



*Universidad Tecnológica de la Mixteca*  
*Instituto de Electrónica y Computación*

# El Proceso Unificado Rational para el Desarrollo de Software.

---

Carlos Alberto Fernández  
y Fernández

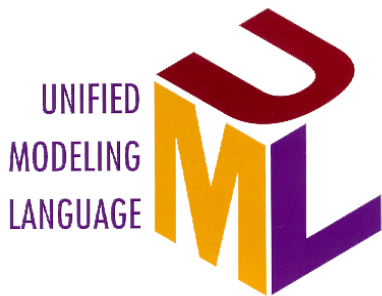
Huajuapán de León, Oaxaca

26 de octubre de 2000

# Objetivo

---

- Proporcionar una visión general del RUP y UML como apoyo para el desarrollo de software de calidad.



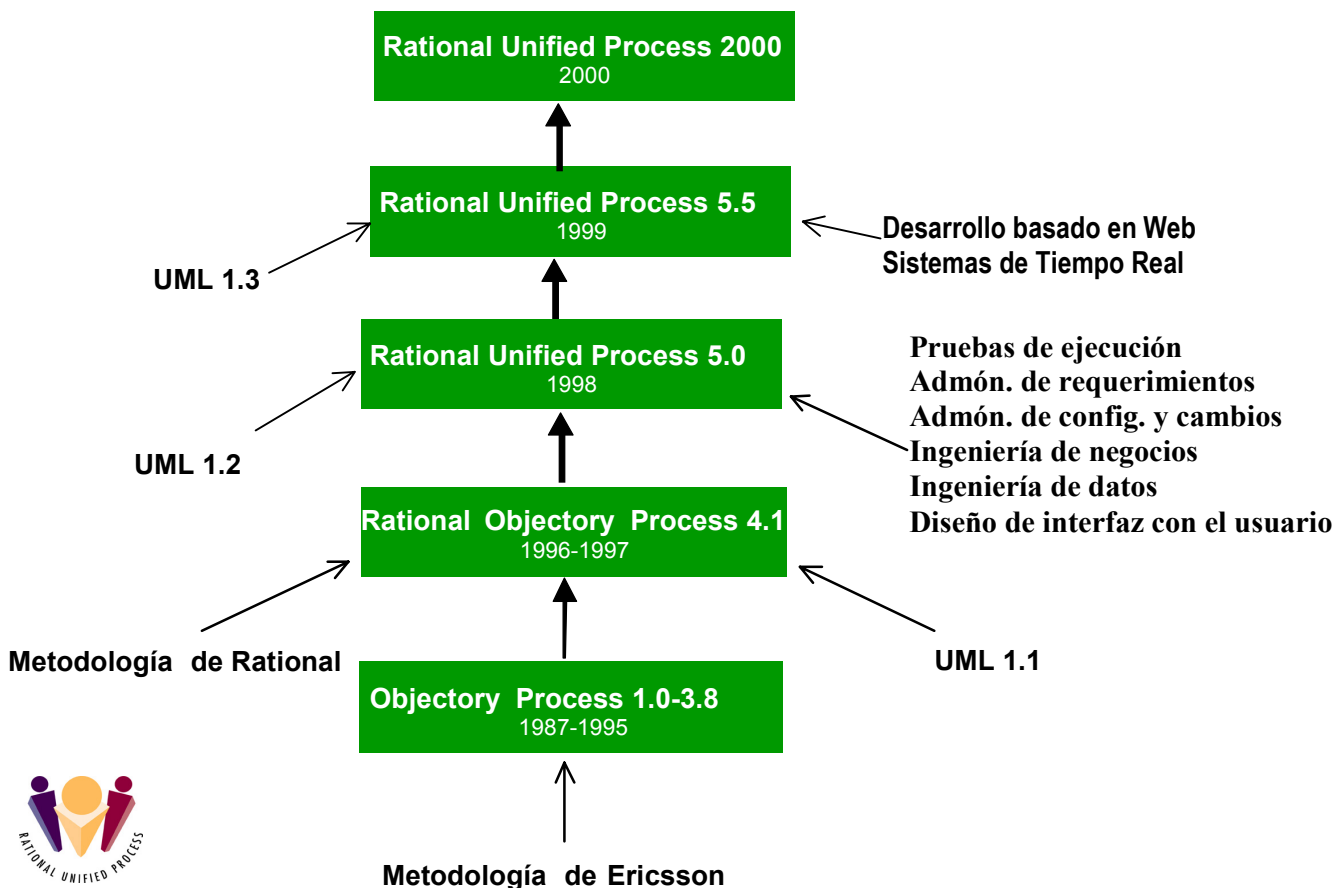
# ¿Qué es un proceso?

