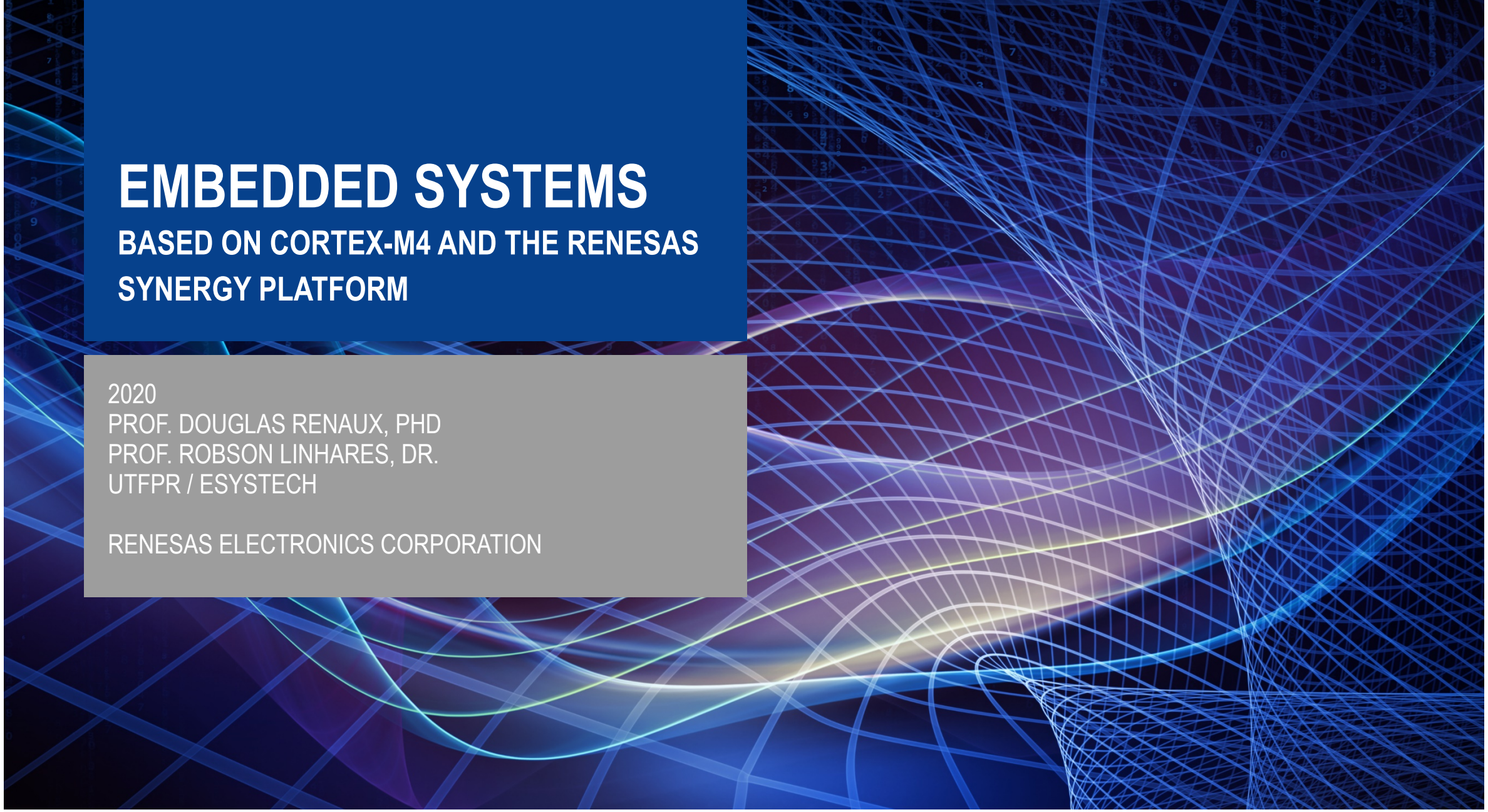


EMBEDDED SYSTEMS

BASED ON CORTEX-M4 AND THE RENESAS SYNERGY PLATFORM

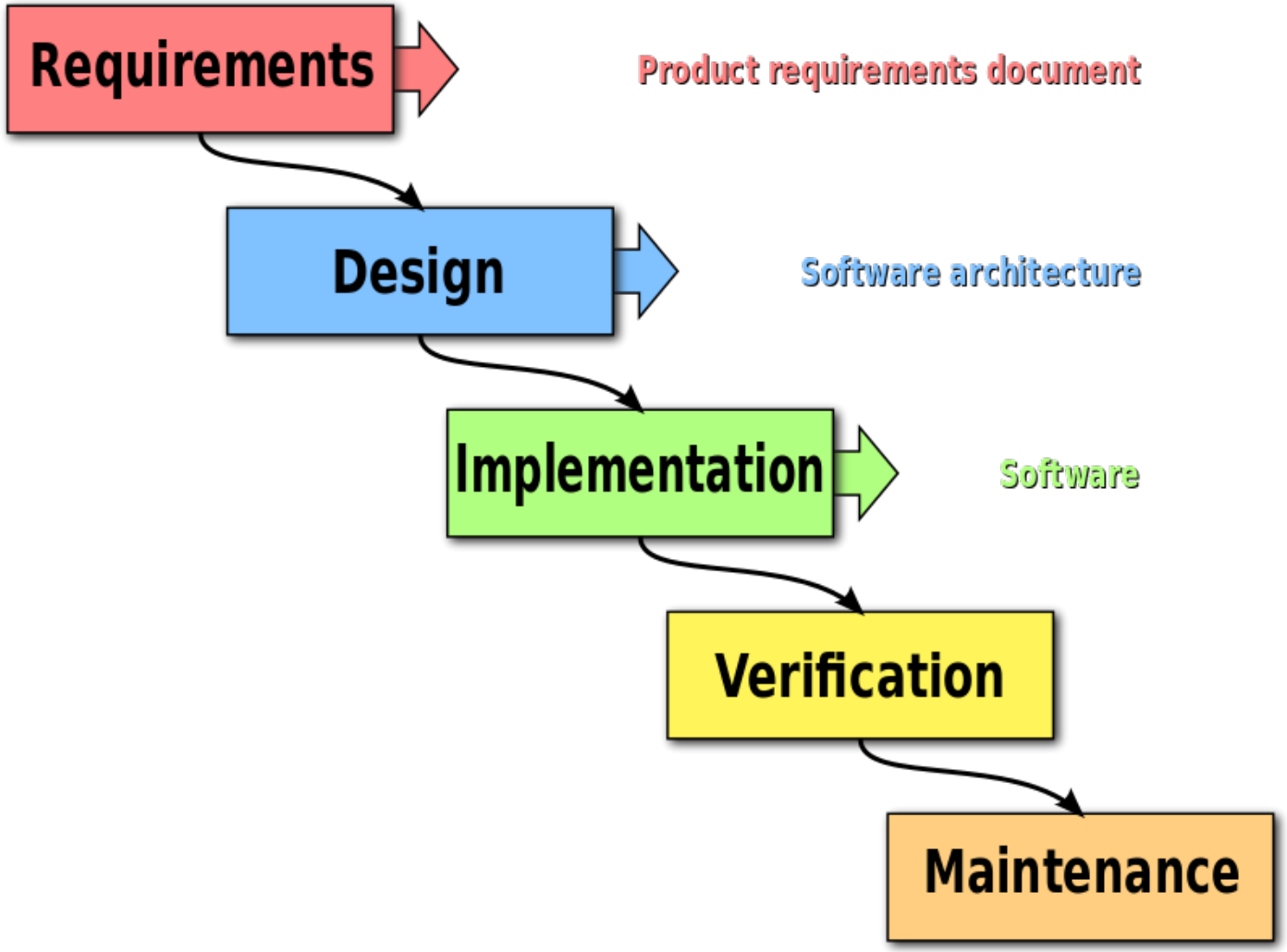
2020
PROF. DOUGLAS RENAUX, PHD
PROF. ROBSON LINHARES, DR.
UTFPR / ESYSTECH

RENESAS ELECTRONICS CORPORATION

12 – SOFTWARE DEVELOPMENT PROCESS

