




# Análisis y Diseño de Sistemas 2

2015

Ing. Luis Alberto Arias Solórzano

Unidad 2

# Esquema del proceso

- ▶ El arquitecto:
    - **Trabaja con el equipo de requerimientos:** el equipo de requerimientos colecta los requerimientos de los stakeholders, el arquitecto los entiende y se asegura que se especifiquen los atributos de calidad para suplir las necesidades.
    - **Trabaja con los stakeholders de la aplicación:** trabaja con los stakeholder aparte de los requerimientos funcionales del cliente para suplir otro tipo de necesidades que puedan surgir (Ej. El administrador del sistema quiere una aplicación fácilmente monitoreable, instalable, actualizable y administrable).
- 

# Esquema del proceso

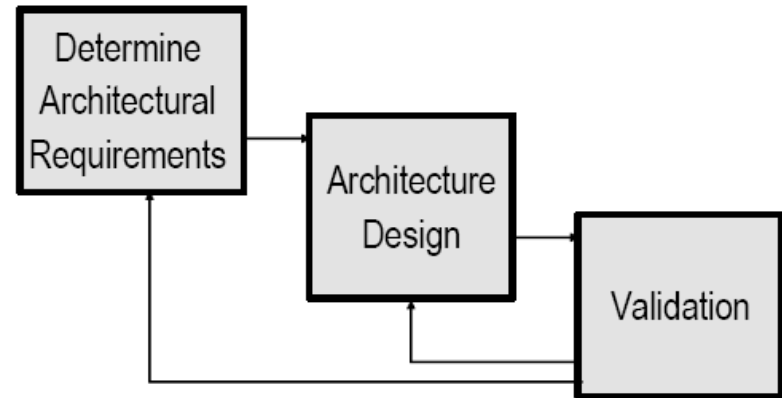
- ▶ El arquitecto:
  - **Lidera el equipo de diseño técnico:** definir la arquitectura de una aplicación es una actividad de diseño el arquitecto lidera el equipo de diseño para producir el esquema general de arquitectura
  - **Trabaja con la administración del proyecto:** asiste en la planificación, estimación y distribución de tareas y calendario.

**Para asistir al arquitecto en la definición de la estructura es de utilidad seguir un proceso de ingeniería de software definido**



# Proceso de tres pasos para definir la arquitectura

- ▶ **Definir requerimientos de arquitectura:** Crear una declaración o un modelo de los requerimientos que dirigirá la arquitectura
- ▶ **Diseño de la arquitectura:** Definir la estructura y responsabilidades de los componentes que abarcan la arquitectura
- ▶ **Validación:** “Probar” la arquitectura

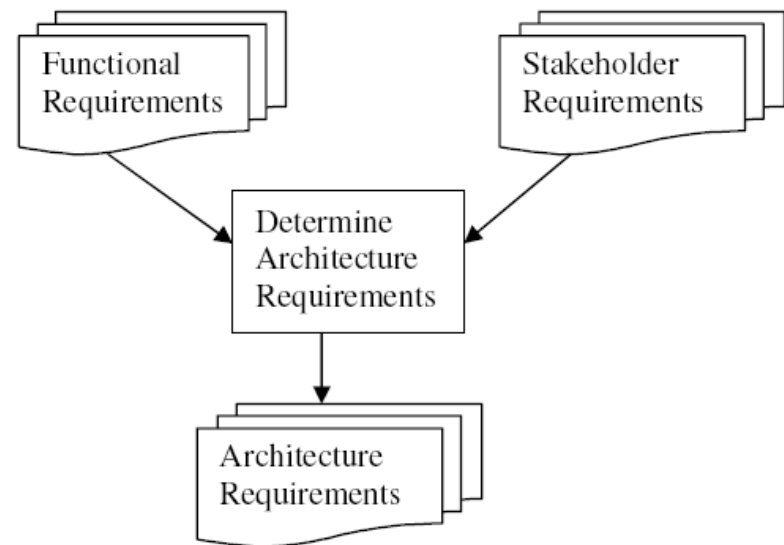


**La naturaleza de este modelo es iterativo!**

# Determinar los requerimientos de arquitectura

## ▶ Ejemplos:

- La comunicación entre componentes debe ser garantizada para no incurrir en pérdida de mensajes (**fiabilidad en la comunicación**)
- El sistema debe utilizar el web server existente, es IIS y Active Server Pages para procesar peticiones web (**restricción**)



