EN
English

Universidad de Oviedo

School of Computer Science

SOFTWARE ARCHITECTURE

Software Architecture

Lab. 02

UML diagrams

PlantUML

Introduction to arc42 docs

2022-23

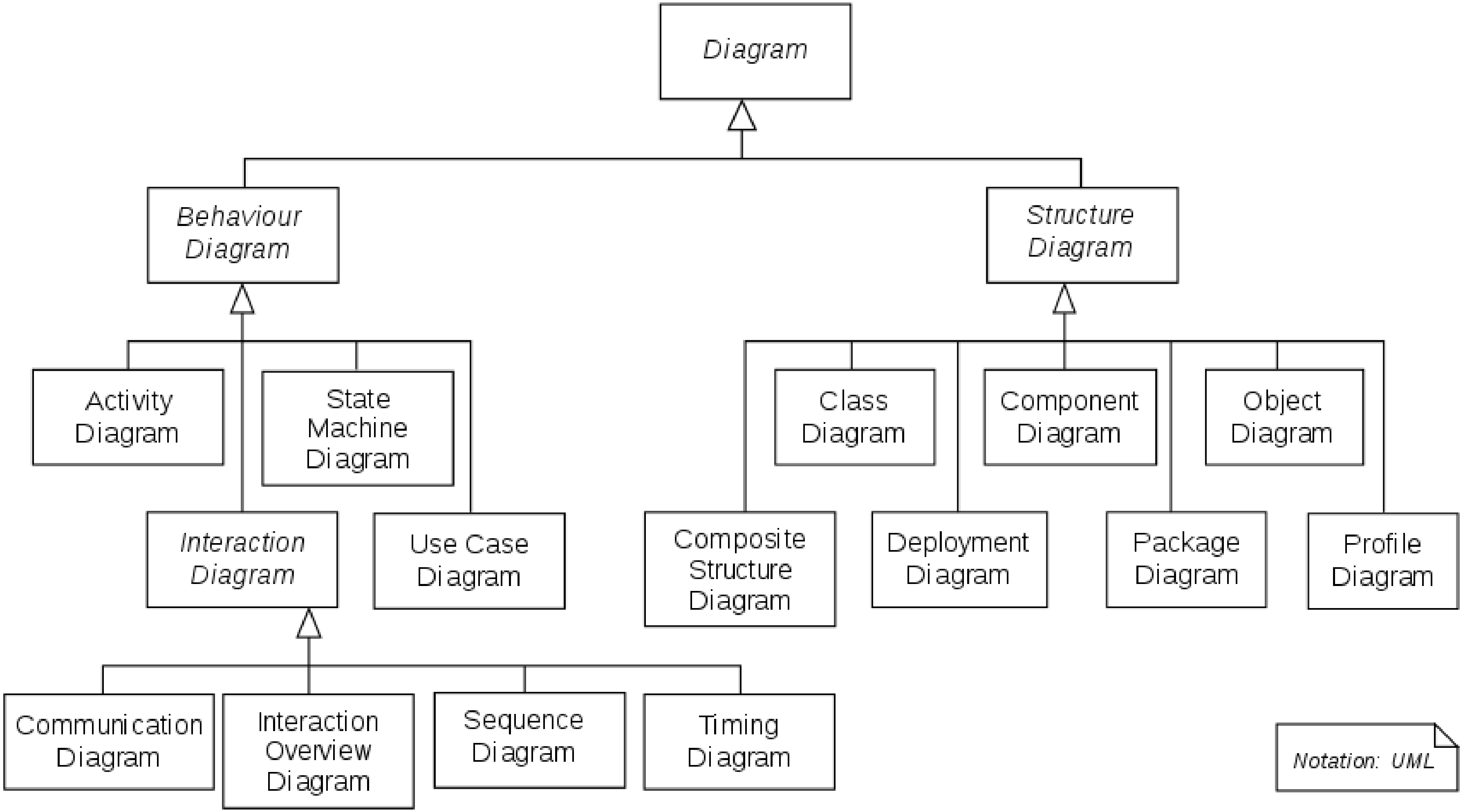
Jose Emilio Labra Gayo
Pablo González
Irene Cid
Cristian Augusto Alonso

School of Computer Science, University of Ovi

UML

- Unified Modeling Language
  - Before UML there were several proposals
  - UML notation unifies them
  - Proposed by OMG (Object Management Group)
  - Current version UML 2.5.1 (2017)
- Model = abstraction of a problem
  - It can have different diagrams
  - Diagram = partial graphic representation of a model
- OCL = Object Constraint Language
  - Constraints between objects using formal language