- Un **proceso** define **quien** está haciendo **que**, **cuando** lo hace, y **como** hacerle para alcanzar un objetivo.



# RUP: Evolución

No es un proceso nuevo, ha ido fortaleciéndose con el tiempo



# RUP: Objetivos

---

1. Proporcionar una **guía** del orden de las actividades de los equipos.
2. Especificar **cuales** artefactos deben ser desarrollados y **cuando** estos deben ser desarrollados.
3. **Dirigir** las tareas de desarrolladores individuales y equipos como una sola.
4. Ofrecer criterios para **monitorear** y **medir** los productos y actividades del proyecto.

# Mejores prácticas

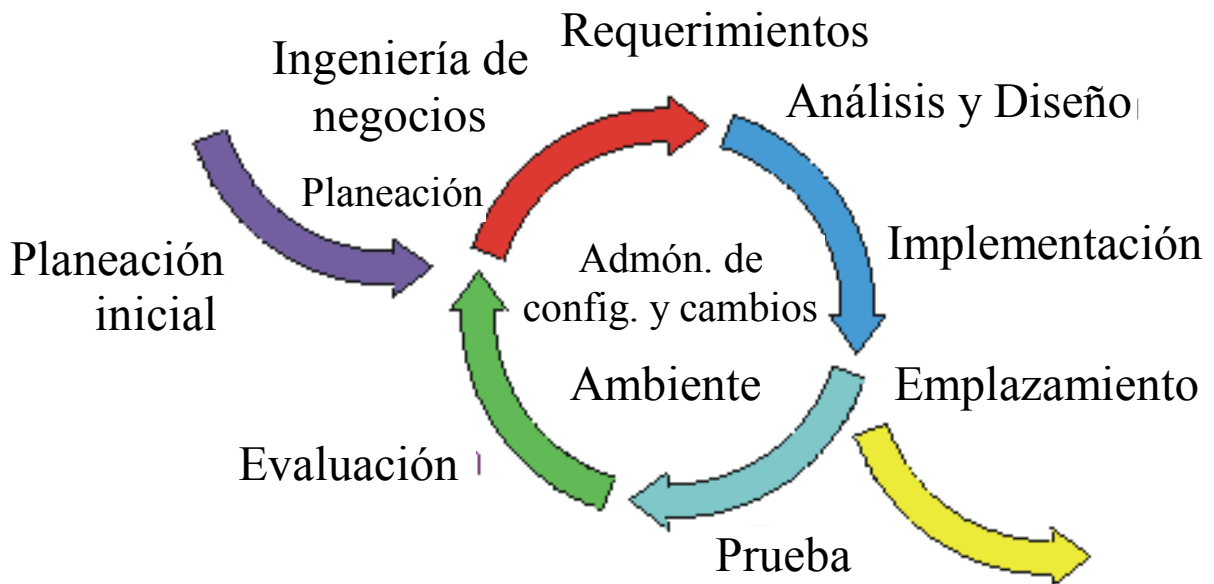
---

1. Desarrollo iterativo.
2. Administración de requerimientos.
3. Arquitectura basada en componentes.
4. Modelado Visual.
5. Verificación de la calidad.
6. Control de cambios.