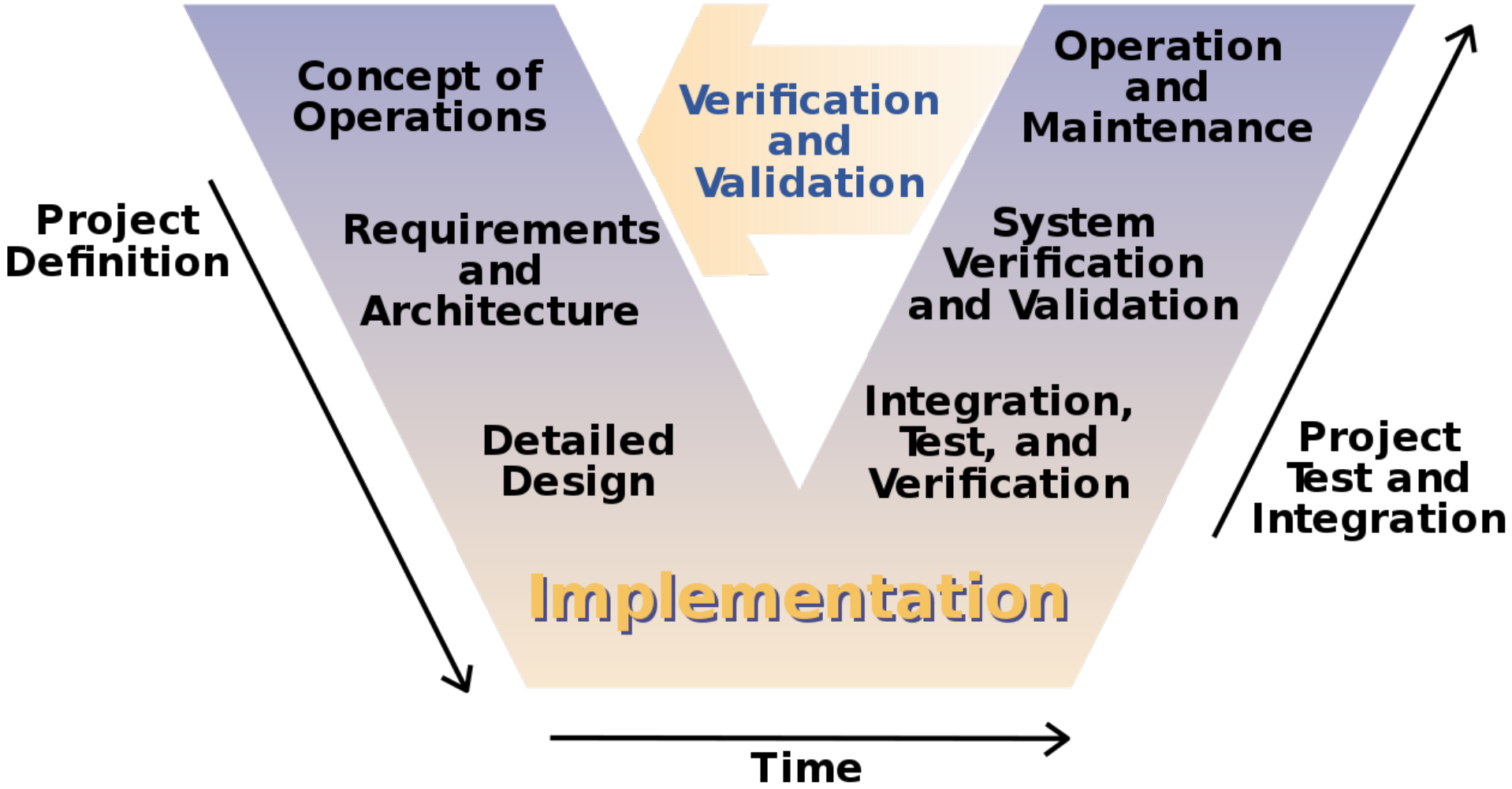
- Software Process Overview
- UML Class Diagram
- UML State Machine Diagram

WATERFALL PROCESS



SOFTWARE PROCESS

V-cycle



UML

Unified Modeling Language

- Originally by Grady Booch, James Rumbaugh and Ivar Jacobson
- Based on the integration of several existing modeling languages
- OMG standard (www.omg.org) in 1998
- Language based on visual models
- Current version: 2.5 (March 2015)
- Non-proprietary language

MODELING

The basic elements of a structural model are:

- THINGS
- Interaction among THINGS

THINGS

Physical things:

- Coffee bean, processor, equipment , vehicle, planet ...

