University of Waterloo

Faculty of Engineering

Department of Electrical and Computer Engineering

StreamingOS: Low Cost Education System

Project Specifications & Risk Assessment

Group 2020.15

Prepared by:
Matthew Milne – 20626854
Vidit Soni – 20627647
Vinayak Sharma – 20585279
Anurag Joshi – 20604210
Surag Sudesh – 20636861

5

Consultant:
Wojciech Golab

June 5, 2019

Scanned by CamScanner

Table of Contents

Scanned by CamScanner

1.0 High-Level Description of Project

1.1 Motivation

As technology improves in the 21st century, using mobile devices for educational purposes is becoming increasingly more common in primary and secondary schools. However, the devices come at a high cost. In the United States, it is estimated that the cost of educational technology ranges from $142 to $490 per student [1]. In addition, this technology becomes quickly outdated and needs to be replaced, forcing schools to spend even more money. With StreamingOS, we aim to alleviate these costs by providing the students and teachers with inexpensive thin endpoint devices, with the resource-heavy applications being streamed to these devices from a backend using container virtualization.

1.2 Project Objective

The objective of this project is to design a powerful and inexpensive device and streaming system that enhances the learning experience. StreamingOS uses an inexpensive Raspberry Pi (or alternative) and container virtualization to visually render and stream applications from a server or the teacher's computer to these devices used by the students. The system design leverages concepts learned in distributed computing, operating systems, database theory, and networking courses. The advantage of this design over current alternatives is that it is scalable while enabling the teacher full control of what software each student views. Due to the inexpensive hardware, it breaks down the barrier of the lack of technology in school settings and empowers teachers to incorporate more modern-day means of learning in their classrooms.

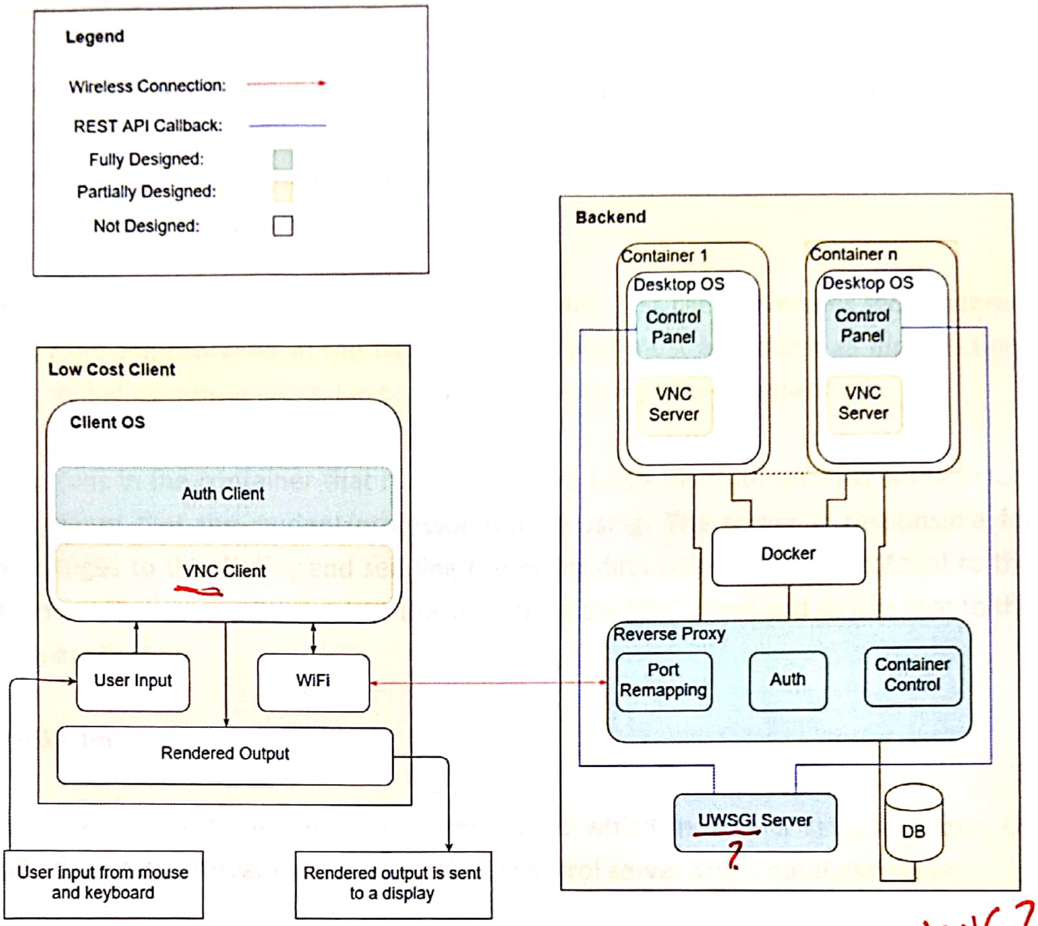
Code or results of exec. of apps.?

