

# ARQUITECTURA DEL SOFTWARE

2025-26

Jose E. Labra

Pablo González

Celia Melendi

Diego Martín

## Laboratorio 2

Documentación  
Diagramas UML  
PlantUML  
Introducción a Arc42



Escuela de  
Ingeniería  
Informática



Universidad de Oviedo

# Arquitectura es más que código

El código no cuenta la historia completa

Preguntas que el código no responde

¿Cómo encaja el software en el entorno existente?

¿Porqué se han elegido ciertas tecnologías?

¿Cuál es la estructura general del sistema?

¿Dónde están desplegados los componentes que se ejecutan?

¿Cómo se comunican los componentes?

¿Cómo y dónde se puede añadir nueva funcionalidad?

¿Qué patrones comunes o principios se utilizan?

¿Cómo funcionan las interfaces con otros sistemas?

¿Cómo se alcanza la seguridad/escalabilidad/... ?

...

Nota:  
Este contenido se ha dado  
en clases de teoría también

# Objetivo de la documentación

Objetivo principal: Comunicar la estructura

Comprender la visión general del sistema

Crear una visión compartida: equipo y otras personas interesadas

Vocabulario común

**Describir** qué software se está construyendo y cómo

Facilitar conversaciones técnicas sobre características

Proporcionar mapa para navegar el código fuente

**Justificar** decisiones de diseño

**Ayudar** a desarrolladores nuevos que se unen al equipo

Nota:  
Este contenido se ha dado  
en clases de teoría también

# Requisitos de documentación

Comprendible por las diferentes personas interesadas

*Stakeholders* técnicos y no-técnicos

Reflejar la realidad

Cuidado con separación modelo-código (model-code gap)

Adaptarse a cambios

Adaptarse a proyectos ágiles

Arquitectura evolutiva

Nota:  
Este contenido se ha dado  
en clases de teoría también

# Reglas para buena documentación

Escribir desde el punto de vista del lector

- Encontrar quienes serán los lectores y sus expectativas

Evitar repeticiones innecesarias (principio DRY)

Evitar ambigüedad

- Explicar la notación (o utilizar notaciones estándar)

- Utilizar leyendas o claves para diagramas

Utilizar una organización estándar o plantilla

- Añadir Pendiente (TBD/To do) cuando sea necesario

- Organizar para referencias/enlaces rápidos

Registrar las justificaciones de las decisiones

Mantener la documentación actual

Nota:  
Este contenido se ha dado  
en clases de teoría también

# Espacio de Problema vs Solución

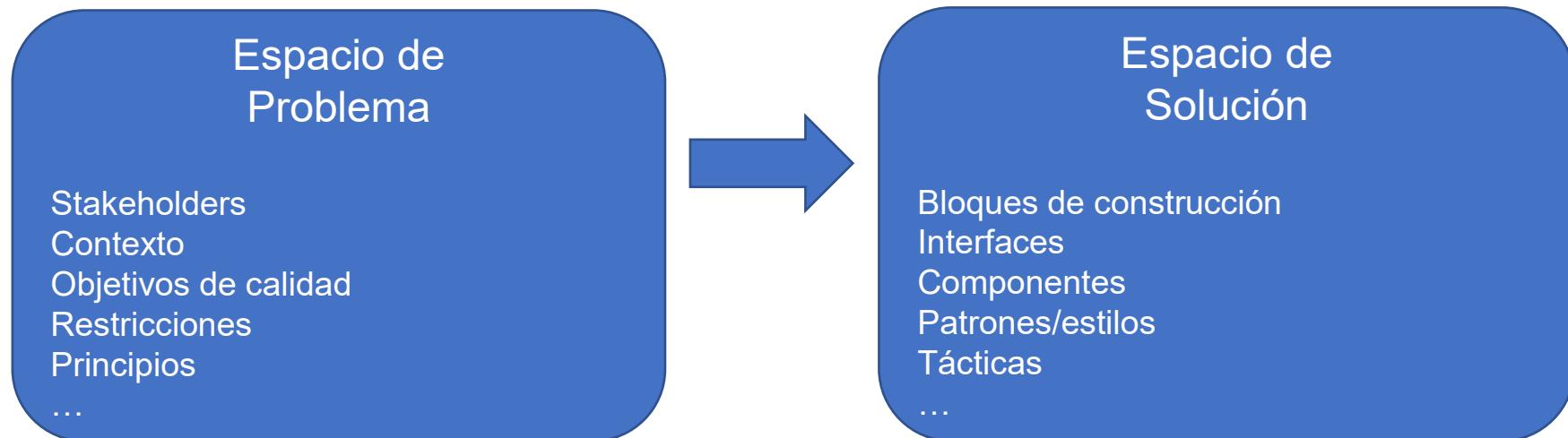
Arquitectura software = camino de problema a solución

