ROUS System Technology Review

Prepared For:
Lonnie Mandigo
By:
Lucien Tamno
CS 461: Capstone fall 2017-18
Oregon State University

Abstract

This document provides technology review for the **ROUS** (`Rodents of Unusual Size`) system. Each group member is responsible to work on three features of the software and Each section of this document covers, each member of th group individual research about getting the most accurate technologies to achieve software functionalities.

November 16, 2017

CONTENTS

1 Overview 2

2 Definitions, Acronyms, and Abbreviations 2

3 Communication Services & Interfaces 2

3.1 Introduction . . . 2

3.2 Deployment communication module (DCM)—Portability . . . 2

3.2.1 description . . . 2

3.2.2 Different Technologies - portability module . . . 3

3.2.3 containerization of choice . . . 3

3.3 Communication Between Nodes - technology of choice (CBN) . . . 4

3.3.1 Description . . . 4

3.3.2 Technologies for the communication between nodes . . . 4

3.3.3 ROUS network technology of choice . . . 4

3.4 ROUS software and System Host interconnection - technology of choice . . . 4

3.4.1 Description . . . 4

3.4.2 Technologies for the interconnection module . . . 5

3.4.3 ROUS Programming Language of choice . . . 5

References 6

1 Overview

After Releasing the software requirement specifications document for the ROUS system, the technology review step is individually written by each member of the group. And the task is for each member of the group takes responsibility for three major components of the ROUS project and for each of these components, three have to be searched and explain how those technologies fit the ROUS system functionalities. finally and after analysis, the group member has to choose in each case the technology that would likely work the best with the system.

2 Definitions, Acronyms, and Abbreviations

TABLE 1
ROUS System: Definitions& Abbreviations

Terms	Definitions
	Collaborative Threat Mitigation
I/O	Input/Output system
Users	Stakeholders (Administrator, client and others)
API	a set of subroutine definitions, protocols, and tools for building software and applications
WSN	Wireless sensor networks
Raspberry Pi	a series of small single-board computer
GUI	Graphical User Interface
VM	Virtual Machine
Hypervisor	software,firware or hardware to create VM

3 Communication Services & Interfaces

3.1 Introduction

Basically,to talk about the ROUS system communications with its environment, what we want to say is that there are three different communication interfaces to be built within the ROUS system. And all provide each different kinds of services as well. Thus we have:

1) Interface ROUS software and transportation container
2) interface communication between Nodes
3) interface ROUS system and host hardware

3.2 Deployment communication module (DCM)—Portability

3.2.1 description

the ROUS system is likely to be deployed on any hardware chosen by its shareholders. By this, I mean that the software can be embedded in any other hardware than the Raspberry Pi or the like which will serve as a testing tool for the developer's team. That is, and while developing the ROUS system, our team has to keep in mind not to tie the ROUS system functionalities to hardware features. Because doing so will eventually establish a strong dependency between any particular hardware and the software to be developed and undermining the software deployment on other hardware with different architecture.Therefore, even the process of choosing a good container in which the ROUS system will be stored and retrieved easily does matter because a secure and flexible container for the ROUS to reside in is an important aspect to ensure the availability of the system.

3.2.2 *Different Technologies - portability module*

The following table content provides the three different technologies associate to the kind of container we want the ROUS system to reside in. To better explain the reasons of the subsequent choice, the current table is labled as such:

- the first is made up with titles with the first column bearing the name of different technologies, and the subsequent columns labled with criteria names used to compare different techonology features.
- the subsequent rows of the table are various instances of techonology considered for the portability module

TABLE 2
ROUS system Portability

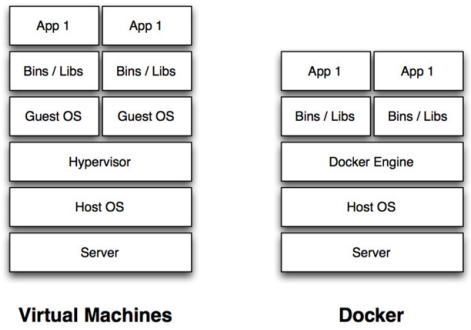
Containerization	Virtualization	ownership	Popularity	API
Docker	Single level of abstraction and directly interacts with Linux Kernel, Isolated VM,No Hypervisor needed ,	yes, (Docker company) - Guarantee projects privacy	widely known by companies	well documented in Linux API
Kubernetes	different level of abstraction and Uses API to provide containers	yes, (Google) - open community	still growing in popularity	Kubernetes API (python)
Mesos	different level of abstraction and Uses API to provide containers	No,(UC Berkeley)- open community and open source project	still growing in popularity	mixed API(JAVA, C++,HTTP)

