Software Architecture Description

YOLO

Group-97
Mert Kaan YILMAZ
2381093

Contents

List of Figures

List of Tables

Revision History

Date	Reason For Changes	Version
22.05.2022	Initial Draft	0.1

Table 1: Revision History of Software Architecture Description Document

1 Introduction

This document is the Software Architecture Description for YOLO project created by a small group of people. The document is written with respect to the specifications of the IEEE Systems and software engineering - Architecture description[1].

1.1 Purpose and objectives of the YOLO social robot

Your Own Living Object (YOLO) is a minimalistic, standalone, portable robot with open-source software that aims to increase children's creativity during playtime. The purpose of this project is to build a small robotic toy named YOLO, which uses light, touching, and movement as inputs and has a reasonable weight and height to be grabbed and easily moved around by the children, just like the other traditional toys. YOLO can provide new ideas for their stories according to the child's movement generated while playing. This way, a social robot that increases children's creativity is designed for and with children.

1.2 Scope

- The system shall make its user feel that the robot is alive when there is no interaction with the robot by doing puppeteer kind of behaviour.
- The system shall move with respect to child's play style.
- The system shall react to child's story with different behaviours by creating different movement patterns.
- The robot shall react with it's LED lights according to if it is touched or its different social behaviour.
- The robot should perform behaviours with its creativity technique.
- Users shall be able to develop new functionalities and add new behaviour models.

1.3 Stakeholders and Their Concerns

There are several different stakeholders for this system. They can be grouped as users, system developers, researchers, and educators-parents.

1. Users
Users can be considered as end-users. They are the people who use the YOLO robot for the main purpose of the project. They interact with the robot, act in different scenarios and boost their creativity. Therefore, the main concerns for the users are the easy usability of the robot, and playing with the robot without any technical problems or knowledge.

2. System Developers
System developers are the team members who are responsible for developing YOLO software. Their main concern is that YOLO software includes several modules that manipulate data at different levels of abstraction, from low-level sensors and actuators to high-level behaviors. Additionally, end-users can be considered system developers, but they actually maintain the system and maybe add new small features to the overall design.

3. Researchers
Researchers are the people whose concerns are to collect data and study on child-robot interaction. The YOLO software serves as a solid platform in academic studies. As societies develop into creativity-based economies, innovative and creative problem solving and the ability to collaborate are becoming must-have skills. Therefore, researchers' concern is to show that everyone has the potential to be creative and that creativity can be nurtured if stimulated with YOLO project.

4. Educators and Parents
These two different types of stakeholders can be considered as one. They both want to test and reveal the potential of the YOLO project, as robot using the software provoke creative narratives in stories which the children created. They mainly concern being an easy-to-use tool and having access to the robot that is easy to prepare.

2 References

[1] IEEE Computer Society, 42010-2011 - ISO/IEC/IEEE International Standard - Systems and software engineering - Architecture description.

[2] Patrícia Alves-Oliveira, Samuel Gomes, Ankita Chandak, Patrícia Arriaga,Guy Hoffman, Ana Paiva. Software architecture for YOLO, a creativity-stimulating robot. (2020)

[3] YOLO software open-source code https://github.com/patricialvesoliveira/YOLO-Software

3 Glossary

API	An application programming interface is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software.
User	An unauthorized person who interacts with the YOLO robot physically.

Table 2: Glossary

4 Architectural Views

4.1 Context View

4.1.1 Stakeholders' use of this view

In this viewpoint, context of the system with its actors are described with their general viewpoint. Context diagram and its description gives general idea of interaction between actors and the YOLO system with the information that's conveyed between each other. Therefore, this view is important for almost all of the stakeholders to get the basic idea.

4.1.2 Context Diagram

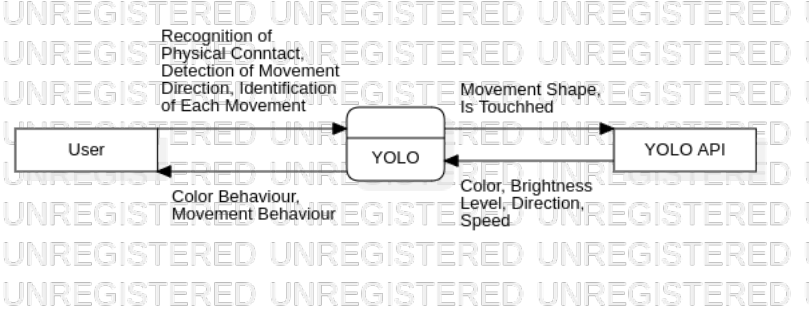

Figure 1: System Context Diagram for YOLO

The YOLO software is not a part of a more extensive system, but it interacts with the physical components of the robot to get input and show behavioral outputs. Inside the robot's body, there is a Raspberry Pi that runs python scripts. In this way, the robot constantly stays connected with the real world with its sensors. There are three different sensors on the YOLO robot, such as touch, shape, and optical sensors, and they provide information to the YOLO system. In the system, these pieces of information are directed into relevant parts of the robot, which are called actuators, and according to the input, the robot shows some behaviors.