# Desarrollo Iterativo

---

El desarrollo iterativo propone una planeación inicial y posteriormente entrar a un **ciclo** en las etapas de desarrollo. Donde para cada iteración resulte una versión ejecutable del sistema.



# Desarrollo iterativo: Ventajas

---

- Tolerable a cambios en los requerimientos.
- Los elementos son integrados progresivamente.
- Los riesgos son mitigados en etapas tempranas.
- Permite a la organización aprender e improvisar.
- Facilita el reuso, porque es fácil identificar partes comunes diseñadas o implementadas.
- Resulta un producto más robusto, ya que los errores se van corrigiendo en cada iteración.
- El proceso puede ser improvisado y refinado en el desarrollo.



# Administración de requerimientos

---

- Un **requerimiento** es una condición o capacidad con el que un sistema debe conformarse.
- La **administración de requerimientos** es una aproximación sistemática para la búsqueda, documentación, organización y seguimiento de los cambios en los requerimientos de un sistema.
- El manejo de los requerimientos de software debe de ser **dinámico**: debe esperarse que estos cambien durante la vida de un proyecto de software.

# Arquitectura basada en componentes

---

Uno de los principales objetivos de las primeras iteraciones es obtener una **arquitectura de software** válida, donde en ciclos iniciales de desarrollo formen un prototipo ejecutable de la arquitectura que gradualmente se vaya convirtiendo en el sistema final en las últimas iteraciones.

# Arquitectura: Ventajas

---

- Permite una arquitectura modular.
- Diseño de componentes reusables.
- Aprovechamiento de infraestructuras comerciales (COM, CORBA, JavaBeans)

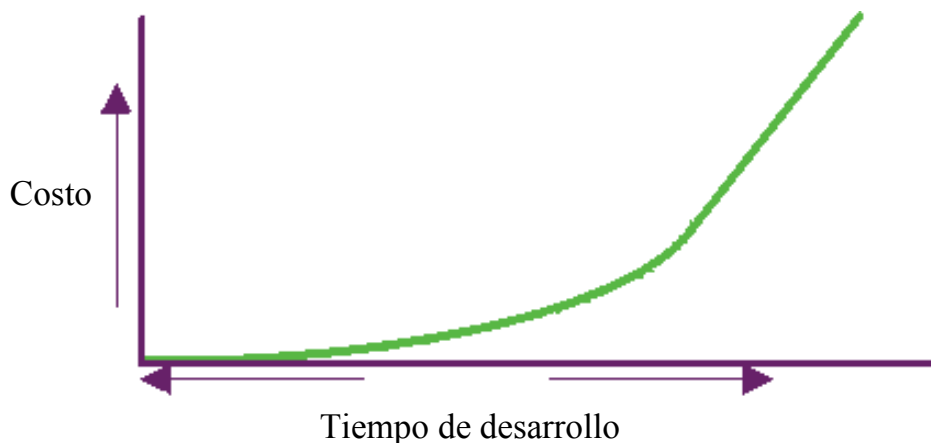


# Control de Calidad

---

Los problemas del software son de 100 a 1000 veces más difíciles de encontrar y reparar (y por tanto más caros) después del desarrollo.

La **verificación** y **administración** de la calidad durante el ciclo de vida del proyecto es esencial para lograr mantener los objetivos y el tiempo estimado de desarrollo.

