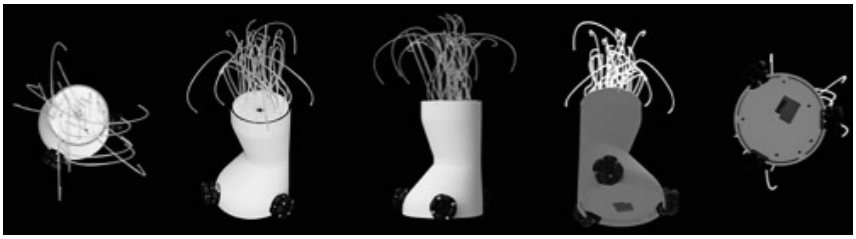

Software Requirements Specification
Your Own Living Object(YOLO)

Göktuğ Ekinci, 2380343
Hazal Güvenkaya, 2380509

15.04.2022

Contents

List of Figures

List of Tables

Revision History

Date	Reason For Changes	Version
08.04.2021	Draft Version	1.0

Table 1: The Revision History of Software Requirements Specification Document

1 Introduction

This document is the (SRS)Software Specification Requirement of Your Own Living Object(YOLO) robot. YOLO is a robot designed and created for social activities. The robot is a storytelling friend for the children to develop their social skills. It brings a new aspect to children's learning and growing journey. Artificial intelligence and several other technologies had been used in the development of YOLO. Since it is a social robot, psychological theories and techniques are often used to develop YOLO. The original paper can be found below:
`https://dl.acm.org/doi/abs/10.1145/3371382.3378395`

1.1 Purpose of the System

The purpose of the project is to bring a new aspect to a child's growing journey with the help of interactive technologies, such as social robots. Social robot mainly aims to benefit children's creativity during playtime.

1.2 Scope

In the scope of YOLO, users can build their own living object YOLO by 3D printing and download the software to the Raspebry Pis' connected to the YOLO. Also, the software is customizable by the community.

YOLO has been used as a research tool for STEAM activities to encourage robotics among young children and parents. It offers a low-purchase and low-maintenance cost robot that supplies the open-source hardware thus provides opportunities for researchers with and without engineering knowledge to produce this robot and further use it to target their own research goals without relying upon complicated robotic platforms.

It is affordable for anyone since a 3D printer can produce it. In addition to that, by being an easy to use tool, educators and parents have access to a robot that is easy to prepare, differing from other existing technological tools that can be unmanageable for non-experts to prepare, and contextual design was carefully designed to involve children in different design stages of the robot.

1.3 System Overview

This document section will provide explicit information about the system, including all elements.

1.3.1 System Perspective

YOLO is not a part of a more extensive system, but it interacts with modules like sensors, actuators, and agents. YOLO uses implicit interaction modalities to communicate with children, such as movements and lights, to maintain playful and imaginative interchanges with children.

