.UBA fiuba

FACULTAD DE INGENIERÍA

75.73 Arquitectura de Software

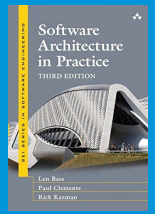
Christian Calónico
Guillermo Rugilo
Mariano D'Ascanio
Nicolás Fernández T.

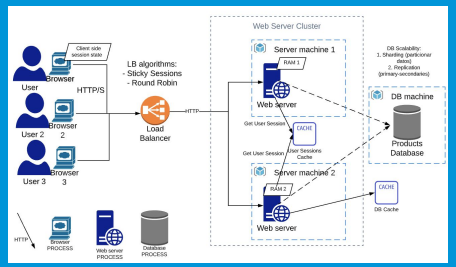
Primer Cuatrimestre 2023

Agenda







Primer Bloque

- El curso
  - Nosotros
  - Comunicación
  - Condiciones de aprobación
- Contenido de la materia
- Material / Bibliografía
- Experiencias nuestras y vuestras
- Introducción a la Arquitectura de Software

Segundo Bloque

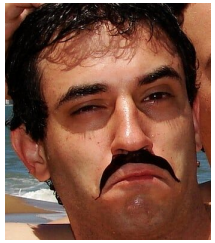
- Estructuras y vistas
- Atributos de Calidad
- Tácticas
- Red conceptual

El curso

Profesor Titular

Christian Calónico

Ayudantes



Guillermo Rugilo

Mariano D’Ascanio

Nicolás Fernández T.

Jueves 18 a 22 hs.

Comunicación

Las clases http://bit.ly/arq-soft-feedback-clases

Slack https://arquitectura-7573.slack.com

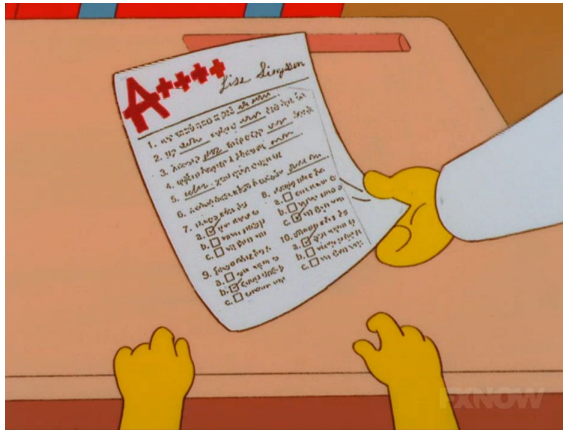
#general
#contenidos
#ejercicios
#tp

Google Drive http://bit.ly/arq-soft-drive

Sí muy lindo, pero decime cómo apruebo

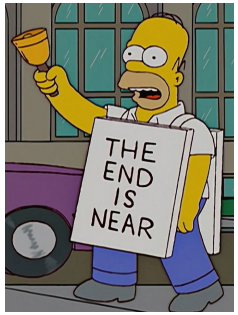
- 1 examen parcial, 2 recuperatorios
  (a ser confirmado)
- 1 Trabajo Práctico, en 2 partes
- **Asistencias**: 75%
- **Hablemos de los condicionales**

- Evaluación Integradora



Pedido especial del plantel docente para las clases virtuales:
¡Por favor compartan el video de sus cámaras siempre que sea posible!

Primer comunicado



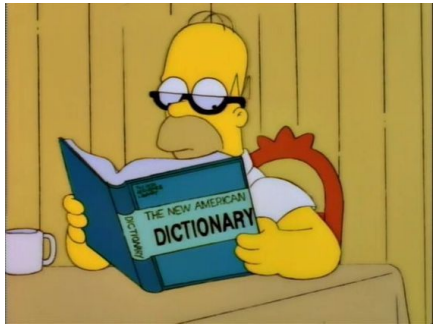
1. Unirse al grupo de Slack https://arquitectura-7573.slack.com

2. Agendar link al Google Drive http://bit.ly/arq-soft-drive

3. Preparativos para el TP

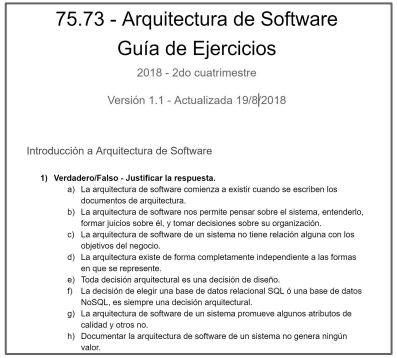
    a. Armar grupo de 4 personas
    b. Github Student Developer pack
        i. https://education.github.com/pack
    c. Gestionar cuenta @fi.uba.ar

Contenido del curso



1. **Introducción a la Arquitectura de Software**
2. **Atributos de Calidad y Tácticas**
   a. Performance, Escalabilidad, Disponibilidad, Confiabilidad, Visibilidad, Modificabilidad, Portabilidad, Seguridad, Interoperabilidad, Testeabilidad, Usabilidad, etc.
   b. Tácticas
3. **Estilos de Arquitectura**
   a. Replicated Repository, Cache, Client Server, Remote Session, Stateless, Layers, Code on Demand, Remote Evaluation, Uniform Pipes & Filters, Events Based Integration, etc.
   b. REST
   c. La Web
4. **Cloud Computing Architecture**
5. **Patterns of Enterprise Application Architecture**
6. **SDGT y SOA**
7. **NoSQL**

Material y bibliografía básica


75.73 - Arquitectura de Software

Guía de Ejercicios

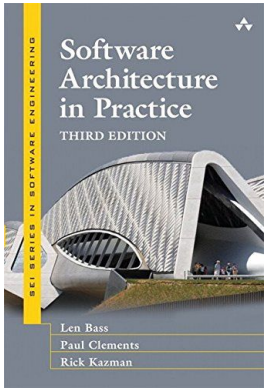
2018 - 2do cuatrimestre

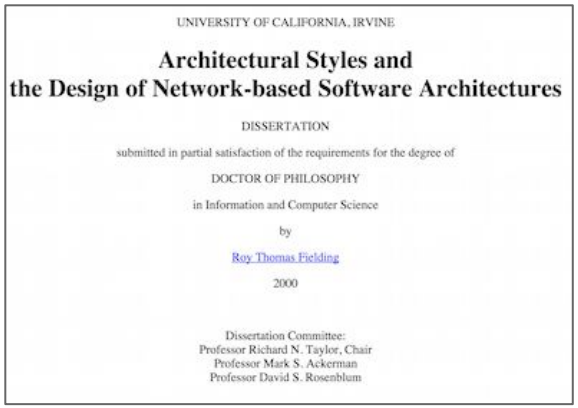
Versión 1.1 - Actualizada 19/8/2018

Introducción a Arquitectura de Software

1) **Verdadero/Falso - Justificar la respuesta.**
   a) La arquitectura de software comienza a existir cuando se escriben los documentos de arquitectura.
   b) La arquitectura de software nos permite pensar sobre el sistema, entenderlo, formar juicios sobre él, y tomar decisiones sobre su organización.
   c) La arquitectura de software de un sistema no tiene relación alguna con los objetivos del negocio.
   d) La arquitectura existe de forma completamente independiente a las formas en que se represente.
   e) Toda decisión arquitectural es una decisión de diseño.
   f) La decisión de elegir una base de datos relacional SQL ó una base de datos NoSQL, es siempre una decisión arquitectural.
   g) La arquitectura de software de un sistema promueve algunos atributos de calidad y otros no.
   h) Documentar la arquitectura de software de un sistema no genera ningún valor.

