**A picture containing clipart

Description automatically generated**

**COMP4910 Senior Design Project 1, Fall 2019**

**Advisor: Gizem Kayar**

**POF: Performance Optimized Fluids**

**High Level Design**

**Design Specifications Document**

**Revision 1.0**

**8.12.2019**

**By:**

**Baran Budak -15070001012**

**Cihanser Çalışkan -16070001020**

**İsmail Mekan -15070001048**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 8.12.2019 | Initial high level design |

# Table of Contents

[Revision History 2](#_Toc501993332)

[Table of Contents 3](#_Toc501993333)

[1. Introduction 4](#_Toc501993334)

[2. POF System Design 4](#_Toc501993335)

[2.1. POF System Architecture 4](#_Toc501993336)

[2.2. POF System Structure 4](#_Toc501993337)

[2.3. POF Environment 4](#_Toc501993338)

[3. POF System Detailed Design: 4](#_Toc501993339)

[4. Testing Design 4](#_Toc501993340)

[References 5](#_Toc501993341)

# 1. Introduction

The purpose of the POF system is to research and apply existed methods to simulate fluids and looking for a better way to simulate it. Various methods will be implemented and tested during the research and development of this project. The main goal is making research and sharing our observations of the project results. One of the major project objectives is to reach a more efficient and better performance fluid simulation system but it is not promised because there is no certain way to achieve it and as mentioned, the project is mainly research-based.

The design is based on The POF system Requirements Specification Document, Revision 2.0 [1]

This design process conforms to the Requirements Specification Document and its diagrams. The project conforms to UML diagrams. Diagrams are describing the project to understand mainly operations of the POF system. Imperceptible parts of the POF system can be changed but the main functioning of the system will remain the same as before. If any change occurs during the development of the POF system, this document and diagrams will be changed.

The system architecture and overall high-level structure of the POF system is given in second section. Detailed design of all system functions and the user interface in terms of are methods of all classes will be given later in third section of this document.

# 2. POF Computer System High Level Design

# 2.1. POF System Architecture

The POF system architecture works with NVIDIA Flex as an outsource asset. NVIDIA flex is mandatory because of the particle positions and axis aligned bounding box (AABB) in three dimensional space. System has a handler between the NVIDIA flex and the POF system. Initially, Flex starts the simulation and creates the particles and AABB. Handler retrieve this data to relevant classes.