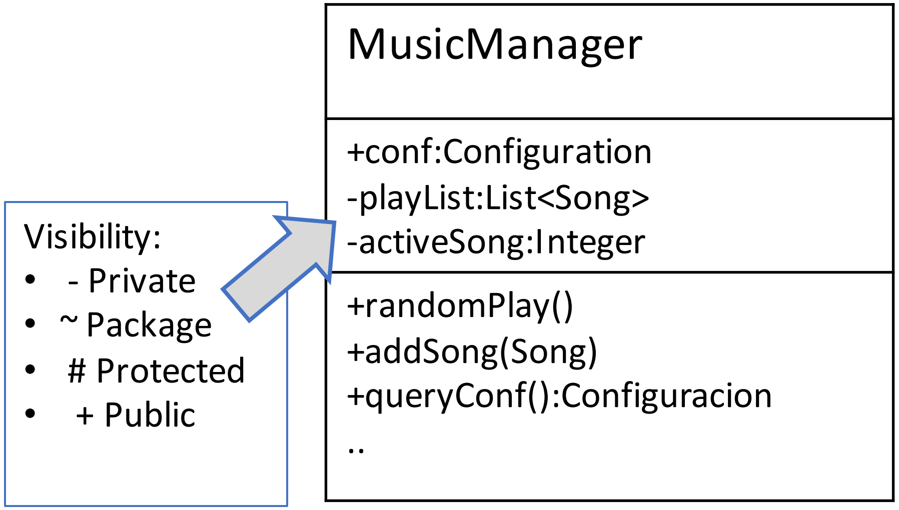
14 UML Diagram types



Fuente: Wikipedia

Class diagrams

- Models the static part of the project, without taking into account the time aspect
- Explains the relationships between the different classes.
- Arc42:8-Concepts



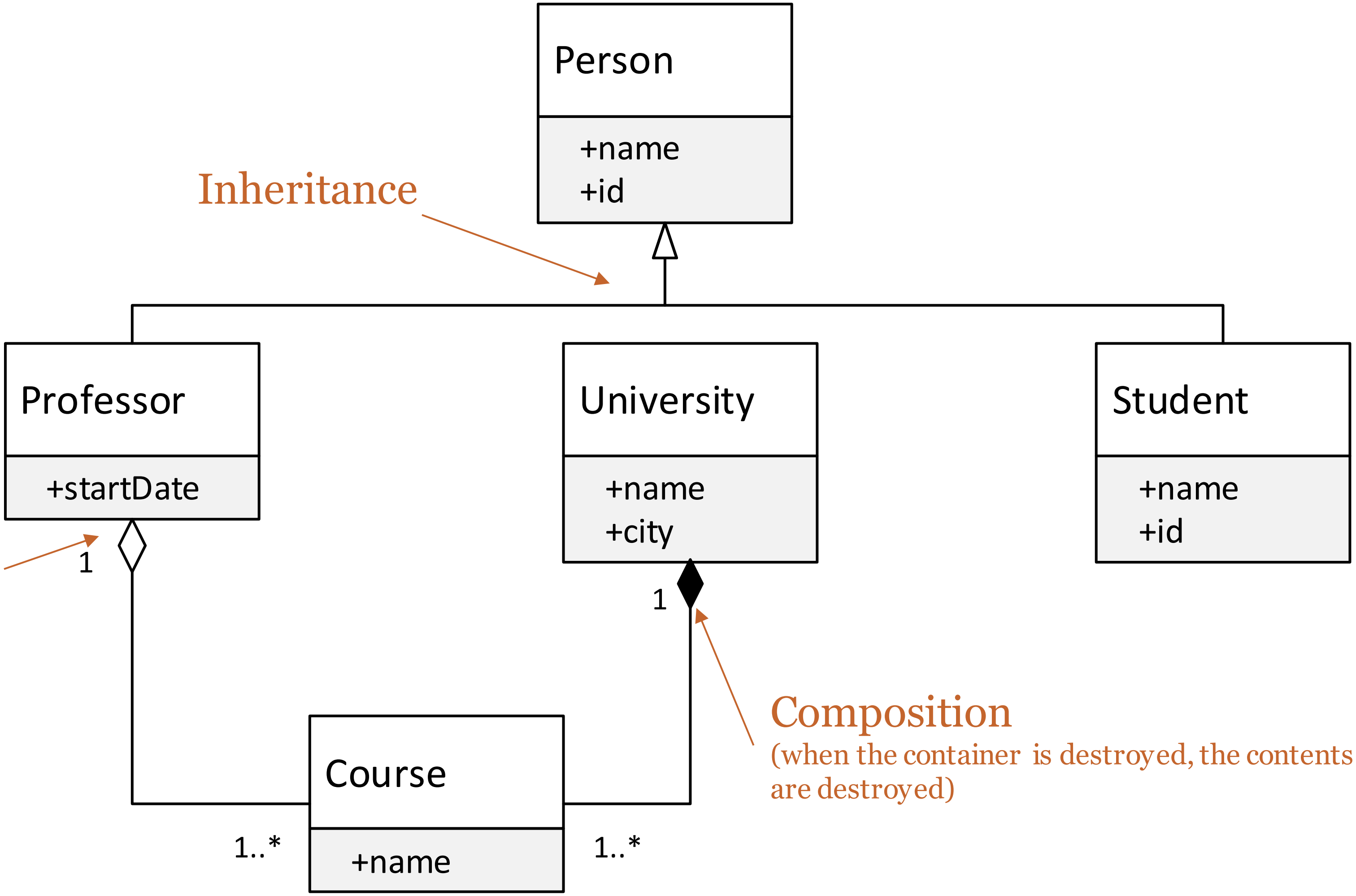
Visibility:
- - Private
- ~ Package
- # Protected
- + Public