- **Guía de ejercicios**
- "**Software Architecture in Practice**", 3rd Edition. Len Bass, Paul Clements y Rick Kazman. 2013.
- "**Architectural Styles and the Design of Network-based Software Architectures**", Roy T. Fielding dissertation, 2000.




UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

2000

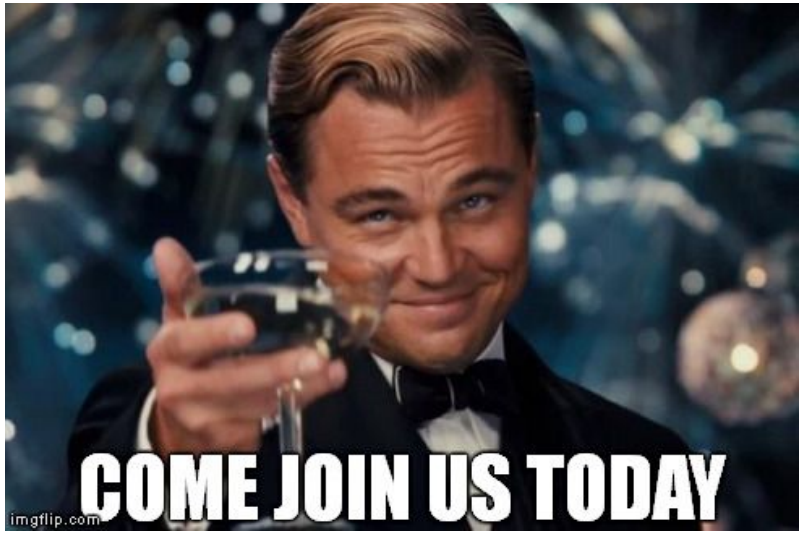
Dissertation Committee:
Professor Richard N. Taylor, Chair
Professor Mark S. Ackerman
Professor David S. Rosenblum

Hola, ¿qué tal?

Ha llegado el momento de conocernos

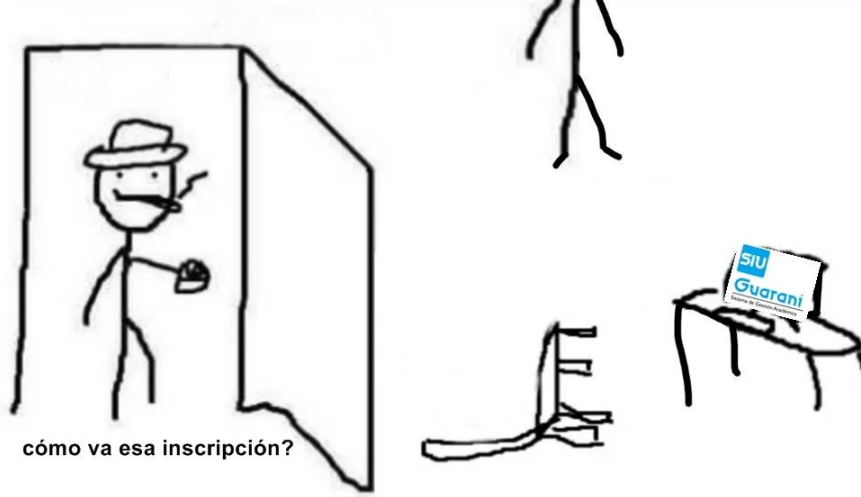
¡Únanse a nuestro Slack! ¡Oh sí!

https://arquitectura-7573.slack.com



Introducción a la Arquitectura de Software

Un hermoso caso de estudio



Para poder anotarse a los cursos brindados en un cuatrimestre, los alumnos de FIUBA utilizan un **web browser** para ingresar al **sitio web** de inscripción a cursos.

Las páginas de dicho sitio son servidas por **3 web servers Apache (3 máquinas físicas)**, a los cuales se accede a través de un **Load Balancer**, instalado en otra máquina.

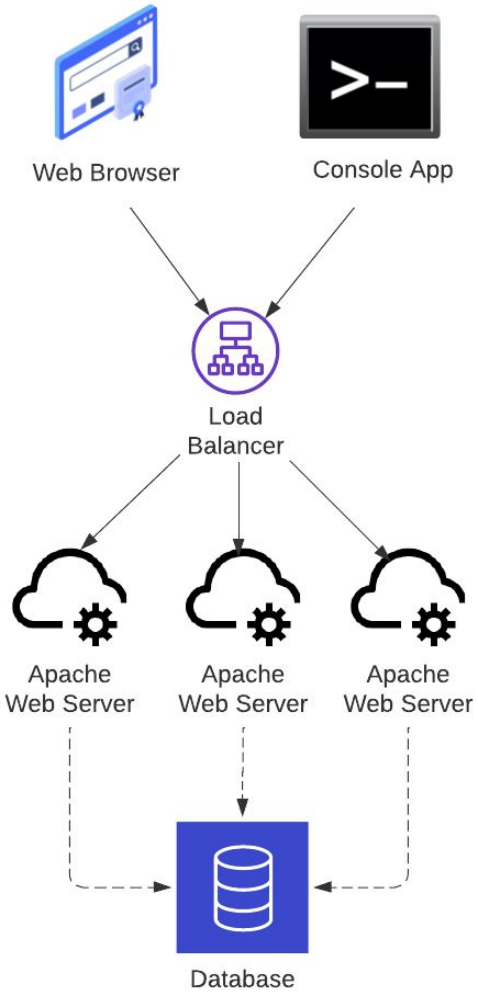
Cada uno de estos web servers acceden directamente a **una base de datos** MySQL para registrar las inscripciones y responder consultas acerca de las mismas.

Además, existe una **aplicación de línea de comandos** que consume algunos de los **servicios expuestos por los web servers**, que es de utilidad a personal administrativo de la facultad.

Arquitectura de Software...

¿Qué demonios es?

¿No es lo mismo que “Diseño”?



¿No es este dibujito la Arquitectura de ese sistema?

Bueno, no, el dibujito no es…

DECIME YA MISMO QUÉ ES, TE LO PIDO POR FAVOR

Ok, veamos algunas definiciones

Martin Fowler

2015 @ OSCON

Paper “Who needs an Architect?”
https://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf

Contrasta 2 definiciones clásicas

1. La descomposición de más alto nivel de un sistema en sus partes fundamentales
2. Conjunto de decisiones de diseño que son difíciles de cambiar

“Al fin de cuentas, Arquitectura se reduce a lo importante - lo que sea que fuera”

Who Needs an Architect?

Martin Fowler

Wandering down our corridor a while ago, I saw my colleague Dave Rice in a particularly grumpy mood. My brief question caused a violent statement, “We shouldn’t interview anyone who has ‘architect’ on his resume.” At first blush, this was an odd turn of phrase, because we usually introduce Dave as one of our leading architects.

The reason for his title schizophrenia is the fact that, even by our industry’s standards, “architect” and “architecture” are terribly overloaded words. For many, the term “software architect” fits perfectly with the smug controlling image at the end of *Matrix Reloaded*. Yet even in firms that have the greatest contempt for that image, there’s a vital role for the technical leadership that an architect such as Dave plays.

