Software Architecture Description (SAD)

for

Your Own Living Object (YOLO)

Group - 100

Berkay Bartuğ Çetin (2309839)

Anıl Berdan Ceylan (2304277)

Version 1.1

METU, 6. June 2022

Table of Contents

List of Figures

List of Tables

Revision History

Version	Date	Explanation
1.0	23.05.2022	Context diagram is drawn. Project purpose, scope, stakeholders and their concerns and general information are added to the document.
1.1	06.06.2022	Rest of the document is prepared, all the necessary parts are added.

Table 1: Revision History of Software Architecture Description Document

1 Introduction

This document is the Software Architecture Design (SAD) of a social-smart robot which is YOLO (Your Own Living Object) developed by four researchers.

1.1 Purpose

The purpose of this project is to create -a new generation of technological toy- a smart robot which can interact with children and help them improve their creativity. The main idea is to increase the creative thought process of children by interacting with them by using alternative but effective interactive modalities.

1.2 Scope

YOLO (Your Own Living Object) is a social robot designed to foster creativity in youngsters through storytelling activities. It appears in children's literature as a character. The YOLO artificial intelligence software tests a range of Creativity Behaviours to see which ones are the most effective in stimulating creativity. YOLO can choose between two different types of creative thinking: convergent and divergent thinking. These tactics were developed based on psychological theories of creativity development as well as research from experts in the field of children's creativity.

In addition to fostering creativity, this program allows for the creation of Social Behaviours, which allow the robot to behave like a real human. The three primary social behaviour qualities that emerged are exuberant, aloof, and harmonious.

These features are meant to facilitate immersive gameplay and character development. During co-design research, the three social behaviours were developed with the cooperation of children and were founded on psychological theories of personality.

In general, this paper examines the design, development, and deployment of social robots that can assist individuals in developing natural human characteristics such as creativity.

1.3 Stakeholders and their concerns

YOLO is an open-source project so there is no commercial gain in the project. On the other hand, since YOLO includes numerous sensors, machine learning algorithms, researches and developers are involved. Also, there is one more main user group which is end users.

- **Researchers and Developers (R&D):** Since researchers are also involved in the development process and vice versa, we cannot separate those two groups of people. These are the people who does research and development for YOLO. Researchers are the people whose concerns are to gather information and analyse the information about YOLO. They are mainly interested in the outcomes and the real world performance of YOLO. Studying and analysing the data gathered from the system might lead to new developments on the smart-robot.

- **End Users:** Users are people who use the project (project's software documents, guidelines on the project's github page etc.). To use the system, users need to complete some steps.
  • Downloading required softwares and documents.
  • Get required hardware tools.
  • Assembly hardware tools according to instructions which are provided by the project page.
  • Set up software in related hardware such as Raspberry Pi.
  Users have mainly 2 concerns:
  • Users need to create their YOLO with low prices, because not all users have high technology software or hardware devices.
  • Creation (setup, assembly phases) of YOLO should not take a long time and the UI of the system should address all kinds of users. Simple and useful system for users.

2 References

This document is written with respect to the specifications of the document below:

42010-2011 - ISO/IEC/IEEE International Standard - Systems and software engineering - Architecture description

Other sources:

- Alves-Oliveira, P., Gomes, S., Chandak, A., Arriaga, P., Hoffman, G., & Paiva, A. (2020b). Software architecture for YOLO, a creativity-stimulating robot. SoftwareX, 11, 100461. https://doi.org/10.1016/j.softx.2020.100461

- Alves-Oliveira, P., Arriaga, P., Paiva, A., & Hoffman, G. (2019). Guide to build YOLO, a creativity-stimulating robot for children. *HardwareX*, *6*, e00074. https://doi.org/10.1016/j.ohx.2019.e00074

- Patrícia Alves-Oliveira, Patrícia Arriaga, Ana Paiva, and Guy Hofman. 2021. Children as Robot Designers. In Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction (HRI '21), March 8–11, 2021, Boulder, CO, USA. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3434073.3444650

3 Glossary

Term	Definition
YOLO	Your Own Living Object
App	Abbreviation for Application
DBMS	Abbreviation for Database Management System
ML	Machine Learning
R & D	Research & Development
API	Application Programming Interface
Researcher	A person who carries out academic or scientific research
Raspberry Pi	Series of small single board computers. It makes YOLO programmable and responsive.
Developer	A person or company that creates new products, especially computer products such as software
LED	A light-emitting diode
GPDI	Github Project Document Interface
CRUD	Abbreviation for Database Management System
HTTPS	Abbreviation for Hypertext Transfer Protocol Secure
TCP	Abbreviation for Transmission Control Protocol

Table 2: Glossary

4 Architectural Views

4.1 Context View

In this viewpoint, the context of the system with all actors are defined in general and detailed viewpoints. In the context diagram, actors and their interaction with YOLO will be explained in general terms. Use case diagrams and the detailed explanations of some possible use cases of the system will be specified below the context diagram.

4.1.1 Stakeholders' use of this view

Your Own Living Object (YOLO) is an intelligent robot developed to increase children's creative thought processes by interacting with them. Hence the target users of YOLO are just children but as children might not be able to create their own YOLO from the given instruction papers, parents of the children might need to use it. Therefore, children should know how to turn on and turn off YOLO and learn the basic behavior patterns that it uses to interact with them for playing with it according to their wishes. Parents should have at least enough knowledge to assemble and initialize YOLO using its API and with the help of a wifi router which is used to connect to the device and send software packages that are needed according to the wishes of users.