MusicManager

+conf:Configuration
-playList:List<Song>
-activeSong:Integer

+randomPlay()
+addSong(Song)
+queryConf():Configuracion
..

Example



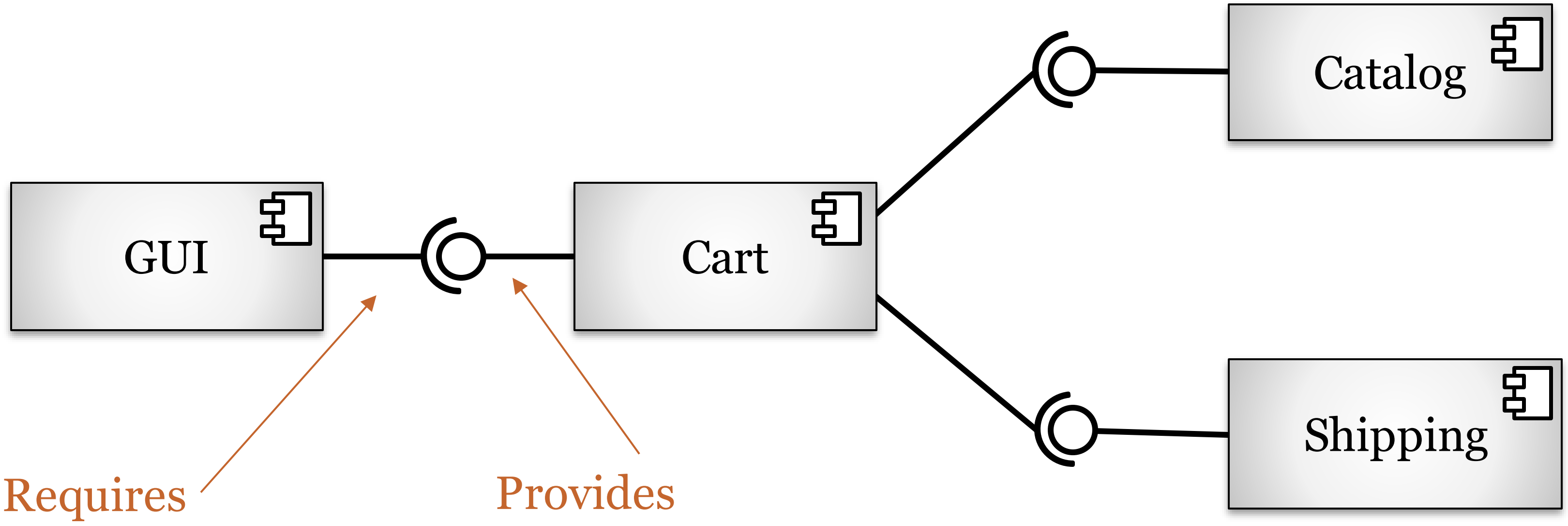
Inheritance

Aggregation

Composition

(when the container is destroyed, the contents are destroyed)