What is architecture?

chitect.) However, as so often occurs, inside the blighted cynicism is a pinch of truth. Understanding came to me after reading a posting from Ralph Johnson on the Extreme Programming mailing list. It’s so good I’ll quote it all.

A previous posting said

> The RUP, working off the IEEE definition, defines architecture as “the highest level concept of a system in its environment. The architecture of a software system (at a given point in time) is its organization or structure of significant components interacting through interfaces, those components being composed of successively smaller components and interfaces.”

Johnson responded:

> I was a reviewer on the IEEE standard that said that, and I argued uselessly that this was clearly a completely bogus definition. There is no highest level concept of a system. Customers have a different concept than developers. Customers do

Leer para la clase que viene

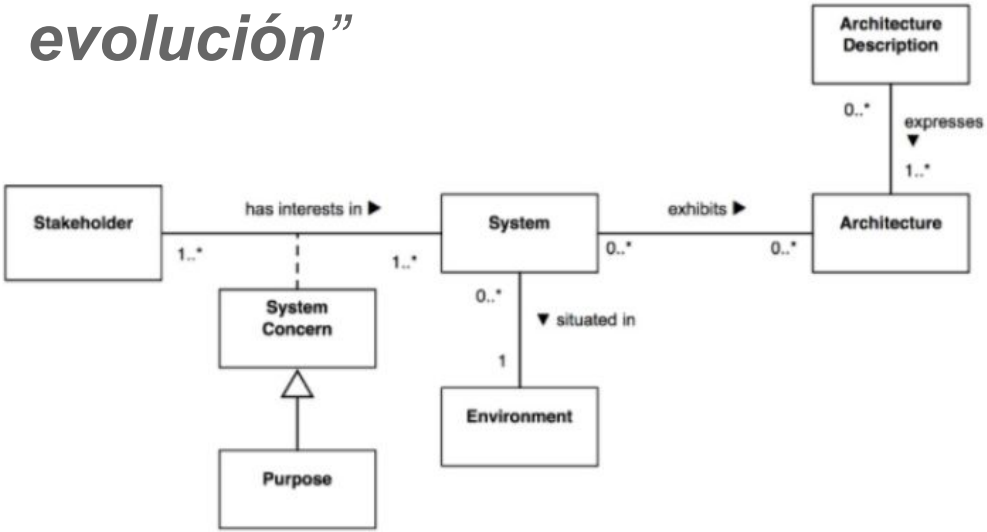
ISO/IEC/IEEE 42010

2011, actualizado 2013

Reemplaza anteriores definiciones de la IEEE e ISO

http://www.iso-architecture.org

*"**Conceptos** o **propiedades** fundamentales de un sistema en su **entorno**, comprendido por sus **elementos**, **relaciones** y **principios** de su **diseño** y **evolución**"*



Roy Fielding

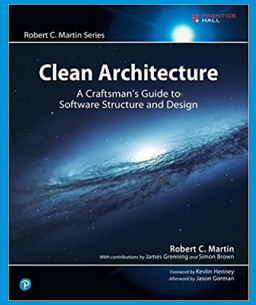
"Architectural Styles and the Design of Network-based Software Architectures", dissertation, 2000

"**Abstracción** de los elementos de tiempo de ejecución (**run-time elements**) de un sistema de software durante alguna fase de su operación.

Un sistema puede tener muchos **niveles de abstracción** y muchas **fases de operación**"

Tipos de elementos: componentes, conectores, y datos.

“Uncle Bob” Robert Martin



Clean Architecture, 2017

*“Es la **forma** dada al sistema por quienes lo construyeron.*

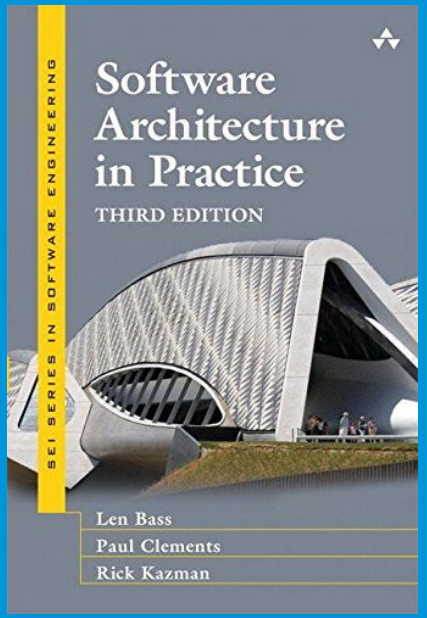
*Esa forma está determinada por la **división en componentes**, la **disposición** de los componentes, y las maneras en que pueden **comunicarse** entre ellos.*

El propósito de esa forma es facilitar el desarrollo, despliegue, operación, y mantenimiento del sistema.

***La estrategia es dejar tantas opciones abiertas como sea posible, por el máximo tiempo posible**.”*

Software Engineering Institute

Carnegie Mellon University



"Software Architecture in Practice", 3rd Edition, Len Bass, Paul Clements y Rick Kazman. 2013.

*"Conjunto de **estructuras** necesarias para razonar sobre el sistema, que constan de **elementos** de software, **relaciones** entre ellos, y **propiedades** de ambos."*

Estructura: conjunto de elementos unidos por una relación

Elementos: módulos (estáticos) y componentes (dinámicos)

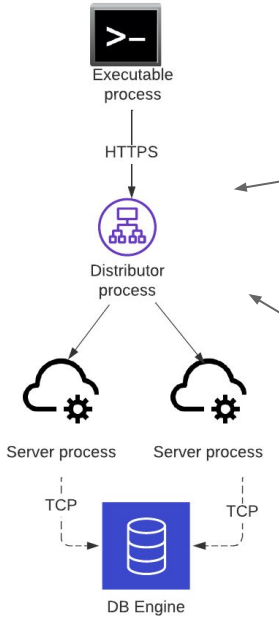
Elegiremos esta definición para nuestro curso

Este… ¿¿qué??

*"Conjunto de **estructuras** necesarias **para razonar sobre el sistema**, que constan de elementos de software, relaciones entre ellos, y propiedades de ambos."*

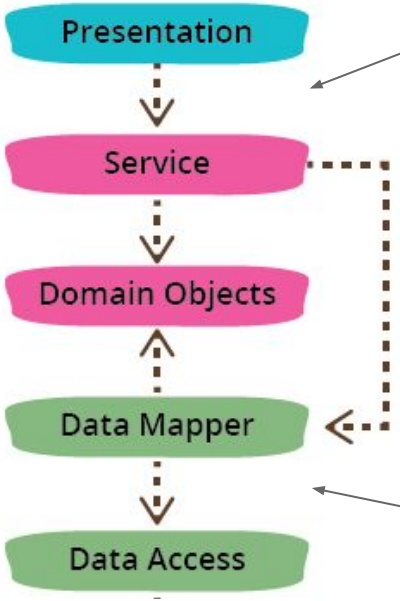
Estructura: conjunto de elementos unidos por una relación.