Logical things:

- Bank account, contract, variable stored in memory ...

INTERACTION AMONG THINGS

Examples:

processor is connected to memory

personA and personB are married

personX is responsible for bank_account_12345

CLASS / OBJECT

A cohesive entity that has attributes, behavior and (optionally) state.

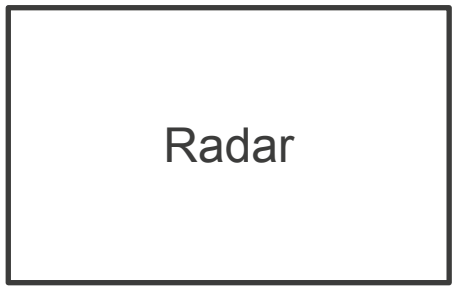
Characteristics of a Class / Object:

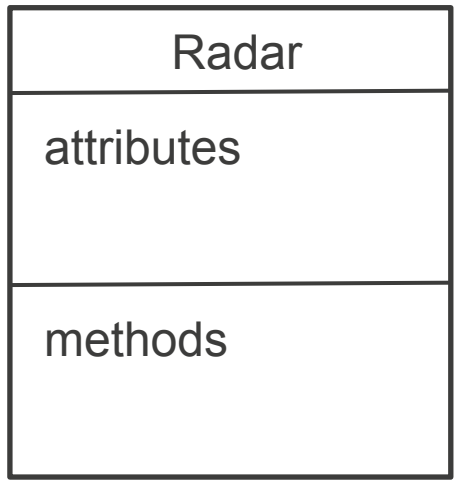
- Data - attributes
- Behavior - operations, methods, services, functions
- State - memory of its past
- Identity - unique identifier
- Responsibilities

CLASS / OBJECT IN PROGRAMMING LANGUAGES

Class	Object
example: a type in C	example: a variable in C
Compile time existence	runtime existence
	Ocupies memory
the design of an object	an instance of a class

CLASS REPRESENTATION



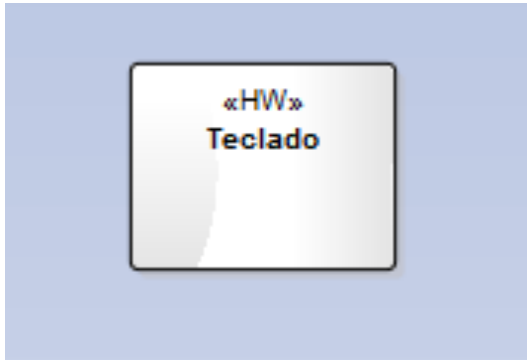


STEREOTYPE

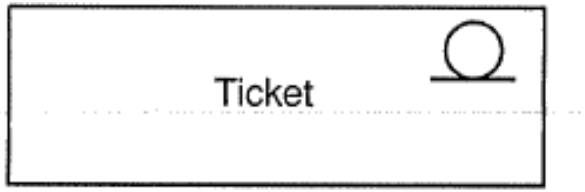
An additional (informal) form of classification.

Notation:

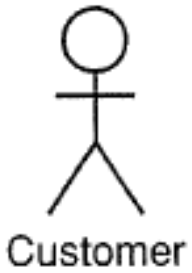
- text: <<stereotype_name>>



- icon



- class is re

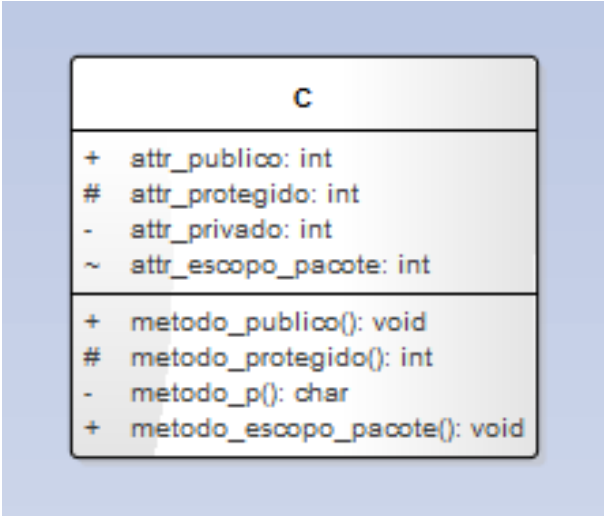


ATTRIBUTES / METHODS – VISIBILITY

A typical class representation has:

