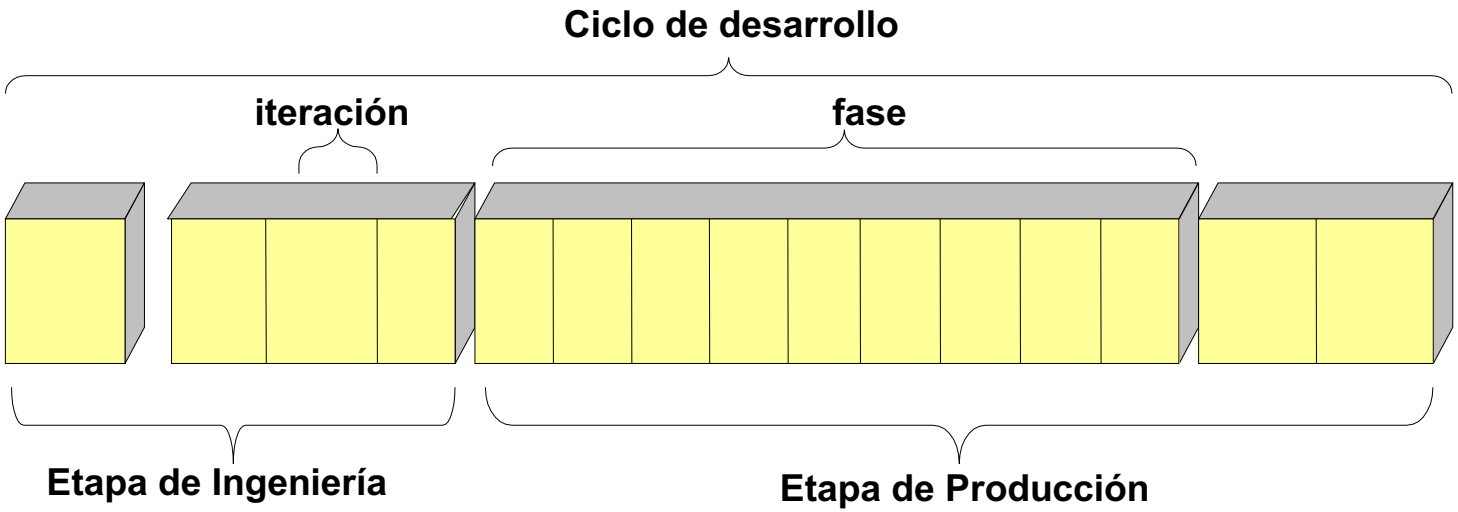
- Organizan las actividades fundamentales de gestión y desarrollo
- Se pueden solapar en el tiempo
- El resultado de las actividades de los flujos de trabajo son los artefactos