Elementos: módulos (estáticos) y componentes (dinámicos)



Representación visual de un grupo de **procesos** relacionados

Representa una **estructura** de **componentes**

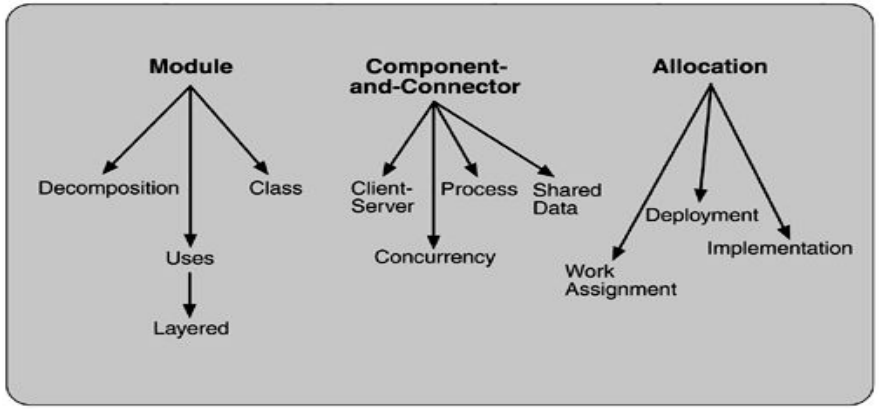


Representación visual de un grupo de módulos relacionados

Representa una **estructura** de **módulos**

Profundizando en la definición

Tipos de Estructuras

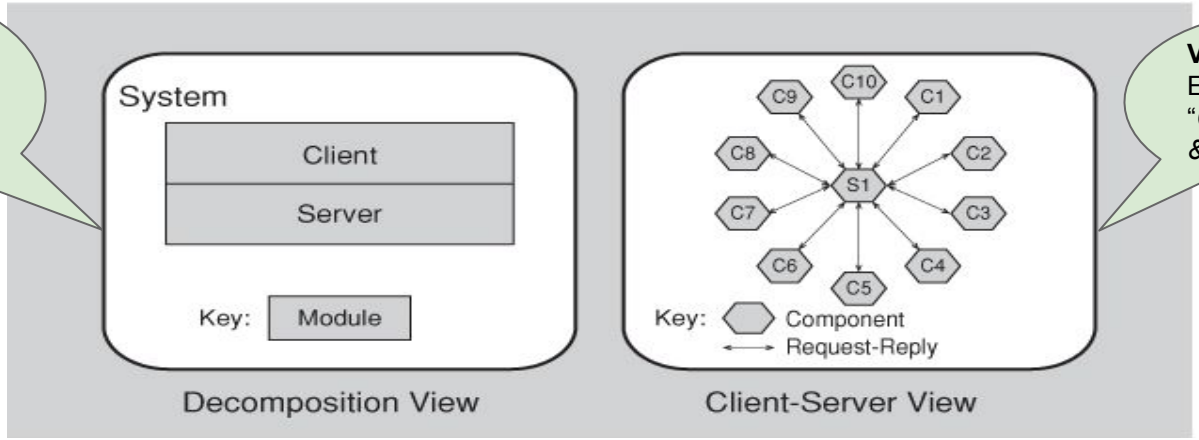


1. **Module**
   a. Estáticas. **Unidades de implementación** a las que se les asigna responsabilidades
   b. Razonar sobre **modificabilidad**
   c. Ej.: Patrón Layers
2. **C&C**
   a. Dinámicas. Elementos en runtime
   b. **Performance, seguridad, disponibilidad**, etc
   c. Ej.: Client-Server
3. **Allocation**
   a. Mapeo entre software y su entorno (equipos, deployment, source code repos)

Vistas de la Arquitectura

Vista: representación de una Estructura

Vista de la Estructura "*Module Decomposition*"



Vista de la Estructura "*Components & Connectors*"

*"Diseñamos **estructuras**, y documentamos **vistas** de esas estructuras"*

La arquitectura es una abstracción

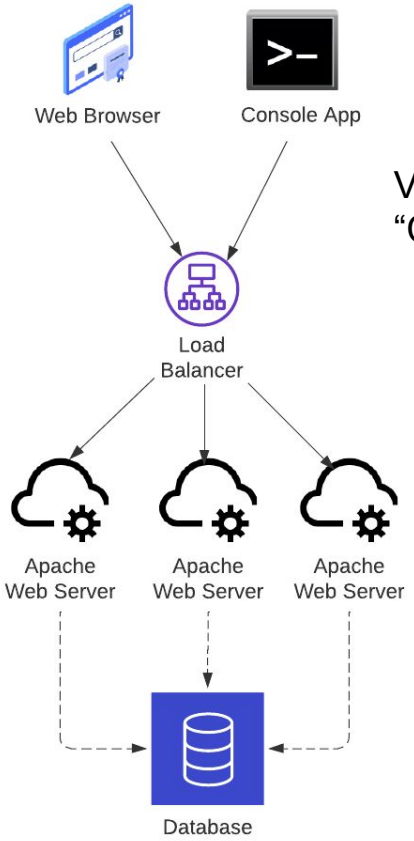
Volviendo al caso de estudio…

Para poder anotarse a los cursos brindados en un cuatrimestre, los alumnos de FIUBA utilizan un **web browser** para ingresar al **sitio web** de inscripción a cursos.
Las páginas de dicho sitio son servidas por **3 web servers Apache (3 máquinas físicas)**, a los cuales se accede a través de un **Load Balancer**, instalado en otra máquina.
Cada uno de estos web servers acceden directamente a **una base de datos** MySQL para registrar las inscripciones y responder consultas acerca de las mismas.
Además, existe una **aplicación de línea de comandos** que consume algunos de los **servicios expuestos por los web servers**, que es de utilidad a personal administrativo de la facultad.

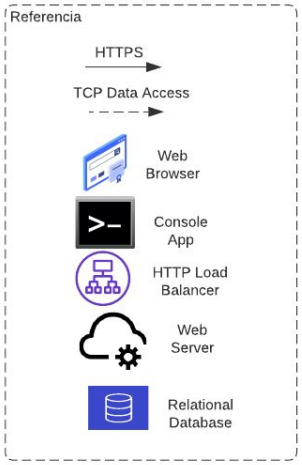
a) Dibujar una vista C&C de la arquitectura del sistema. Identificar cuáles son los **componentes**, cuáles los **conectores**, y qué **datos** están involucrados.

b) Sabiendo que los arquitectos del sistema dividieron el mismo en **3 capas lógicas** llamadas “Presentación” (UI), “Lógica de Negocio” (Business Logic) y “Acceso a Datos” (Data Access), generar una vista de módulos de la arquitectura del sistema.

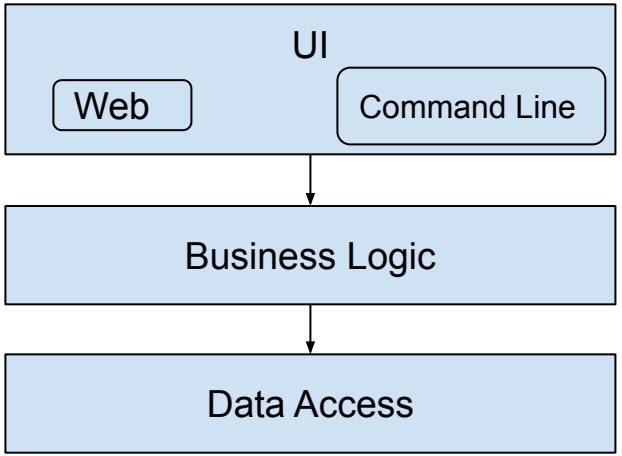
Vistas

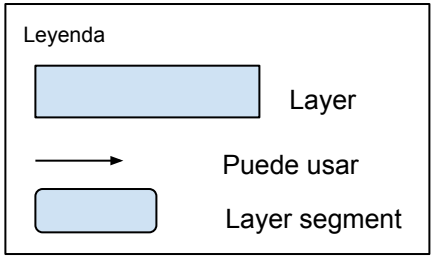


Vista de la Estructura
“Components & Connectors”



Vista de la Estructura “Layers”





Hermosos conceptos.