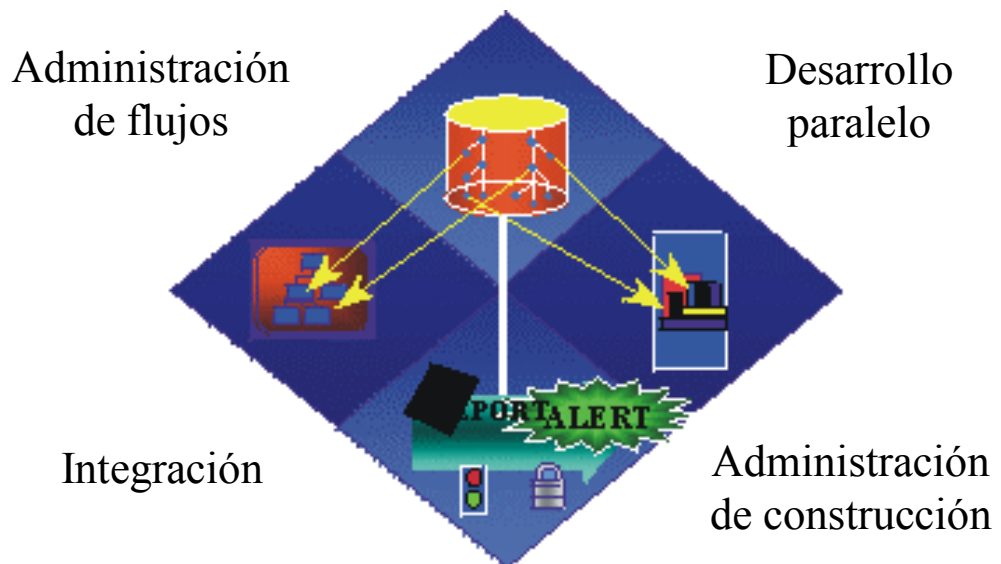


# Control de Cambios

---

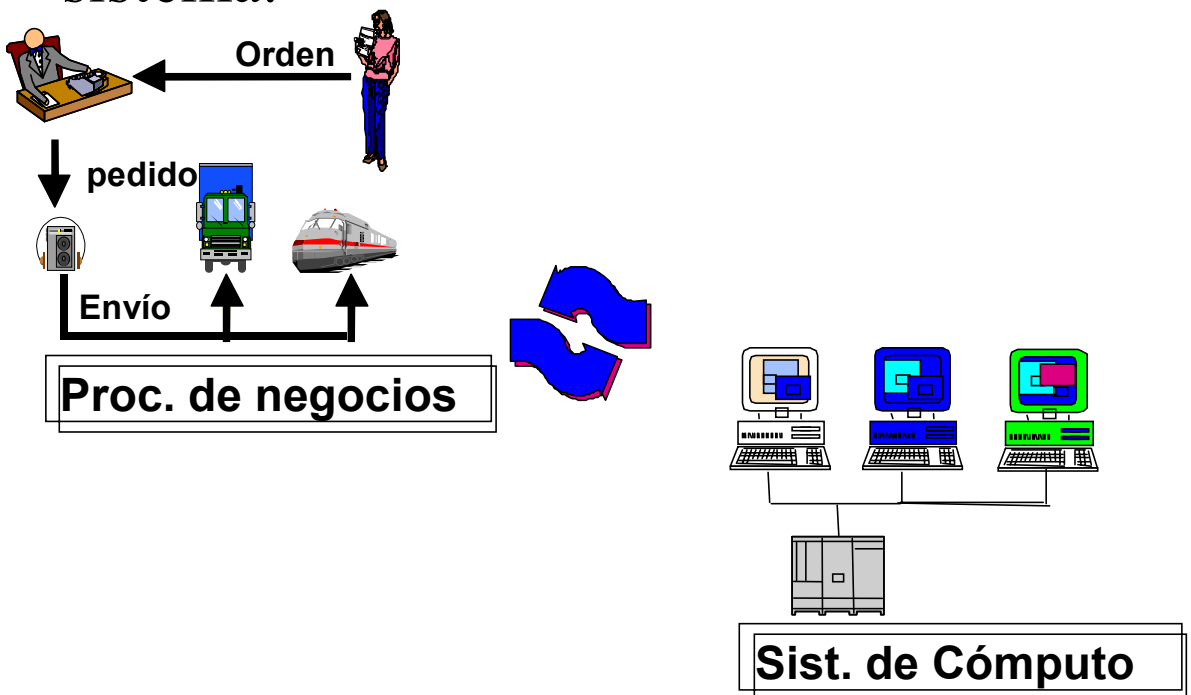
- Si no existe una disciplina de control, el proceso de desarrollo rapidamente degenera en **caos**.
- La coordinación de las actividades y artefactos de los desarrolladores y equipos, involucra **establecer flujos repetibles** para administración de cambios al software. Esta coordinación permite una mejor identificación de los recursos básicos en las prioridades y riesgos del proyecto.