Artefactos

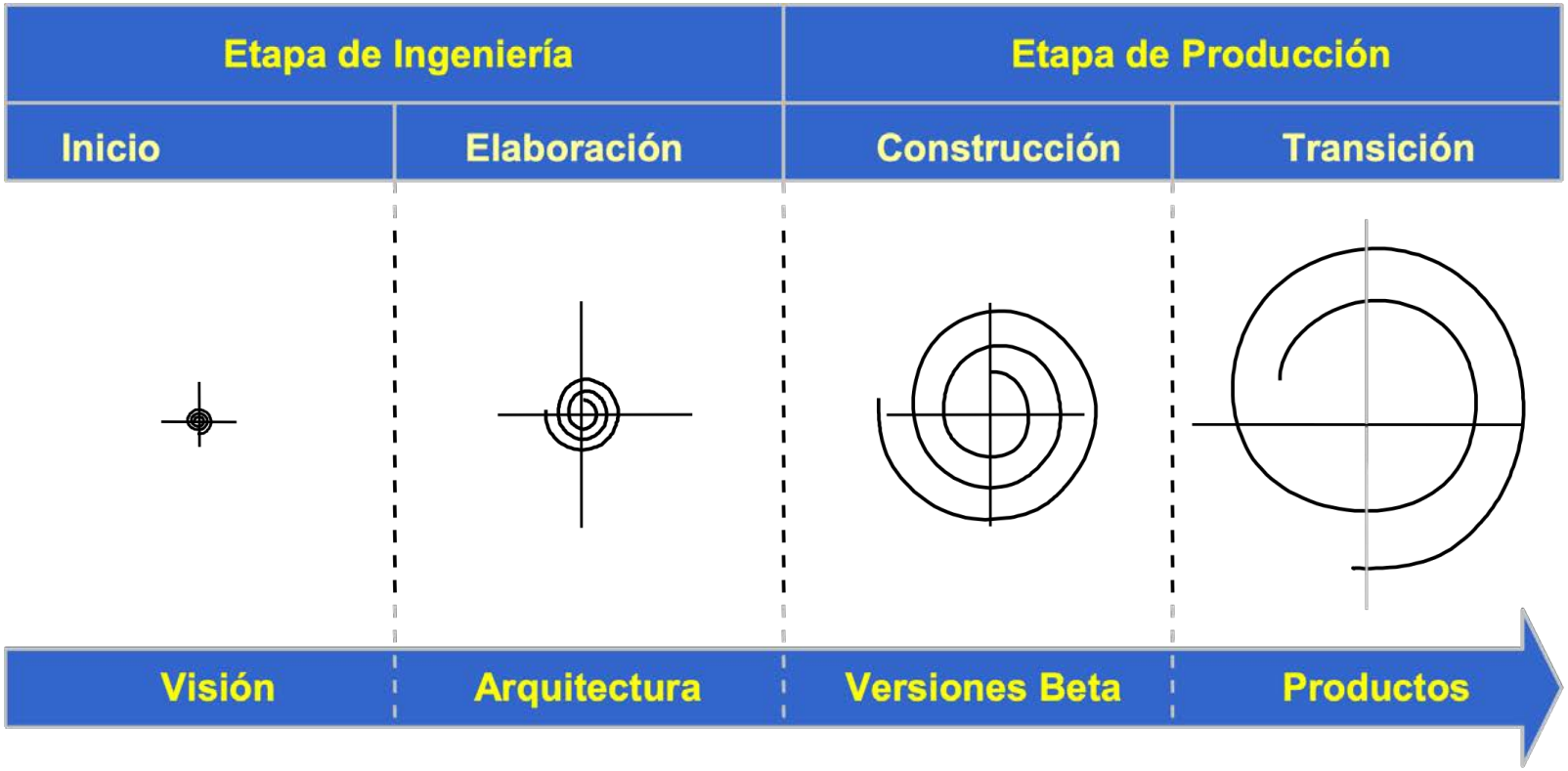
- Cualquier tipo de información producida por los desarrolladores de un sistema (diagramas UML, código, ejecutables, casos de prueba...)
- Se construyen de forma incremental