¿Tienen relación alguna con el mundo real?

Arquitectura y Negocio

Arquitectura de software de un sistema

Conjunto de

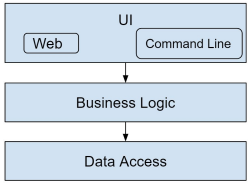
Estructuras

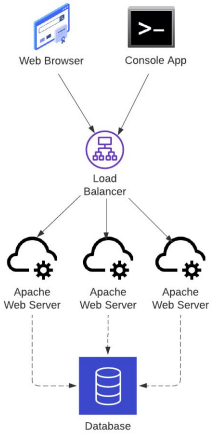
Conjunto de

Elementos arquitecturales

Estáticos: Módulos
Dinámicos:
- Componentes
- Conectores
- Datos

Estáticas: módulos
Dinámicas: C&C, Allocation





Se representan a través de

Vistas Arquitecturales

?

RF1. Inscripción online a cursos
RF2. Admin online de cursos y materias
...
RQA1. Responder en menos de 2s
RQA2. Disponible "7x24"
...

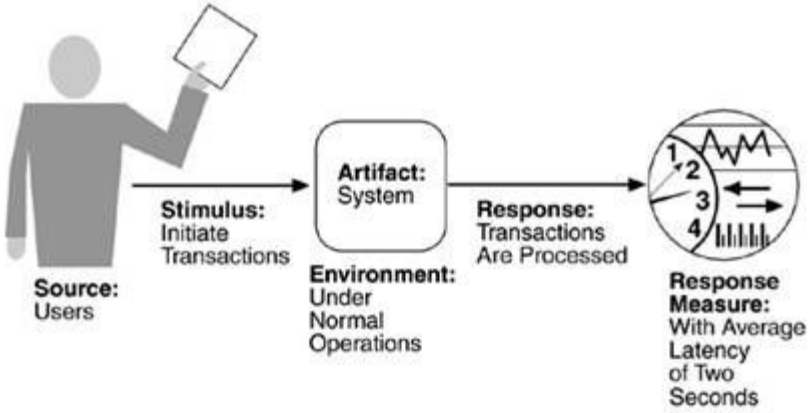
Requerimientos del sistema

- Funcionales
- Atributos de Calidad
- Restricciones

Objetivos del negocio

"Permitir realizar el 100% de los trámites administrativos de forma online"

Atributos de Calidad y Tácticas



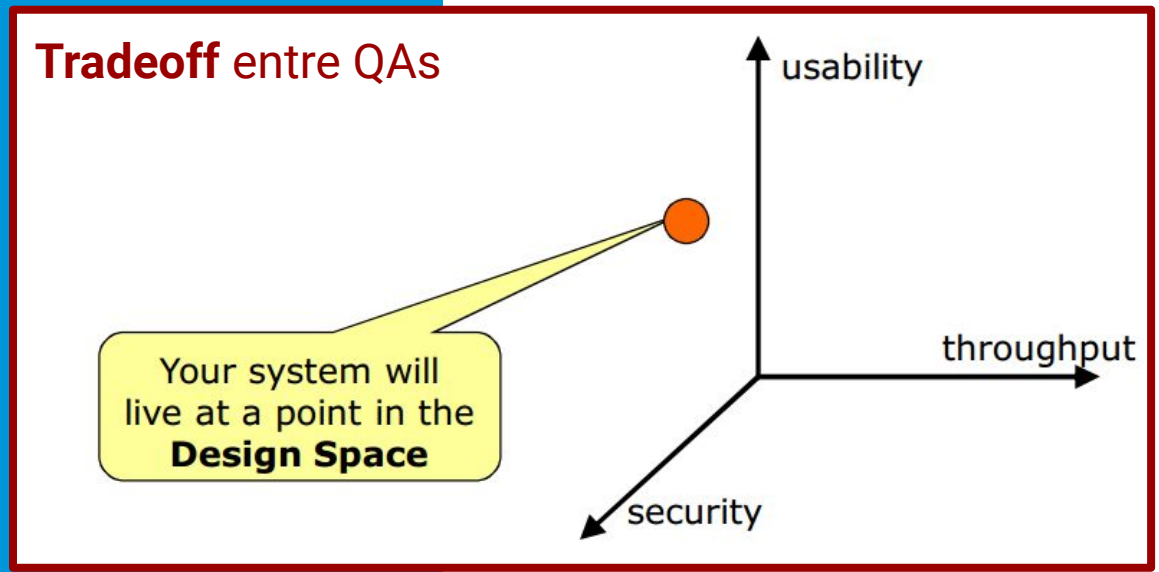
Atributos de Calidad

QAs - Quality Attributes

Propiedades de Arquitectura
- Funcionales
- “No funcionales” aka “Extra-funcionales”

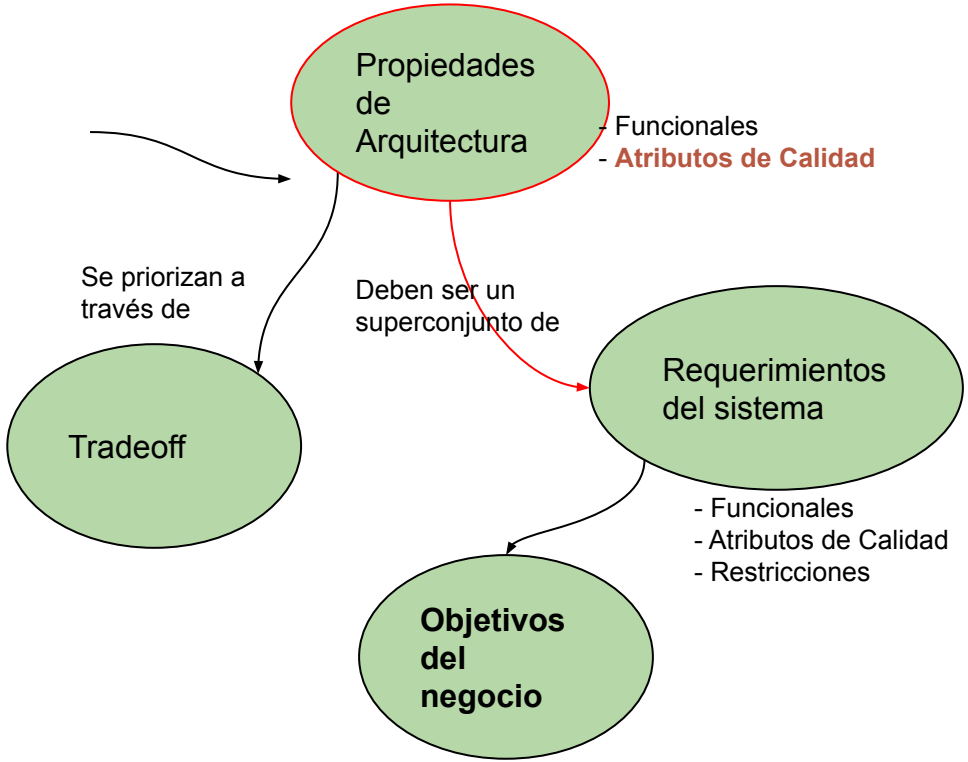
Propiedad *mensurable o testeable del sistema*