4.1.3 External Interfaces

bla bla bla

4.1.4 Interaction scenarios

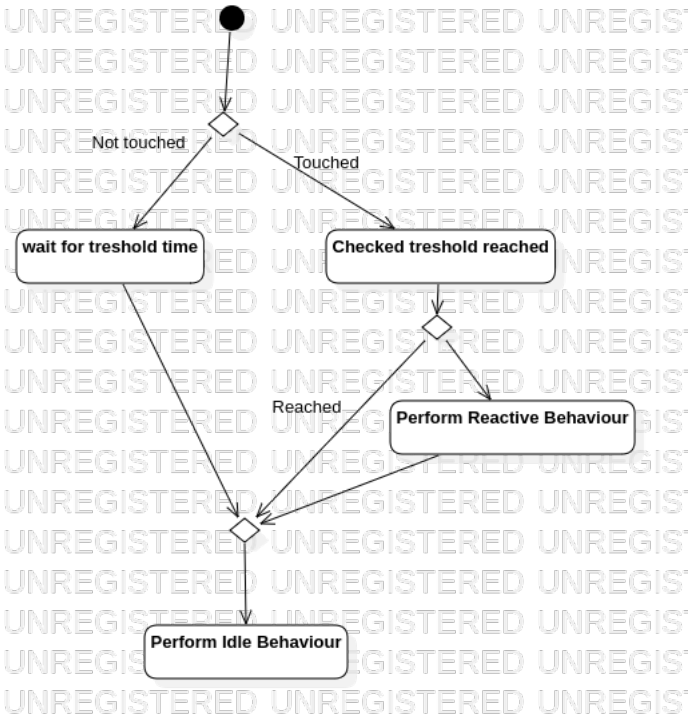

Figure 2: An activity Diagram of YOLO

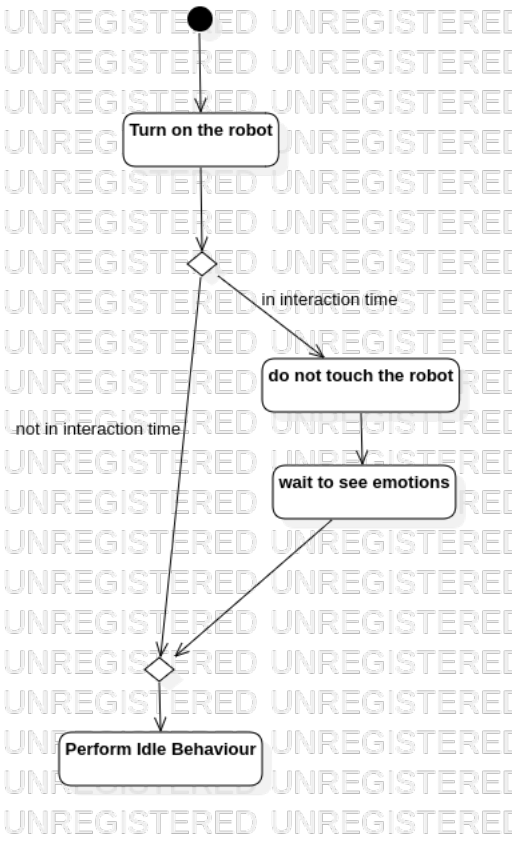

Figure 3: An activity Diagram of YOLO

4.2 Functional View

4.2.1 Stakeholders' use of this view

This view is a component-level view of the system as it is running. Namely, main functions of the system are described and modeled here. Therefore, this view is important for users and developers. Users might get the general understanding of how the system works and developers might use this to further improve the system.