4.1.2 Context Diagram

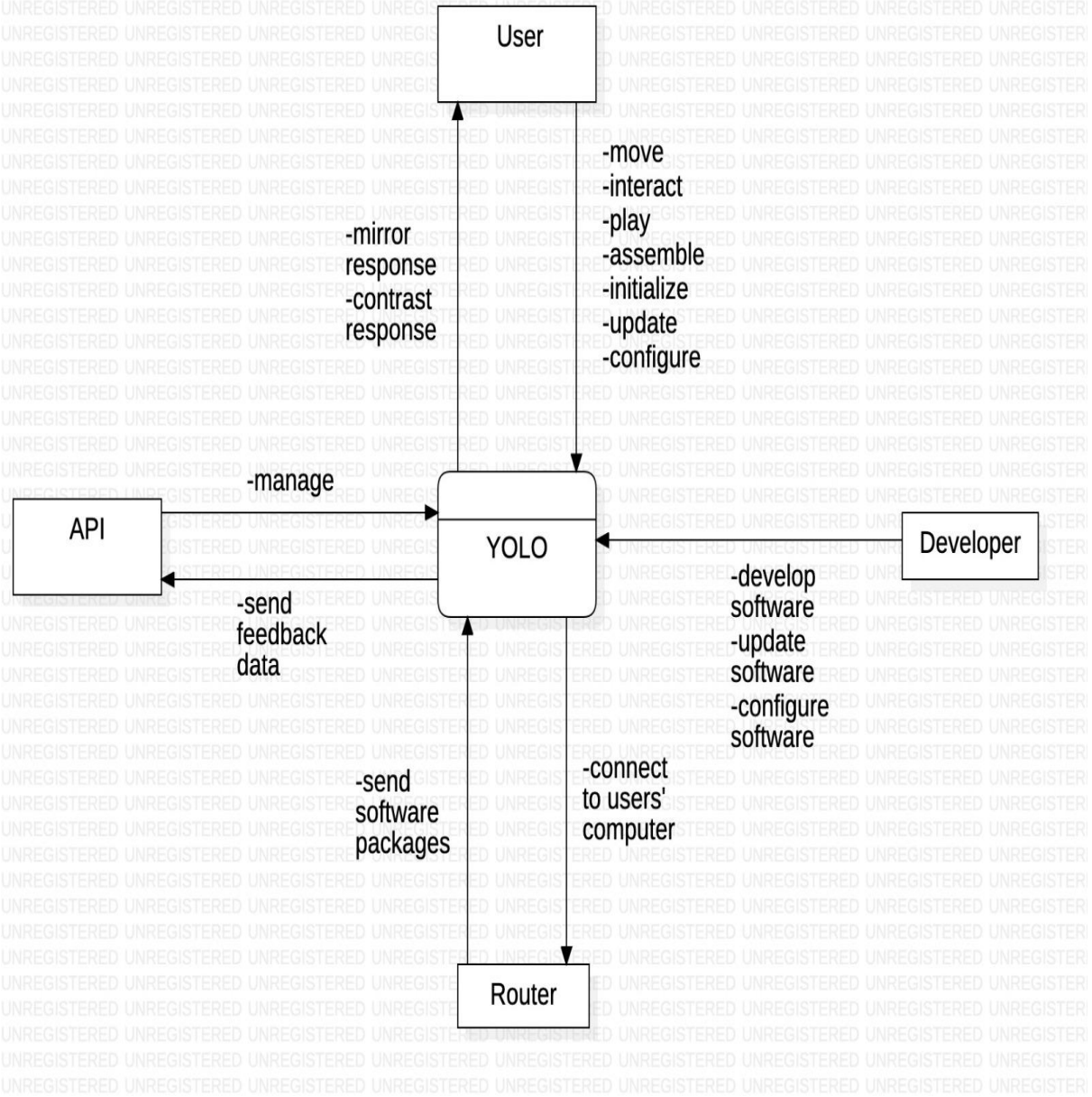

Figure 1: Context Diagram for YOLO

YOLO (Your Own Living Object) is neither a part of a large system nor a simple toy. YOLO is a social robot designed and developed to stimulate creativity in children through storytelling activities. Children use it as a character in their stories and use it as a helper to their creative thought process.

The software for the YOLO hardware should be exclusively installed on YOLO hardware. All customers shall follow the installation guide in order to install the software of their own living object's Raspberry-Pi. YOLO uses different kinds of sensors and camera management systems to get the data that is used for learning the environment and reacting to the outer world.

(YOLO Software website is https://github.com/ElsevierSoftwareX/SOFTX_2019_242 and the installation guide can be found in https://github.com/patricialvesoliveira/YOLO-Software/wiki)

4.1.3 External Interfaces

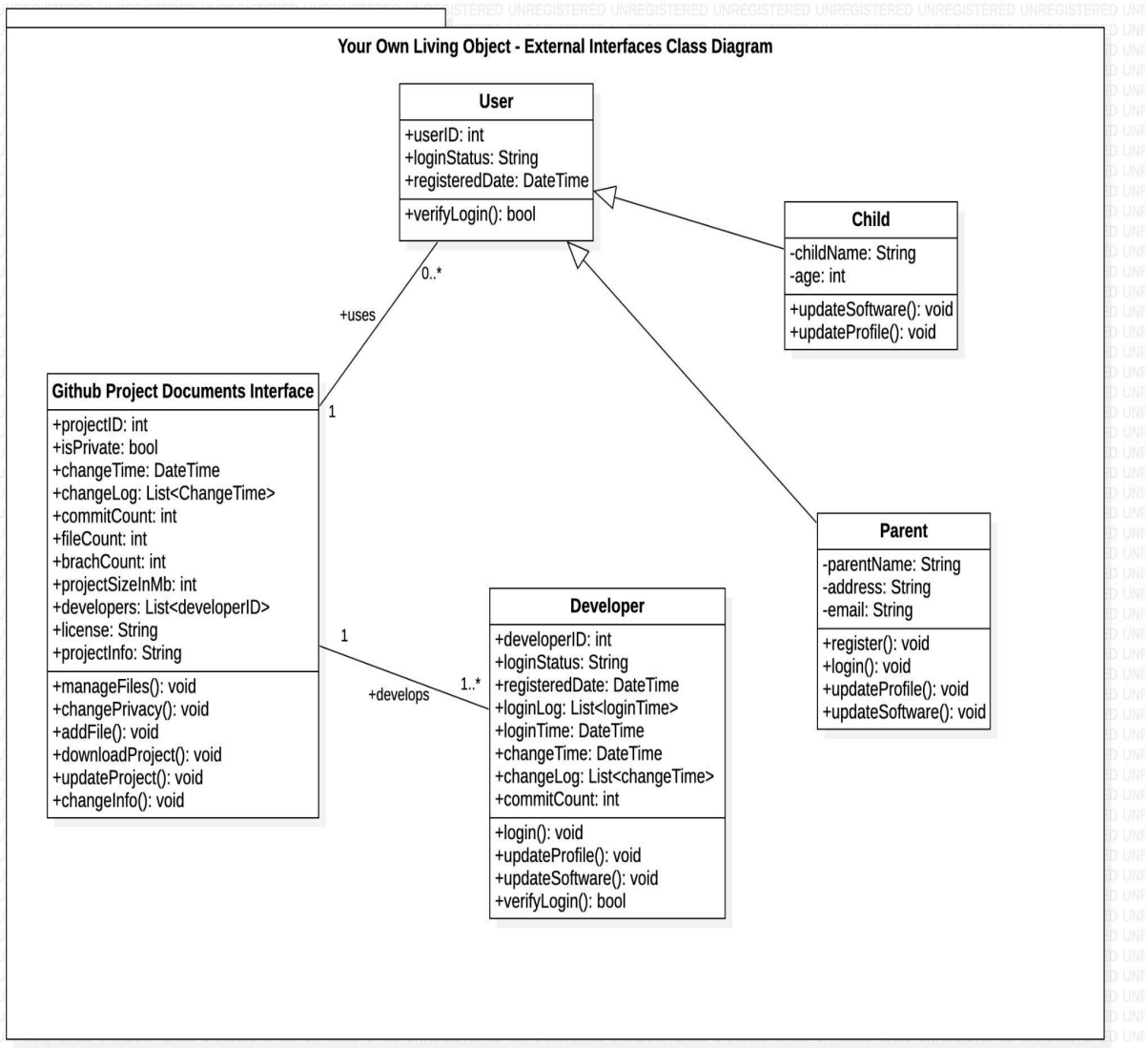

Figure 2: External Interfaces Class Diagram for YOLO

Operation	Description
verifyLogin	This operation checks whether the user or the developer is trying to login and verifies entered user information.
updateSoftware	This operation is used by parent and child to replace (update) the currently installed software on YOLO with the new software that is developed by the developer and it is used by developer to update (make changes on) the current version of the YOLO software.
updateProfile	This operation is both used by developer and user (parent and child) to change their profile information that is created to identification.
register	This operation is used by the parent (user) to register to the YOLO system and create a profile for both him/herself and his/her child.
login	This operation is used by users and developers to get into the YOLO system to make changes on their profile or on the system software.
manageFiles	This operation is used by the Github Project Document Interface (GPDI) to manage existing project files.
changePrivacy	This operation is used by the Github Project Document Interface (GPDI) to change the privacy of existing project files in order to decide who will have access to see the project files.

addFile	This operation is used by the Github Project Document Interface (GPDI) to add new files to current project files.
downloadProject	This operation is used in the Github Project Document Interface (GPDI) to download project files in order to install necessary software to YOLO.
changeInfo	This operation is used by the Github Project Document Interface (GPDI) to change the information page provided by developers to give users information about the YOLO project.