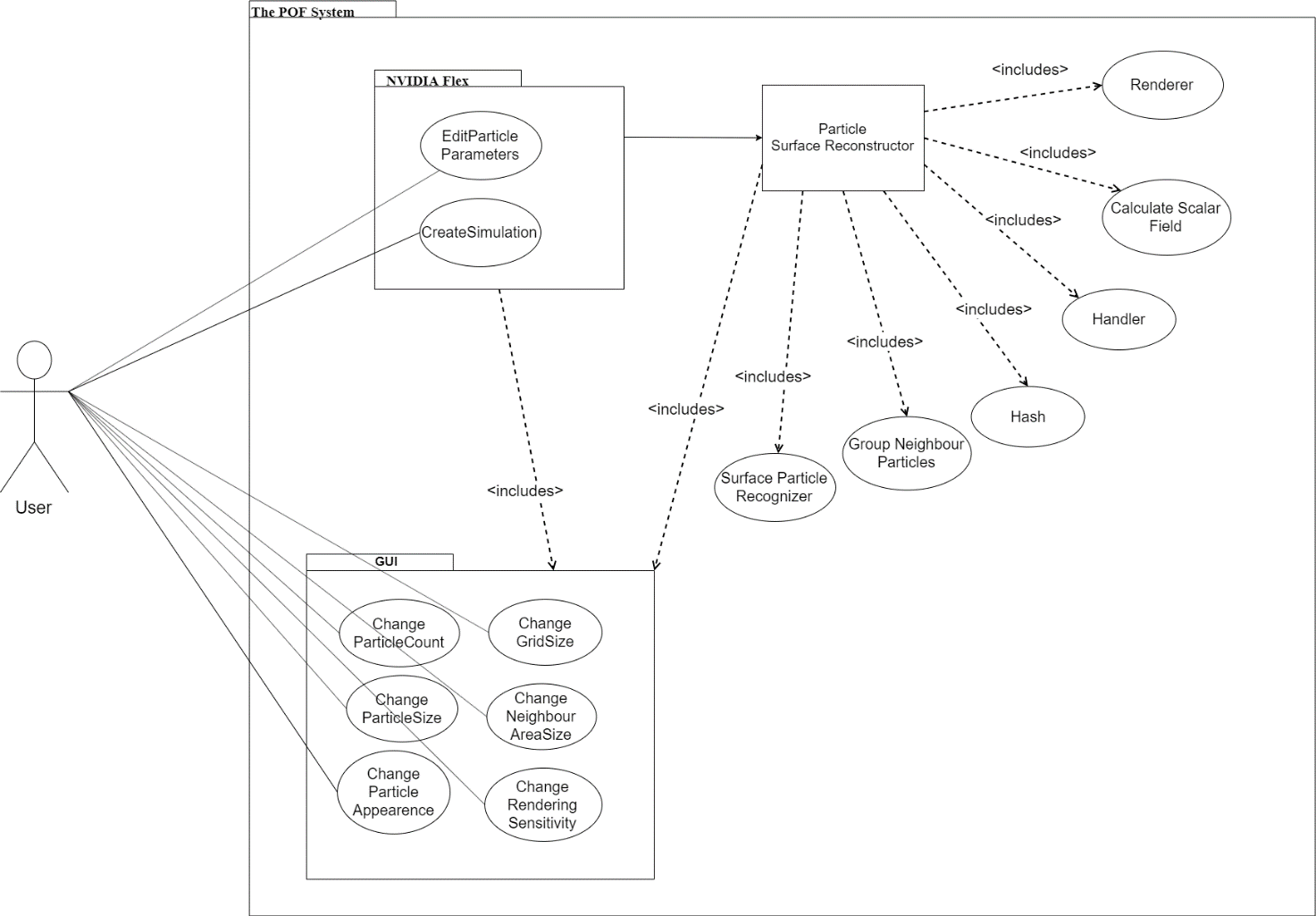
/////////////////////////////////////////////////////

* Overall computer system architecture
* Architecture of large system components, if such components with different architecture/structure exist. For example, a client-server system may have complex client and server side components with different architectures. Similarly for peer-to-peer and/or large distributed systems/
* Discussion and justification of design decisions to choose specific architectures.

# 2.2. POF System Structure

**\*\* activityleri detailed design a koymalı mıyız?\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**2.2.1 Use Case Diagram**



**Description**

|  |  |
| --- | --- |
| **Title** | **Description** |
| **Calculate Scalar Field** | Calculates a constant value of a particle in a given. |
| **Change grid size** | Interval of grid size of axis aligned bounding box can be changed with this function. |
| **Change neighbor area size** | Neighbor particle range of volume can be changed which affects the visualization of particles and changes the shape. |
|  |  |
|  |  |
|  |  |
|  |  |

**Calculate Scalar Field:** Calculates a constant value of a particle in a given.

**Change grid size:** Interval of grid size of axis aligned bounding box can be changed with this function.

**Change neighbor area size:** Neighbor particle range of volume can be changed which affects the visualization of particles and changes the shape.

**Change particle appearance:** Particle color, texture and light settings can be changed with this function.

**Change particle count:** Particle number in the scene canbe edited.

**Change particle size:** Radiusof the particle can be changed with this function. Increase in the particle size will result slow performance compared to less particle size.

**Change rendering sensitivity:** Rendering sensitivitycan be changed with this function. If sensitivity increase, fluid visualization will be more precise however processing time will increase.

**Create Simulation:** NVIDIA Flex simulation initialize when this function called.

**Edit particle parameters:** Particle attributes can be edited by user from GUI. Parameters can be maximum particle number, particle size, friction, adhesion etc.