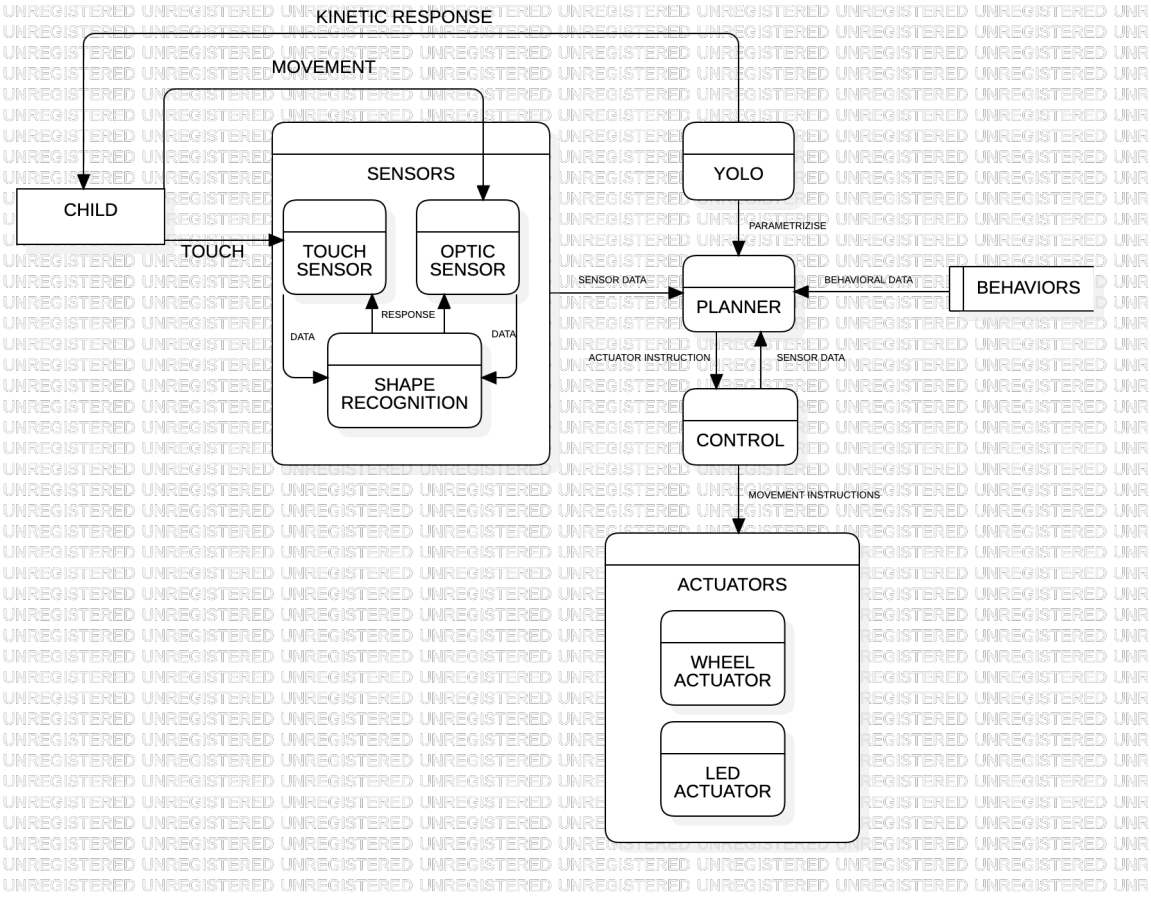

Figure 1: Context Diagram for YOLO

1.3.1.1 Hardware Interfaces

Sensors: There are used two kinds of sensors. Touch sensors are used for recognizing when the robot is being touched, and optical sensors are used to determine the direction of the play

patterns of children while operating the robot.

Actuators: There are used two kinds of actuators, wheel and led actuators. Wheel actuators are used to provide navigation to the robot, and led actuators are used to provide lights to the robot.

1.3.1.2 Software Interfaces

Control: Extract data into a programmable format and demands actuators.

Planner: Schedules and performs behaviors.

Behavior: Behaviors are classified as composed and simple behaviors. Simple behaviors are the ones that directly use the actuator data, and composed behaviors are the ones that unite several simple ones.

1.3.1.3 Communications Interfaces

Lights and movement were chosen as the primary interaction modalities between the robot and actors as this combination was identified as an efficient nonverbal multi-modal communication way.

1.3.1.4 Memory Constraints

Memory is not a huge issue for YOLO. However, system should have enough memory to sustain basic operations.

1.3.1.5 Operations

There are two operations of YOLO. One of them is moving and other one is turning on the lights.

1.3.2 System Functions

Function	Summary
Download the Software	User can download the software.
Install the Software	User can install the software to the Raspberry Pi.
Modify Movements & Reactions	User can customize the movements and reactions in a desired way.
Move the YOLO	User can play with YOLO to trigger the behaviors by moving it.
Create Storyline	User can play with YOLO to trigger the behaviors by creating a storyline.
Activate Puppeteer Mode	User can activate pupeteer mode when YOLO detects user's physical contact.

Table 2: Product Functions

1.3.3 Stakeholder Characteristics

Users Characteristics: The target users of YOLO is mainly children who are at their growing age, and secondly their parents. According to researches, creativity is an ability that can be nurtured if stimulated, and the main target of YOLO is the children who are open to increasing their creativity by using social robots as easy to use toys to be included in children's spaces, such as schools, with the primary goal of creative growth via play.

Researchers Characteristics: The target researchers are the ones that want to use YOLO's API to study child–robot interaction. The researcher doesn't need to have the engineering knowledge to construct this robot and further use it to target their research goals without relying upon complicated robotic platforms. They basically work with children as co-researchers to support sharing, gathering, and analyzing data from their exercise during robot usage.

1.3.4 Limitations

- **Regulatory Requirements and Policies:** These policies are very localized and thus vary per institution (e.g., school district, university, specific school policies). Privacy and confidentiality require embracing alternative approaches for data collection that protect a child's identity.
- **Hardware Limitations:** A stable and adequate connection is needed to connect the system. Also, there should be enough battery power, memory, and storage space on the hardware side.
- **Audit Functions**: There is no audit limitation.
- **Control Functions:**Only system admins can regulate and configure servers. Users are not authorized to change server settings.
- **Higher-Order Language Requirements**: Users can use python to regulate the hardware of the system.
- **Criticality of the Application**: Failures could cause adverse effects on child development. Since YOLO is used for to expand the imagination of children and contribute to their development, failures should be solved as soon as possible.
- **Safety and Security Considerations:** No safety guidelines for personal fabrication have been formally specified for YOLO, and misusages have been regarded as users' responsibility.
- **Physical/mental considerations:** There is no such limitation on this subject.

1.4 Definitions

- **YOLO:** Your Own Living Object.
- **API:** Application Programming Interface, a software intermediary that authorizes two applications to communicate to each other.
- **STEAM:** Science, Technology, Engineering, Art, and Math. It's a common acronym to reference these essential subjects and skills in the education world.

2 References

This document is written with respect to IEEE 29148-2011 standard:
29148-2011 - ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering.

[1] Alves-Oliveira, Patrícia & Gomes, Samuel & Chandak, Ankita & Arriaga, Patricia & Hoffman, Guy & Paiva, Ana. (2020). Software architecture for YOLO, a creativity-stimulating robot. 11. 100461. 10.1016/j.softx.2020.100461.

