1. Introduction

1.1 Purpose of this Document

The purpose of the Software Requirements Specification is to outline the requirements for the MQTT-DrawBoard. It will be built using HTML5 with CSS and JavaScript. It will be operating system independent and accessible with any standard compliant browser on Desktop, Laptop or Mobile devices.

1.2 Scope of the Development Project

MQTT-DrawBoard will be a web platform for conveying illustrative representations of messages to the subscribed clients using a virtual canvas.

1.3 Definitions, Acronyms and Abbreviations

HTTP – Hyper Text Transfer Protocol
MQTT – Message Queuing Telemetry Transport
IoT – Internet of Things
M2M – Machine to Machine
TCP – Transmission Control Protocol
Ack - Acknowledgement
QoS – Quality of Service
Broker – Server in MQTT protocol
png – Portable Network Graphics
JSON – JavaScript Object Notation
JS - JavaScript

1.4 References

https://slides.com/sachinkmr/iot

https://www.eclipse.org

https://github.com/mcollina/mosca

1.5 Overview of Document

This document contains all of the software requirement specifics. It contains a general description of the types of users who will be using our application, how it is going to work, and what technologies we are using to make it work. We will also outline and describe specific components of the project.

2. General Description

2.1 User Personas and Characteristics

Creator – The creator is given the control of the draw-board. Creator is the one who can draw illustrations on the canvas which are viewed by the subscribed spectators.
Spectator(s) – The spectator(s) can only view the draw board with real-time updates on their devices.

