**MAKERERE UNIVERSITY**

COLLEGE OF COMPUTING AND INFORMATION SCIENCES

DEPARTMENT OF NETWORKS

BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING (YEAR 2)

RECESS TERM 2 (BSE 2301)

TECHNICAL DESIGN DOCUMENT FOR:

**MULTI SALIENT OBJECT ANALYSIS PROJECT.**

**Version 1.0**

**PROJECT MEMBERS (GROUP 12)**

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **REG No.** | **STD No.** | **SIGNATURE** |
| ISABIRYE TAIBU | 16/U/20495 | 216022278 |  |
| NABUUFU ERETH | 16/U/8227/EVE | 216009341 |  |
| NAKIRANDA PROSCOVIA | 16/U/8785/PS | 216002865 |  |
| KANDOLE LAILA | 12/U/6171/PS | 212011932 |  |

Table of contents

[1 Introduction 4](#_Toc518368389)

[1.1 Purpose 4](#_Toc518368390)

[1.2 Scope 4](#_Toc518368391)

[1.3 Intended audience 4](#_Toc518368392)

[1.4 System overview 5](#_Toc518368393)

[2 Design considerations 5](#_Toc518368394)

[2.1 Assumptions and Dependencies 5](#_Toc518368395)

[2.2 General Constraints 6](#_Toc518368396)

[**2.3** **Goals and Guidelines** 6](#_Toc518368397)

[**2.3.1** **Module dependence.** 6](#_Toc518368398)

[**2.3.2** **Portability.** 6](#_Toc518368399)

[**2.3.3** **Reliability.** 6](#_Toc518368400)

[**2.3.4** **Correctness.** 7](#_Toc518368401)

[**2.3.5** **Speed.** 7](#_Toc518368402)

[**2.3.6** **Storage requirements.** 7](#_Toc518368403)

[**2.4** **Development Methods** 7](#_Toc518368404)

[3 Architectural Strategies 7](#_Toc518368405)

[4 System Architecture 8](#_Toc518368406)

[4.1 Architectural diagram 8](#_Toc518368407)

[4.2 Decomposition description 9](#_Toc518368408)

[4.2.1 Context diagram 9](#_Toc518368409)

[4.2.2 Level 1 Data flow diagram 10](#_Toc518368410)

[5 Detailed System Design. 10](#_Toc518368411)

[5.1 Image Upload component 10](#_Toc518368412)

[5.2 Image processing module. 12](#_Toc518368413)

[5.3 Display Results Component. 12](#_Toc518368414)

[5.4 Help Components 13](#_Toc518368415)

[6 Glossary. 14](#_Toc518368416)

[7 Bibliography 15](#_Toc518368417)

[7.1 References: 15](#_Toc518368418)

Table of figures

[Figure 1 MSO System architecture 8](#_Toc518368105)

[Figure 2 MSO system context diagram 9](#_Toc518368106)

[Figure 3 MSO system level 1 data flow diagram 10](#_Toc518368107)

# **Introduction**

## Purpose

The purpose of the MSO Technical design document is to provide a low level description of the system providing an insight into the structure and design of the system to allow software development to proceed with understanding of the system requirements from the user’s point of view.

## Scope

This document gives a detailed description of the software architecture of the MSO system. It specifies the structure and design of some of the modules discussed in the SRS. It also contains some of the constraints that affect the overall system development process and guide the development team on how to implement the system.

## Intended audience

The intended audience of the document is the individuals directly involved in the development of the MSO system. These include the following.

* Developers of the system who will use this document to get the details of the intended project capabilities and more easily understand where their efforts should be targeted in implementing the system features.
* Project testers who will use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. This way testing will be able to do the validation and verification tests to make sure the developed system satisfies the user requirements.
* Project supervisors will use this document to assess the quality of the implemented system

## System overview

MSO system is a web based system for analyzing and processing of images. It receives transaction data from the user then process and produce analysis results to the user.

MSO presents quick analysis of images by providing a graphical user interface where the user can easily interact with the system.

The user provides image input and then the system does the analysis of the image and return both the image and the number of salient objects contained in the image to the user.

The system is divided into four subsystems which include the Image input module, Image analysis module, The display image analysis results module and the help module. Each of these performs a certain role as described below.

Image Upload Subsystem. This module enables users of the system to input an image into the system.

Help Subsystem. This module enables users who get difficulties in using the system to access system documentation.