- Priorizar los requerimientos de arquitectura:
  - Alta: Este requerimiento dirige el diseño de la arquitectura
  - Media: Este requerimiento debería ser soportado eventualmente
  - Baja: Wish list. “Sería bueno tenerlo”

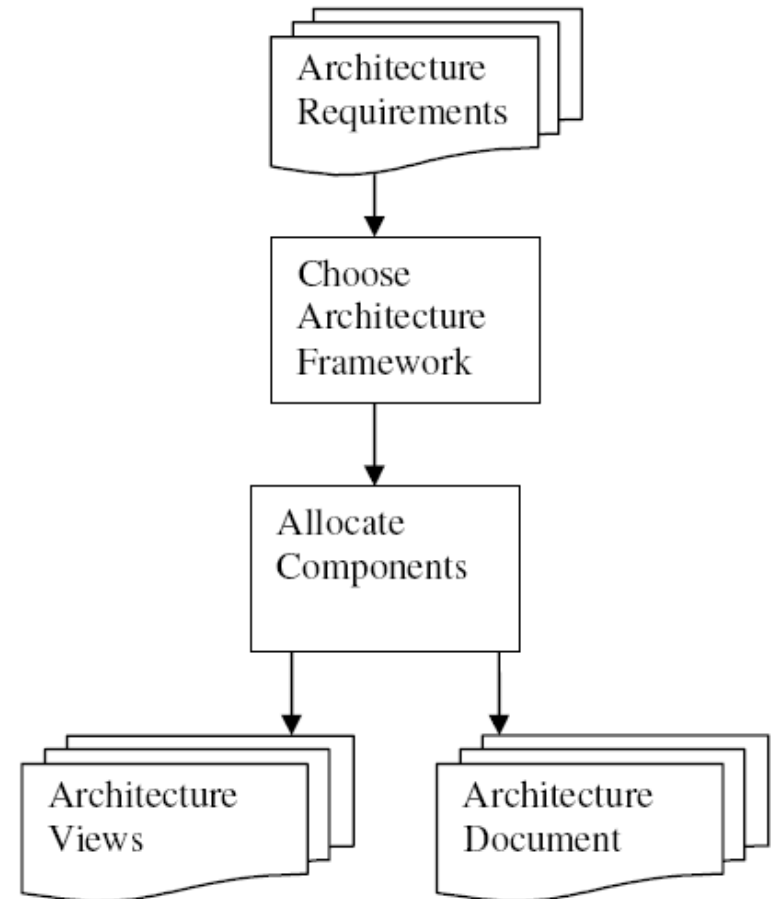
# Diseño de la arquitectura

## ► Escoger un framework de arquitectura:

- N-Capas
- Messaging
- Publish Suscribe
- Broker
- Process Coordinator

## ► Distribuir componentes:

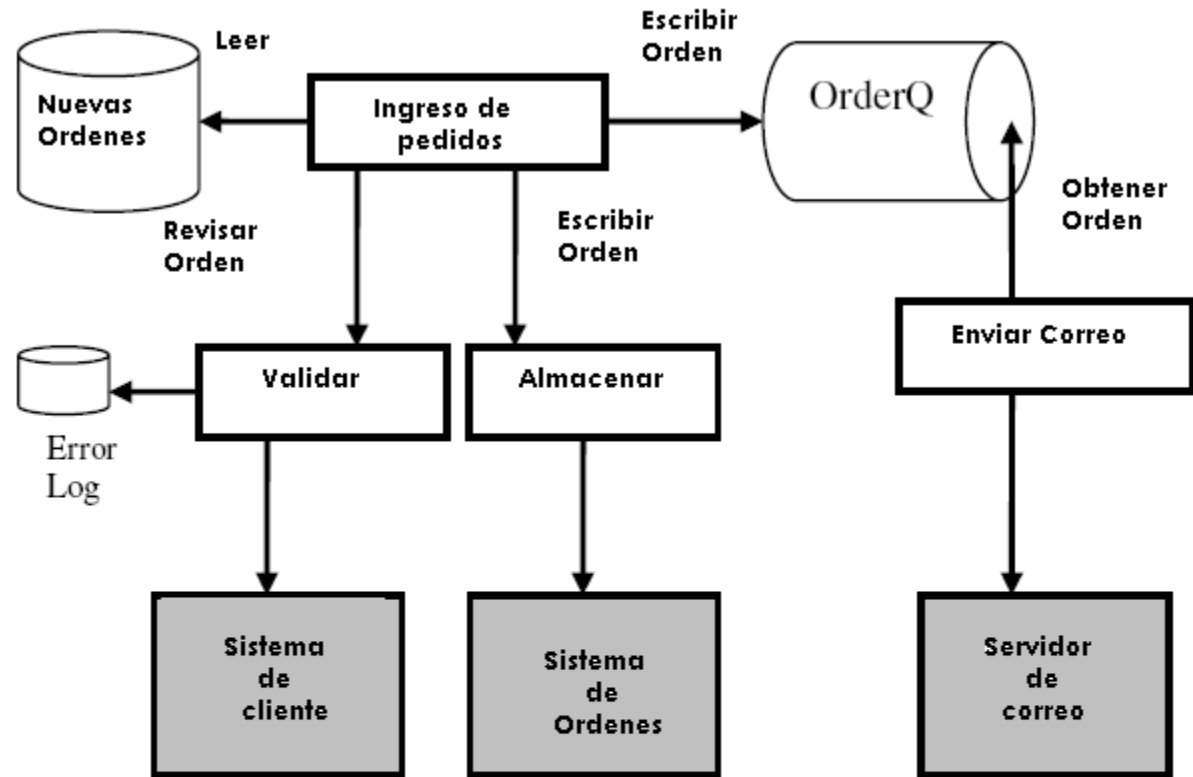
- Id. Los componentes esenciales
- Id. Interfaces entre ellos
- Id. Responsabilidades de c/u
- Id. Dependencias entre ellos
- Id. Particiones en la arquitectura que la hagan candidata para ser distribuida en varios servidores una red



# Guías en el diseño de componentes

- ▶ Minimizar dependencias entre componentes (loosely coupled). Cambios en un componente no deben tener un alto impacto en nuestra arquitectura.