PROCESO ITERATIVO E INCREMENTAL

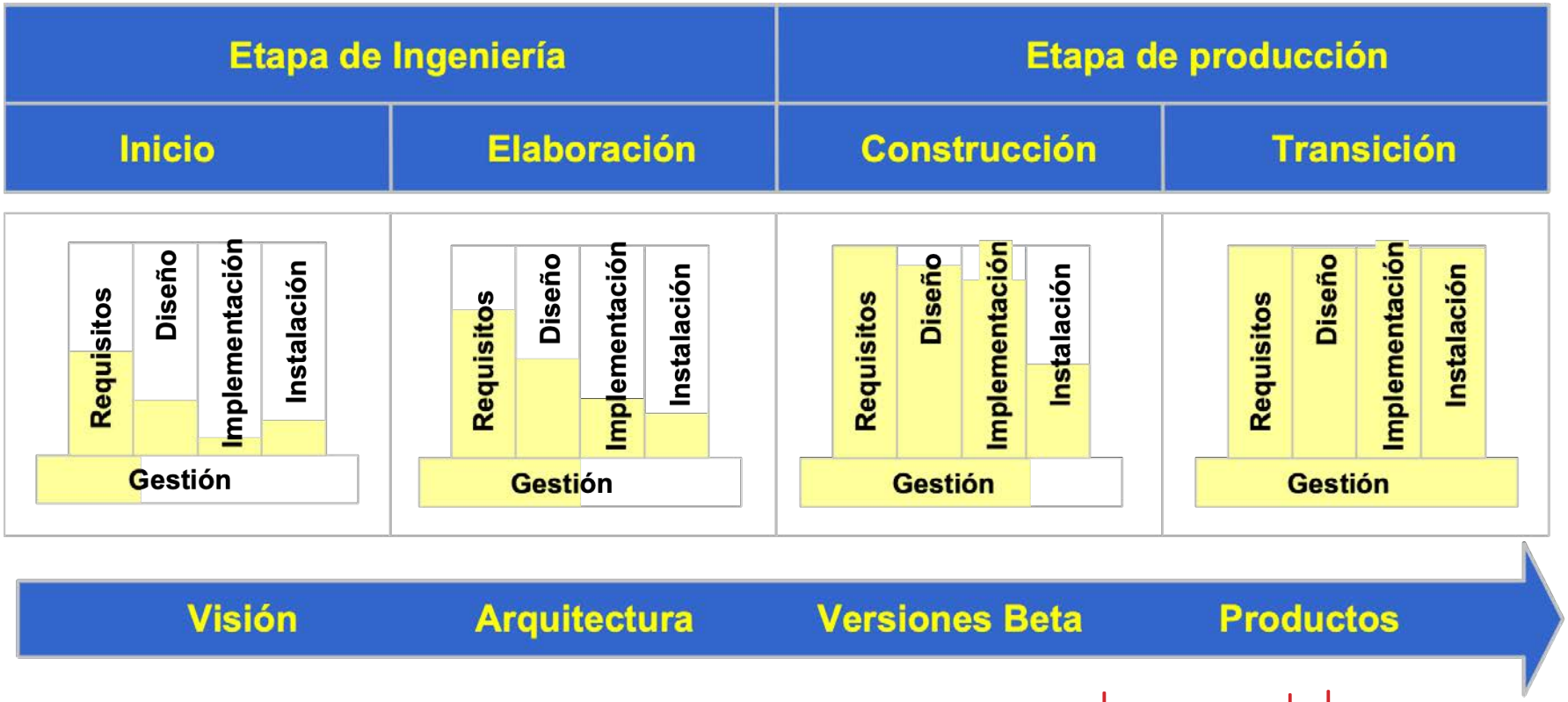
- El Proceso Unificado propone una serie de ciclos de desarrollo
  - Hay que separar claramente la etapa de Ingeniería de la etapa de Producción
  - Cada una de las dos grandes etapas se dividen en fases
  - Las fases se dividen en iteraciones