Comprender el problema

Diseñar una solución

Justificar las soluciones propuestas

Registros diferentes alternativas de diseño



Nota:  
Este contenido se ha dado  
en clases de teoría también

# UML

## Unified Modeling Language

Antes de UML había varias propuestas

Notación UML los unifica

Propuesta por OMG (Object Management Group)

Versión actual: UML 2.5.1 (2017)

## Modelo = abstracción de un problema

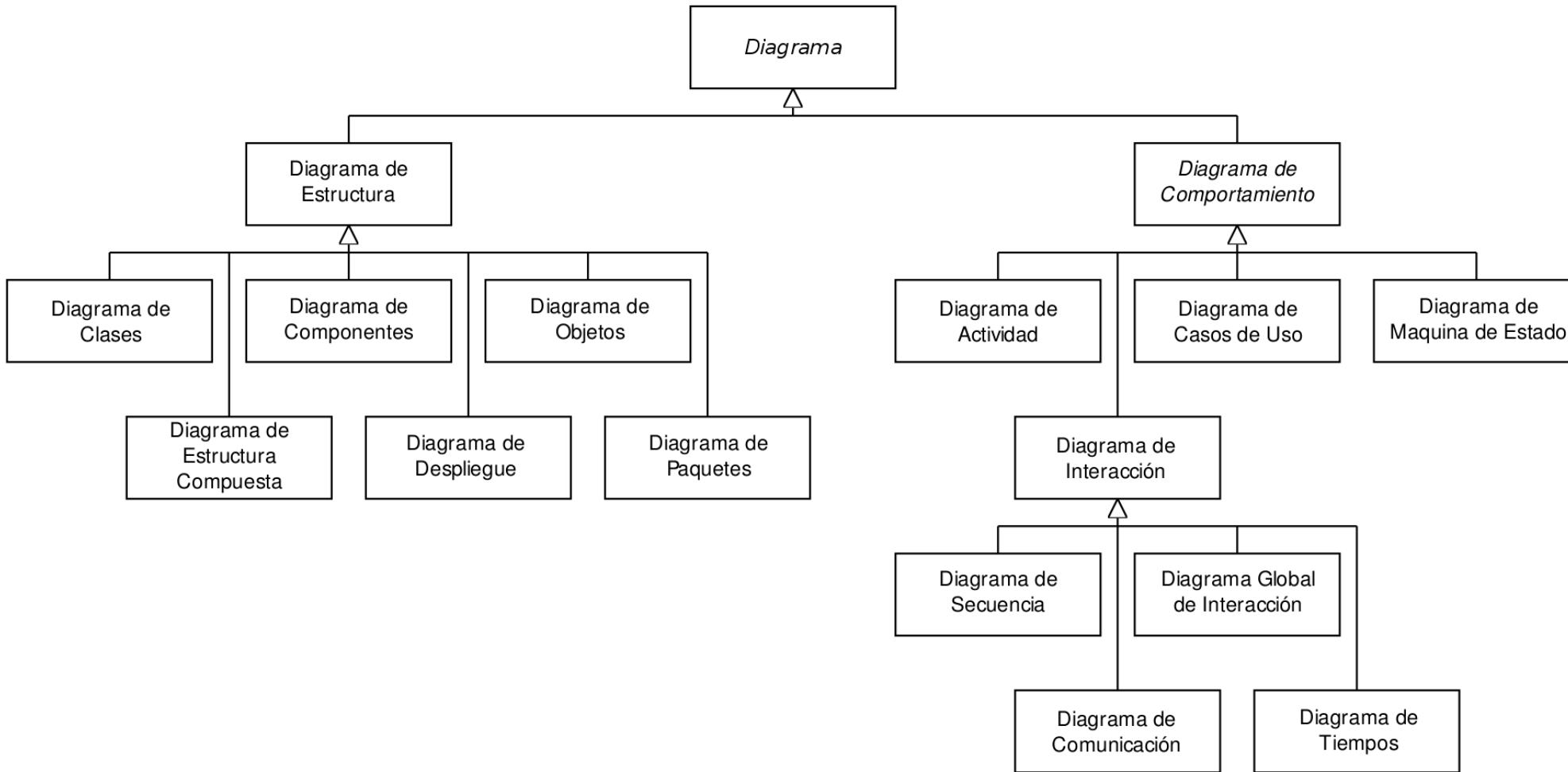
Puede tener varios diagramas diferentes

Diagrama = representación gráfica parcial de un modelo

## OCL = Object Constraint Language

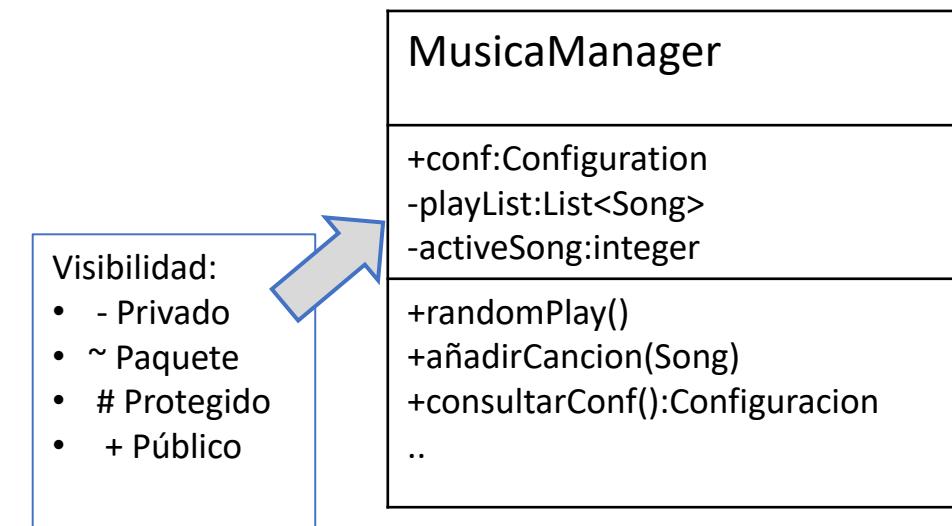
Restricciones entre objetos usando lenguaje formal