- 
- El control de cambios es más que revisar entradas y salidas en los archivos. Este incluye **administrar** los flujos, el desarrollo paralelo, la integración y la construcción del software.



# Modelado Visual

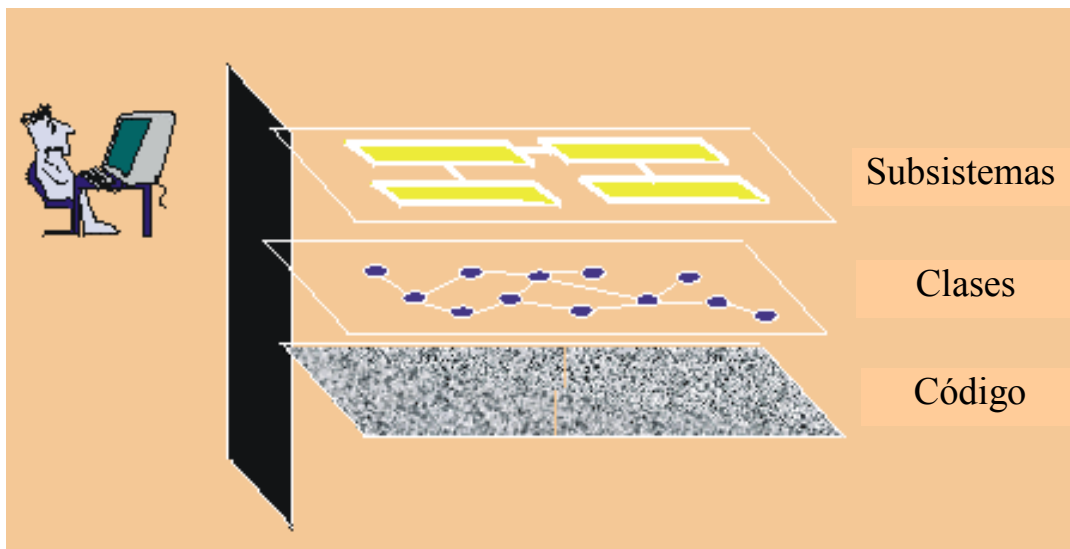
- Un modelo es una simplificación de la realidad que describe completamente un sistema desde una perspectiva particular.
- El modelado es importante porque ayuda al equipo a visualizar, especificar, construir y documentar la estructura y el comportamiento de la arquitectura del sistema.



---

Un Modelo, correctamente diseñado usando tecnología de objetos:

- **Es fácil de entender.** Claramente corresponde a la realidad.
- **Fácil de modificar.** Cambios en un aspecto en particular concierne únicamente al objeto que representa ese aspecto.