[2] Alves-Oliveira, Patrícia & Arriaga, Patricia & Paiva, Ana & Hoffman, Guy. (2019). Guide to build YOLO, a creativity-stimulating robot for children. HardwareX. 6. e00074. 10.1016/j.ohx.2019.e00074.

[3] Alves-Oliveira, Patrícia & Arriaga, Patricia & Paiva, Ana & Hoffman, Guy. (2021). Children as Robot Designers. 399-408. 10.1145/3434073.3444650.

3 Specific Requirements

3.1 External Interfaces

Following class diagram represents the relationship between interfaces and their functionalities. For explanation of interfaces, please refer to section 1.3.1.2.

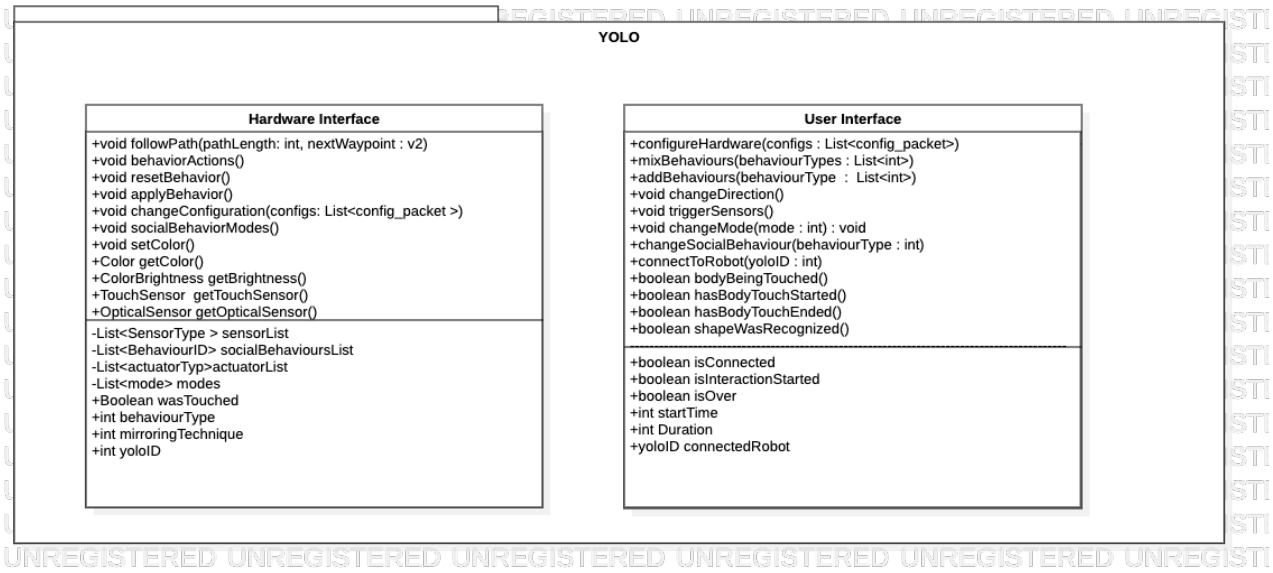

Figure 2: External Interface Class Diagram for YOLO

3.2 Functions

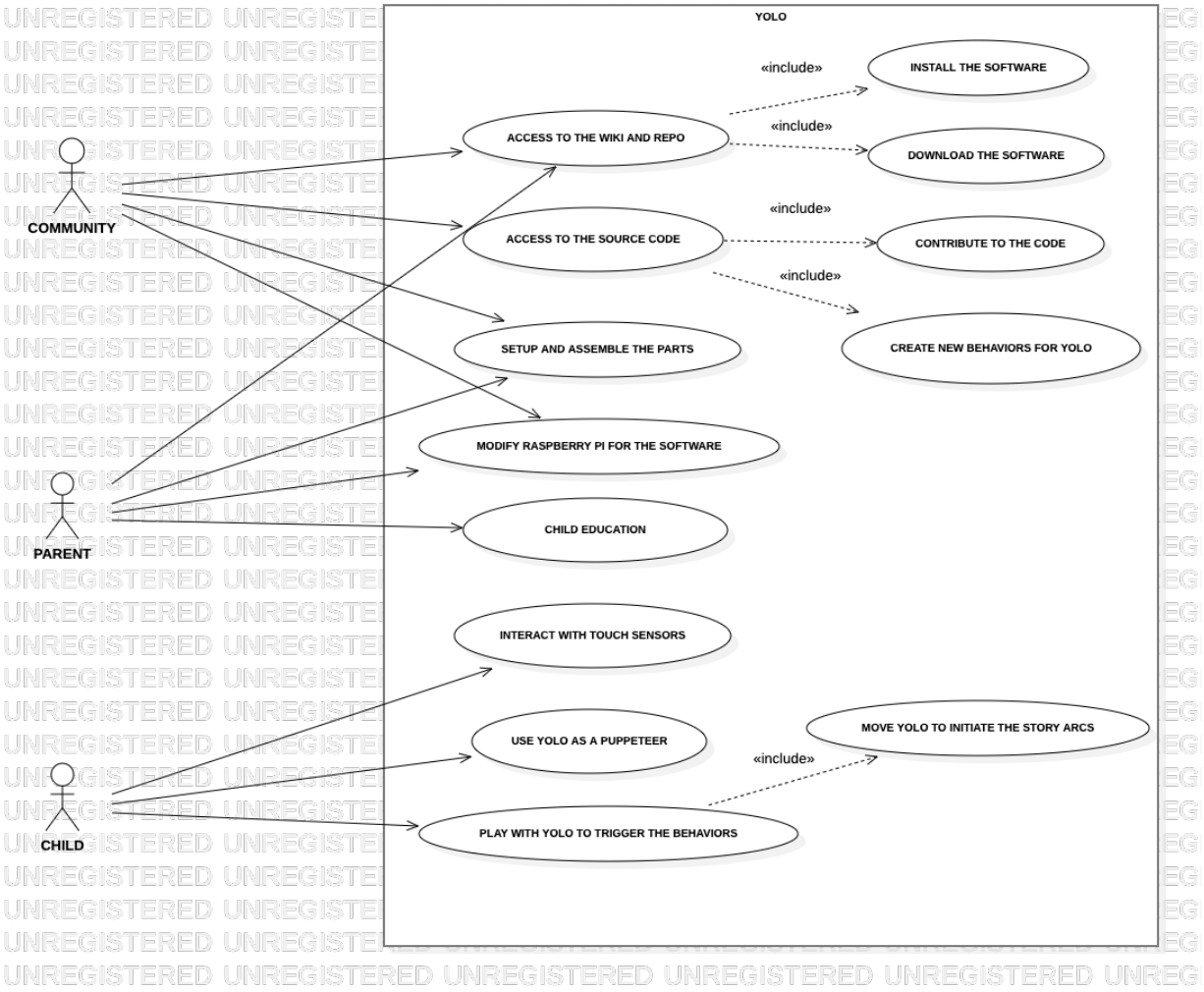

Figure 3: Use-Case Diagram for YOLO

Use Case ID	1
Use-Case Name	Access to the Wiki and Repo
Actors	Community, Parents
Description	The user can access to the repository which can see the the software and the wiki about the project YOLO.
Pre-conditions	A proper internet connection.
Post-conditions	-
Basic Flow	Step 1 – The user goes to the www.github.com Step 2 – The user finds the repository that includes wiki and the code. Step 3 – The user can search through tabs for what they need.
Exceptions	Bad internet connection.

Table 3: Access to the Repo

Use Case ID	2
Use-Case Name	Download the Software
Actors	Parents, Community
Description	The user downloads the software from the repository where the project lies.
Pre-conditions	The user must have access to the repository. A working machine with memory.
Post-conditions	-
Basic Flow	Step 1 – The user accesses to the repository. See: 3 Step 2 – Switch to the tab that software can downloadable. Step 3 – Download the software to the machine.
Exceptions	Machine's storage could be full.

Table 4: Download the software

Use Case ID	3
Use-Case Name	Install the Software on the Raspberry Pi
Actors	Parents, Developers
Description	User can install the software to the machine that YOLO is going to work on.
Pre-conditions	A raspberry pi that works properly. Software must be downloaded from the repository to install.
Post-conditions	A raspberry pi that can store and run the software properly.
Basic Flow	Step 1 – Download the software. See the 4. Step 2 – Set up the software.
Exceptions	Requested dependencies might not work properly on the machine.