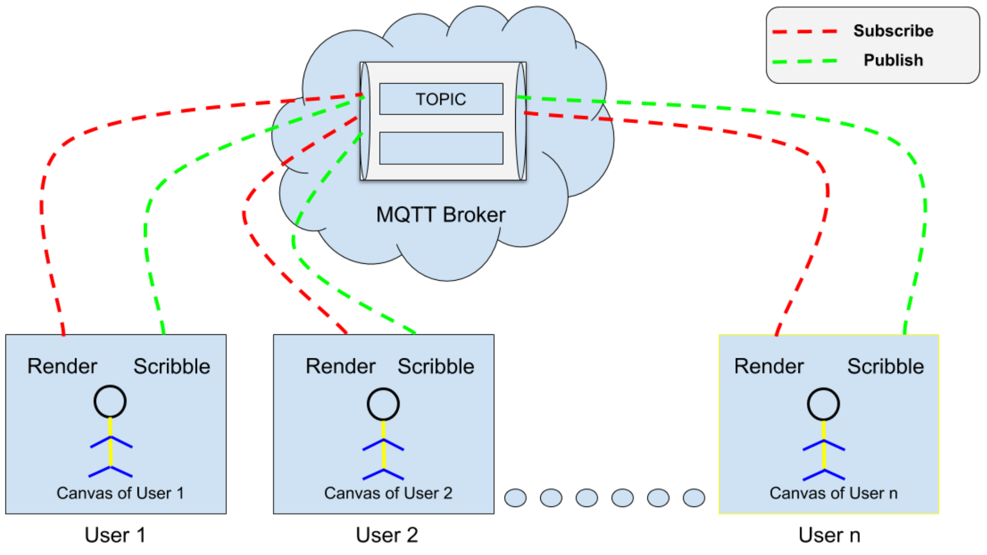


Figure 1: Block Diagram

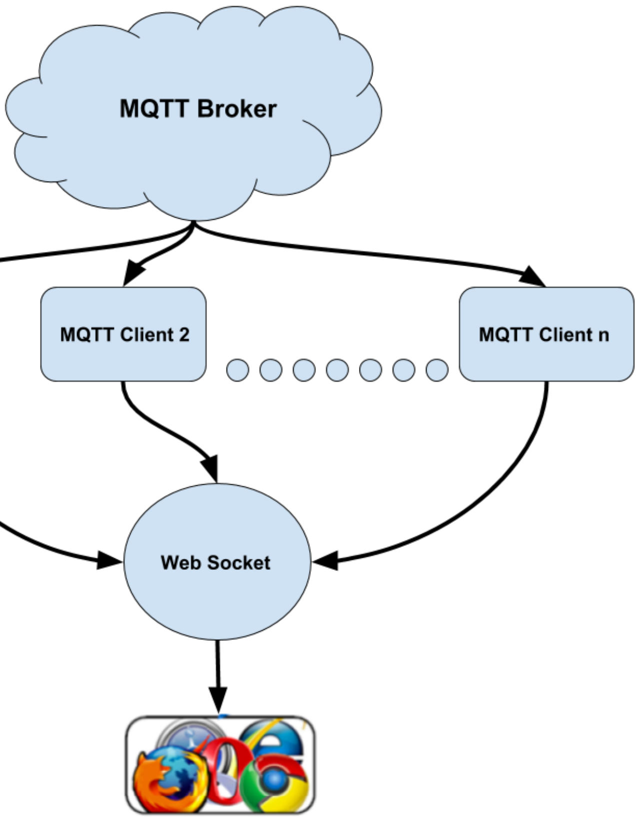


Figure 2: Browser acting as MQTT Client

2.2 Product Perspective

This product is designed to run on any system that is capable of receiving mouse or touch input. This covers a wide variety of machines, including operating systems like Mac OS X, Windows XP, Windows Vista, Linux, Android OS and iOS. The sole requirement for the user is a web browser (Google Chrome, Opera, Safari, Firefox or Internet Explorer) with an active Internet connection.

2.3 Overview of Functional Requirements

- Must be able to receive input in the form of mouse clicks and/or touch.
- Must be able to connect clients to MQTT broker.
- Must contain an easy interface.

2.4 General Constraints, Assumptions, Dependencies, Guidelines

This program will work with virtually any system that can connect to the internet and browse web pages.

2.5 User View of Product Use

The user comes to the website:

Canvas:

- The user can draw the illustrations on the canvas using a mouse or touch input.
- He/She can use the color picker to change the color of the brush or the slider to change the size of the brush strokes.
- Clear button is be used to clear the canvas.
- Undo button undoes the last stroke.
- Save button will save the drawing on the canvas in png format.

Objectives:

The purpose of this website will be to allow creators to communicate with spectators using drawing illustrations. These can be used for real time broadcast teaching remotely as well as for personal purposes. For example, playing Pictionary with friends and/or family.

3. Specific Requirements

3.1 External Interface Requirements

- Operator/user interface characteristics from the human factors point of view
  - This will interface with most browsers, especially Internet Explorer, Firefox and Safari. As long as the user has knowledge to operate a web browser, they will be able to operate our website efficiently.
- Characteristics required of the interface between the software product and each of the hardware components
- An MQTT broker. EMQ 1.0 is being currently used by us. It is currently the most scalable and advanced open source MQTT broker in global market, which is wildly used in thousands of IoT, M2M, Smart Hardware and Mobile App projects

3.2 Detailed Description of Functional Requirements

3.2.1 Template for Describing Functional Requirements

This is a typical template you can use to describe each of the functional components that were identified in Section 2.3. These sections should be at least the following:

Purpose	A description of the functional requirement and its reason(s)
Inputs	Which inputs; in what form/format will inputs arrive; from what sources input will be derived, legal domains of each input element
Processing	Describes the *outcome* rather than the *implementation*; include any validity checks on the data, exact timing of each operation (if needed), how to handle unexpected or abnormal situations
Outputs	The form, shape, destination and volume of the output; output timing; range of parameters in the output; unit measure of the output; process by which the output is stored or destroyed; process for handling error messages produced as output

3.3 Performance Requirements

Since the project involves using a MQTT management system, some decisions on performance are highly correlated to IoT broker. Issues such as the number of connections to the brokers are based on the current settings of the MQTT broker and the capabilities of the web browsers used.