Table 3: Operation Descriptions for External Interfaces Class Diagram

4.1.4 Interaction scenarios

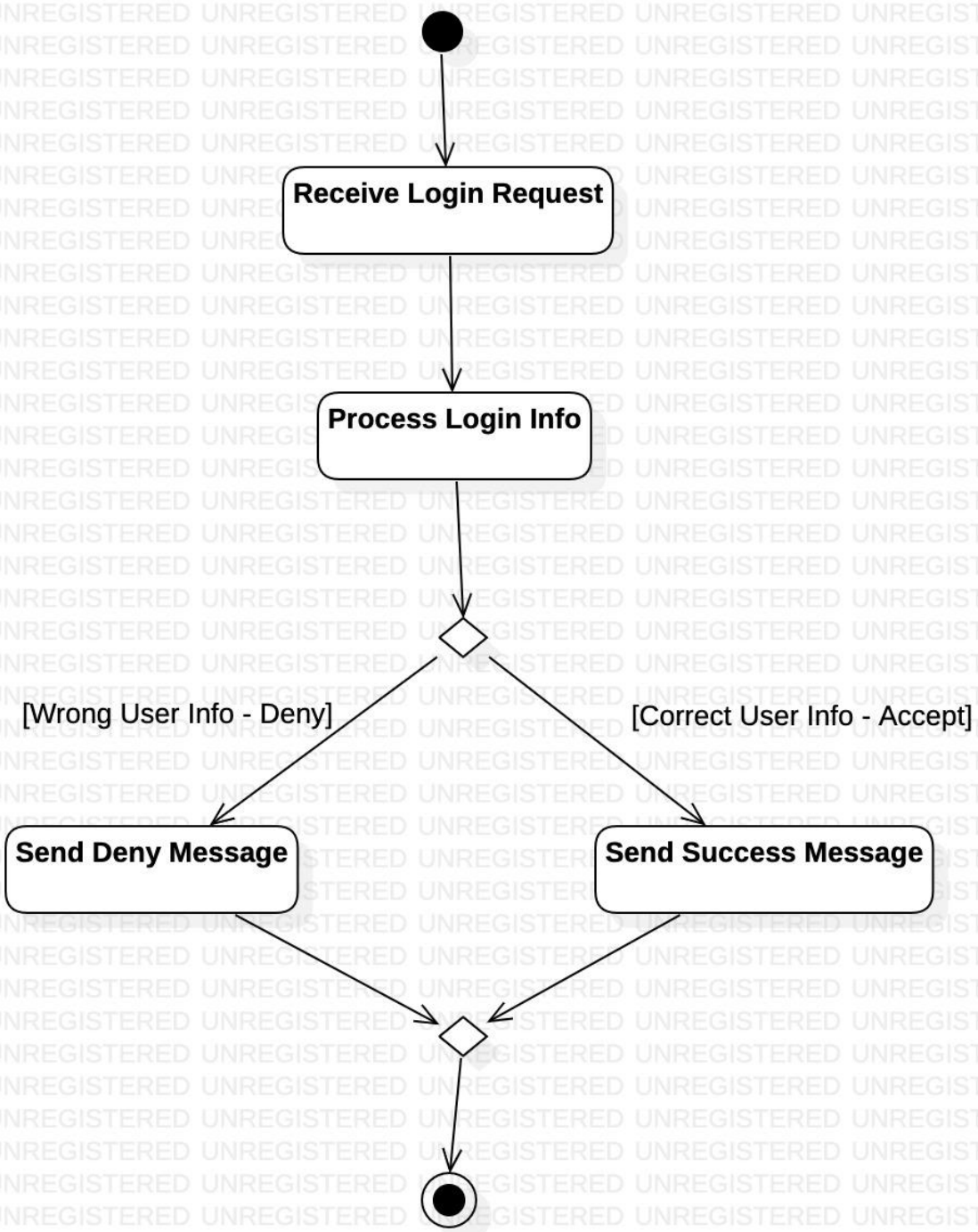

Figure 3: Login Process Activity Diagram

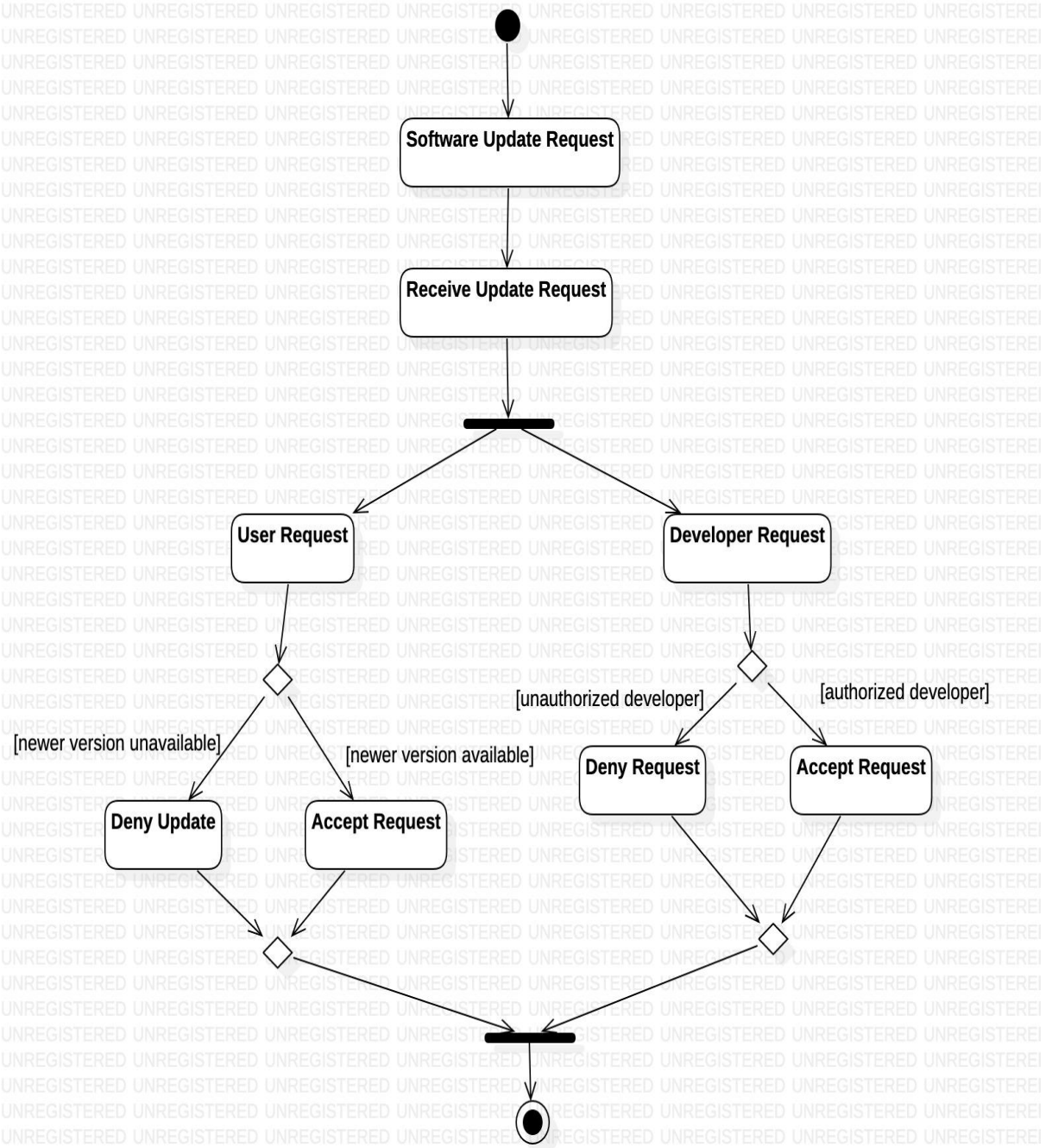

Figure 4: Software Update Activity Diagram

4.2 Functional View

In this view, components and interfaces between them are presented in a high level view. Detailed explanations are given in the related sections.

4.2.1 Stakeholders' use of this view

The runtime functional pieces of the system, their roles, interfaces, and major interactions are described. Other system structures, such as the component structure are shaped by it. It also has a substantial impact on the system's quality properties, such as its capacity to modify, secure, and execute at runtime.

4.2.2 Component Diagram

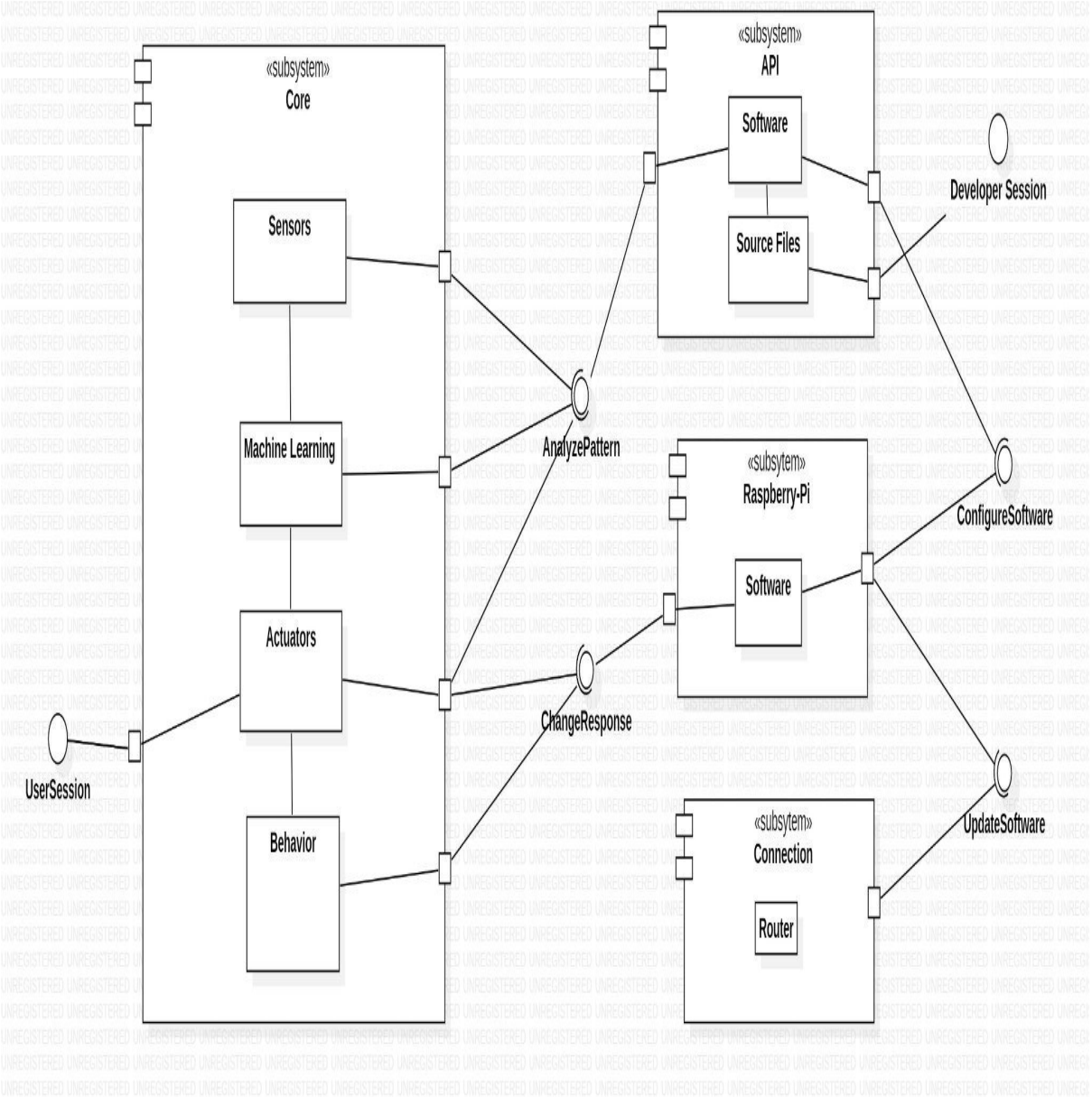

Figure 5: Component Diagram of YOLO

- In the Core component there are 4 parts. Namely sensors, machine learning, actuators and the behavior.
- Core component mainly senses the motion and analyzes the patterns and creates behaviors accordingly.
- API components consist of 2 parts, which are software and the source files .
- Users or developers use API to create new configuration files or update the software.
- Raspberry-Pi component has only one part which is software.
- It is used for changing the software or updating it.
- Connection component has only the router.
- It ıs used for connecting YOLO to the API.
- To analyze patterns and change the software API needs core. Therefore, AnalyzePattern is provided as an interface for core.
- To configure software and sending it to YOLO, API needs Raspberry-Pi. Hence, ConfigureSoftware is a required interface for the API.