Table 5: Install the Software on the Raspberry Pi

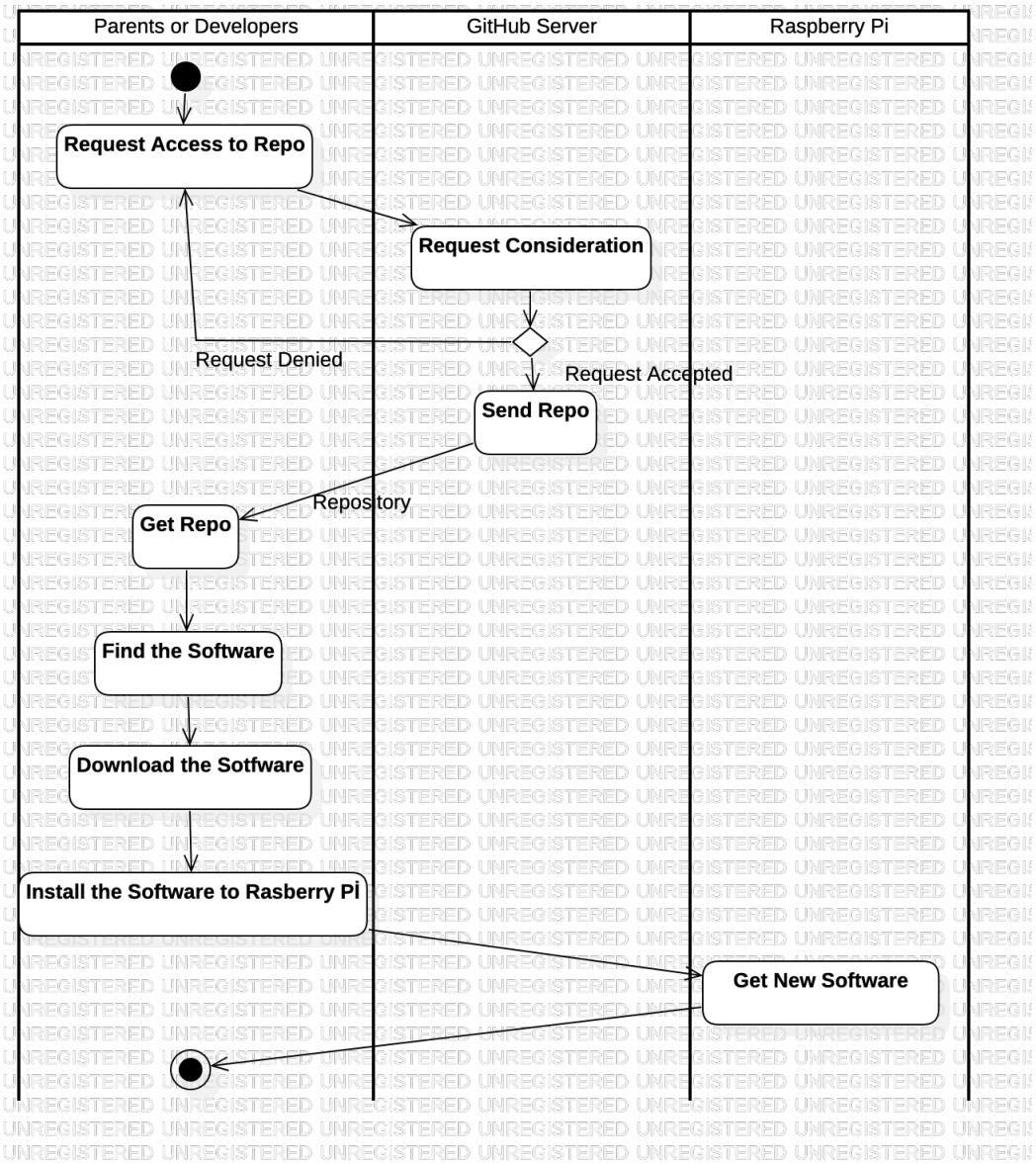

Figure 4: Install the Software on the Raspberry Pi

Use Case ID	4
Use-Case Name	Access to the Source Code
Actors	Community
Description	By accessing to the repository in GitHub. Users can see the source code itself.
Pre-conditions	The user must access to the access to the repository
Post-conditions	-
Basic Flow	Step 1 – Access to the repo. See: 3 Step 2 – The user switches to the code tab Step 3 – The user goes into the source file folders. Step 4 – The user can see the source code of the project YOLO.
Exceptions	Connection to microscope may not be established, if client fails.

Table 6: Connect to Server

Use Case ID	5
Use-Case Name	Contribute to the source code
Actors	Community
Description	The user contributes to the source code of the project YOLO.
Pre-conditions	Valid access on the code. Good software knowledge. A written code that might improve or change the project YOLO.
Post-conditions	Code must be compatible to the current source code.
Basic Flow	Step 1 – The user observes and understands the code of the project YOLO. Step 2 – The user writes a code that can contribute to the current source code. Step 3 – Contributed code can be run.
Exceptions	The user's code might not be compatible. The user's code might corrupt some aspects of the code.

Table 7: Contribute to the source code

Use Case ID	6
Use-Case Name	Create New Behaviors for YOLO
Actors	Community
Description	Users can create and add new behaviors to YOLO.
Pre-conditions	Valid access on the code.
Post-conditions	New behavior must be suitable for behavioral structure.
Basic Flow	Step 1 – The user may create a new behavior. Step 2 – The user writes a code to add this behavior.
Exceptions	The user's code might not be compatible. The user's code might corrupt some aspects of the code.