- Dimensión de una **cualidad**
- Ayudan a evaluar en qué medida el sistema satisface las necesidades de los stakeholders



Atributos de Calidad y Requerimientos

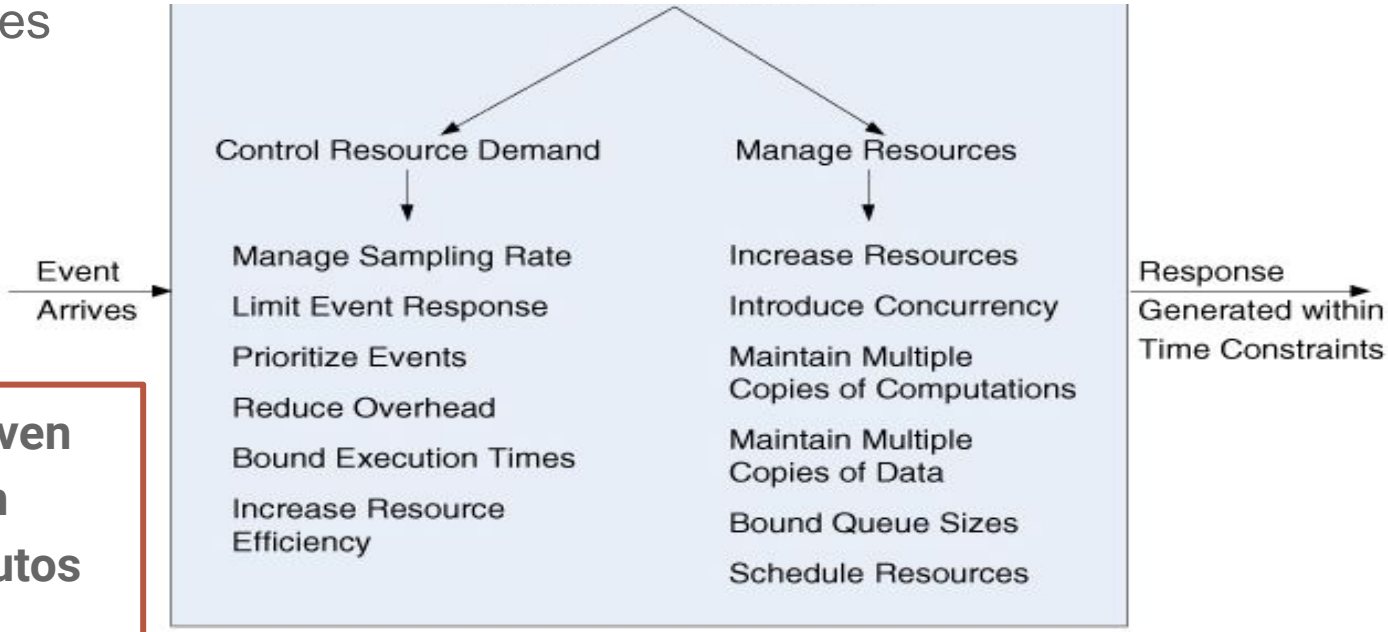
Ok, bueno…
¿Pero cómo actúo sobre los Atributos de Calidad?



Tácticas

aka Decisiones de Diseño, Restricciones

Ejemplos relacionados con el QA “Performance”

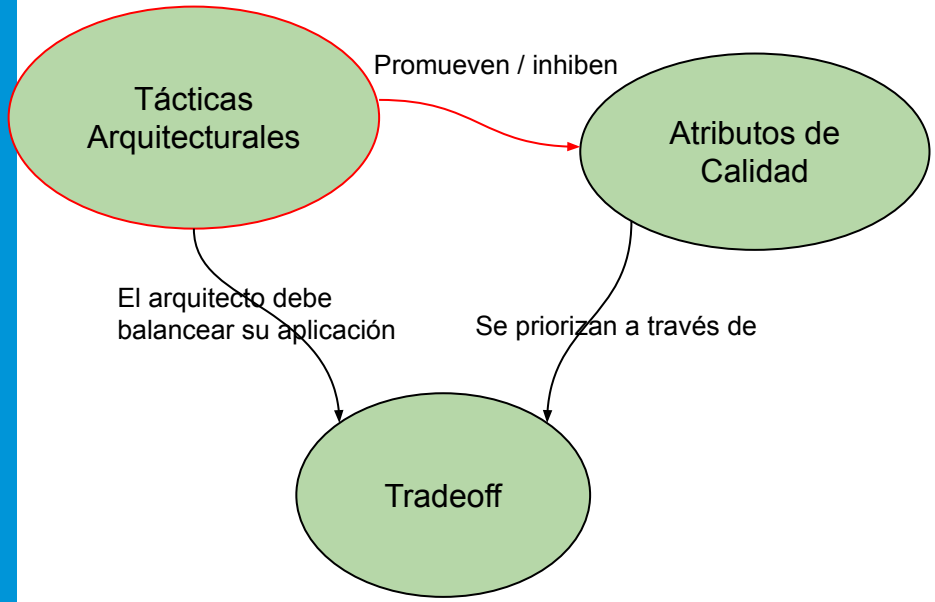


Las Tácticas promueven (favorecen) ó inhiben (desfavorecen) atributos de calidad

Tácticas

Decisiones de diseño

Restricciones



Una categorización posible

- Allocation of responsibilities
- Coordination model
- Data model
- Management of resources
- Mapping among architectural elements
- Binding time decisions
- Choice of technology

Ajá… ¿y entonces?

La red conceptual ejemplificada

Objetivos del negocio

"Permitir realizar el 100% de los trámites administrativos de forma online"

Motivan

Requerimientos del sistema

RF1. Inscripción online a cursos
RF2. Admin online de cursos y materias
...
RQA1. Responder en menos de 2s
RQA2. Permitir 500 inscripciones por minuto
RQA3. Disponible días hábiles 8 a 20hs
RQA4. Facilitar la inscripción a través de nuevas apps cliente en el futuro

Se aplican sobre

Propiedades de Arquitectura

- Funcionales
  - UI y API de inscripción
  - UI de admin
- **Atributos de Calidad**
  - **Performance**
  - **Disponibilidad**
  - **Extensibilidad**

Promueven o inhiben

Tácticas

- Permitir concurrencia
- Múltiples web servers
- Redundancia activa de web servers
- Encapsular funcionalidad de inscripción en una API
- Modularizar con Layers

Determinan relaciones, responsabilidades y roles de los

Elementos

Estáticos: Módulos
- **UI Layer**
- **Business Logic Layer**
- **Data Access Layer**
- **Web module**
- **CLI module**