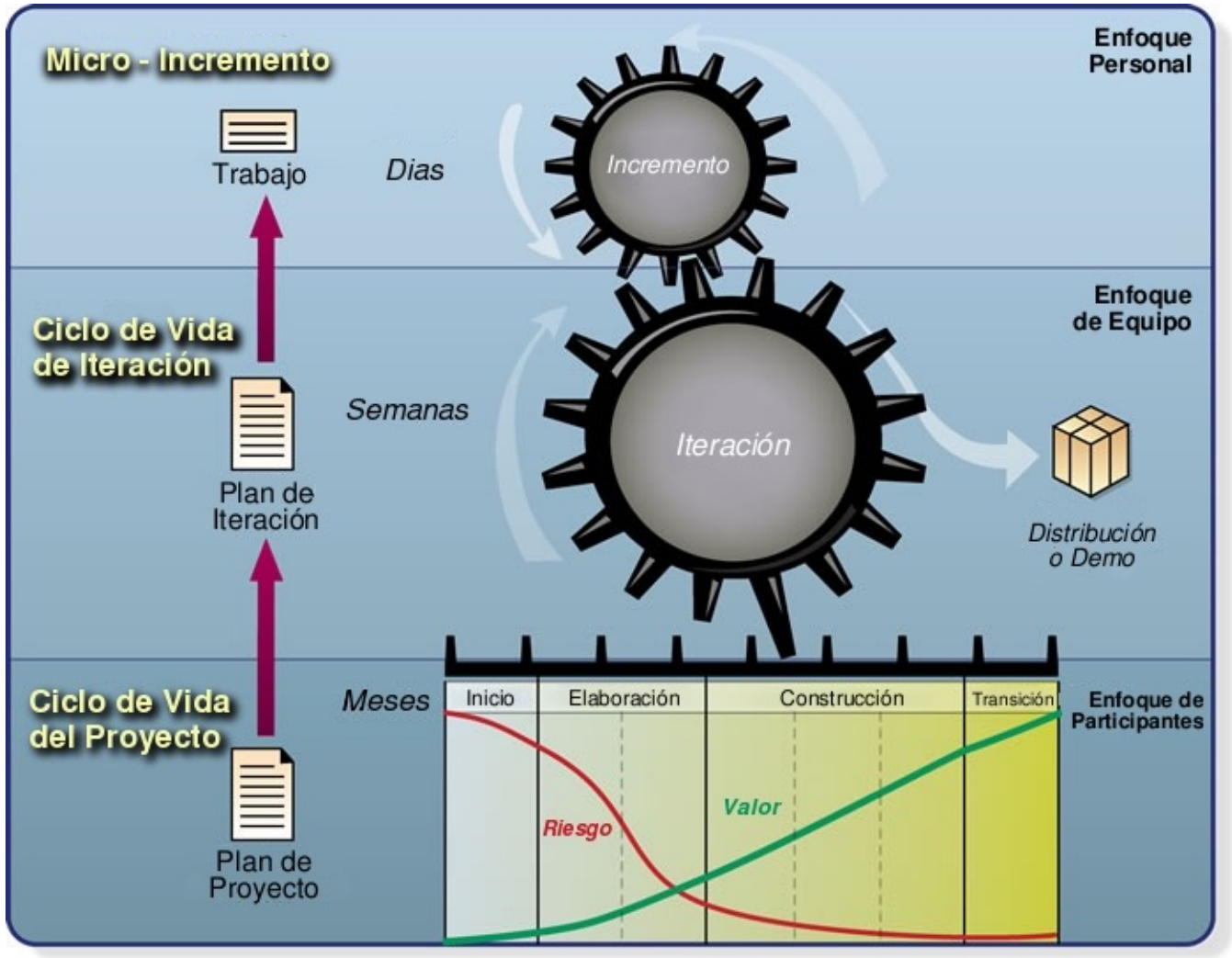
PROCESO ITERATIVO E INCREMENTAL



PROCESO ITERATIVO E INCREMENTAL



ORGANIZACIÓN DEL TRABAJO



https://upload.wikimedia.org/wikipedia/commons/b/bc/Es_three_layers.jpg

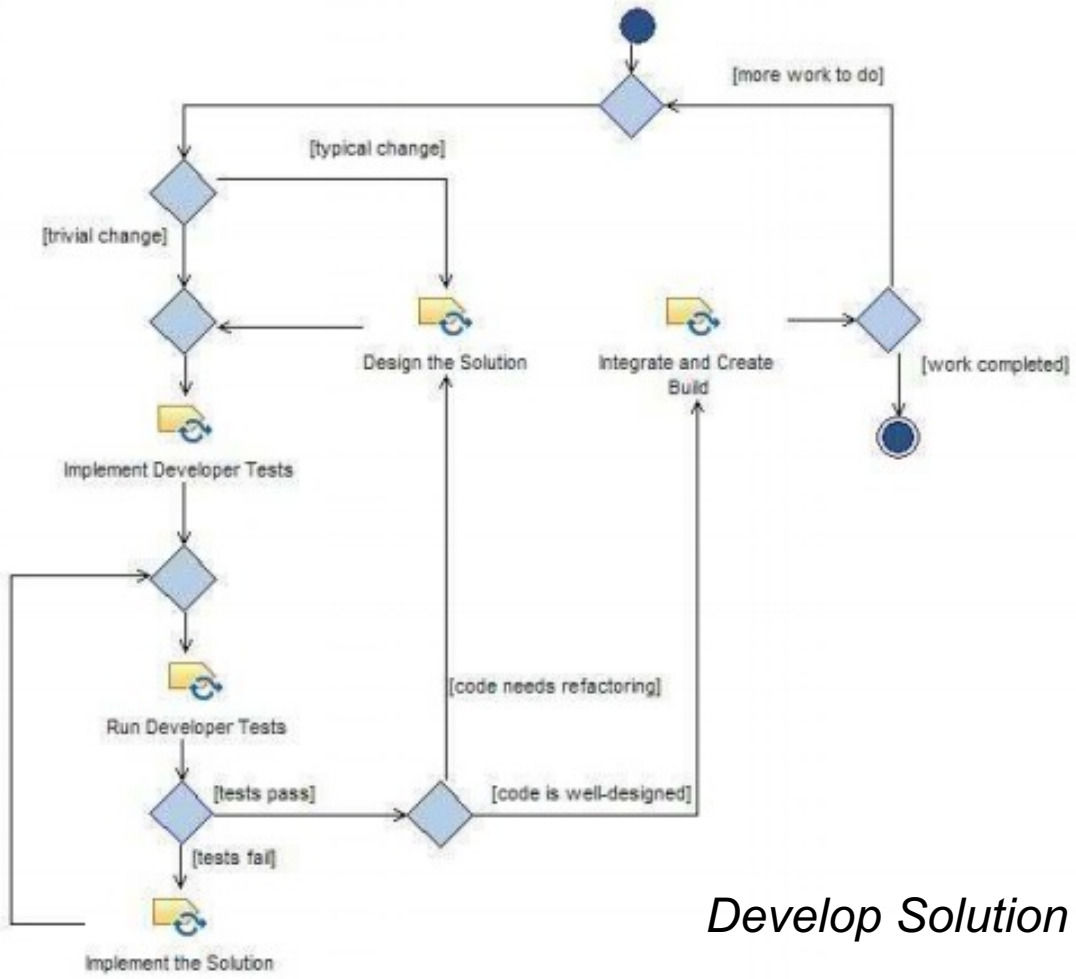
PATRONES DE CAPACIDAD

- El método de Proceso Unificado se crea de forma separada a su aplicación en procesos
- Los métodos ofrecen explicaciones paso a paso para describir cómo se consiguen los objetivos específicos de desarrollo independientemente de su ubicación en un ciclo de desarrollo concreto
- Los procesos toman estos métodos y los relacionan en secuencias semi-ordenadas que se personalizan para los proyectos concretos
- Los elementos de los métodos se organizan en piezas reutilizables que se denominan **patrones de capacidad** (*capability patterns*) que ofrecen una aproximación consistente de desarrollo a las necesidades comunes de los proyectos *software*

PATRONES DE CAPACIDAD

Iteration template patterns	Phase objectives
Inception Phase Iteration Initiate Project Plan and Manage Iteration Identify and Refine Requirements Agree on Technical Approach	▪ Understand what to build ▪ Identify key system functionality ▪ Determine at least one possible solution ▪ Understand the cost, schedule and risks associated with the project
Elaboration Phase Iteration Plan and Manage Iteration Identify and Refine Requirements Define the Architecture Develop Solution Increment Test Solution Ongoing Tasks	▪ Get a more detailed understanding of the requirements ▪ Design, implement, validate, and baseline an Architecture ▪ Mitigate essential risks, and produce accurate schedule and cost estimates
Construction Phase Iteration Plan and Manage Iteration Identify and Refine Requirements Develop Solution Increment Test Solution Ongoing Tasks	▪ Iteratively develop a complete product that is ready to transition to its user community ▪ Minimize development costs and achieve some degree of parallelism
Transition Phase Iteration Plan and Manage Iteration Develop Solution Increment Test Solution Ongoing Tasks	▪ Beta test to validate that user expectations are met ▪ Achieve stakeholder concurrence that deployment is complete