4.2.3 Internal Interfaces

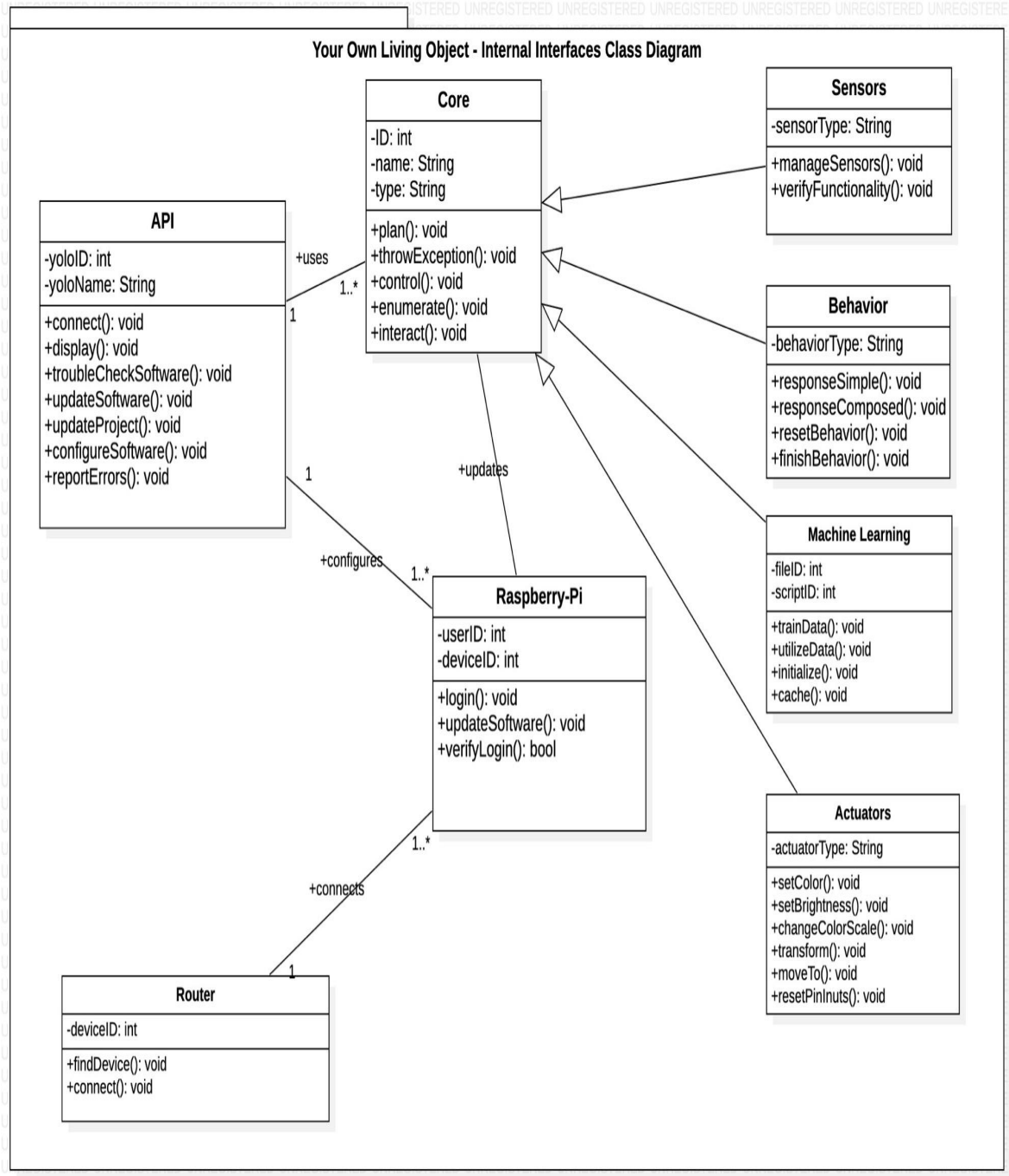

Figure 6: Internal Interfaces Class Diagram for YOLO

Operation	Description
verifyLogin	This operation checks whether the user or the developer is trying to login and verifies entered user information.
updateSoftware	This operation is used by parent and child to replace (update) the currently installed software on YOLO with the new software that is developed by the developer and it is used by developer to update (make changes on) the current version of the YOLO software.
connect	This operation is used to connect the API to Raspberry-Pi.
display	This operation is used user to see interface
troubleCheckSoftware	This operation is used to check if there is any trouble with the software.
Configure Software	This operation is used by developers or users to configure software.
reportErrors	This operation is used to report errors.
plan	This operation is used to plan Core parts to work in harmony.
throwException	This operation is used to show errors in the Core.
control	This operation is used to control Core parts.
enumerate	This operation is used to enumerate operations in the Core components' parts.
interact	This operation is used to make the connection between Core parts.
manageSensors	This operation is used to control sensors.

verifyFunctionality	This operation is used to check whether the sensors are functional or not.
responseSimple	This operation is used to give a simple response.
responseComposed	This operation is used to give a composed response.
resetBehavior	This operation is used to reset the behavior.
finishBehavior	This operation is used to finish behavior.
trainData	This operation is used to train data for analyzing patterns.
utilizeData	This operation is used to create responses according to analyzed data.
initialize	This operation is used to start machine learning and accordingly start analyzing patterns.
cache	This operation is used to keep the data that has been used lately.
setColor	This operation is used to set the color of Leds.
setBrightness	This operation is used to set the brightness of Leds.
changeColorScale	This operation is used to change the color scale of the Leds.
transform	This operation is used to learn the position and the angle of the YOLO.
moveTo	This operation is used to move YOLO to the wanted location.
resetPinInputs	This operation is used to reset pin inputs.