UML Component diagram



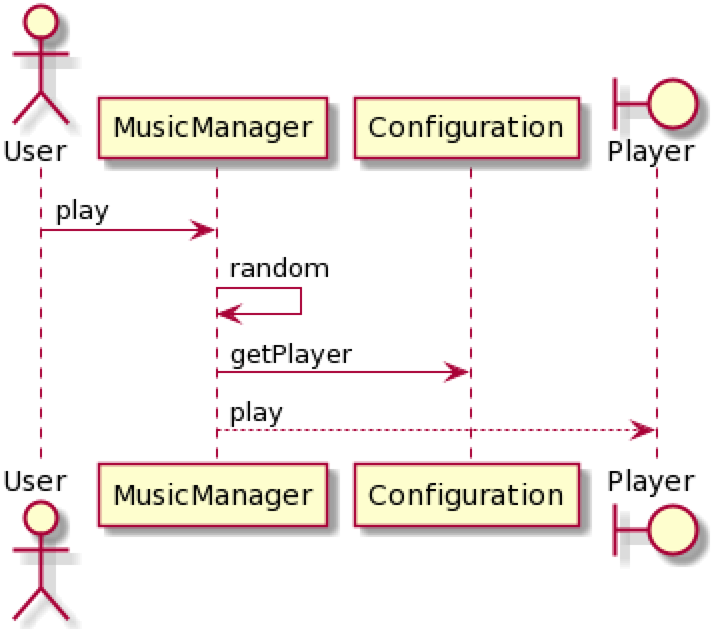
- Component digram represents components relationships
- Useful for Complex Systems with many components

Sequence diagram

Models communication between some objects at a given time

Objects can send two types of messages: synchronous or asynchronous

Arc42:6-RuntimeView



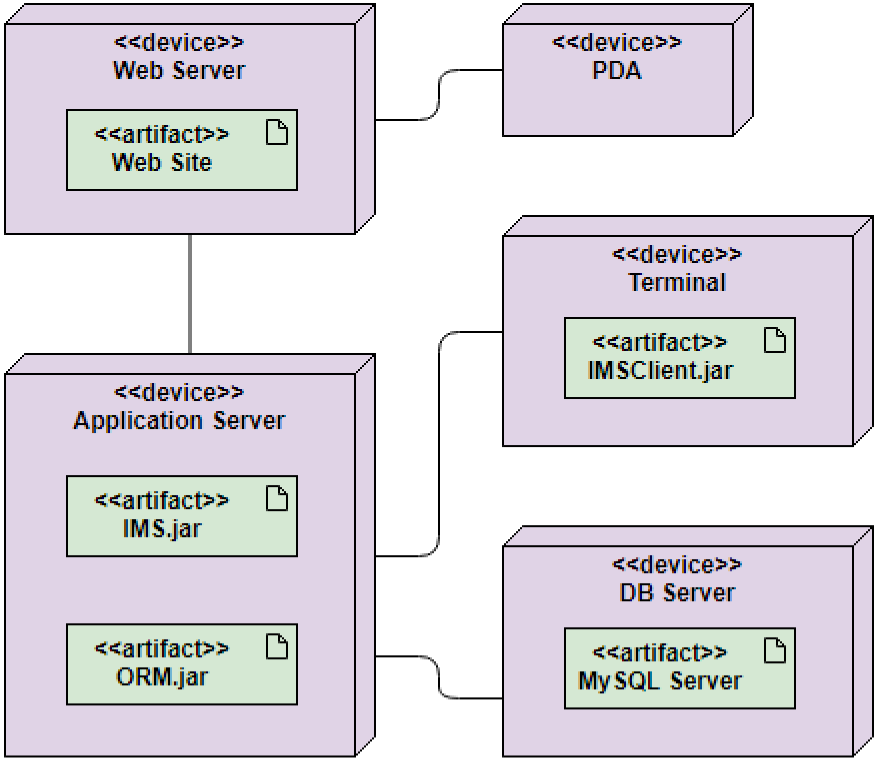
Deployment diagrams

Represents the final location of the components in an app

Elements:

- Nodes , Components, relationships

Arc42: 07.DeploymentView

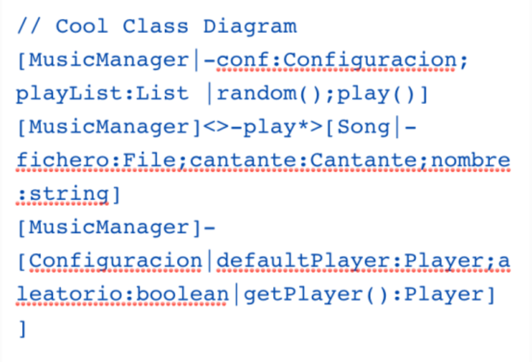


Fuente:VisualParadigm

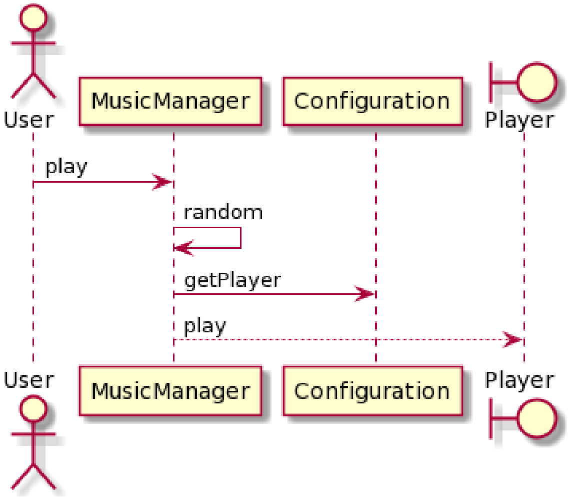
Text-based tools

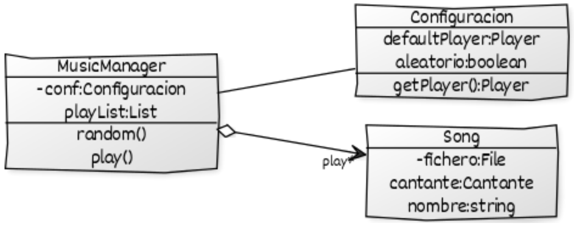
PlantUML

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



```
// Cool Class Diagram
[MusicManager|-conf:Configuracion;
playList:List |random();play()]
[MusicManager]<>-play*>[Song|-
fichero:File;cantante:Cantante;nombre
:string]
[MusicManager]-
[Configuracion|defaultPlayer:Player;a
leatorio:boolean|getPlayer():Player]
]
```





School of Computer Science, University of Oviedo

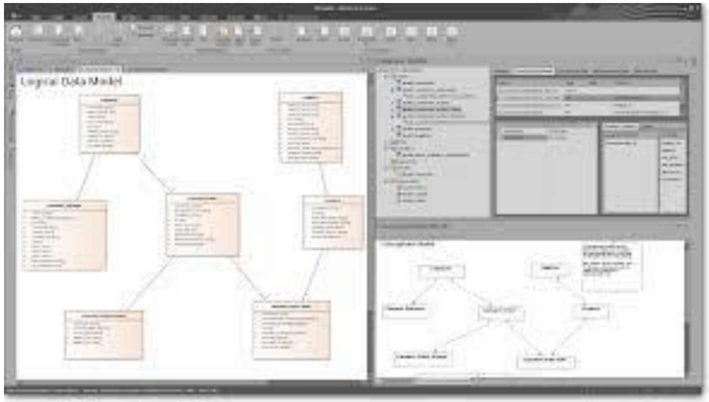
Drawing tools

- Powerpoint
- MsVisio
- UMLet (https://www.umlet.com/)

CASE tools

EnterpriseArchitect

- Reverse Enginering with Java/C++
- Oracle connection for relational databases
- Word, HTML templates



MagicDraw

- Java based
- UML diagrams
- Reverse Engineering Java , C++

Visual Paradigm

- Commercial (student license)

Modelio

- Open source
- Java based
- Reverse Engineering Java , C++

School of Computer Science, University of Oviedo

Diagramming the architecture

Video:

https://www.youtube.com/watch?v=wgpSdpny-0c

Checklist:

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

Arc42 templates

Arc42

- https://arc42.org/

LoMap already follows the template:

- https://arquisoft.github.io/LoMap_0/

Generation of docs:

- `> cd docs`
- `> npm install (only first time)`
- `> npm run build`

Documentation deployment

Documentation is deployed using GitHub Pages

- GitHub Pages allows users to publish a simple website directly on GitHub.
- docs website will be pushed to the branch **gh-pages**
- asciidoc files will be pushed to develop branch of repository (not automatically)
- The npm package **gh-pages** takes care of pushing the doc website to the gh-pages branch
- Everything is automatized with the following command:

```
> npm run deploy
```

GitHub Pages

End of presentation