**Group Neighbor Particles:**

**Handler:** Handler transmits data between layers and relevant classes. Handler manage data transmission.

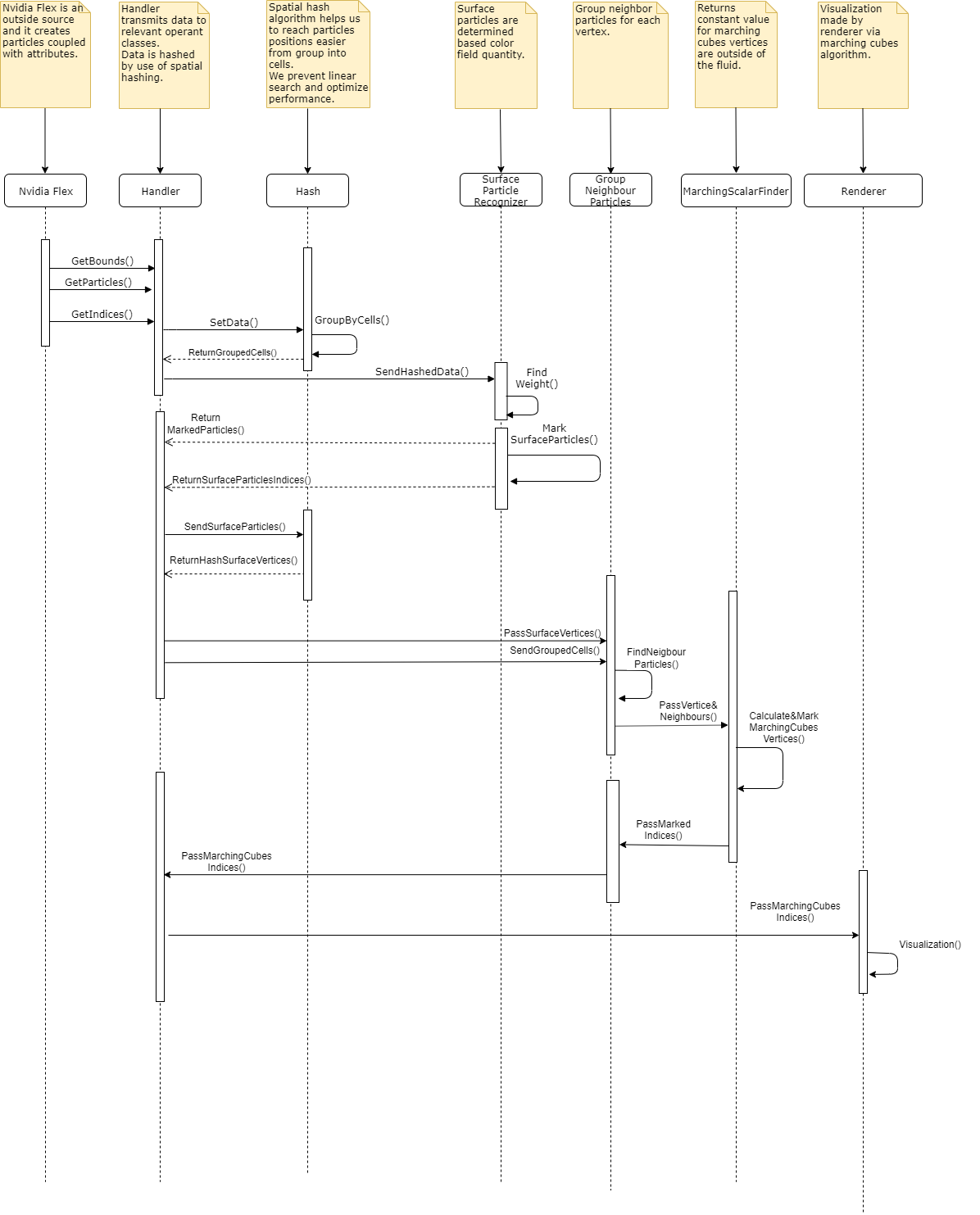
**Hash:** Hashes the particle and cell position