Image Processing subsystem. This module enables the processing and analysis of the image inputs by the system

Display Analysis module. This module enables users to see the image analysis results on the screen.

# **Design considerations**

## Assumptions and Dependencies

One assumption about the product is that it will always be used on platforms were R can be installed and that the system has enough performance. It is assumed that the users have working and well configured R environment version 1.6 or above is on their machine. This is because most of the features of the system depend on R environment packages and will work as intended only if these packages are installed.

It is also assumed that the users have a browser on their computer since MSO has a web based interface.

Beyond this, no other assumptions and dependencies are necessary to run the system.

## General Constraints

MSO system is platform independent and will be written in R. Its user interface will be written with R shiny package functions, so anyone who wishes to work on further development of the system has to know this programming language.

The memory requirements of Multi Salient Object detection system will be small since it is a light weight application.

MSO is meant to be quick and responsive, even when dealing with large number of transactions, so each feature must be designed and implemented with efficiency in mind.

## Design Goals and Guidelines

Below are the goals of the MSO system and the guidelines to the development team in order to achieve the goals

### **Module dependence.**

One of the goals of the MSO system design is to generate modules that are as closely working together as possible. None of the modules of the MSO system is independent of one another as the image Processing module depend solely on input from the image input module. The output / results display module depend on what the processing module has produced as output

In any case, the goal should be to minimize the independence between the modules.

### **Portability.**

The MSO system is intended to be portable and usable on any environment including windows, Mac OS and Linux. Developers should ensure that the system is environment independent

### **Reliability.**

The MSO system should be implemented in such a way that it ensures reliability to the user. The probability of failure to free software operation for a specified period of time in a specified environment should be minimal. Responses and the work done by the system should be consistent.

### **Correctness.**

The MSO system will work correctly if all the requirements and assumptions are met. It will give the same result regardless of time or environment.

### **Speed.**

Speed of operation is one of the major goals of the MSO system. The system must be interactive and delays involved must be less or must be minimal so in every detection process the delay is based on the number of salient objects available and so there is a probability that there, will be a delay of less than 20 seconds.

### **Storage requirements.**

The MSO is a light weight system that needs very few system resources in order to work. It is designed not to delay the system from other key processes and the response time of the program is direct. The main goal here is to enable users with limited system resources to also use the system

## Development Methods

We used the Unified Modeling Language (UML) to visualizing and documenting the systems design.

We considered using UML because it uses object-oriented design concepts, and it is independent of any specific programming language and can be used to describe business processes and requirements generally. This enables developers to use the same design to implement the system using various programming languages rather than being constrained to one language

UML provides various graphical tools, such as use case diagrams and sequence diagrams. These helps us in representation of the system from a user’s point of view.

# Architectural Strategies.

The proposed system is designed using client server architecture.

The Client application subsystem collects the users input and send it to the server via a network connection. The server analyses the data submitted and produces the image analysis results.

The analysis results are then sent back to the web based interface through the connection and displayed to the user.

Other architectural models that we considered include;

Layered architecture which is based on interfacing of subsystems and organize them in layers. Though it allows replacement of entire layers so long as the interface is maintained, providing a clean separation between layers is often difficult and a high-level layer may have to interact directly with lower-level layers rather than through the layer immediately below it. Performance can be a problem because of multiple levels of interpretation of a service request as it is processed at each layer.

Pipe and filter architecture where functional transformations process their inputs to produce outputs. Though it is easy to understand and supports transformation reuse, the format for data transfer has to be agreed upon between communicating transformation

This design document is prepared with system flexibility in mind. This is because the system will have upgrades and updates in the future to ensure quality and reliability which are some of our design goals. Developers should consider future modifications while implementing the system. They should provide support material for people who will maintain the system.