- class name (+ stereotype)
- attributes
- methods

The visibility of the methods and attributes is indicated by:



Symbol	Meaning	Visibility
+	public	accessible to all
#	protected	accessible by derived classes
-	private	accessible only by this class
~	package scope	accessible to the other classes in this package

INTERACTION AMONG THINGS

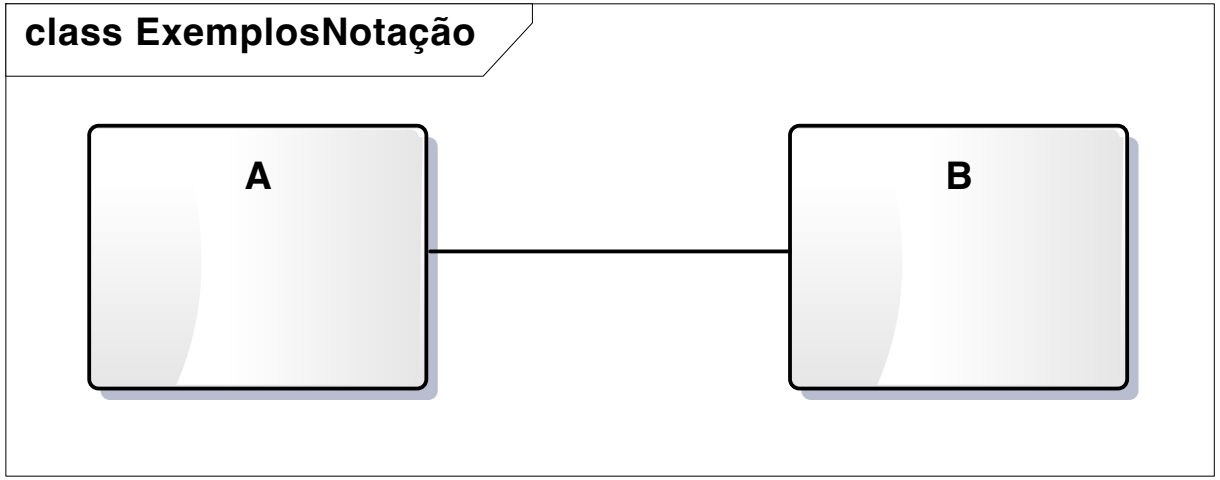
Relationships:

- Association
- Aggregation
- Composition
- Generalization
- Dependency

ASSOCIATION

A and B know of the existence of each other and may interact:

- access to public methods
- access to public attributes

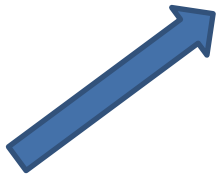


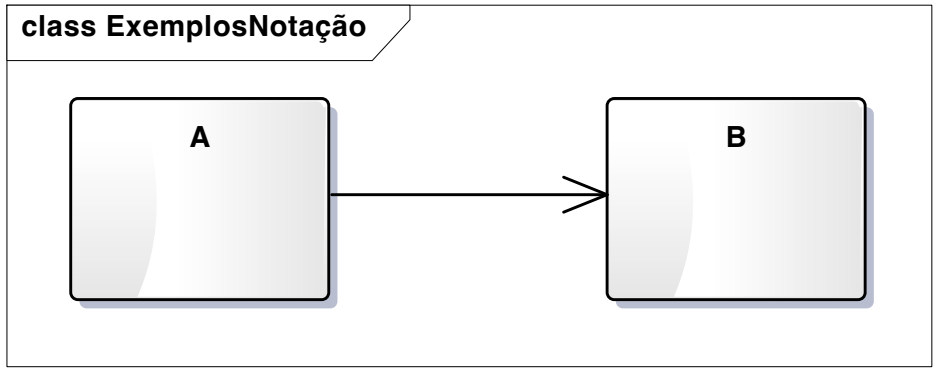
NAVIGABILITY

Who has access to whom?