4.2.2 Component Diagram

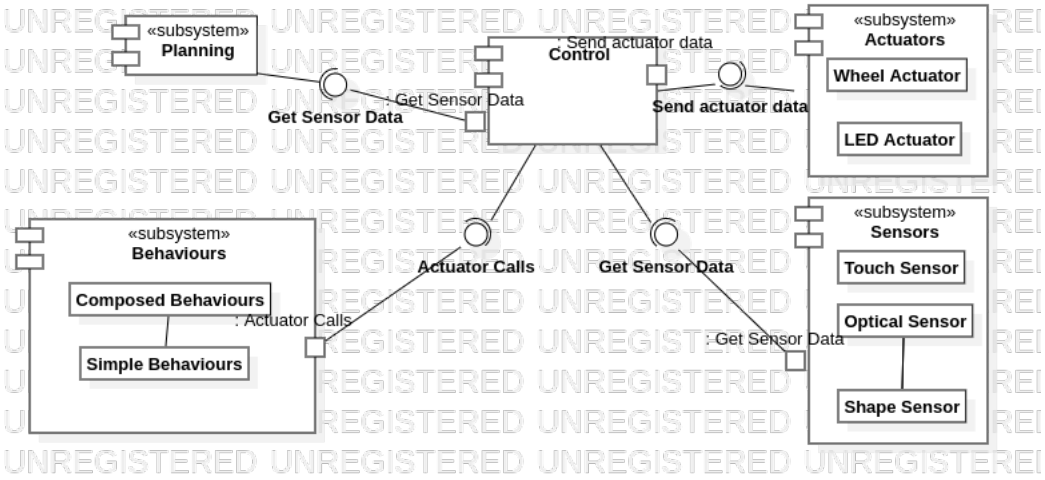

Figure 4: Component Diagram of YOLO

In the whole system, there are multiple different subsystems in YOLO project and they are related with each other via some interfaces. We can generally say that control subsystem manages almost all the action, data transmission in the system, and so on. Other components are responsible to do their job with the information they got or sent to control subsystem.