# **System Architecture**

## Architectural diagram

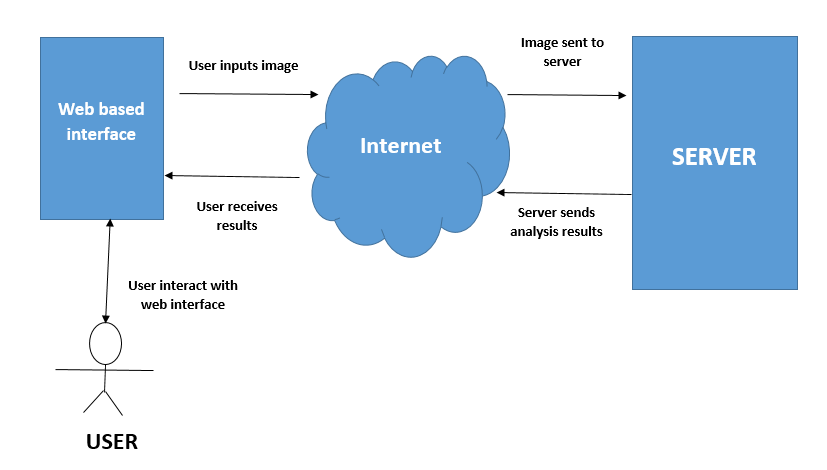


Figure 1 MSO System architecture

Double headed arrows mean that communication takes place in both directions.

## Decomposition description

### Context diagram

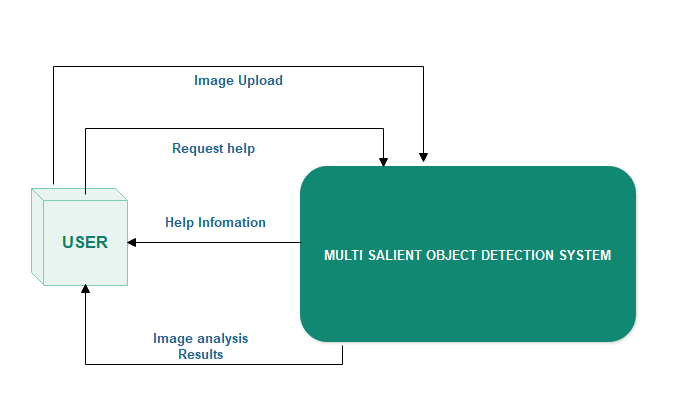


Figure 2 MSO system context diagram

### Level 1 Data flow diagram

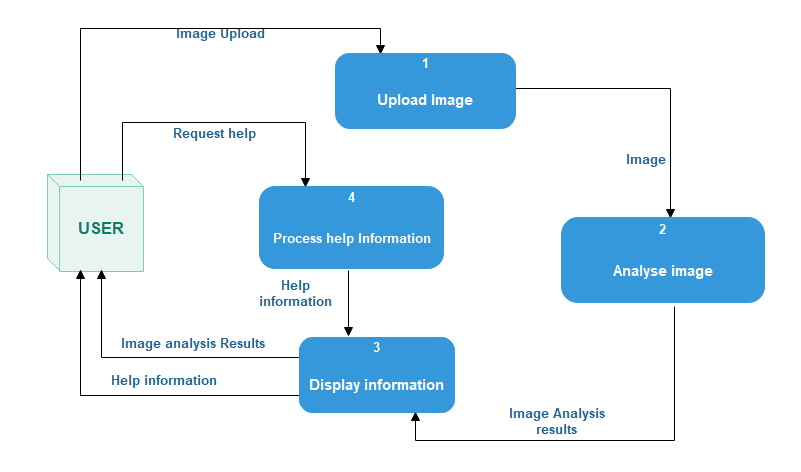


Figure 3 MSO system level 1 data flow diagram

# **Policies and Tactics**

## Choice of language

In this documentation we considered the use of R data analysis programming language because of the following reason.

R's functionality is developed with statisticians in mind, thereby giving it field-specific advantages such as great features for data visualization. It focuses on better, User friendly data analysis, graphical models and statistics. With R, statistical models can be written with only a few lines of code. Other data analysis programming languages like python where considered though were not taken because R has bigger community support than these options.

## Plans for ensuring requirements traceability

Requirements traceability will be ensured with the SRS document that was provided and all the requirements specified in the document have been applied in the design. Also models like DFDs are created to make sure that all the functionality of the system will be implemented according to the requirements.

## Plans for testing the software.

Al the system features will be tested with the test scenarios that will be created. These will be provided in an additional test document.

## Plans for maintaining the software.

Periodically, bug reports from users will be reviewed. Any reported problems will be fixed. This document will need to be updated if there are any new additional requirements involved.

## How to build and/or generate the system's deliverables.

The system will be compiled from the development machine and placed into the shiny web server.

# **Detailed System Design.**

In this section, system architecture of MSO which is explained in section 4 will be covered and most important points about design will be detailed. While doing this, main components and their classes and most important functions will be handled.