A has access to B, but

B does not have access to A

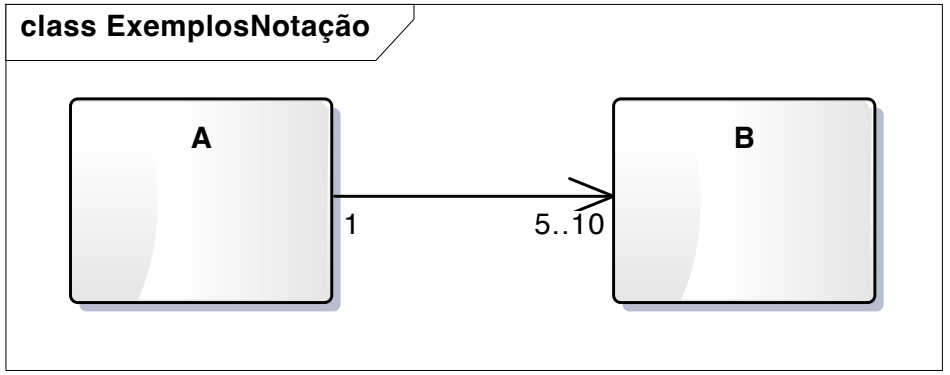





```
class A;
class B;

class A {
    B * pB;
    int a_attr;
};

class B {
 public:
    int b_attr;
};
```

MULTIPLICITY

How many objects of each class participate of this relationship?



1	exactly one
*	many (0 or more)
n	many (0 or more)
1..*	1 or more
3..20	from 3 to 20
4,6,8	4 or 6 or 8

```
class A;
class B;

class A {
    B * pB[10];
    int a_attr;
};

class B {
 public:
    int b_attr;
};
```

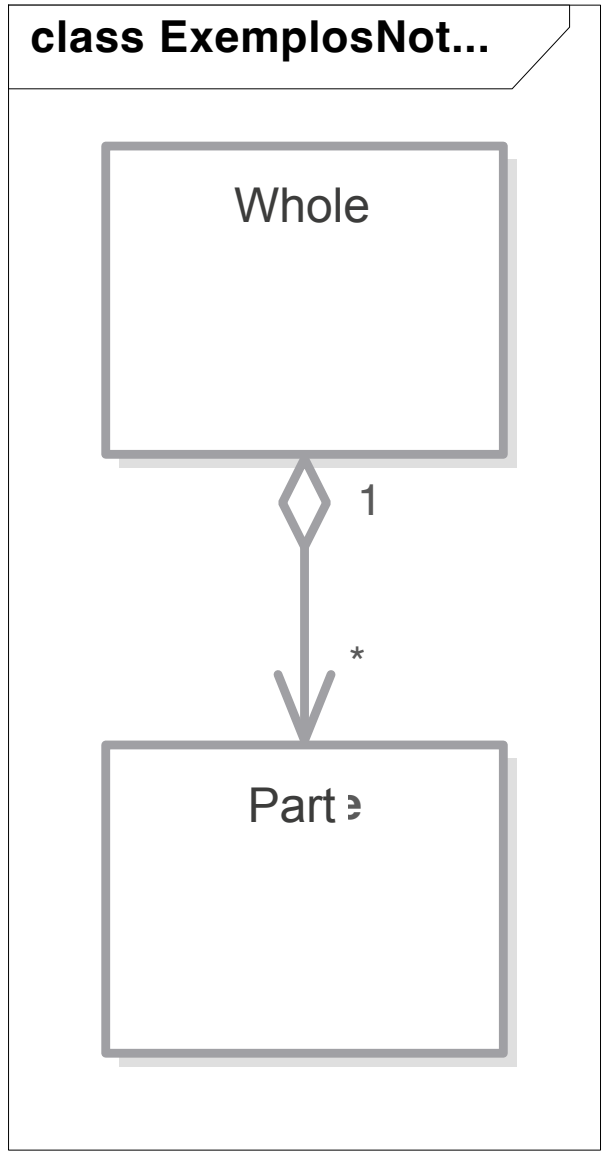
AGGREGATION

This relationship represents a weak part-whole or part-of.

The parts lifetime are different from the lifetime of the whole.

Read as:
Whole has a Part.