# 14 tipos de diagramas UML

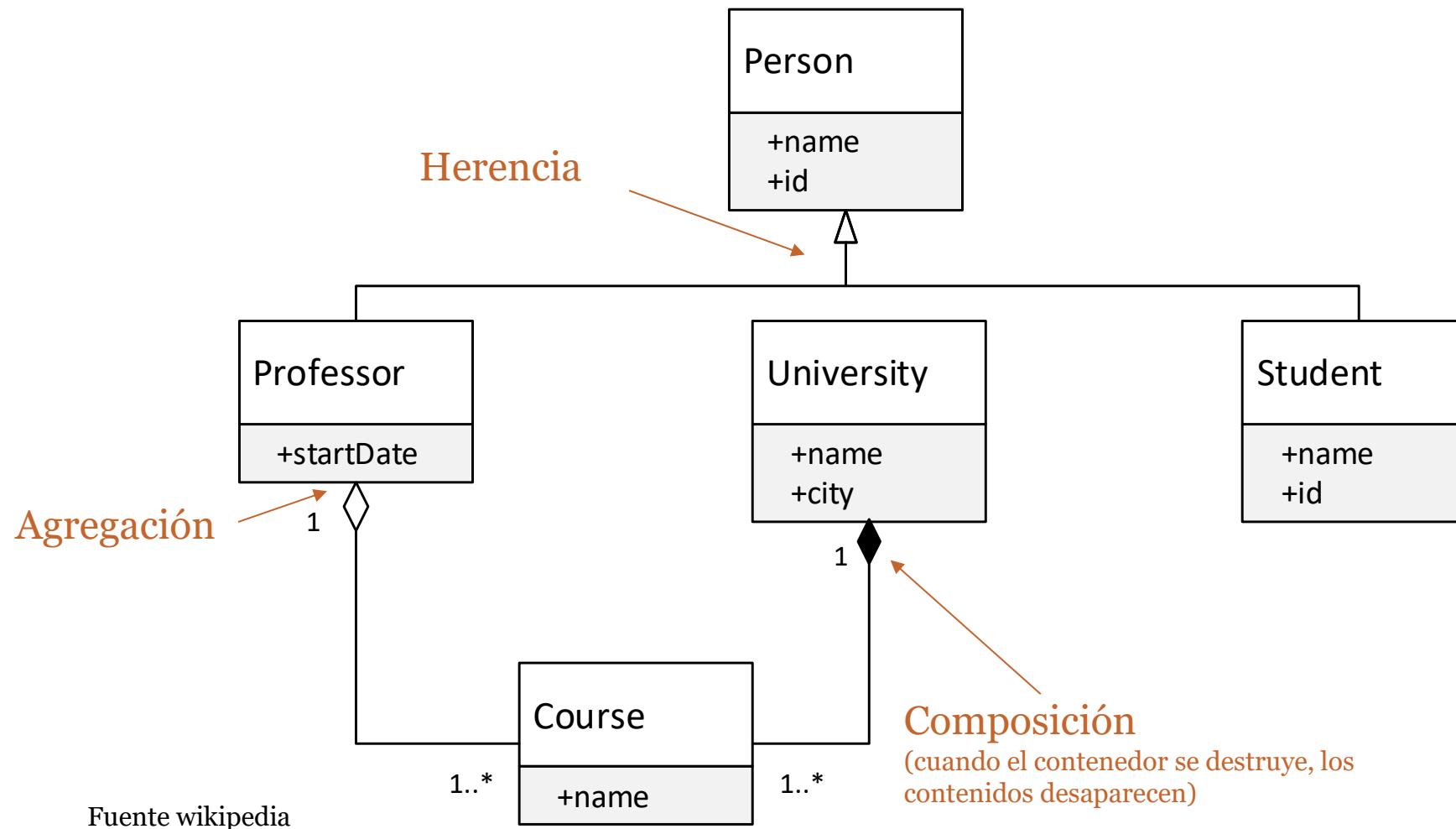


# Diagramas de clase

- Modela la parte estática del proyecto, sin tener en cuenta la situación del sistema en un tiempo.
- Explica las relaciones que hay entre las distintas clases.
- Arc42: 8-Concepts