Performance is measured in terms of delay experienced in transferring the strokes data over MQTT. The appreciable the delay, lower is the performance of our DrawBoard and vice versa.

Code will be clear and well commented, preferably following the established guidelines for easy integration with some other project.

The number of different simultaneous users and connections per second should only be limited by MQTT broker's configurations, specified by TAA and irrelevant to our project.

3.4 Quality Attributes

The webpage should be easily accessible and connectable, allowing multiple users to connect with the same DrawBoard.

The page will be available to anyone, but it can be restricted by creating rooms for different types of users. For example, a separate room can be created especially for teaching purposes in which the teacher is given admin privileges and their students can join in by using room ID.

The code will be clear and well commented to allow modification if necessary. All commonly changed settings should provide an administrator interface to apply the changes. The code should be modular in nature and incremental so that changes and tweaks are easy.

The webpages we work on should provide good error output on forms the clients will complete. Warning messages should appear when something is incomplete, and Error messages will appear in the unlikely event that something doesn't work as intended, as well as information on how to resolve the issue relevant to the user.

3.5 Data Model and Description

In this section a brief description of each data object is given. For each data object, function and semantics associated with it are summarized. This section also describes major attributes of data objects.

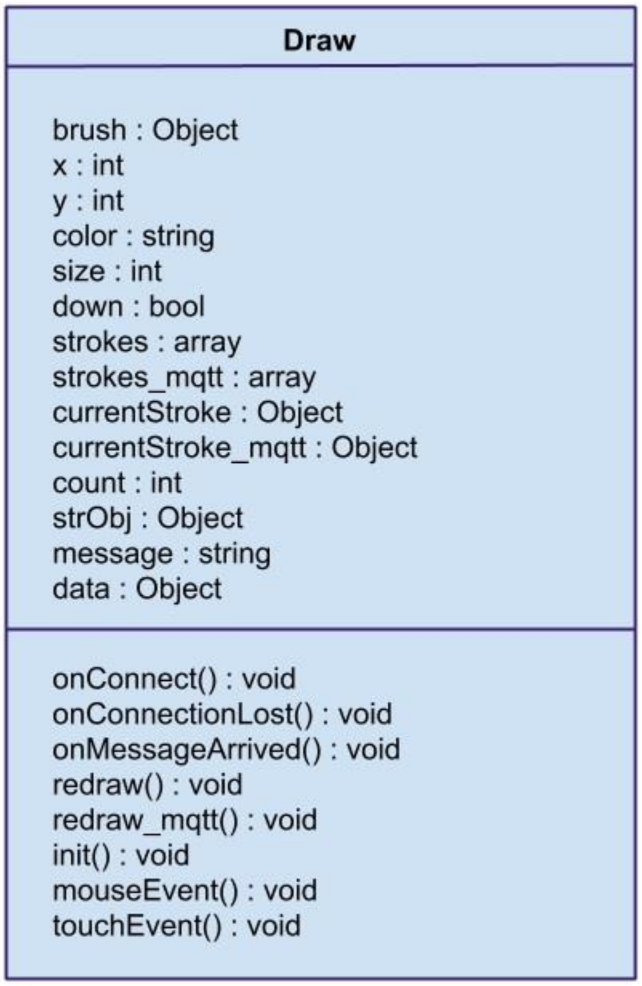


Figure 3: Class Diagram

Draw: This data object represents our DrawBoard.

brush: The object for our brush that is used to make strokes across the canvas.

x: The x-coordinate of the canvas.

y: The y-coordinate of the canvas.

color: The color of the brush stroke.

size: The size of the brush stroke.

down: The boolean value which specifies whether or not the brush is in contact with the canvas.

strokes: The array of strokes.

strokes_mqtt: The array of strokes that is sent over MQTT.

currentStroke: The value of the most recent stroke. It contains the x-y coordinates, size and color of stroke.

currentStroke_mqtt: The currentStroke that is sent over MQTT.

count: A counter.