4.2.3 Internal Interfaces

bla bla bla

4.2.4 Interaction Patterns

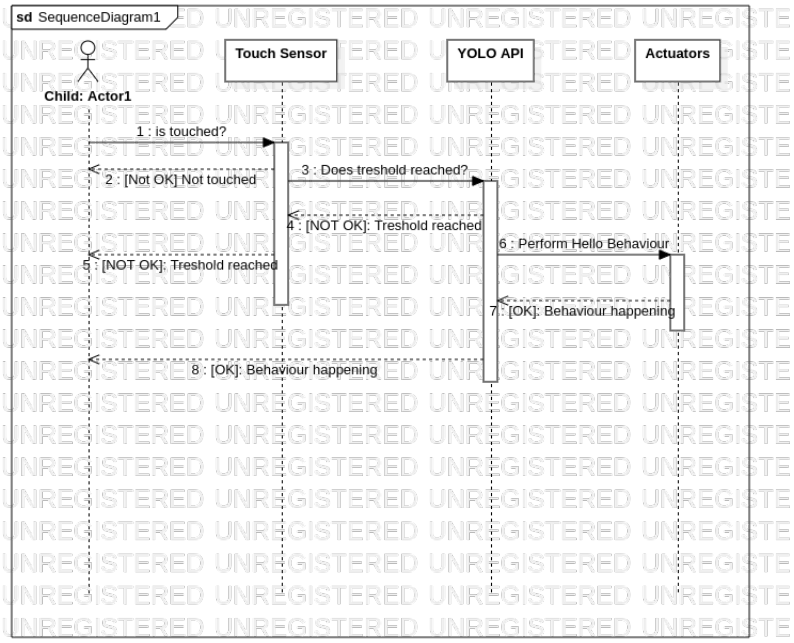

Figure 5: Sequence Diagram for an interaction

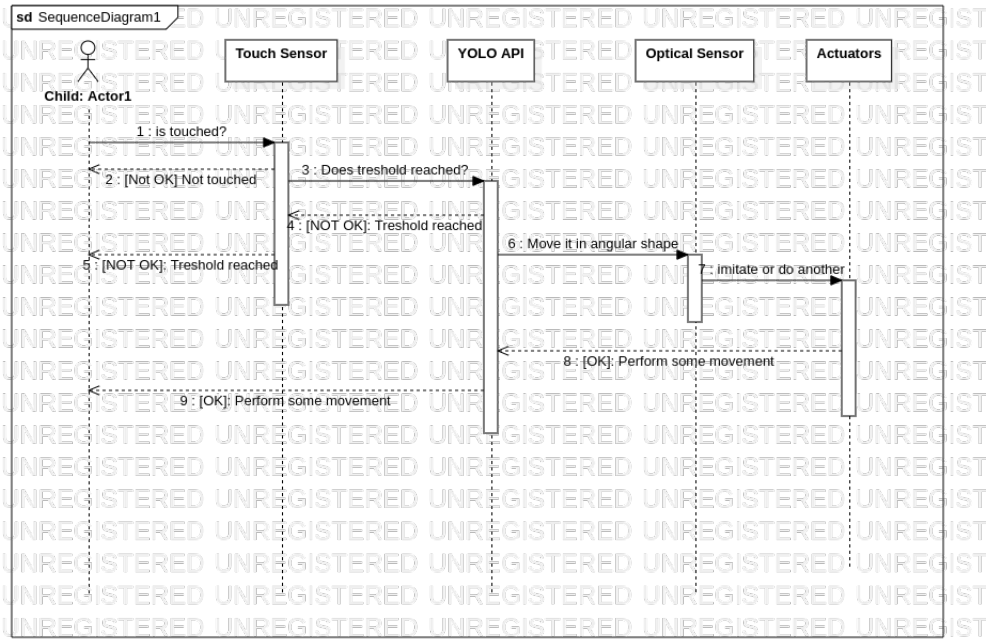

Figure 6: Sequence Diagram for an interaction

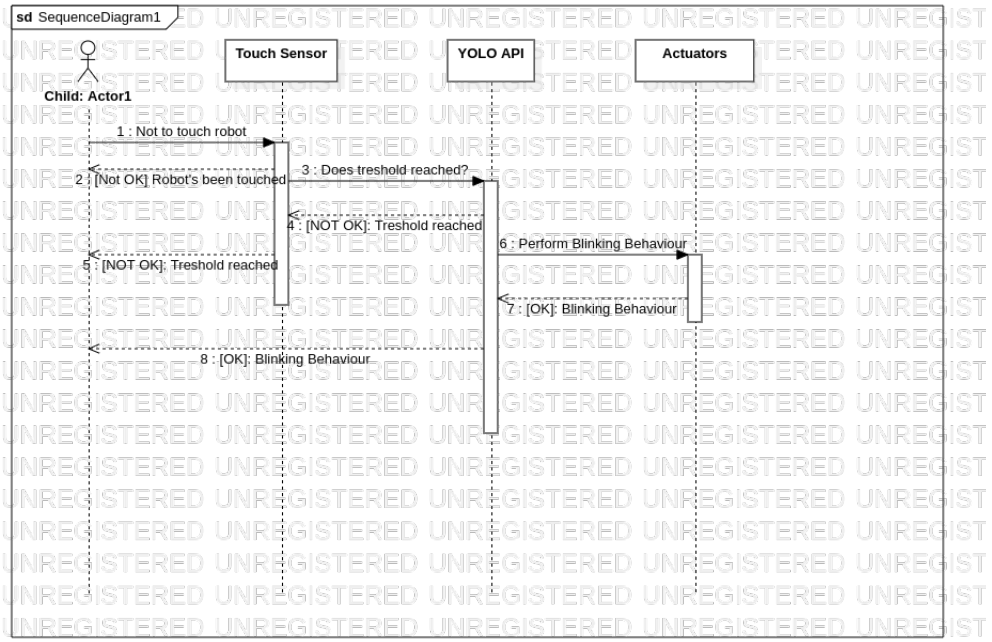

Figure 7: Sequence Diagram for an interaction

4.3 Information View

4.3.1 Stakeholders' use of this view

The information view might mainly concern two stakeholders. One of them is system developers, and the other one is researchers. When you build a system, it does not matter if hardware-heavy or software-heavy, you need to deal with data transmission in your project. Therefore, while developing the system, all the data operations should be considered. On the other hand, this view might be useful for researches, too. When they try to manage a social research on robot and human interaction, they need to take care of information handling between two of those actors. Therefore, researchers are also concerned about this view as much as developers do.