Table 4: Operation Descriptions for Internal Interfaces Class Diagram

4.2.4 Interaction Patterns

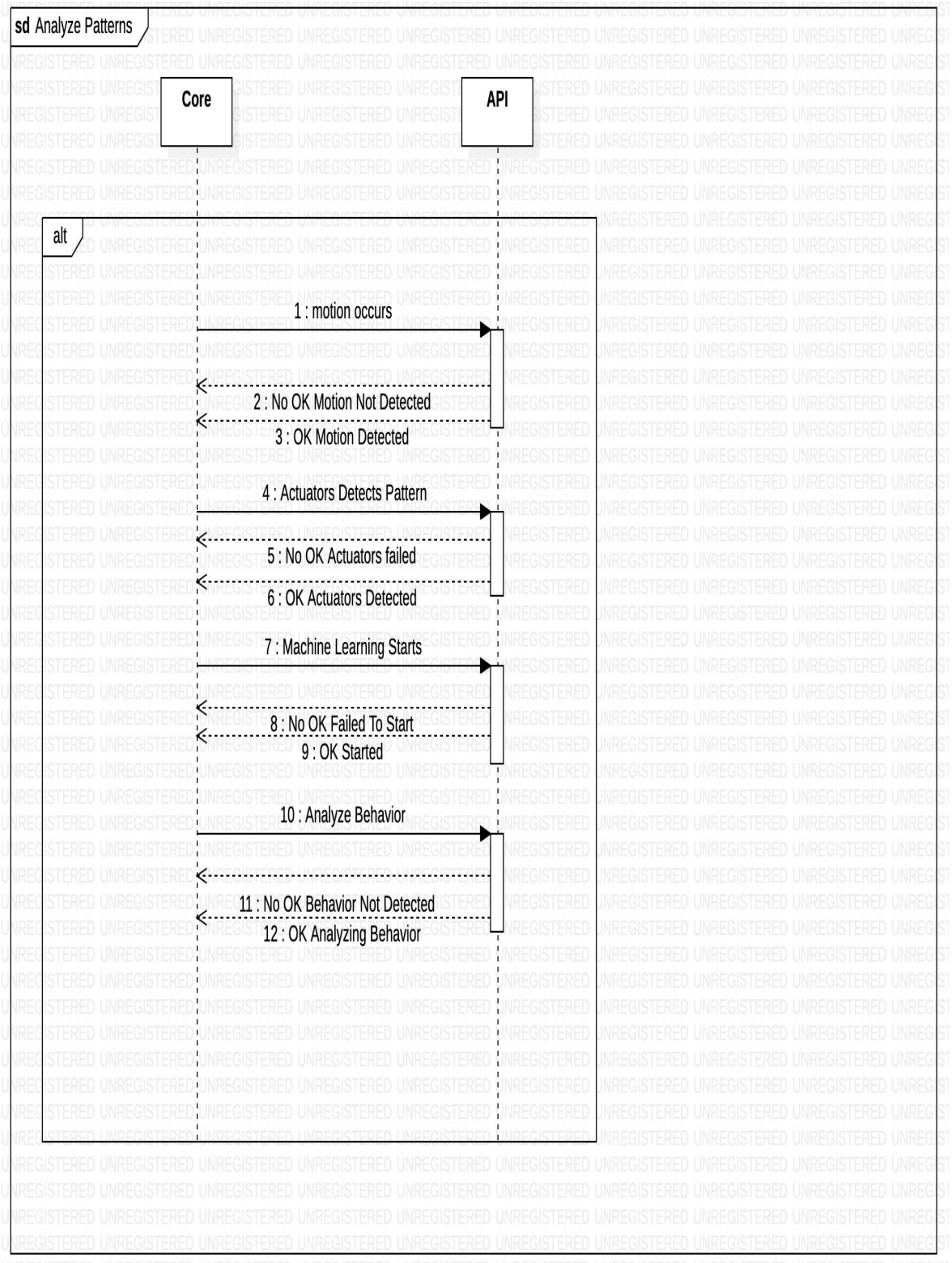

Figure 7: Analyze Patterns Sequence Diagram

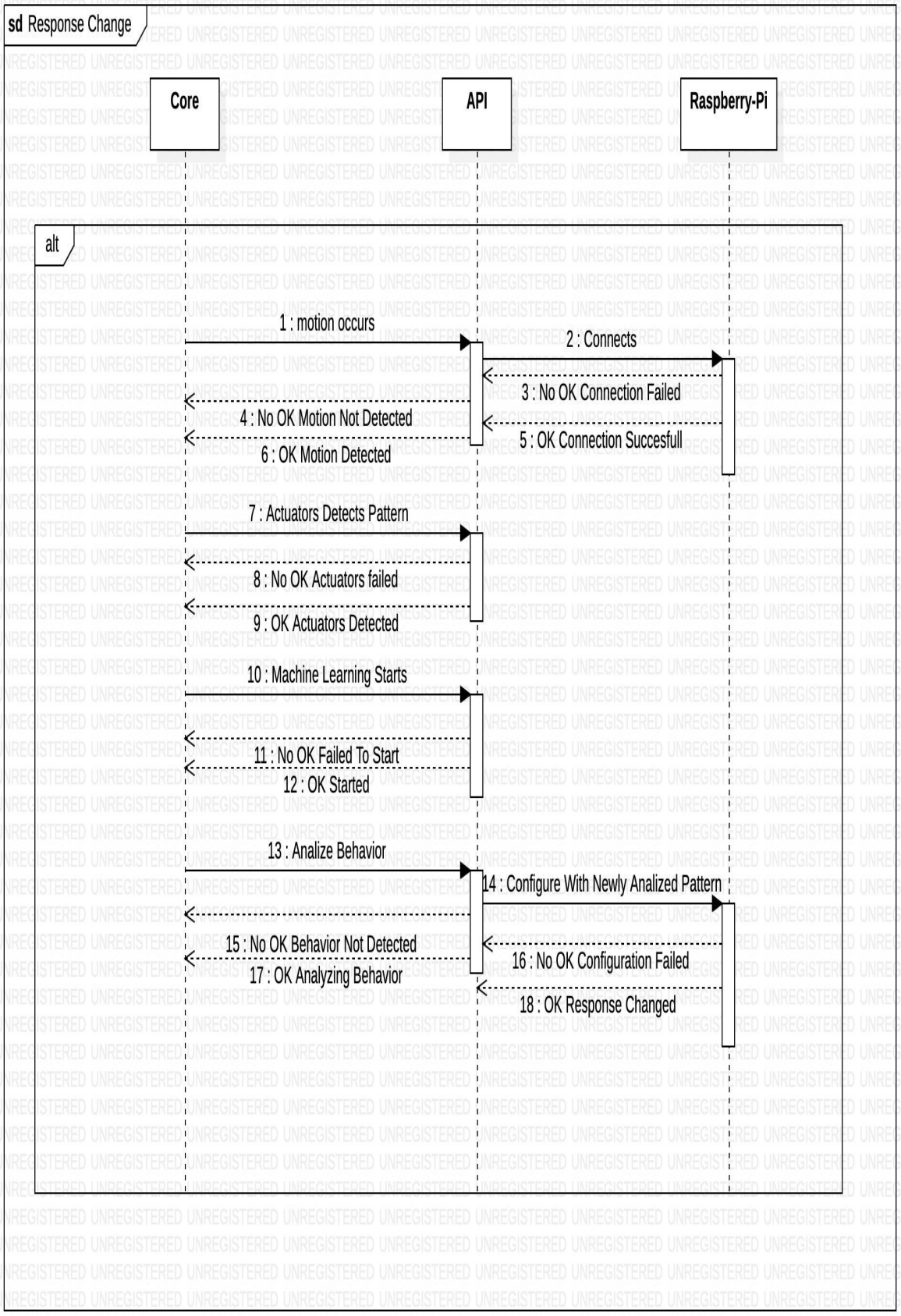

Figure 8: Response Change Sequence Diagram

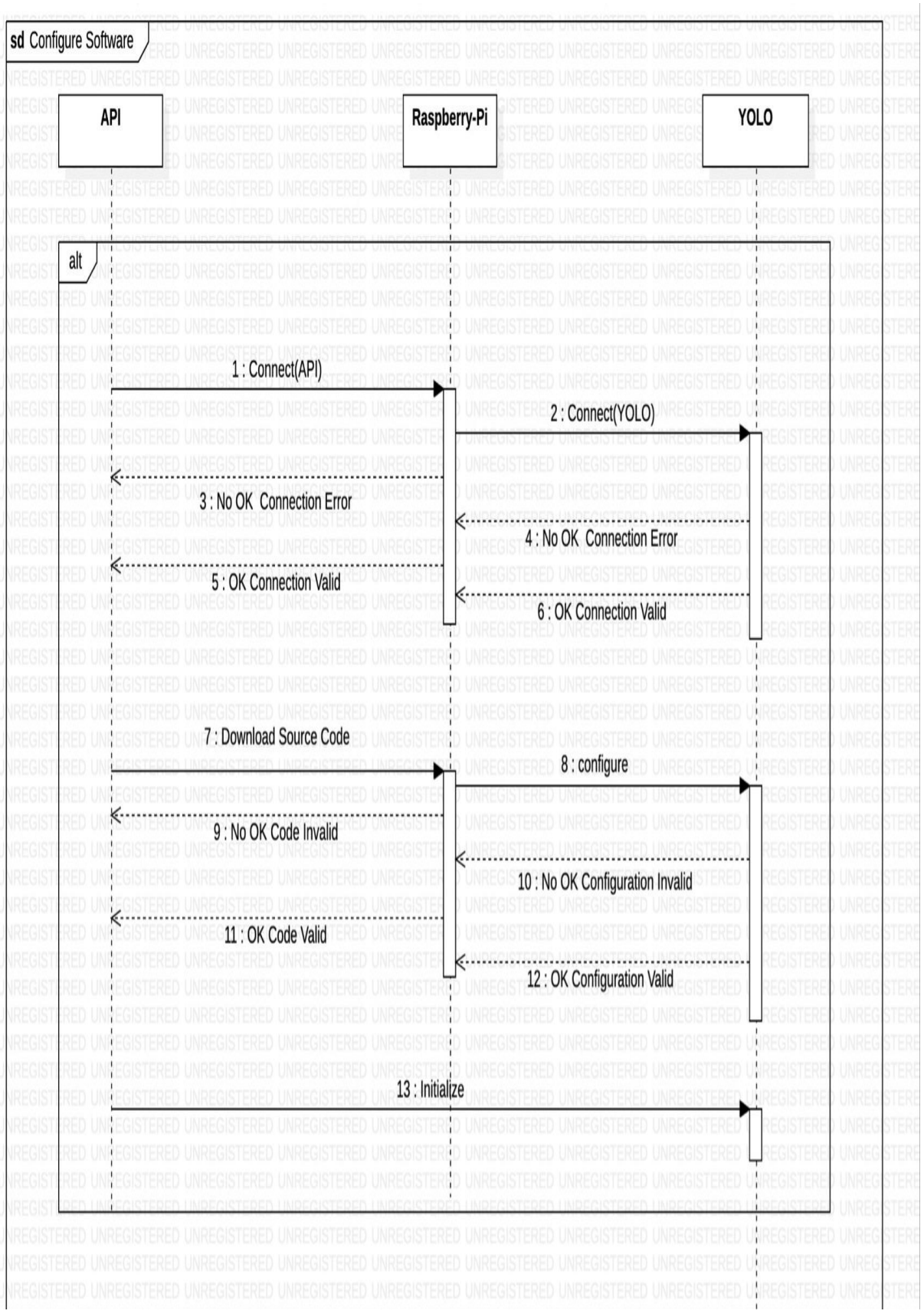

Figure 9: Configure Software Sequence Diagram

4.3 Information View

The system, as well as the relationships between the data to be stored and the system's activities to produce, utilize, alter, and remove the data, will be detailed in this view. The consequences of system actions on data will also be investigated in terms of effect type: create, read, update, and delete.

4.3.1 Stakeholders' use of this view

This viewpoint shows to stakeholders, how the architecture stores, manipulates, maintains, and distributes data. In addition, it creates a comprehensive but high-level picture of static data structure and information flow.

4.3.2 Database Class Diagram