Table 8: Create New Behaviors for YOLO

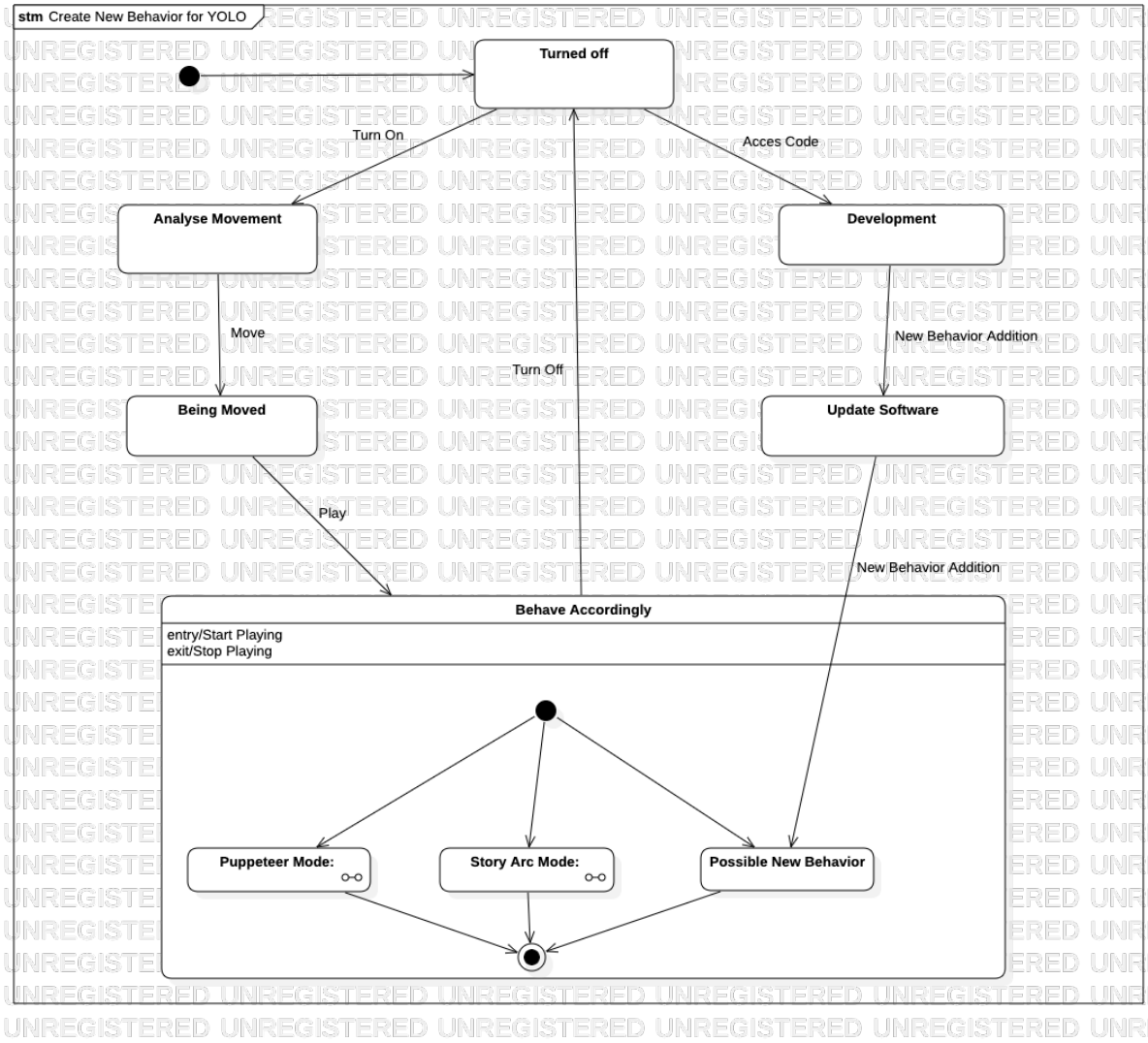

Figure 5: State Diagram for Behavior Creation Use Case

Use Case ID	7
Use-Case Name	Setup and Assemble the Parts
Actors	Parents, Community
Description	Ordering the parts from regarding sites and assembling YOLO together.
Preconditions	All the parts of the YOLO must be ready for setup and assembly. There must be no corrupted or wrong parts.
Post Conditions	The store is opened.
Basic Flow	Step 1 - Order the parts. Step 2 - Download and set the software to the machine. Step 3 - Assemble the parts of the YOLO together.

Table 9: Setup and Assemble the Parts

Use Case ID	8
Use-Case Name	Child Education
Actors	Parents
Description	YOLO is built to improve the creativity of children. Parents can use YOLO as a toy and tool for children to improve their creativity.
Preconditions	A perfectly working YOLO robot. Child, free space for the child
Post Conditions	–
Basic Flow	Step 1 – The user orders the YOLO robot. Step 2 – The user assembles the robot. Step 3 – The user sets up the required software. Step 4 – Create a free space for the child and leave them with the YOLO for child to discover their creativity.

Table 10: Child Education

Use Case ID	9
Use-Case Name	Modify Raspberry Pi for the Software
Actors	Community, Parent
Description	The user can modify Raspberry Pi for the software.
Pre-conditions	A raspberry pi that works properly.
Post-conditions	A raspberry pi that can store and run the software properly.
Basic Flow	Step 1 – The user may change the existing script. Step 2 – The new script works in the raspberry pi.
Exceptions	The new script does not works properly in the raspberry pi.