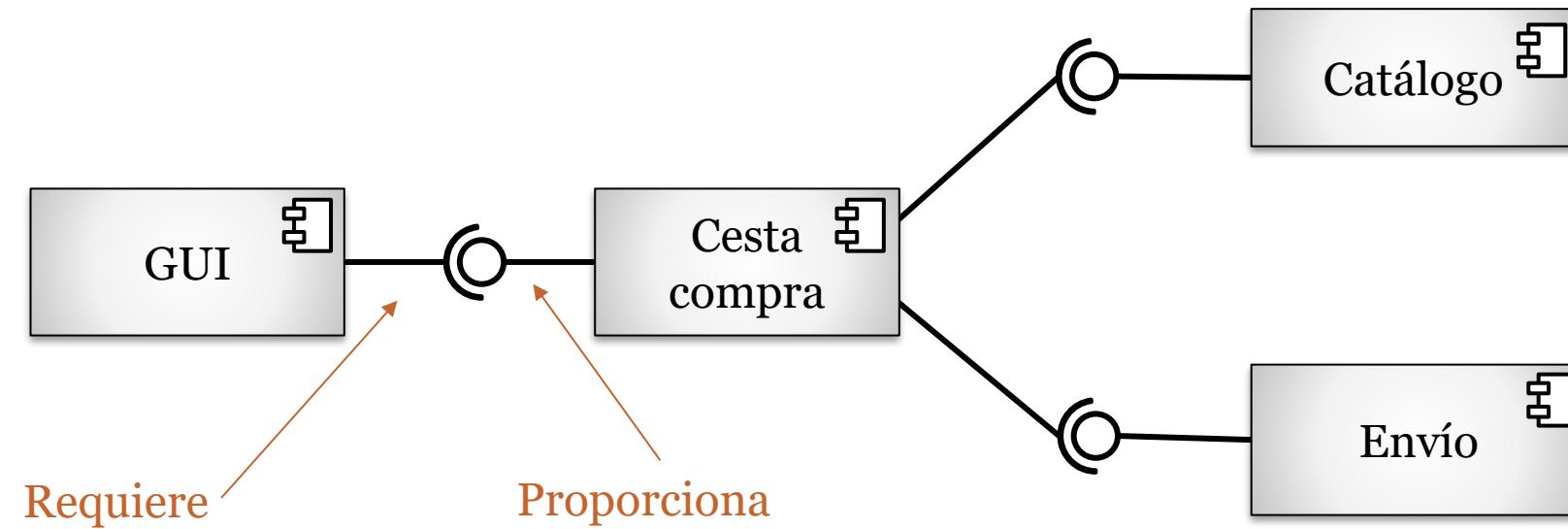
# Ejemplo



**Composición**  
(cuando el contenedor se destruye, los contenidos desaparecen)

# Diagrama de componentes

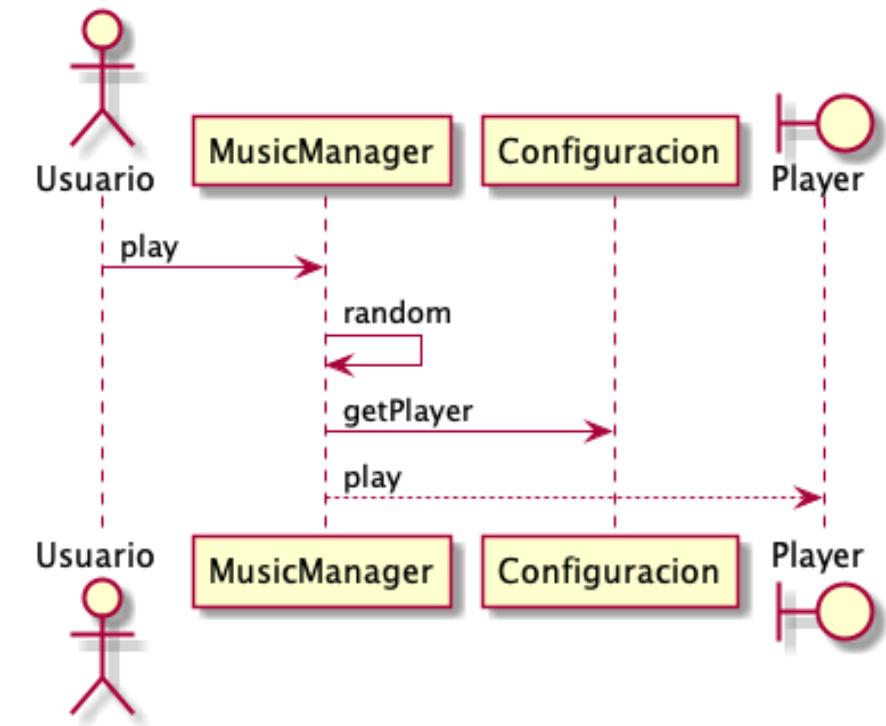
Representa relación estructural de componentes del sistema de software  
Sistemas complejos que tienen muchos componentes  
Interfaz suele representarse mediante notación *lollipop*