## Image Upload component

**Classification:** Module

**Definition:** The purpose of this component is to collect image inputs from the user

**Responsibility:** The module provides a graphical user interface where the user can select an image that he wants to analyze from his computer and then upload the image to the server.

**Uses/Interactions:** This component uses the web based interface for the purpose of selecting the image to be uploaded. Component provides an interface to the user that allows them to search the image by means of a search box.

**Processing:** Component initiates the system and gets prepared everything for the user to let him/her pick an image and start uploading. Every user-based event processed and according to these events, all actions are handled in order to make system ready for the analysis of the image.

Algorithm

1 Click search button

2 Select image to upload and click ok

3 **If** input file type equal to image

**Then**

Accept image

Issue upload success message

**Else**

Issue error message

4 End

After the image is loaded into memory, this module then sends the image to the server through an internet connection

**Constraints:** It is assumed that all the input from the user are images and not any other files. These images must be any of the three formats of JPG, PNG and TIFF.

**Resources:** The resources required by this module are CPU time and Computer memory to hold the images temporarily before sending them to the server.

## Image processing module.

**Classification:** Module

**Definition:** The purpose of this component is to analyze image inputs from the user

**Responsibility:** The module provides an environment for processing and analyzing the images through segmentation. It uses the R packages like EBImage and flsr to perform the image analysis and processing

**Uses/Interactions:** This component does not provide any user interface to the user as it performs all its tasks on the server not the client side.

**Processing:** ThisComponent receives image inputs from the client side. The module then decomposes the image into segments and analyze the image in form of pixels. After the image is loaded onto the server, this module then analyzes the image pixels and identify the number of salient objects in the image.

Algorithm

1 Segment the image.

2 Analyze the objects in the segments.

3 Store number of objects.

4 End

**Constraints:** It is assumed that all the input from the user are images and not any other files. These images must be any of the three formats of JPG, PNG and TIFF.

**Resources:** The resources required by this module are all server resources and not the user’s computer resources.

## Display Results Component.

**Classification:** Module

**Definition:** The purpose of this component is to display the image analysis results on the web interface.

**Responsibility:** The module provides a graphical user interface where the user can see the image analysis results. It receives the analysis results from the server after processing.

**Uses/Interactions:** This component uses the web based interface for the purpose of displaying image analysis results to the user. On this interface, the user is able to see the number of salient objects in a scene and the scene its self.

**Processing:** The component uses an Http connection to receive the image analysis results from the server and then display it onto the browser window.

Algorithm

1 Click on a given tab tab .

2 Display images and number of objects they contain.

3 End

**Constraints:** It is assumed that the user has a browser installed on their computer

**Resources:** The resources required by this module are CPU time and Computer memory to hold the image analysis results as they are displayed to the user.

## Help Components

**Classification:** Module

**Definition:** The purpose of this component is to display the system documentation on the web interface.

**Responsibility:** The module provides a graphical user interface where the user can see the system documentation.

**Uses/Interactions:** This component uses the web based interface for the purpose of displaying image analysis results to the user. On this interface, the user is able to see the number of salient objects in a scene and the scene its self.

**Processing:** The component uses an Http connection to receive the system documentation from the server and then display it onto the browser window.

Algorithm

1 Click on help tab.

2 Display help information.

3 End

**Constraints:** It is assumed that the user has a browser installed on their computer

**Resources:** The resources required by this module are CPU time and Computer memory to hold the documentation information as they are displayed to the user.

# **Glossary.**

## Abbreviations

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| MSO | Multi salient object |
| OS | Operating System |
| UI | User Interface |
| Http | Hypertext transfer protocol. |
| JPG | Joint Photographic Experts. A commonly used method of compression for digital images. |
| TIFF | Tagged Image File Format. A computer file format for storing graphics images, popular among graphic artists, the publishing. |
| API | Application Programming Interface. |
| PNG | Portable Network Graphics- A raster graphics file format that supports lossless data compression. |
| DFD | Data flow Diagram |

# **Bibliography**

## References:

**[1]** B Gary, J Harry Systems analysis and design Ninth Edition. Course Technology, Cengage learning,

USA, 2012.

**[2]** J Hunaizu, Y. Zejian and S Li, Salient Object Detection: A Discriminative Regional

Feature Integration Approach.,2011.

**[3]** A. Borgi, M.M. Cheng, Salient Object detection: A survey. In CVPR, 2017.