4.3.2 Database Class Diagram

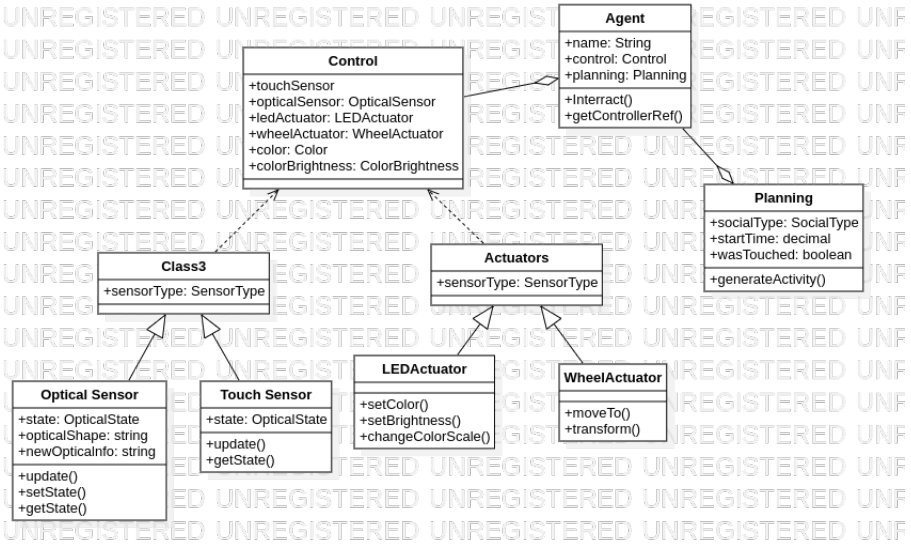

Figure 8: Database Class Diagram for YOLO

4.3.3 Operations on Data

Operation	Description
getControlRef	This method returns the Control instance associated to the agent.
update	This method updates the robot's sensors, actuators, and the behaviour schedule.
setColor	This method applies a certain color tone to the robot's LEDs
setBrightness	this method applies a certain color brightness to the robot's LEDs using the LEDActuator.
setWheelMovement	This method orders the wheelActuator to direct the robot to a certain point in space at a certain speed as defined in the WheelActuator class.
predictShape	A shape is returned based on the array of points given by the OpticalSensor used in the Control.

Table 3: Operation Descriptions

4.4 Deployment View

4.4.1 Stakeholders' use of this view

This view basically captures the hardware environment that your system needs, the technical environment requirements for each element, and the mapping of the software elements to the run-time environment that will execute them. Therefore, the main stakeholders for this view are system developers. Because the design of the overall deployment of the system is important for initial development and future improvements, system developers might be concerned about this view.

4.4.2 Deployment Diagram

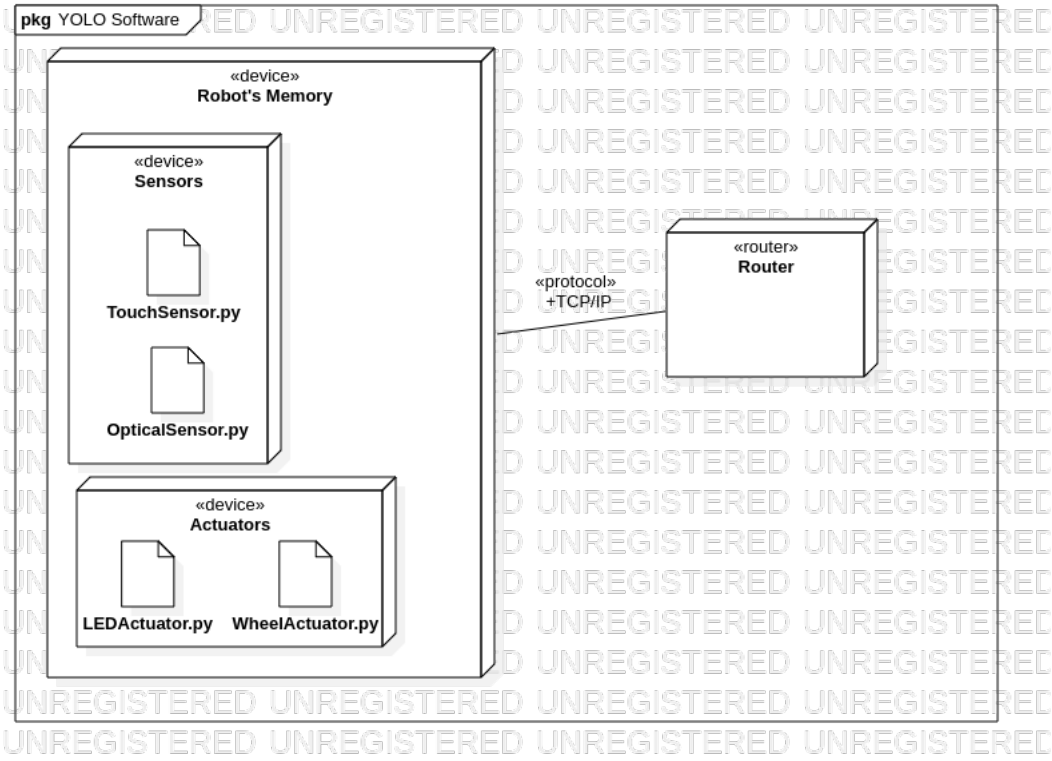

Figure 9: Deployment Diagram of YOLO

4.5 Design Rationale

- Context View: General structure, actors and cases shown.
- Functional View: Functionalities of the system is described in more detail. Different cases are handled with methods step by step.
- Information View: In order to make clear database design, one need to show information with a well structure. With database models and its diagrams, relationship between data elements and methods are also considered.

- Deployment View: It show the where software and some related components of the system are deployed the be used as a complete product.