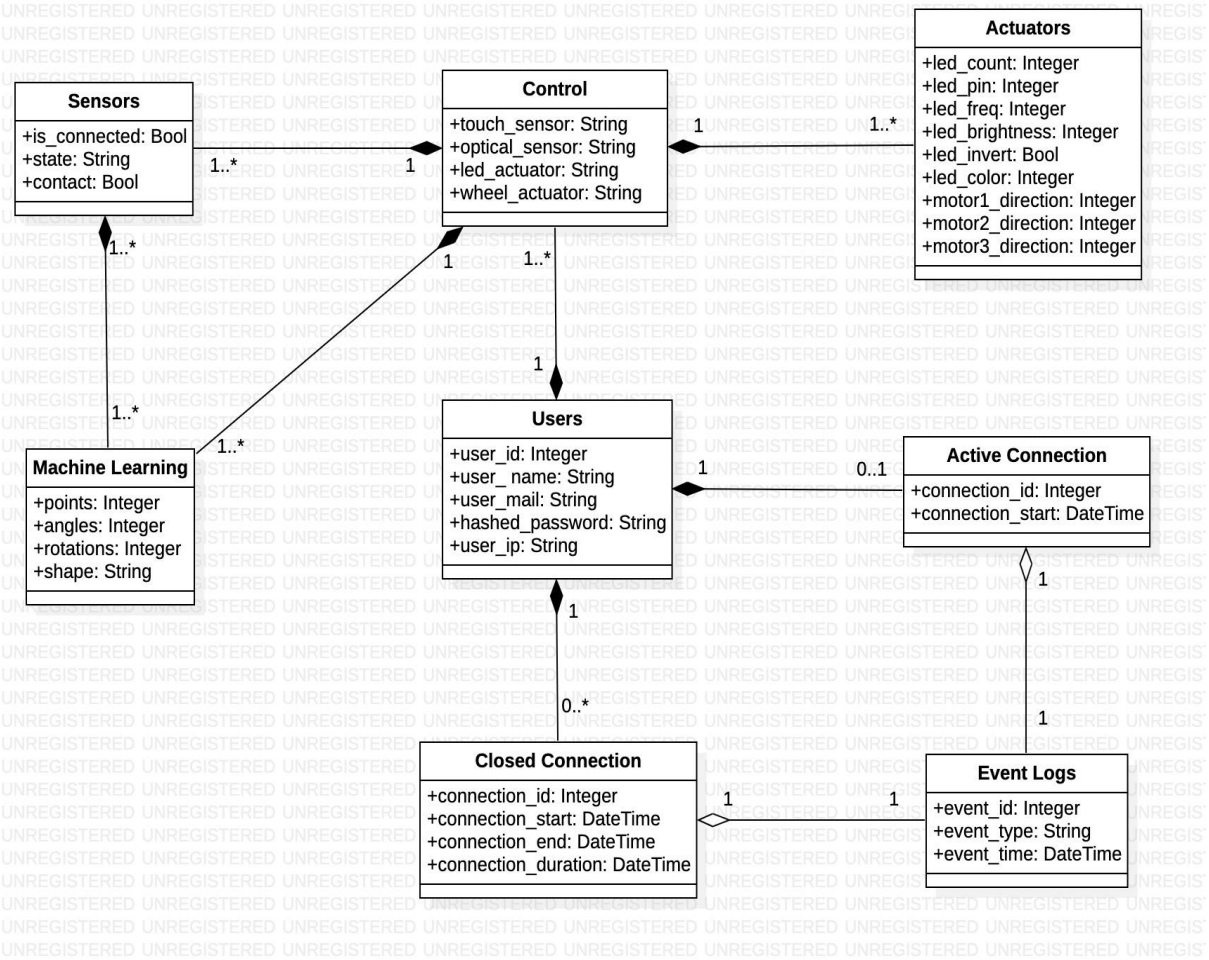

Figure 10: Database Class Diagram of YOLO

Name	Description
Sensors	Sensors are components that YOLO uses to analyze its position and surroundings.
Machine Learning	It is used by YOLO to decide on how to behave to the user
Actuators	They are the managing components of the YOLO.
hashed_password	It is the password generated by processing the original password with hashing algorithms.
points	Points are the actual positioning points of the YOLO which are used to perform ML tasks.
shape	Shape is the approximate shape of the YOLO's route which is performed by playing with it by the user.
angles	Angles are the approximate angels of the YOLO's moving directions which are performed by playing with it by the user.
rotations	Rotations are the approximate rotations of the YOLO's moving directions which are performed by playing with it by the user.
motor1_direction - motor2_direction - motor3_direction	They are the directions that the YOLO will move to when the motors launch.