- ▶ Diseñar componentes que encapsulen un conjunto de responsabilidades con un alto grado de cohesión (highly cohesive). La cohesión es una medida de que tan bien las partes de un componente trabajan juntas.
- ▶ Aislar dependencias en middleware y cualquier tecnología COTS utilizada, mientras menos componentes dependientes entre si hay, mas fácil será realizar cambios para actualizar el middleware u otros servicios de infraestructura.
- ▶ Minimizar llamadas entre componentes, ya que estos pueden afectar el performance si hay componentes distribuidos.

# Ejemplo



## Figure Key

Existing Component



New Component



Dependency



Database



Persistent Queue

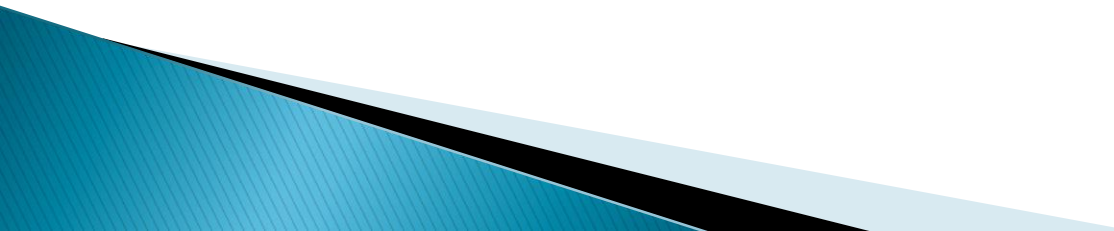




# Validación

- ▶ Incrementa la confianza al equipo de diseño de que la arquitectura cumple con su propósito, debe ser alcanzada en una agenda y presupuesto definido.
  - Técnicas
    - Escenarios
    - Construcción de prototipos

El propósito es identificar potenciales defectos y debilidades en el diseño para que pueda ser mejorado previo a su implementación.