**Renderer:** Visualize fluid by drawing the given polygons.

**Surface Particle Recognizer:** Surface particles marked andprocessed for the necessary calculations in this function.

**User:** Users can be anyone who has access to the program.

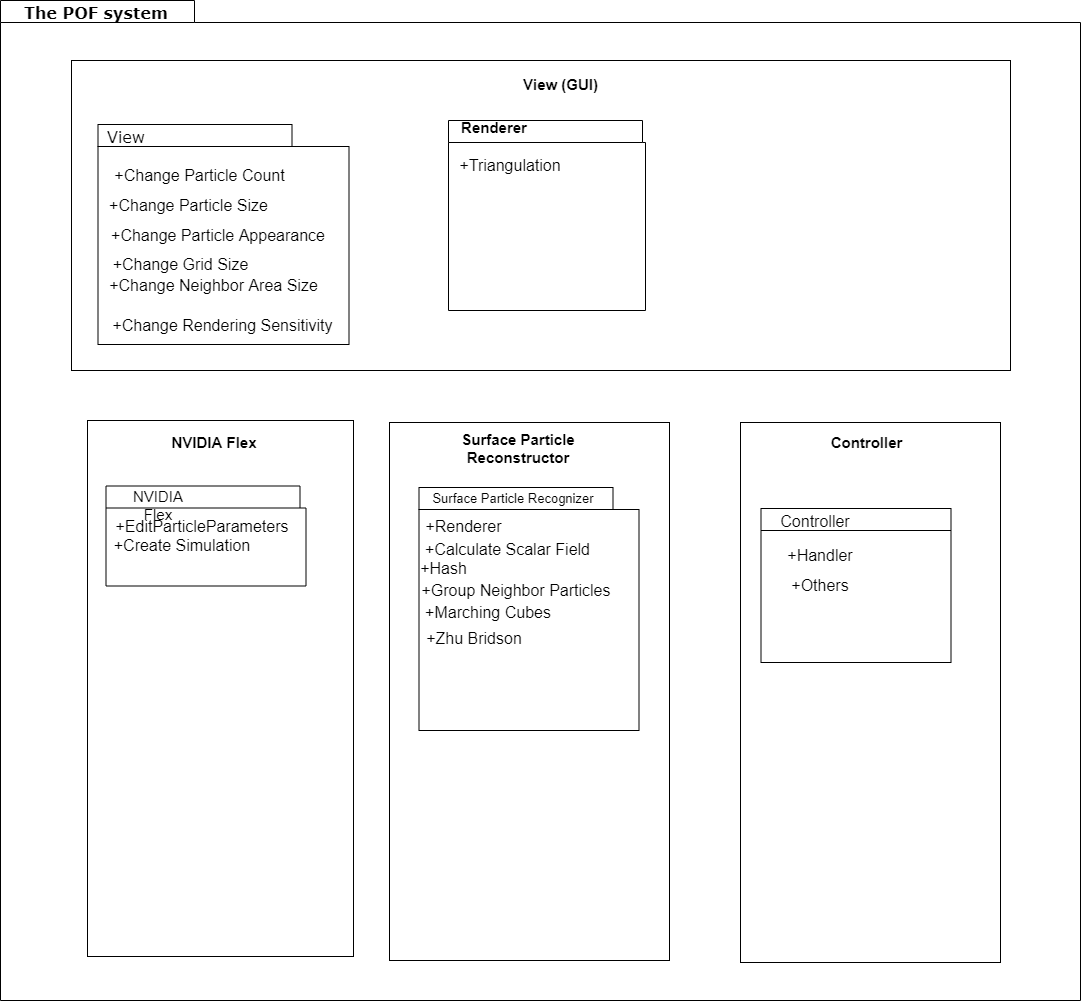