# Diagrama de secuencia

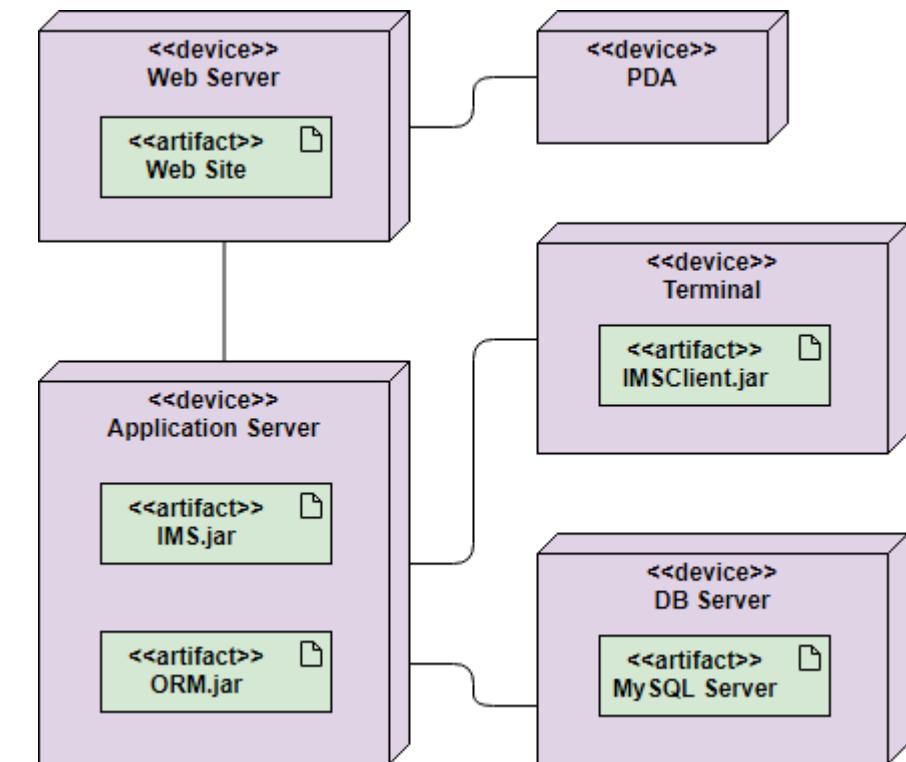
Modela comunicación entre los objetos de sistema en un determinado momento  
Los objetos pueden enviarse dos tipos de mensajes: síncronos y asíncronos

Arc42: 6 - RuntimeView



# Diagrama de despliegue

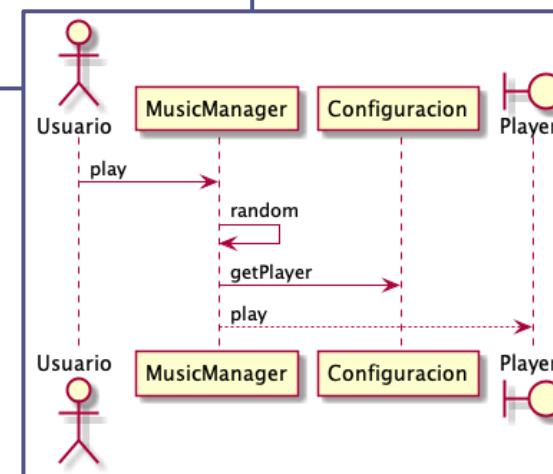
- Representa la localización final de los componentes de la aplicación
- Elementos : Nodos , Componentes, relaciones
- Arc42: 07.DeploymentView



# Herramientas Textuales

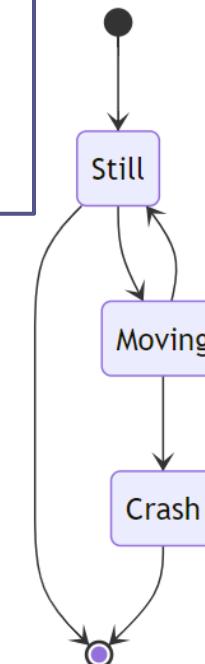
## PlantUML

```
@startuml component  
actor Usuario  
participant MusicManager  
participant Configuration  
boundary Player  
Usuario -> MusicManager: play  
MusicManager -> MusicManager: random  
MusicManager -> Configuration : getPlayer  
MusicManager --> Player : play  
@enduml
```



## Mermaid

```
stateDiagram-v2  
[*] --> Still  
Still --> [*]  
Still --> Moving  
Moving --> Still  
Moving --> Crash  
Crash --> [*]
```



# Herramientas de dibujo

PowerPoint

Visio (Microsoft)

UMLet (<https://www.umlet.com/>)