# Prototipos

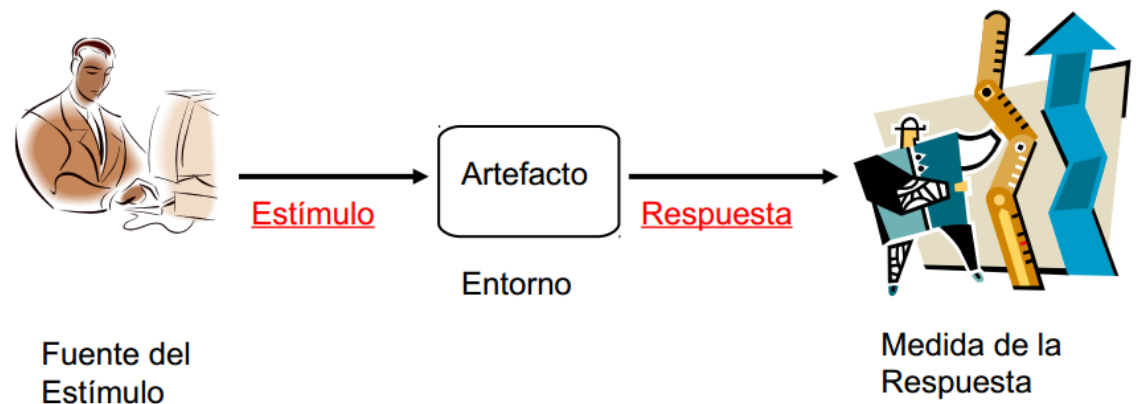
- ▶ Son creados para probar algunos aspectos de diseño de alto riesgo y que no hallan sido entendidos completamente
- ▶ Se usan específicamente para dos propósitos:
  - **Prueba de concepto:** ¿puede el diseño de la arquitectura ser construido y satisfacer así los requerimientos?
  - **Prueba de tecnología:** la tecnología seleccionada ¿Se comporta como se esperaba?

# Escenarios de calidad

Quality Attribute	Stimulus	Response
Availability	The network connection to the message consumers fails.	Messages are stored on the MOM server until the connection is re-stored. Messages will only be lost if the server fails before the connection comes back up.
Modifiability	A new set of data analysis components must be made available in the application.	The application needs to be rebuilt with the new libraries, and the all configuration files must be updated on every desktop to make the new components visible in the GUI toolbox.
Security	No requests are received on a user session for ten minutes.	The system treats this session as potentially insecure and invalidates the security credentials associated with the session. The user must logon again to connect to the application.
Modifiability	The supplier of the transformation engine goes out of business.	A new transformation engine must be purchased. The abstract service layer that wraps the transformation engine component must be re-implemented to support the new engine. Client components are unaffected as they only use the abstract service layer.
Scalability	The concurrent user request load doubles during the 3 week enrollment period.	The application server is scaled out on a two machine cluster to handle the increased request load.

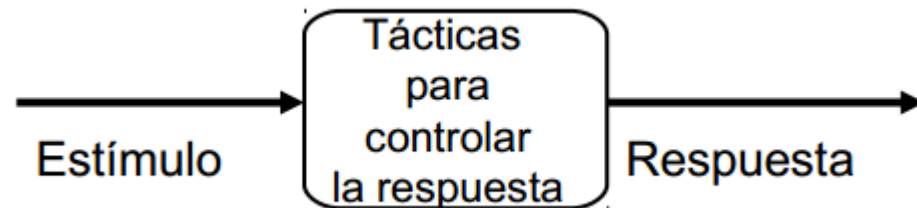
# Tácticas

- ▶ Los requerimientos de calidad (escenarios) especifican las respuestas del software a determinados estímulos.
- ▶ El arquitecto debe aplicar ciertas tácticas para poder diseñar una arquitectura de software que permita brindar estas respuestas.




# Tácticas


- ▶ **Definición:** Decisión de diseño que afecta el control de una respuesta a un atributo de calidad. Es un mecanismo arquitectónico, sinónimo.
- ▶ **Estrategia Arquitectónica:** Conjunto de tácticas.