1.3 Block Diagram

The proposed solution consists of a variety of different components which enable the functionality of streaming the operating system to a cost-efficient device that is provided to students and teachers. There will be individual client devices (provided to students and teachers) running instances of a custom VNC client which connects to a Docker container running a full desktop Linux operating system through the usage of a reverse proxy. Each of the components mentioned above is discussed in detail through sections 1.3.1 and 1.3.4. The relation between each of the modules is shown in Figure 1.

expand

Scanned by CamScanner



VNC ?

Figure 1: Design Overview

1.3.1 Low Cost Client

The low-cost client is implemented through the means of a low-cost microcontroller. The low-cost microcontroller is used as a connectivity medium to the backend system. It runs a lightweight Linux based distribution. On the operating system, an authentication client and a custom designed VNC client are implemented. The microcontroller will be running an instance of a custom designed VNC client, allowing for the streaming of the operating system to the device. The microcontroller will connect to a low-cost monitor, have keyboard and mouse input/output capabilities, networking capabilities, and may support running on a battery or a direct power source.

Scanned by CamScanner

1.3.2 VNC Client and Server

Virtual Network Computing (VNC) is a graphical desktop sharing system that uses the remote frame buffer (RFB) protocol, developed by *Olivetti & Oracle Research Lab*, to control a computer [2]. This is implemented using a simple client-server configuration. The VNC clients and servers are implemented using *libVNC* [3].

The OS running on the client device runs a simple VNC Client. This client oversees the rendering of the desktop from the container in the backend. The client must also catch all input actions from the client, including mouse clicks, keyboard clicks, touch input (non-essential).

The VNC server runs in the container that runs the chosen Linux distribution. This is the actual desktop environment that the student/professor will be using. The server is responsible for catching any changes to the display and sending the event data using the RFB protocol to the client for rendering. The server also receives the input from the VNC client and passes that to the currently active application.

1.3.3 Backend System

The backend system consists of four main components which include a reverse proxy, OS containers, a UWSGI (Web Server Gateway Interface) control server and a database server.

The reverse proxy allows for packet forwarding between the client, and the assigned Docker container running the desktop operating system. The reverse proxy uses a dynamically modified IP table to allow for differential packet forwarding. The reverse proxy also has the ability of dynamic scaling. If a new server is added and registered with the reverse proxy module, it can spin up new containers on multiple servers in a distributed manner.

The database server runs an instance of MySQL with proper schemas in place allowing for functionality such as adding/revoking students from the system, storing session information, application access control on student OS containers, and miscellaneous client information.

The containers run a non-resource intensive desktop Linux OS that has a GUI. This ensures that a large quantity of these OS containers can be spun up on one backend machine. The teacher's OS container will be running a special version of an application developed using Electron (referred to as Control Panel in Figure 1, which allows them to have additional functionality such as controlling student's screens, sharing their own screens, revoking/enabling application access for students, revoking or enabling system access for new or existing students, and more. The

Scanned by CamScanner

backend system will be easy to set up, such that it can be enabled quickly on a teacher's computer or in a server.

The purpose of the control server is to allow the teacher's client to control the student client sessions. The communication between client and server will be done using REST API callbacks. For example, the teacher's client could send a POST request to the server with a specific student ID and a list of available applications. The server would then inform the respective client of the updated permissions using a second REST call. Finally, a response will be returned to the teacher's client showing that the call was successful.

1.3.4 Frontend Desktop Application

The front-end portion of this project consists of an interactive desktop application that will be built using Electron. This is referred to as the *Control Panel* in the block diagram. This application will have a simple yet powerful user interface (UI). This will be implemented using a Bootstrap template which consists boilerplate code for the HTML, JavaScript and CSS components of this application. The UI mainly consists of two sets of dropdowns, the first one being a list of students currently enrolled in the class and the second being the list of applications each student has access to. In addition, teachers will also be able to view personalized information and data about the class. The primary user of this front-end system in our context will be the teacher. Once the teacher wishes to give a student access to a specific application, he/she will need to click the submit button after which the desktop application will call API endpoints on UWSGI Control Server sitting in the backend. School administrators will also be able to use this application to add or delete students.

Scanned by CamScanner

2.0 Project Specifications

The project specifications are classified into two broad categories, functional and non-functional specifications. The functional specifications refer to the functions that the project is required to implement; these are listed in Table 1. The non-functional specifications describe the characteristics of the product; these are listed in Table 2. Each specification is further sub-categorized into 2 categories, essential or non-essential components. This is based on the importance of the specifications, in order to meet the project prototype objective. Essential specifications are those which must be met for the project prototype to be satisfactory, whereas the non-essential specifications make the final prototype more than just complete when they are met. In other words, these specifications are considered to be the non-critical aspects of the system. The system will still function properly if the non-essential specifications are not met. Each specification is rated as essential or non-essential based on their importance to the overall design.