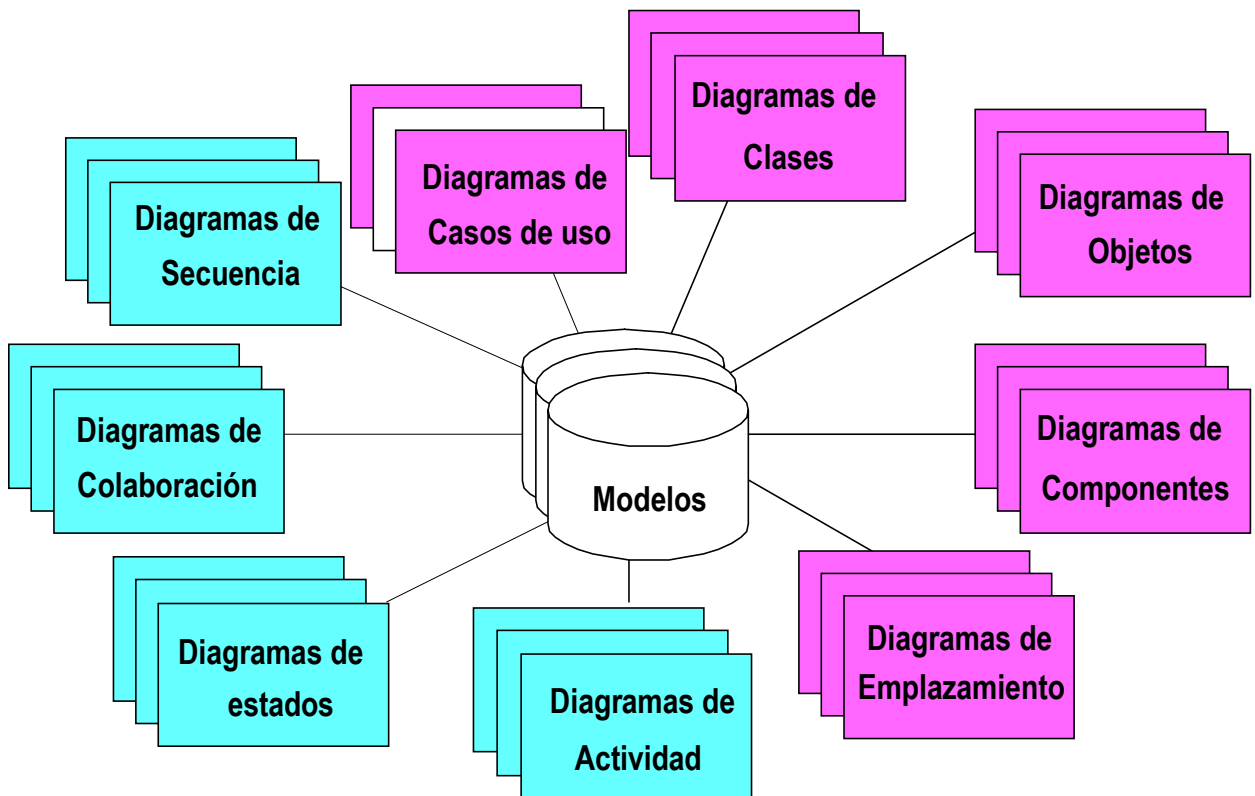
- Se implementa a través de **UML**



# UML

---

- Siglas de *Unified Modeling Language*, resulta de la **unificación** de los principales métodos de análisis y diseño orientado a objetos.