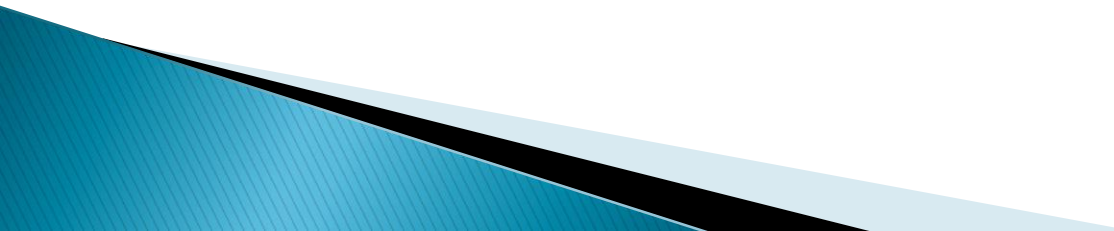
# ¿Porqué documentar la arquitectura?

- ▶ Otros puedan entender y evaluar el diseño
  - ▶ Podamos entender el diseño luego de un buen tiempo.
  - ▶ Otros miembros del proyecto puedan aprender de la arquitectura digiriendo el pensamiento detrás del diseño
  - ▶ Se pueda analizar el diseño para quizá mejorar el desempeño o generar métricas
- 

# ¿Porqué es problemático documentar la arquitectura?

- ▶ No hay estándar en documentación de arquitectura universalmente aceptada
  - ▶ Una arquitectura puede ser muy compleja y documentarla de forma entendible consume mucho tiempo
  - ▶ Una arquitectura tiene múltiples vistas, documentarlas todas consume mucho tiempo y dinero
  - ▶ Mantener la documentación de una arquitectura al día puede consumir muchos recursos de tiempo y calendario
- 

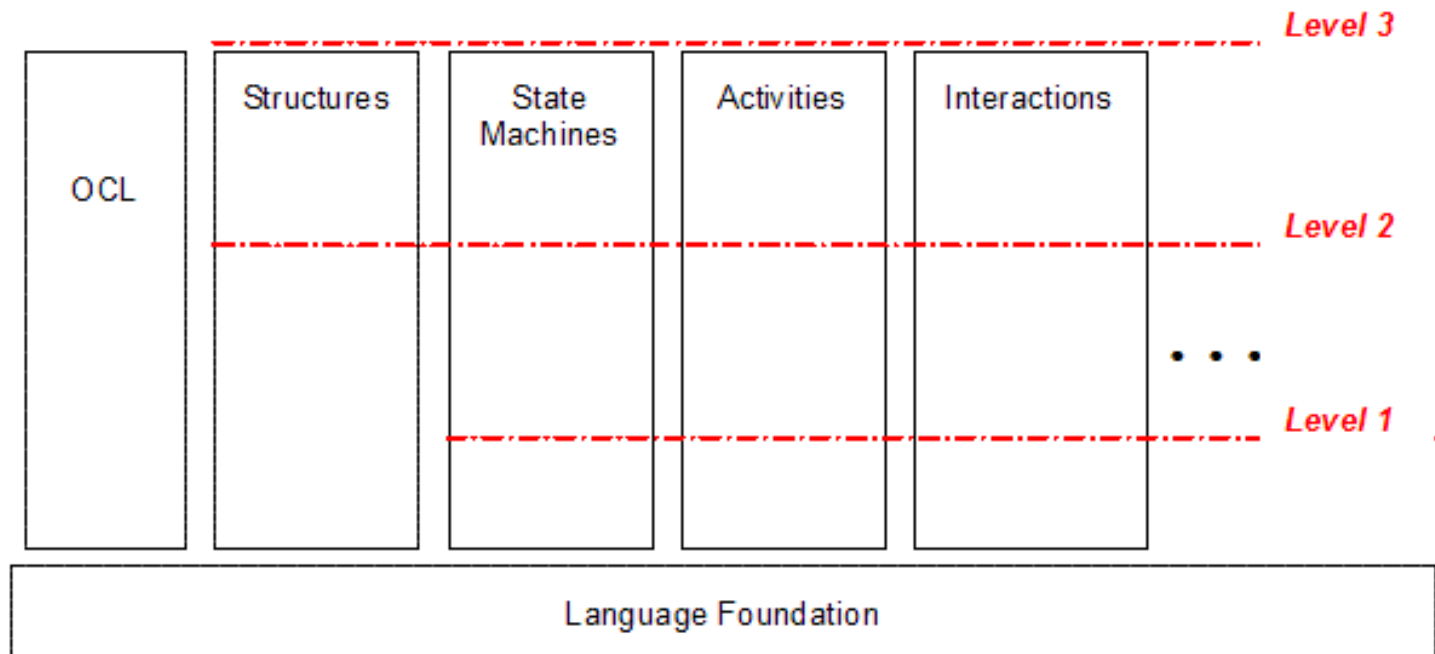
# ¿Qué deberíamos documentar?