2.1 Functional Specifications

Table 1 – Functional Specifications

This is non-functional

Specification	Description	Classification
Latency	The user should be able to interact with the desktop operating system with a latency that does not exceed 300ms.	Essential
Teacher Control	Teachers should be able to see what is on the students' screens, as well as revoke access to certain applications.	Essential
Multi-User Support	The central node should be able to support at least 30 systems (Teachers and students)	Essential
Authentication System	The client-server setup requires an authentication system.	Essential
Network Connection	We should follow the TCP/IP protocols correctly so that packets can be transferred between the server and many clients.	Essential
Performance	The desktop operating system should be able to run browsers and other compatible applications with reasonable reactivity.	Essential
Student Birds Eye View	The teacher should be able to see the screens of all the students in one convenient location in the control panel application.	Non-essential
Secure tunnel for data transfer	The VNC client and server will use an encrypted tunnel to transfer data packets.	Non-essential

Electrical and mechanical specs of the device?

Will the proposed use of Raspberry Pi pose any bottleneck?

2.2 Non-Functional Specifications

Table 2 – Non-Functional Specifications

Specification	Description	Classification
Cost of Device	The cost of a single educational device should not exceed $75.	Essential
Reliability ??	The system should transfer data from the backend systems to the client's systems without any interruptions in a reliable fashion.	Essential
Backend server uptime	Our servers should be up for the duration of the school day.	Essential
Portability	The dumb terminals should be portable with a touchscreen display.	Non-essential
Security	The student and teacher systems should be reasonably secure from malicious attacks.	Essential
Usability	The desktop application should be well polished and user friendly.	Essential
Heterogeneity	The application is developed in such a way that it can only support multiple operating systems and platforms.	Non-essential

What are The main challenges here?

Scanned by CamScanner

3.0 Risk Assessment

Every engineering project contains risks associated with it. In Table 3, these risks are outlined, along with methods for mitigating them. Additionally, the probability of the events and the impact the events have on the successful completion of the project have been calculated.

Table 3 – Risk Assessment

Risk	**Problem**	**Mitigation**	**Probability & Impact**
Scheduling Meetup of Group Members	Course work and job interviews could limit the amount of times the group can meet throughout the term. In addition, group members will likely be separated for the duration of the coop term.	Schedule meetings when the entire group is free as much as possible, otherwise keep the rest of the group updated on Facebook Messenger or Slack.	**Probability**: Medium **Impact**: Medium
Parts Procurement	The design of this project will require purchasing of many hardware components.	We will purchase the parts that we require early enough so that our project isn't delayed.	**Probability**: Low **Impact**: High
System Integration	This project uses many different software products, and this combination may not integrate well together.	Spend lots of time on integrating the software to lessen the impact of integration issues.	**Probability**: High **Impact**: High
Technical Knowledge	Our project covers a wide knowledge base, including database theory, distributed systems, REST APIs, web development, virtualization, operating systems, networking and server development.	Delegate the team member(s) with the most knowledge in a specific area to perform tasks that require that knowledge. We also chose a consultant with knowledge in many of these areas.	**Probability**: High **Impact**: Medium

Scanned by CamScanner

References

[1] B. B. a. R. L. Ross, "The Cost of School-Based Educational Technology Programs," 3 March 2013. [Online]. Available: https://www.rand.org/pubs/monograph_reports/MR634/index2.html. [Accessed 1 June 2019].

[2] "Virtual Network Computing," [Online]. Available: https://en.wikipedia.org/wiki/Virtual_Network_Computing. [Accessed 2 June 2019].

[3] "LibVNC," [Online]. Available: https://libvnc.github.io/.

Scanned by CamScanner

ECE498A: Marking Sheet for Project Specifications and Risk Identification Document			
Group number: 2020. 15			Marks
Presentation and layout	Follows required formatting, layout, structure, page limit: Correct spelling, grammar, captioning, referencing, units: Readability, professionalism of language, flow, graphics:	up to 2 marks up to 2 marks up to 2 marks	5 / 06
Block diagram	Includes well-labelled inputs, outputs, subsystems; graphics are crisp with a good layout and easy to follow: Clearly indicates which subsystems will be designed and which will not be designed: Overall structure is clearly explained; the block diagram layout is logical with a reasonable level of granularity:	up to 2 marks up to 3 marks up to 7 marks	10 / 12
Specifications	Specs are reasonably categorized as functional/ non-functional and essential/non-essential: Each specification is verifiable, open-ended, & appropriate: Set of specifications is complete: The specifications are at a good level of challenge for a capstone engineering project:	up to 4 marks up to 12 marks up to 5 marks up to 5 marks	22 / 26
Risk assessment	Includes 3 situations that could hinder project completion: Nature, potential impact, and probability of each risk are included and seem reasonable: Good suggestions for what can be done to reduce likelihood of and to reduce damage from high impact/probability risks:	up to 2 marks up to 2 marks up to 2 marks	6 /06
	Deduction for late submission: (-10 marks if miss deadline, and -10 marks off for every additional 24 hours)		
	Final Mark:		43 / 50

Additional comments from course instructor:

Scanned by CamScanner