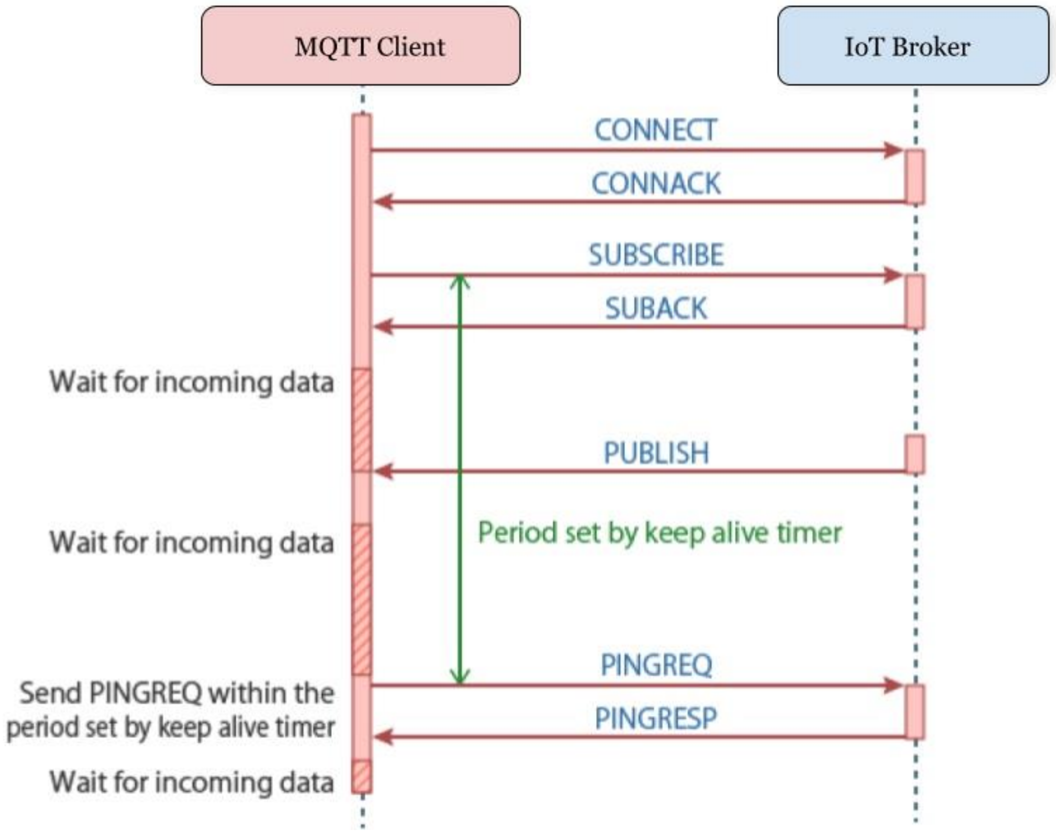
strObj: A data object used in transmission over MQTT.

message: A data object used in transmission over MQTT.

data: A data object used in transmission over MQTT.

3.6 Interaction Model and Description

In this section a brief description of interactions among the different elements of the model is given. The interaction and communication between objects are summarized. It consists of two main parts, which will be client and server side of the editor. First one, client side, is to produce activities and connections on the canvas area. Activities will be created by scribbling on the canvas area generating a certain shape. And the server side is responsible for collecting the points drawn on canvas area and sending those points to other subscribed clients.



3.7 Other Requirements

None at this time.

4. Conclusion

As we examined in the introduction part, we described everything about the project MQTT-Drawboard clearly and well-organized. We gave brief information about the document in the “Introduction” part. Then we described product perspective and product functions in the “Overall Description” part. In the “Specific Requirements” part, we explained all the software requirements of the project. In the “Data Model and Description” part, we described our data model. Inside of the “Interaction Model and Description” part, we presented a description of interactions among the different elements of the software. And now we are at the end of our SRS. We hope that this demanding document will be a good basis for our future works.