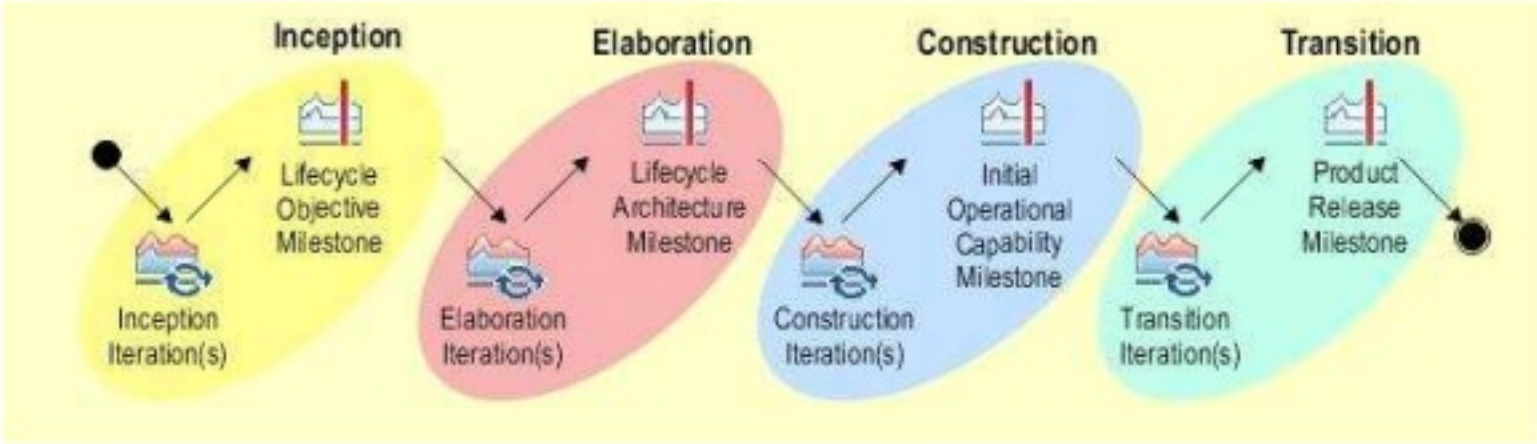
PATRONES DE CAPACIDAD



Develop Solution Increment

PROCESO DE ENTREGA

- Cuando se culmina una iteración por los patrones de capacidad (lo cual puede hacer tantas veces como se planifique) se cierra un ciclo de desarrollo y se da lugar a un proceso de entrega



BIBLIOGRAFÍA

- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Introducción al Proceso Unificado," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021. [Online]. Disponible en: https://bit.ly/396xpCz. doi: 10.5281/zenodo.4420263.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Flujos de trabajo del Proceso Unificado," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021. [Online]. Disponible en: https://bit.ly/2Lb1tFd. doi: 10.5281/zenodo.4423145.
- G. Booch, J. Rumbaugh y I. Jacobson, 2nd, Ed. *The Unified Modeling Language User Guide* (Object Technology Series). Upper Saddle River, NJ, USA: Addison-Wesley, 2005.
- I. Jacobson, G. Booch y J. Rumbaugh, *The Unified Software Development Process* (Object Technology Series). Reading, Massachusetts, USA: Addison Wesley, 1999.
- Object Management Group, "Unified Modeling Language specification version 2.5.1," Object Management Group, Needham, MA, USA, formal/17-12-05, 2017. Disponible en: https://goo.gl/kaE82a
- P. B. Kruchten, "The 4+1 View Model of architecture," *IEEE Software,* vol. 12, no. 6, pp. 42-50, 1995. doi: 10.1109/52.469759.

PROCESO UNIFICADO

INGENIERÍA DE SOFTWARE I

2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2020/2021

Francisco José García Peñalvo / fgarcia@usal.es
Alicia García Holgado / aliciagh@usal.es
Andrea Vázquez Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca

VNiVERSiDAD D SALAMANCA
CAMPUS OF INTERNATIONAL EXCELLENCE

GRIAL Research Group University of Salamanca