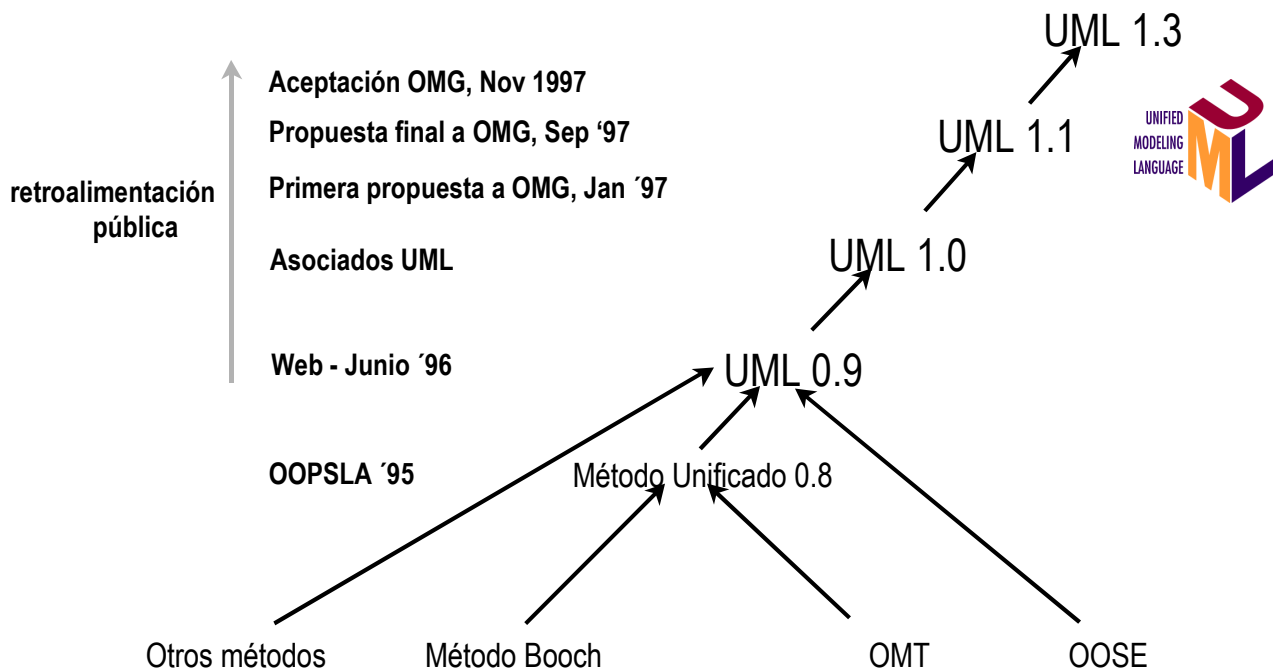
# UML: Características

---

- Desplegar los **límites de un sistema** sus principales funciones mediante casos de uso y actores.
- Representar la **estructura estática** de un sistema usando diagramas de clases.
- Modelar los **límites de un objeto** con diagramas de estados.
- Mostrar la **arquitectura de la implementación** física con diagramas de componentes y de emplazamiento o despliegue.