- ▶ Interfaces entre componentes
  - ▶ Restricciones de subsistemas
  - ▶ Escenarios de prueba
  - ▶ Decisiones de compra de componentes third party
  - ▶ Estructura del equipo y dependencias de agenda
  - ▶ Servicios externos que ofrece la aplicación
- 



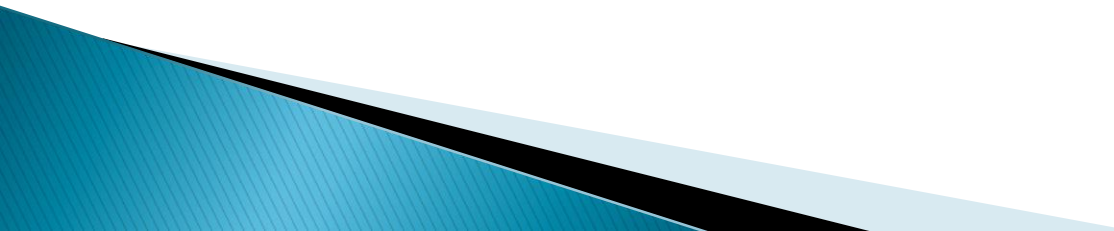
# UML 2.0

- ▶ La notación UML 2.0 cubre aspectos tanto de comportamiento como de estructura en un sistema de software



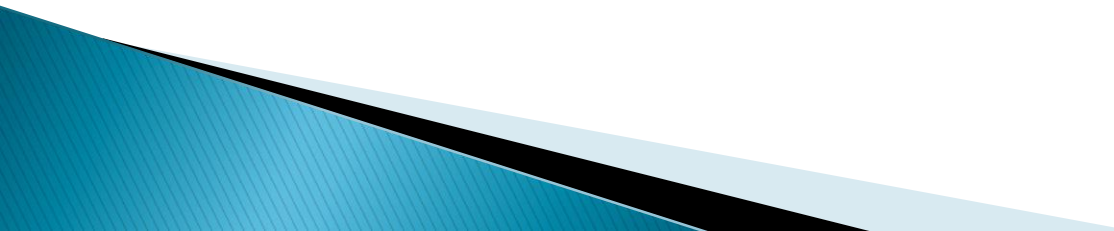
# UML 2.0

## Aspectos estructurales

- ▶ Diagrama de clases
  - ▶ Diagrama de componentes
  - ▶ Diagrama de paquetes
  - ▶ Diagrama de deployment
  - ▶ Diagrama de objetos
  - ▶ Diagrama de estructuras de composición
- 

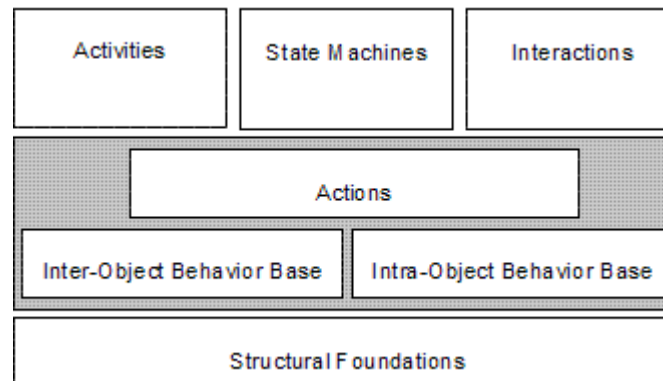
# UML 2.0

## Aspectos de comportamiento

- ▶ Diagrama de actividades
  - ▶ Diagrama de secuencias
  - ▶ Diagrama de comunicación
  - ▶ Diagrama de estado de maquinas
  - ▶ Diagrama de descripción general
  - ▶ Diagrama de tiempos
  - ▶ Diagrama de casos de uso
- 

# Lectura

- ▶ [http://www.ibm.com/developerworks/rational/library/05/321\\_uml/](http://www.ibm.com/developerworks/rational/library/05/321_uml/)



**Gracias**