**2.2.2 Sequence Diagram**

****

**Description:**

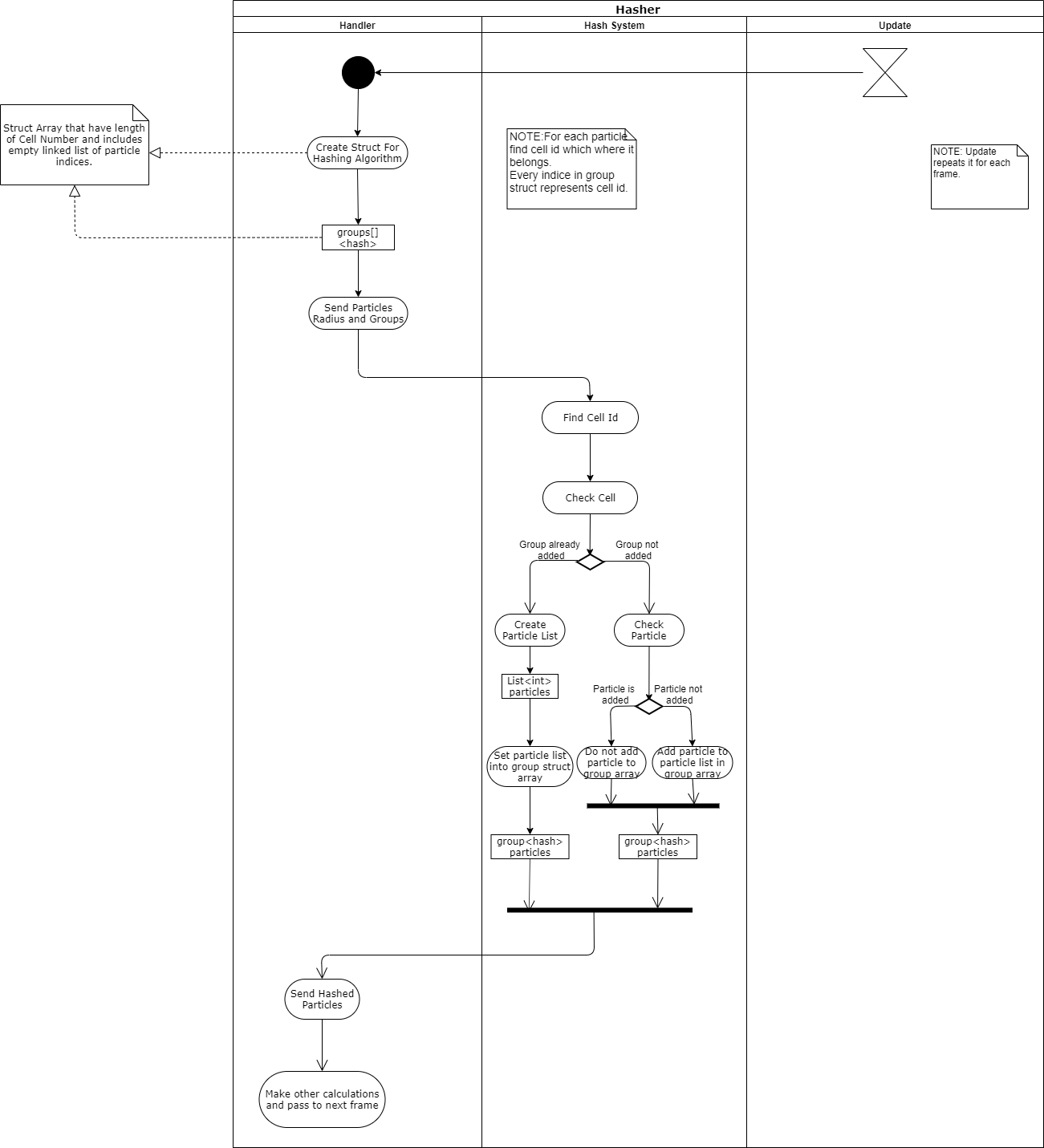
Handler gets the boundaries, indices and particle positions from NVIDIA Flex. Handler retrieves these data to Hash and hash returns a grouped cells by looking data. This cells are making easier to access particles. Cell id’s are send to surface particle recognizer and