Dinámicos:
- Componentes: web **browsers, Load Balancer, Web Servers, FIUBA DB**
- Conectores: **protocolos HTTPS, TCP**
- Datos: **interfaces en HTML y JSON**

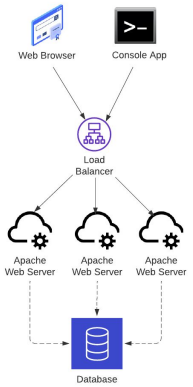
Arquitectura de software de un sistema

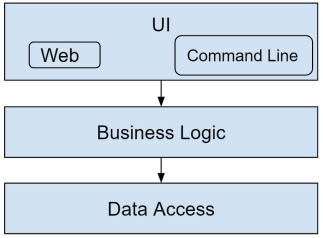
Estructuras

Estáticas: módulos
Dinámicas: C&C, Allocation

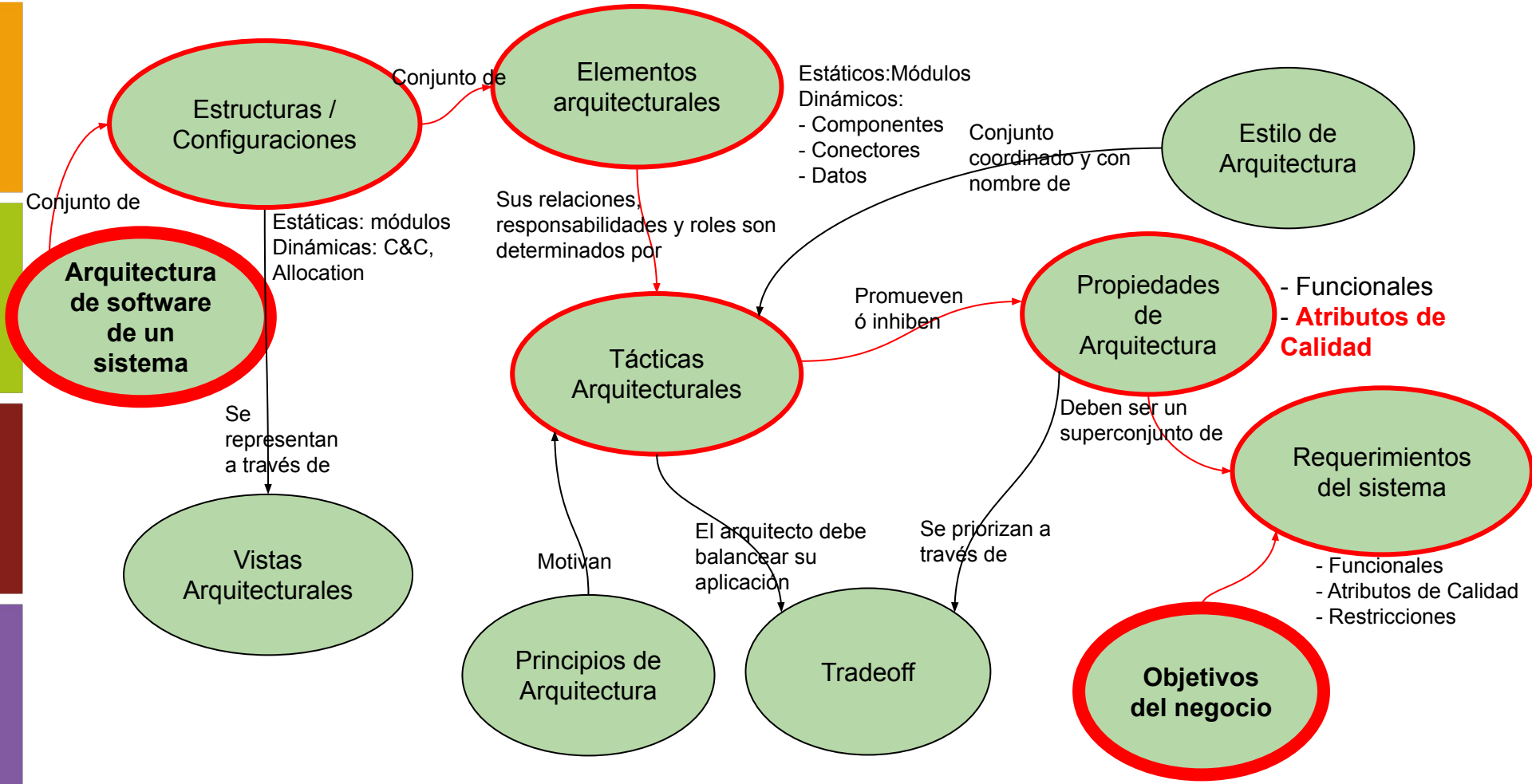
Se representan a través de

Vistas





La red conceptual completa



Una aclaración…

Objetivos de Negocio

Requerimientos

Ejemplos de Objetivos de Negocio

1. *“Ganar más porción del mercado”*
2. *“Mantener una reputación alta”*
3. *“Agregar nuevas funciones fácilmente”*
4. *“Proveer un framework amigable para programadores”*
5. *“Integrarse con otros sistemas fácilmente”*
6. *“Superar a la competencia en términos de velocidad de procesamiento”*

Atributos de Calidad sobre los cuales podría haber requerimientos

1. Modificabilidad, Usabilidad
2. Disponibilidad, Usabilidad, Performance
3. Modificabilidad, Disponibilidad
4. Modificabilidad
5. Interoperabilidad, Portabilidad, Modificabilidad
6. Performance, Escalabilidad

Bueno. ¿Ya nos podemos ir?

Arquitectura, sí. ¿ó Diseño?

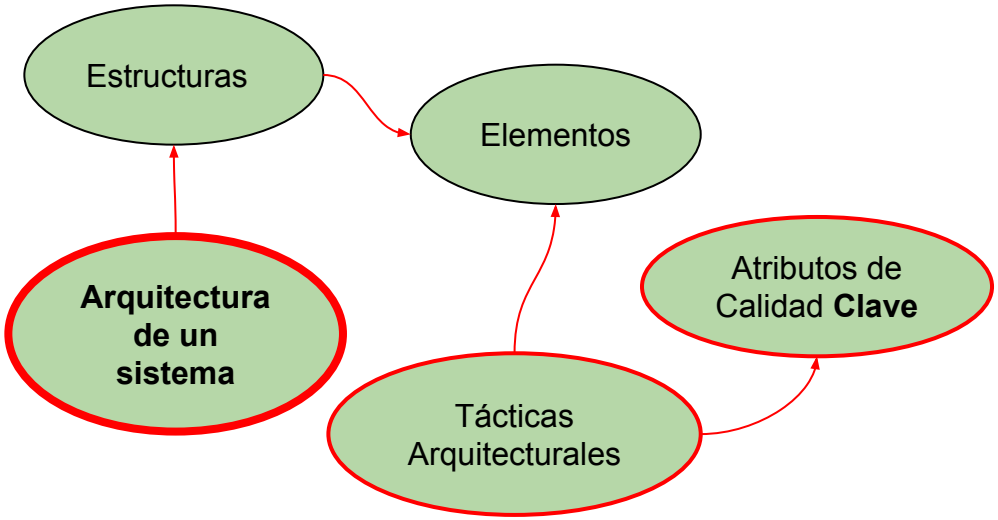
*“El término “arquitectura” suele ser usado en el contexto de **algo a alto nivel que está separado de los detalles de bajo nivel**, en tanto que “diseño” parece usarse para implicar estructuras y decisiones de un nivel más bajo.*

*Pero ambas cosas son parte del mismo todo. Forman un **continuo** que define la forma del sistema. No se puede tener uno sin el otro, no hay una línea que los separe claramente. Es simplemente un continuo de **decisiones** desde el más alto hasta los más bajos niveles.”*

Arquitectura vs. Diseño

Definamos la cuestión

- Una táctica arquitectural es aquella que afecta a al menos 1 Atributo de Calidad Clave (ACC) del sistema
  - Ninguna táctica es inherentemente arquitectural
  - Ej.: Dependiendo si performance es un ACC o no, procesar un input incrementalmente (streaming) puede ser una táctica de arquitectura
- Arquitectura es diseño, pero no todo diseño es arquitectura



¡Ya casi!