# Herramientas CASE

## Enterprise Architect

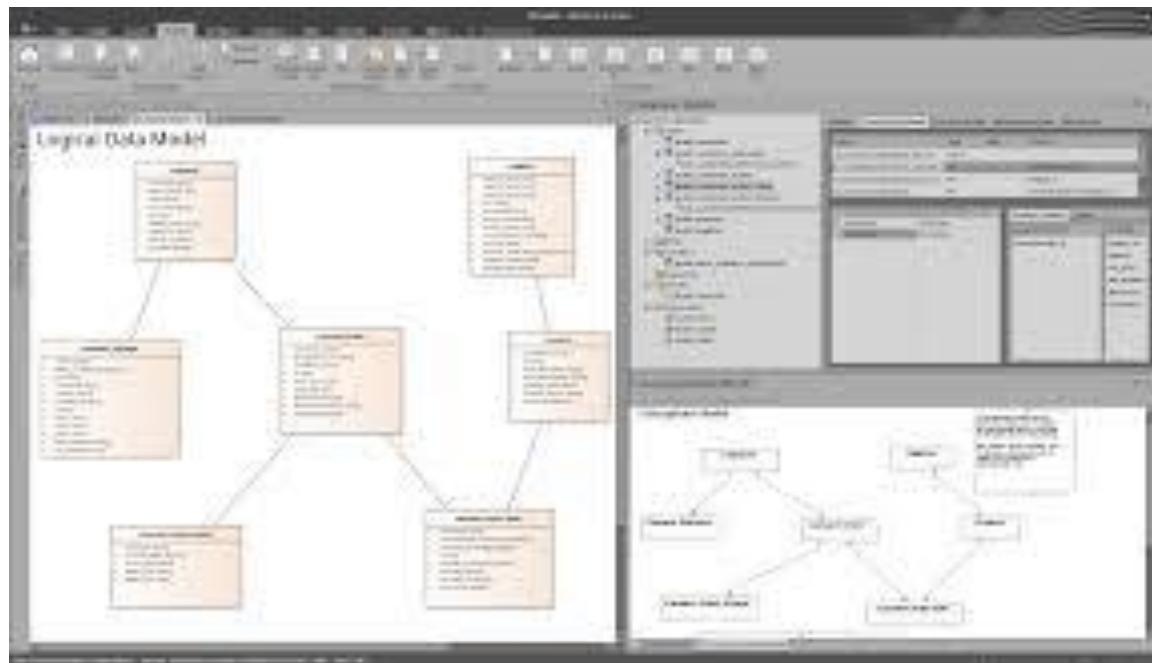
Solo para Windows

Entiende todo tipo de diseño

Ingeniería Inversa con Java/C++

Conecta con Oracle modelos datos  
relacionales

Plantillas editables para Word, HTML



## MagicDraw

Para todo sistema con Java

Diagramas UML

Ingeniería Inversa Java , C++

## Visual Paradigm

Comercial (Licencia estudiantes)

## Modelio

Código abierto

Java based

Ingeniería Inversa

# Dibujando la arquitectura

Vídeo con pautas para diagramas

<https://www.youtube.com/watch?v=wgpSdpny-oc>

Checklist utilizado en C4

<https://c4model.com/assets/software-architecture-diagram-review-checklist.pdf>

# Plantillas arc42



Arc42: <https://arc42.org/>

wiq\_XXX ya sigue la plantilla:

[https://arquisoft.github.io/wiq\\_o/](https://arquisoft.github.io/wiq_o/)

Generación de documentación (en local):

```
$ cd docs  
$ npm install          (sólo la primera vez)  
$ npm run build
```

# GitHub Pages

- GitHub permite crear sitios web
- Útil para información personal
- Despliega lo que se encuentra en la rama de repositorio **gh-pages**

# GitHub Pages - ejemplos

- Nivel Organizativo
  - Repositorio:
    - <https://github.com/Arquisoft/Arquisoft.github.io>
  - Desplegado:
    - <https://arquisoft.github.io/>
- Muy útil para tener páginas personales
  - <http://pglez82.github.io>

# Despliegue de la documentación

Utilizaremos GitHub Pages para desplegar la documentación

GitHub Pages permite a usuarios publicar un sitio web sencillo en GitHub.

El sitio web de la documentación se enviará a la rama **gh-pages**.

Archivos asciidoc se enviarán a rama `develop` del repositorio (Manualmente)

Paquete npm de **gh-pages** envía documentación generada a `gh-pages`

Todo esto está automatizado ejecutando el siguiente comando:

```
$ npm run deploy
```