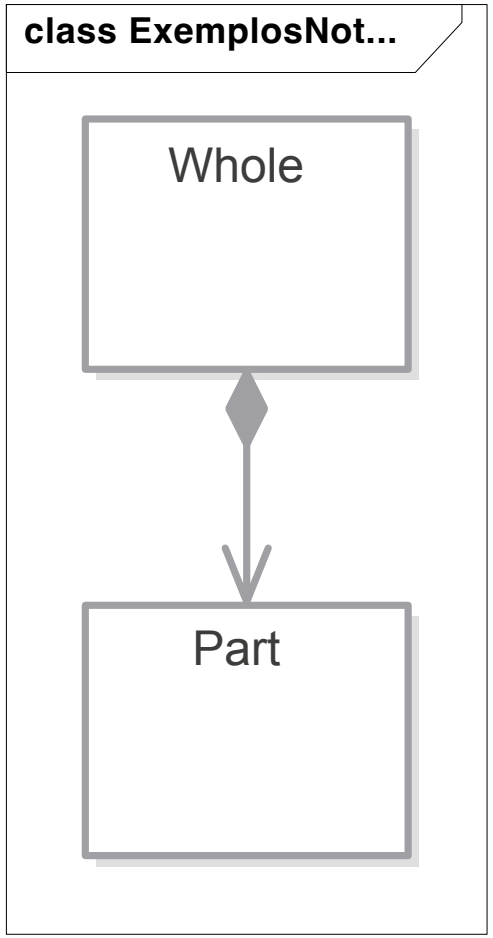
```
class Whole;
class Part;

class Whole {
  Part * pP[10];
  int a_attr;
};

class Part {
  public:
   int b_attr;
};
```

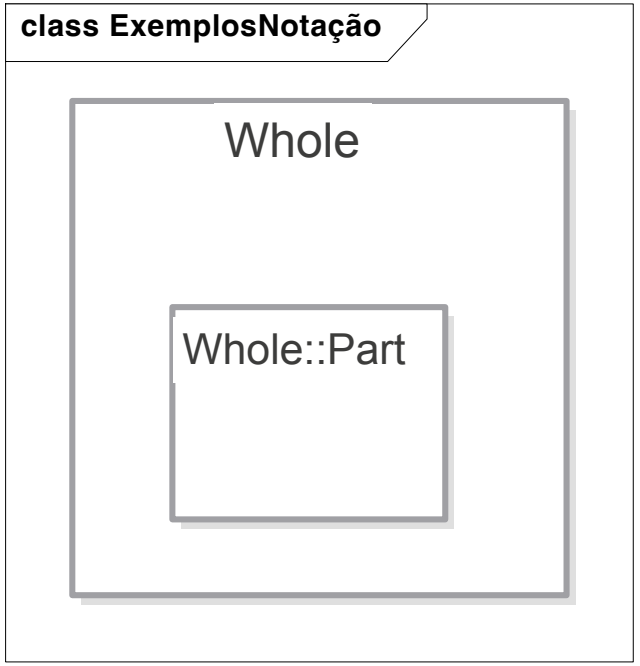


COMPOSITION

- Strong whole-part relationship.
- The whole and the parts form a single entity.
- Same lifespan for the whole and the parts.

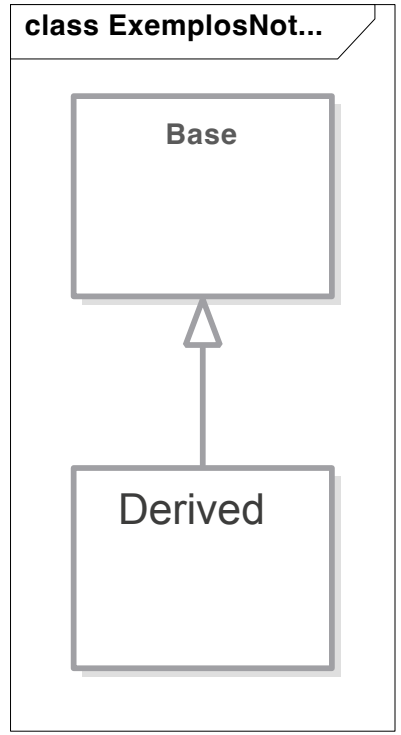
```
class Part {
  public:
    int b_attr;
};

class Whole {
    Part obj;
    int a_attr;
};
```