**2.2.3 Package Diagram**

****

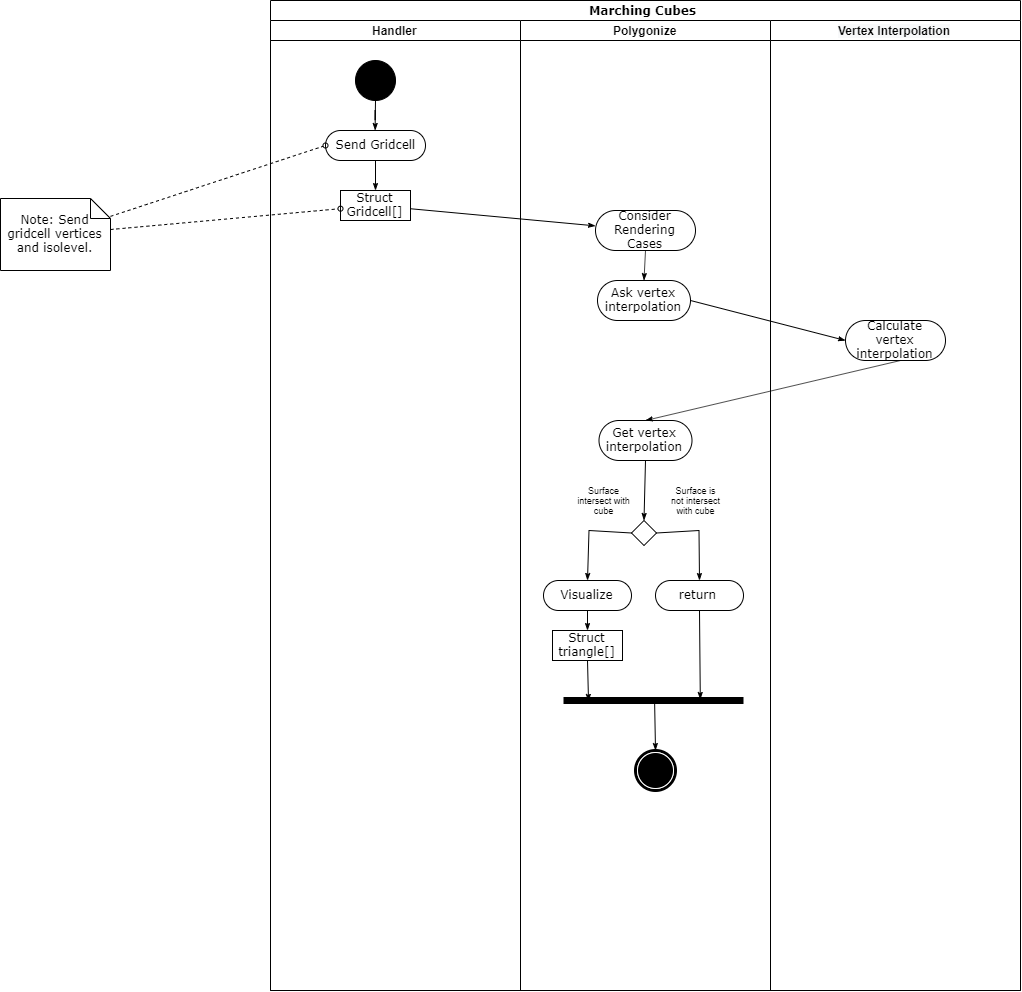
**Description:**

**2.2.4 Activity Diagram of Hasher**

****

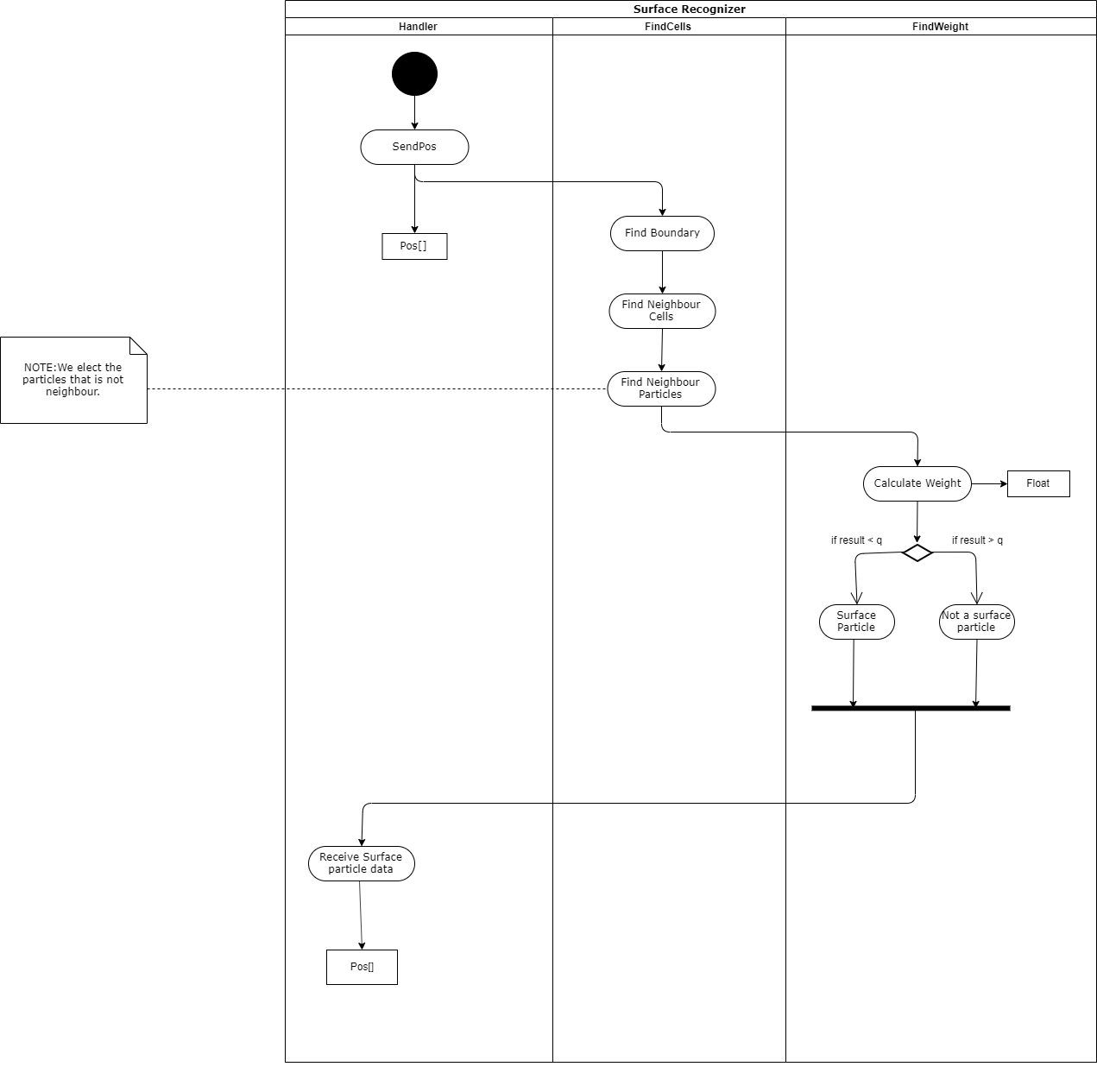
**Description:**

**2.2.5 Activity Diagram of Marching Cubes**

****

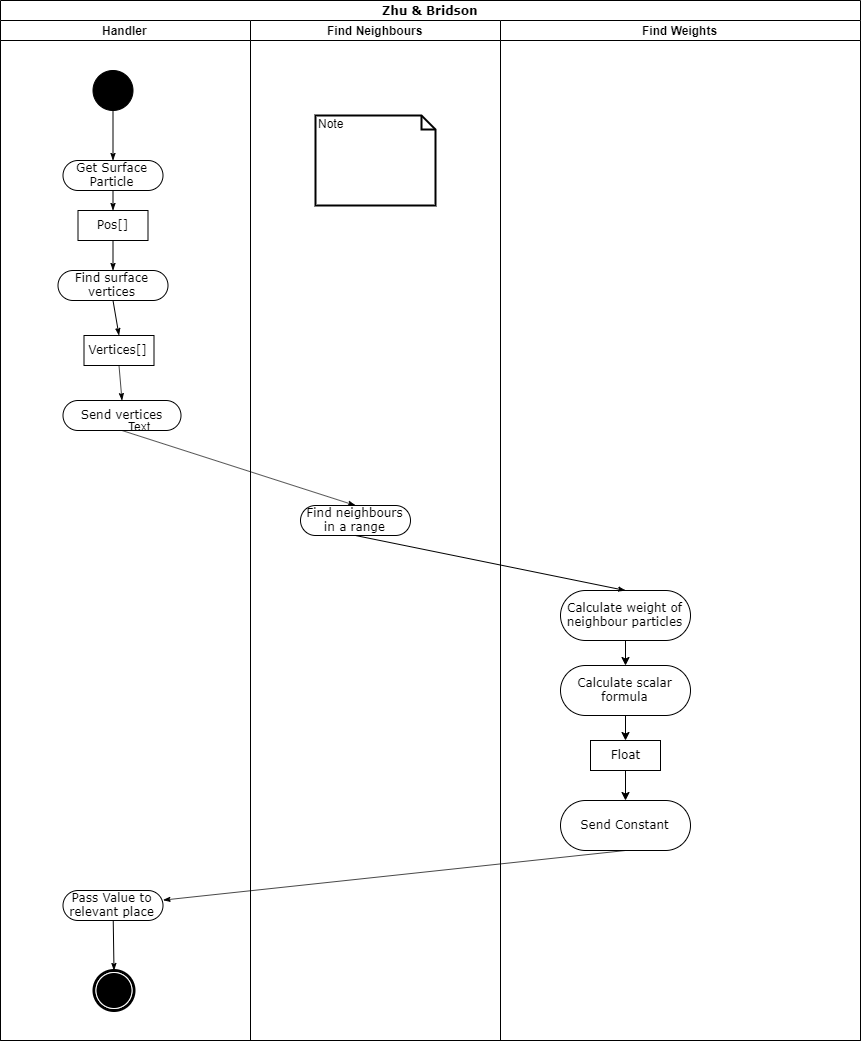
**Description:**

**2.2.6 Activity Diagram of Surface Recognizer**

****

**Description:**

**2.2.7 Activity Diagram of Surface Recognizer**

****

**Description:**

# 2.3. POF System Environment

The POF system environment constraints:

* D3D11 capable graphics card
* NVIDIA: GeForce Game Ready Driver 372.90 or above.
* AMD: Radeon Software Version 16.9.1 or above.
* Microsoft Visual Studio 2013 or above.
* G++ 4.6.3 or higher