Table 11: Modify Raspberry Pi for the Software

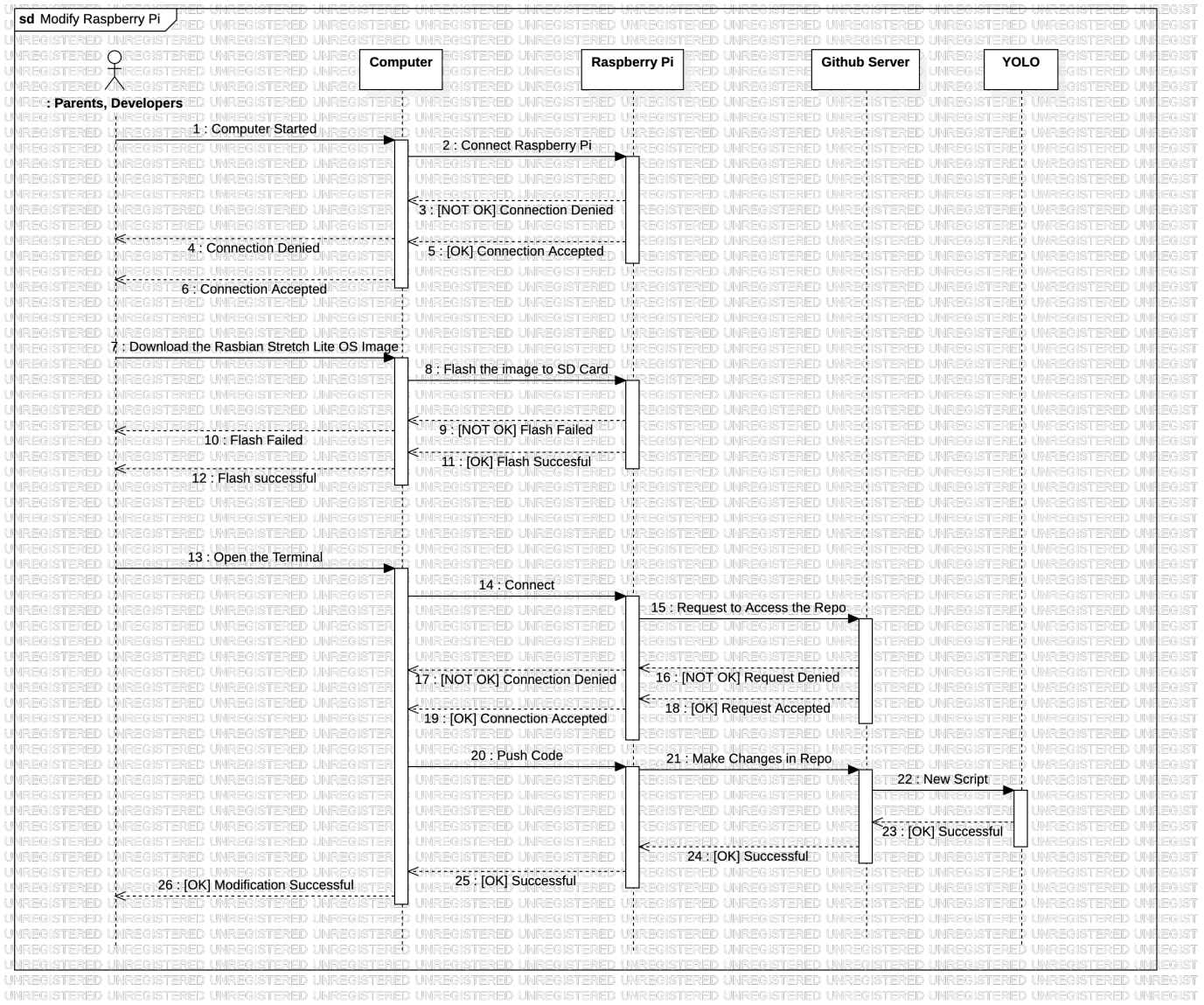

Figure 6: Modify Raspberry Pi for the Software

Use Case ID	10
Use-Case Name	Interact with Touch Sensors
Actors	Children
Description	User can touch YOLO to interact with it via touch sensors.
Pre-conditions	Properly working touch sensors.
Post-conditions	Sensors recognize the user's touch.
Basic Flow	Step 1 – User should touch YOLO Step 2 – YOLO should perceive the touch by the sensors.
Exceptions	Sensors do not recognize the user's touch.

Table 12: Interact with Touch Sensors

Use Case ID	11
Use-Case Name	Use YOLO as a Puppeteer
Actors	Children
Description	YOLO can be puppeteered by the user mimicking a traditional toy.
Pre-conditions	The user should touch the YOLO to activate puppeteer mode.
Post-conditions	The user is in full control of YOLO.
Basic Flow	Step 1 – The user grabs YOLO. Step 2 – The user maintains traditional play formats.
Exceptions	YOLO may not detect the puppeteer mode.

Table 13: Use YOLO as a Puppeteer

Use Case ID	12
Use-Case Name	Play with YOLO
Actors	Children
Description	The user can play with YOLO to trigger the behaviors.
Pre-conditions	Properly working Wheels, Charged Battery
Post-conditions	Creative playing environment for the user.
Basic Flow	Step 1 – User may move the YOLO to activate the story arcs. Step 2 – User may create a storyline for YOLO.
Exceptions	Corruption of the actuators. Corruption of the sensors.

Table 14: Play with YOLO

Use Case ID	13
Use-Case Name	Move YOLO
Actors	Children
Description	The user can move the YOLO to initiate the story arcs.
Pre-conditions	Properly working Wheels, Charged Battery
Post-conditions	-
Basic Flow	Step 1 – User should touch the YOLO to activate touch sensors. Step 2 – User may push YOLO to another place in a path.
Exceptions	Corruption of the actuators. Corruption of the sensors.

Table 15: Move YOLO

3.3 Usability Requirements

- Users must be informed properly in the wiki.
- Users should be able to download and install the software easily.
- Users must be able to update the software when it is a need.
- Users should be able to contribute to the project by ease and could be able to see the results.
- Users must be able to customize the movements or reactions of the YOLO in every way they want.
- Users should be able to see and order the required parts of the YOLO.

3.4 Performance Requirements

- The system's code will be written in Python.
- The system must analyze the movements and touches and give a respond to the user simultaneously.
- The system must be able to run without internet.
- The system must move without a problem and should not bump into objects or obstacles.
- YOLO must go through the story arc in the right moments.