GENERALIZATION

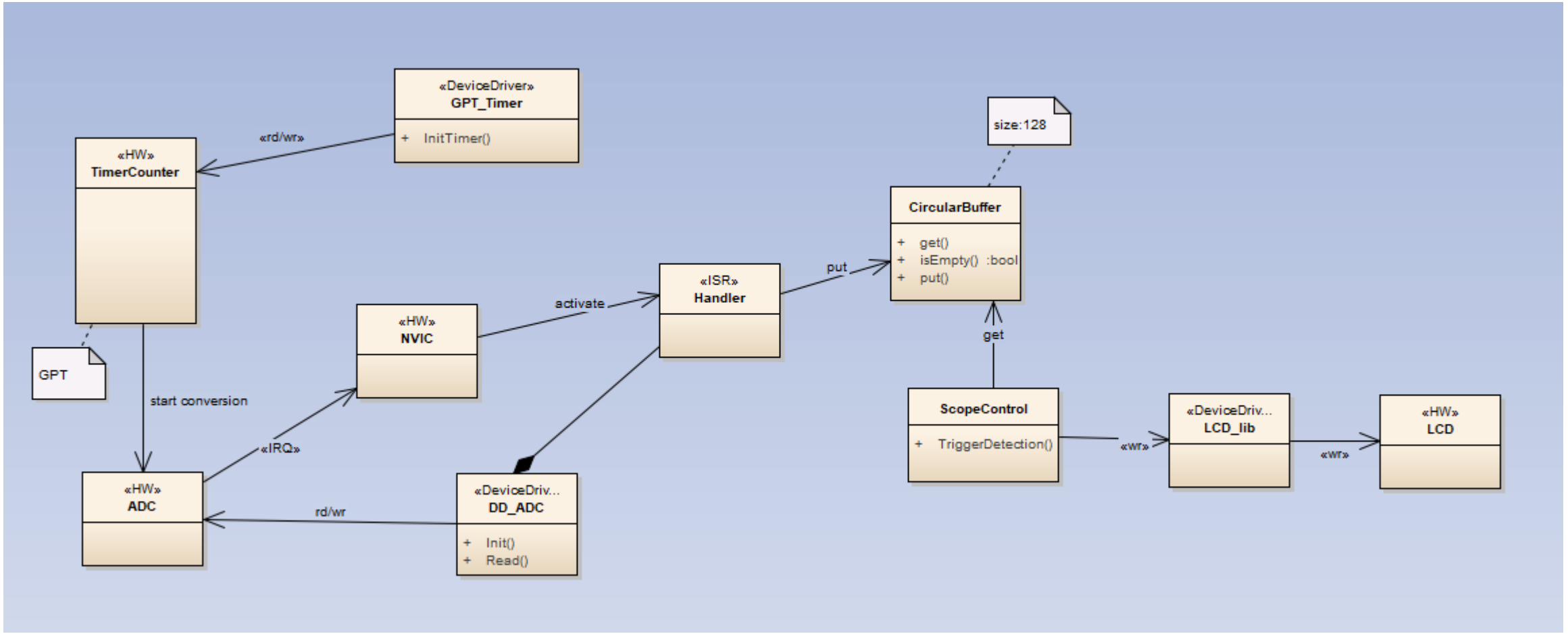
- Represents inheritance.
- The derived class inherits all methods and attributes of the base class.
- Reading:
  Derived **is-a** Base.




```
class Base {
  protected:
    int base_attr;
    void base_meth(int);
};

class Derived : public Base {
    int der_attr;
};
```

USING UML TO DESIGN AN EMBEDDED SOLUTION

- Classes may represent: classes (e.g. C++), set of cohesive functions in C, hardware components, ...
- Since classes represent such a variety of things, the use of stereotypes is encouraged to inform the type: «HW», «ISR», «device driver»,...
- Associations also have several possible interpretations, from actual physical connections to pointers (e.g. in C). The use of stereotypes is also encouraged.
- A common mistake is to consider the navigation adornment as an indication of the direction of flow of data. It is not!



This is an example of a class diagram representing the components of a very simple digital scope.
Stereotypes are used to document what type of thing is being represented by each class: HW, Device Driver, ISR, ...

Renesas.com