* CUDA 8.0.44 or higher
* DirectX 11/12 SDK
* Windows 7 (64-bit) or higher.
* Unity 3D 2017.3 version or higher

The main project made on the system:

* Operating System: Windows 10 (64-bit)
* Processor: Intel Core i7-4700 HQ CPU
* Memory: 16 GB RAM – DDR3L-1600 Mhz
* GPU: NVIDIA GeForce GTX850M 4GB DDR3

This system has low performance on this project because it can handle very small amount of particles. The optimal system should be workstation defined in final report [3].

# 3. POF System Detailed Design:

To be completed in COMP 4920, in detailed design specification document.

# 4. Testing Design

Testing Design will be in this section.

Unity ortamında constant değerleri değiştirip en iyi performans ve sonuç veren değer ve methodlara bakıp onu implemente edeceğiz. Testte temel alacağımız faktörler, memory and performance efficiency, Frame Rate etc. Çeşitli uygulanan methodların tablo olarak performans karşılaştırılmaları vs.

# References

1. Requirement Specification Document revision 2.0 (RSD 2.0)
2. Use case, sequence, package diagrams in RSD 2.0
3. Final Report revision 1.0
4. Other references to additional documents, like other internal organizational documents, software project management documents, software design tool documents, etc.
5. References to additional bibliographic sources, like professional books, textbooks, handbooks, patents, standards, technical reports, journal/conference papers, etc.