3.5 Logical Database Requirements

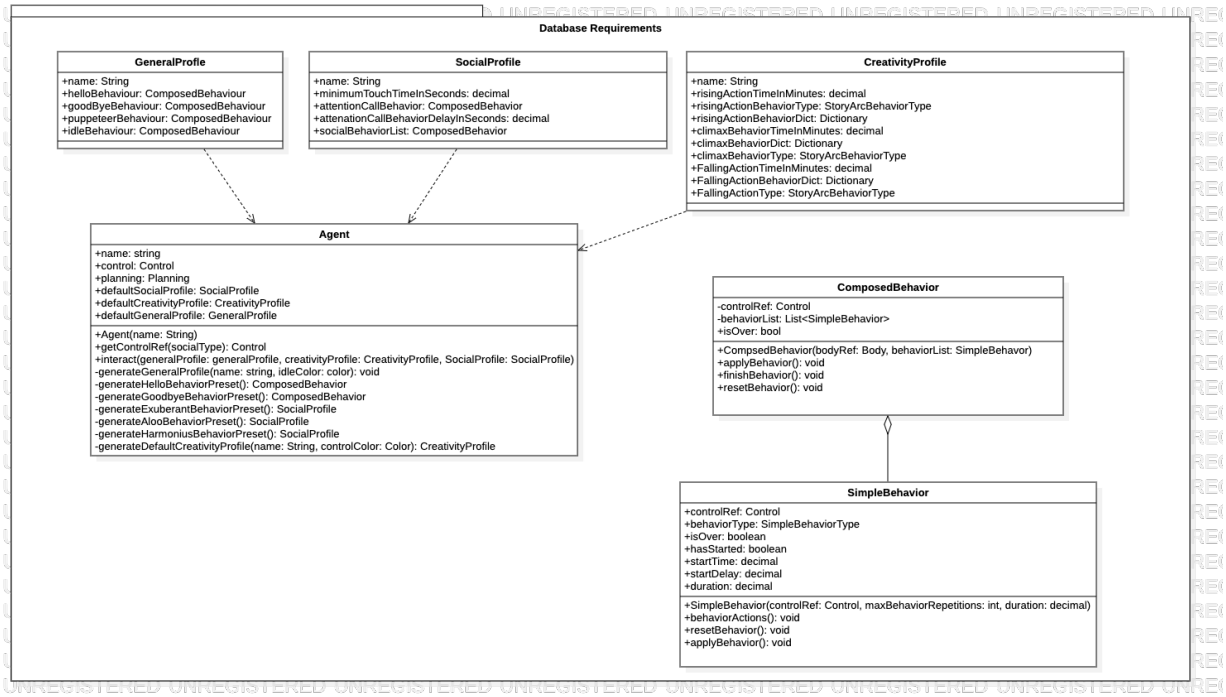

Figure 7: Logical Database Requirements Diagram for YOLO

- When YOLO starts working, its social, creativity and general profiles are kept in the agent to lead the behaviors of YOLO.
- YOLO might have various behaviors and these behaviors' mother class is ComposedBehavior.
- YOLO keeps track of the active simple behaviors with the variable behaviorList in ComposedBehavior.
- YOLO can decide the story arcs by using the variables of the CreativityProfile of the system. Rising action, climax and falling action, all of their attributes and duration are being kept.
- SimpleBehavior is a inheritance of the ComposedBehavior class.
- These all data are flows to the agent itself and needed functions such as generateGoodByeBehavior or generateHarmoniusBehavior can be called accounting the knowledge of the profiles of the user and behaviors.
- YOLO might check whether is a behavior is going on or it is ended using the value isOver.
- YOLO keeps track of time with the attributes like startTime, climaxBehaviorTimeInMinutes.

3.6 Design Constraints

- The system must be easily installable to the raspberry pi.
- The system must consume low energy so that the system can run for enough time.
- The system's wiki and sources must be organised regularly.
- A proper environment must be provided to the open source community so that people can easily contribute to the system.

3.7 Software System Attributes

3.7.1 Reliability

- YOLO is an open source project. This means limitless people are involved in the process, and this minimizes the risk and exploits of the project.
- YOLO's movements are designed by analyzing the results gathered from 100+ children. This means a reliable decision mechanism for the movements.

3.7.2 Availability

- The system is available 24/7. It is not depends to a local or global time.
- The system can be accessible through the internet. Build and charging is a must.
- In any case of failure, all parts are replaceable and customizable.

3.7.3 Security

- System doesn't require an active internet connection to system work. This ensures lots of security concerns are not needed.
- System is physically stable and slow. Even system can still hurt a child while moving or falling off, it is unlikely.

3.7.4 Maintainability

- The document and the wiki for the project must be perfectly clear.
- All parts are replaceable and customizable.
- Software might be updated some times. User must be updated to upgrade the version of the YOLO.

3.7.5 Portability

- System is totally portable to anywhere. It can run in anywhere that its wheels work properly.
- The system can be disassembled to be assembled in somewhere else.

3.8 Supporting Information

Since the purpose of the YOLO is to stimulate creativity in children with the development of techniques and mechanisms for child-centered robot design, the design should be as easy as possible to build and use.