Table 5: Description Table for the Database Class Diagram

4.3.3 Operations on Data

Below table describes the operations on data.

Operation	**CRUD Operations**
getUser	Create: - Read: Users, user_id Update: - Delete: -
getUserAction	Create: - Read: Users, event_id Update: - Delete: -
newUser	Create: User Read: - Update: Users Delete: -
developerLogin	Create: - Read: Developer Update: - Delete: -
connectDevice	Create: connection_id Read: - Update: connection_start Delete: -
getActuator	Create: - Read: Actuators Update: - Delete: -

disconnectDevice	Create: - Read: connection_id Update: connection_duration Delete: connection
getSensors	Create: - Read: Sensors Update: - Delete: -
getControlData	Create: - Read: Control Update: - Delete: -
getActiveConnection	Create: - Read: connection_id, connection_start Update: - Delete: -
deleteUser	Create: - Read: user_id Update: Users Delete: User
getSensorStatus	Create: - Read: touch_sensor, optical_sensor, led_actuator, wheel_actuator Update: - Delete: -

Table 6: CRUD Operations

4.4 Deployment View

In this viewpoint, the environment in which the system will be deployed, as well as the system's dependencies on various parts of that environment are shown.

4.4.1 Stakeholders' use of this view

With the help of deployment view, users and developers can see the hardware environment that the system requires, as well as the technical environment requirements for each element and the mapping of software elements to the runtime environment in which they will be executed.

4.4.2 Deployment Diagram

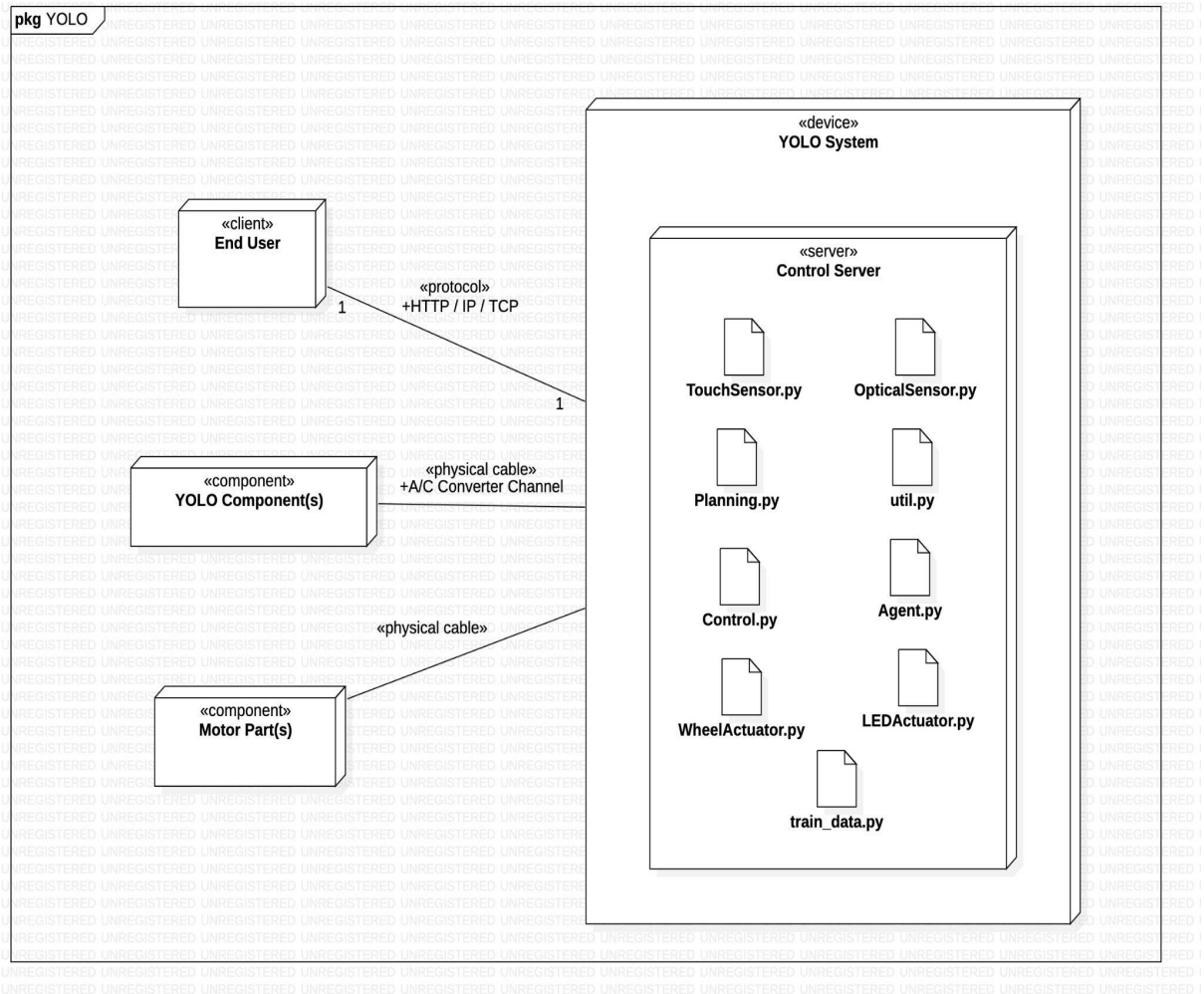

Figure 11: Deployment Diagram

- Python is used in the YOLO System's Control Server: YOLO Controller, Sensor Scripts, Machine Learning Scripts, Actuator Scripts are all written in Python.
- Connection between End User (Client) Server and YOLO System is provided via HTTP, IP and TCP protocols. HTTP is the protocol for requests and responses. IP is used for broadcasting of YOLO, and it is also the main interface of it. For the HTTP requests and responses, the data is transferred with TCP packages in order to prevent possible data losses.
- YOLO components are connected to the YOLO System and the Control Server via a physical cable which carries analog electrical signals. These analog signals are then converted into digital signals in the Control Server.
- YOLO motors are connected to the YOLO System and the Control Server via a physical cable for sending necessary commands to motors in order to make YOLO do the needed taks.

4.5 Design Rationale

- Relationships and dependencies in the YOLO system are visible with the help of this point of view. Therefore, understanding responsibilities of the components like API, Core and Raspberry-Pi is facilitated.
- If an error occurs in any part of the processes, it will be reported back to the API. With the help of this viewpoint, error handling is easy and doable always.
- Every entry that is inserted to the database will be kept for future investigation, in other words, developing, and making research would be much easier.
- Seeing the requirements for YOLO to work functionally, eases the assembling of YOLO and configuring the software.