Un mini checklist para despertarnos

A ver, a ver… (Verdadero/Falso)

1. La arquitectura de software comienza a existir cuando se escriben los documentos de arquitectura
2. La arquitectura de software nos permite pensar sobre el sistema, entenderlo, formar juicios sobre él, y tomar decisiones sobre su organización
3. La arquitectura de software de un sistema no tiene relación alguna con los objetivos del negocio.
4. La decisión de elegir una base de datos relacional SQL ó una base de datos NoSQL, es siempre una decisión arquitectural.
5. La arquitectura de software de un sistema promueve algunos atributos de calidad y otros no
6. Una estructura de tipo C&C (Components & Connectors) nos sirve para entender cómo interactúan los elementos arquitecturales en runtime.
7. Todos los atributos de calidad son igualmente importantes para todos los sistemas.

¿Consultas?

http://bit.ly/arq-soft-feedback-clases

grugilo@fi.uba.ar

¡Gracias!

www.**ingenieria**.uba.ar

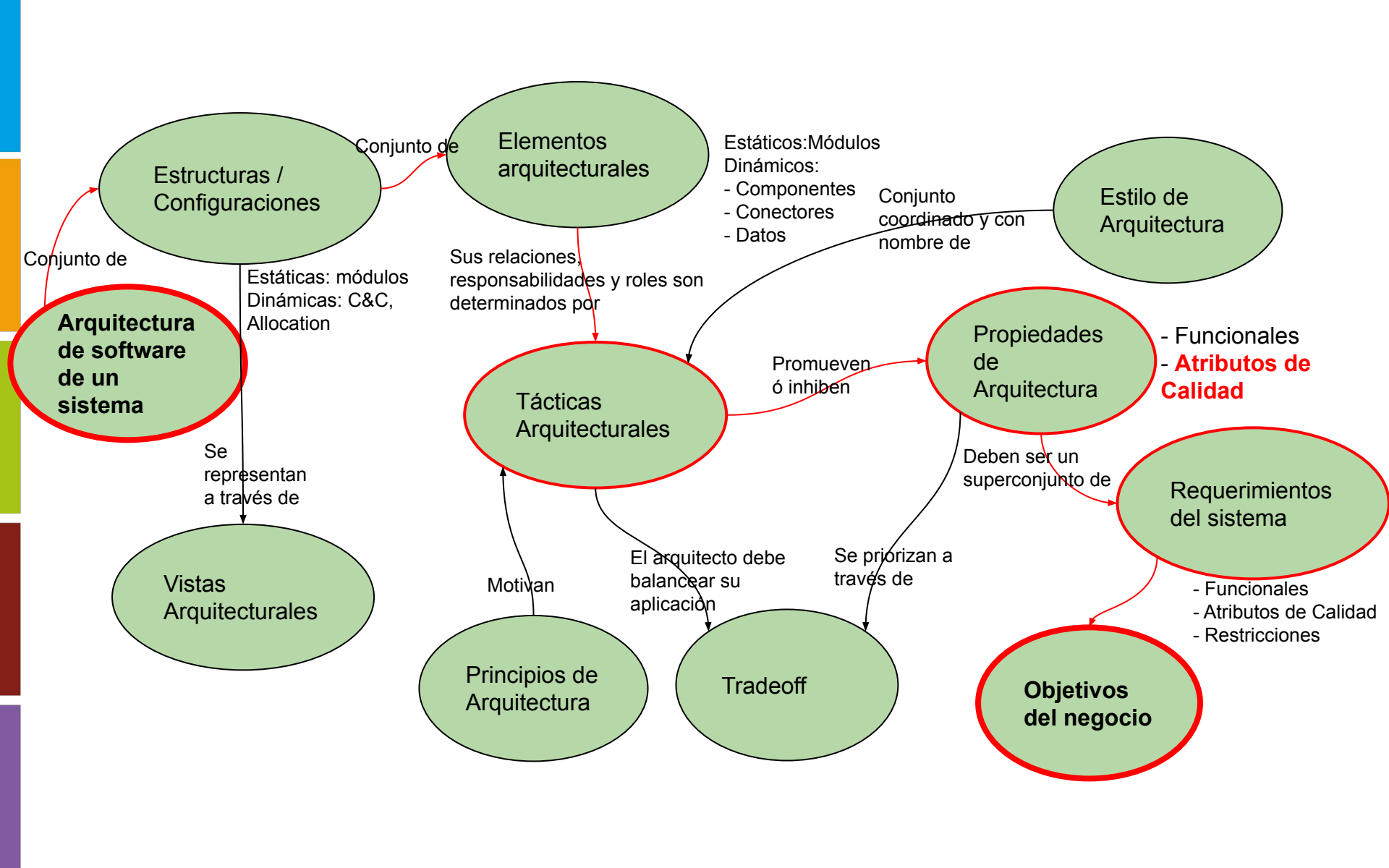
/ingenieriauba

/FIUBAoficial

Una más, y no insistimos más

Los requerimientos funcionales: (marque todas las respuestas correctas)

1) Se pueden desestimar al tomar cualquier decisión de arquitectura.
2) Representan funciones que el sistema debe implementar, y la totalidad de estas funciones deben ser implementadas por los elementos arquitecturales.
3) Deben proveerse en su totalidad previamente a comenzar el diseño de la arquitectura de cualquier sistema.
4) Proveen información necesaria para la asignación de responsabilidades a los elementos arquitecturales.
5) Si son ignorados al tomar decisiones de arquitectura, la probabilidad de que el sistema fracase aumenta.


Estructuras / Configuraciones
Conjunto de
Elementos arquitecturales
Estáticos:Módulos
Dinámicos:
- Componentes
- Conectores
- Datos
Conjunto coordinado y con nombre de
Estilo de Arquitectura
Conjunto de
Arquitectura de software de un sistema
Estáticas: módulos
Dinámicas: C&C, Allocation
Sus relaciones, responsabilidades y roles son determinados por
Tácticas Arquitecturales
Promueven ó inhiben
Propiedades de Arquitectura
- Funcionales
- Atributos de Calidad
Se representan a través de
Vistas Arquitecturales
Motivan
El arquitecto debe balancear su aplicación
Se priorizan a través de
Deben ser un superconjunto de
Requerimientos del sistema
- Funcionales
- Atributos de Calidad
- Restricciones
Principios de Arquitectura
Tradeoff
Objetivos del negocio