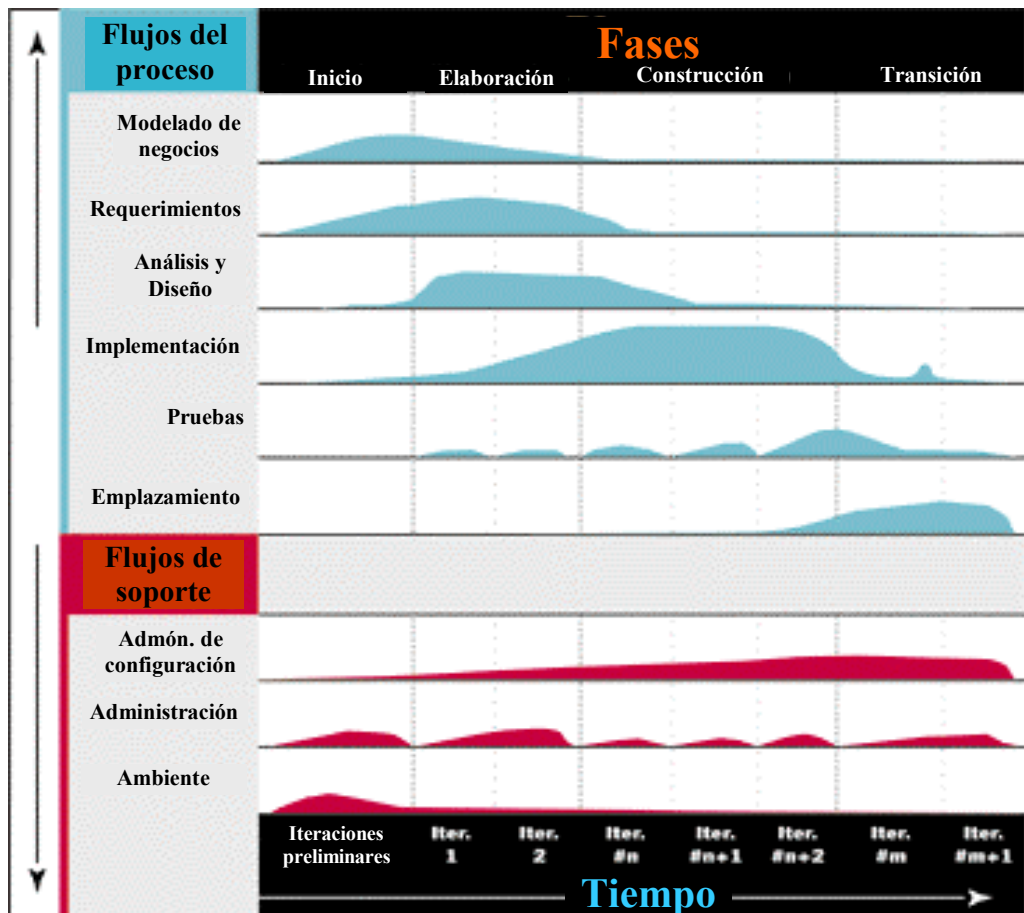
# UML: Historia

- UML es un estándar de **OMG** (*Object Management Group*) a partir de noviembre de 1997, para la visualización, especificación, construcción y documentación de sistemas de software.



# RUP: Esquema

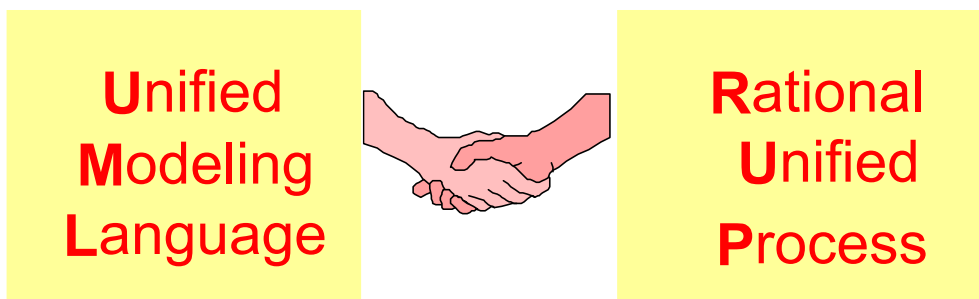
El RUP organiza a los proyectos en términos de **flujos** de trabajo y **fases**, las cuales consisten de una o más iteraciones. En cada iteración, el énfasis en cada flujo de trabajo variará a lo largo del ciclo de vida.



# Conclusiones

---

- Es un esqueleto del proceso a desarrollar.
- Iterativo e incremental.
- Maneja Casos de Uso.
- Es diseñado para ser flexible y extendible:
- Permite una variedad de estrategias de ciclos de vida.
- Elegir que "artefactos" producir.
- Define actividades y trabajadores.
- **No** es un Proceso Universal.



**Dos partes de un Conjunto Unificado.**

# Referencias

---

- [Booch, 1999] BOOCH, Grady. **Software Architecture and the UML**. Rational Software Corporation. 1999. de julio de 2000.
- [Jacobson, 1999] JACOBSON, Ivar. **Applying UML in The Unified Process**. Rational Software Corporation. 1999.[Fowler, 1999]
- [Fowler, 1999] FOWLER, Martin. **UML Gota a Gota**. México. Ed. Addison Wesley. 1999.
- [Larman, 1999] LARMAN, Craig. **UML y Patrones**. México. Ed. Prentice Hall. 1999.
- [Rational, 1997] **Analysis and Design with UML**. Rational Software Corporation. USA. 1997.
- [Rational, 1999] **Inside the Unified Modeling Language**. Rational Software Corporation. USA. 1999.
- [Rational, 1999b] **Object-Oriented Analysis and Design Using the UML. Student Manual**. Rational Software Corporation and Rational University. USA. 1999.
- [Rational, 2000] **Rational Unified Process**. Versión 2000. Rational Software Corporation. USA. 2000.