3.2.3 *containerization of choice*

In addition to information found in table 2above, Docker has the following features:

- Docker has engine that can be run on any virtual machine.
- Docker system doesn't pre-allocate resources but sparingly uses those resources when needed(RAM, Disk space), reason why Docker is being called lightweight virtual machine.
- Docker has been devised to run on Linux system which is run by pretty much 99 percent of the entire Internet.

which ultimately ollow it to be the best choice for the ROUS system project.



source: Comparing a VM to Docker [1]

Fig. 1. Architecture Docker technology

3.3 Communication Between Nodes - technology of choice (CBN)

3.3.1 Description

At the ROUS software level of communication between nodes, a protocol that works the best and complies with the system requirements is required. In this case, the communication happening between nodes has to be isolated to any external interference. In other words, to say that no outside device (outsider) shouldn't be able sends signal which is processed like a normal stream of data by the system. And to mitigate this first layer or threat, the system must avoid any kind relationship with internet communications, and eventually if possible the use of IP addresses either private because could be found running on private wireless systems others than the ROUS system or public because these types IP addresses are visible over the internet which we don't to connect with.

So, the following table that describes the three kinds technology, we will somehow want to work with and the subsequent choice will be made based on the comparable criteria presented.

3.3.2 Technologies for the communication between nodes

The table here underneath provides samples network technologies with each having a set of different characteristics to be compared as followed:

- the first is made up with titles with the first column bearing the name of different technologies, and the subsequent columns labled with criteria names used to compare different techonology features.
- the subsequent rows of the table are various instances of techonology considered for the communication nodes

TABLE 3
Technologies: ROUS system nodes interconnection

Protocol	IP address usage	Wireless	System exposure	standards
ad hoc	yes via ARP/RARP	yes	external environment vulnerability via IP address	open to industrial applications
WSN	No, signal sensor	yes	better security via controlled sensors to	primarily reserved for military applications but currently open to industrial applications
Bluetooth	No, UHF radio waves	yes	less secure and used by many mobile devices	open to industrial applications

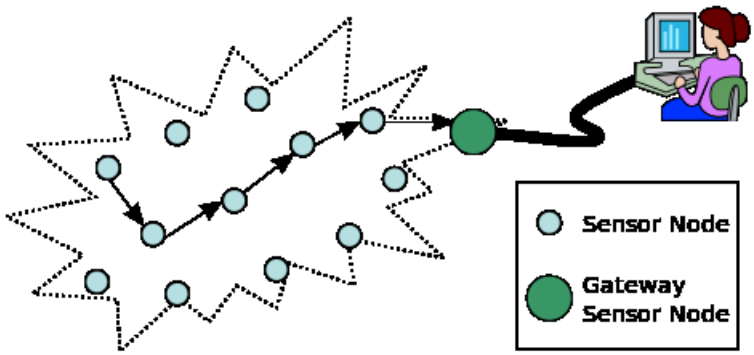
3.3.3 ROUS network technology of choice

Based on data put on display by the table 3 above, and also considering the fact that our system has to be freed from any wireless interference communication, we would definitely use as a protocol like: the wireless sensor network to implement the communication between ROUS nodes and Hereafter is a figure of what a WSN will be like.

3.4 ROUS software and System Host interconnection - technology of choice

3.4.1 Description

The ROUS software has to interact with its environment through different kinds of interfaces.here we will have the User interface (UI), one or more API insuring any communication between the ROUS software and a an embedded hardware.For each of the interfaces, the programming language to implement is crucial because of the reasons described in the table 4, part of the section 3.4.2 underneath.



source: Typical multi-hop wireless sensor network architecture [2]

Fig. 2. Architecture wireless sensor network

3.4.2 Technologies for the interconnection module

This section provides a table content of three possible choices of technology which are the programming languages of choice to provide better interaction between the ROUS system and any host hardware environment. The choice is made with the criteria that the system has to avoid any additional external piece of software in order to run in its hosting environment. To do so, in our table below, we will compare three programming languages to allow the ROUS system to easily be deployed on a hardware without any additional piece of software in order to run. And this table like the 2 previous puts in display information in this way:

- the first is made up with titles with the first column bearing the name of programming languages, and the subsequent columns labled with criteria names used to compare different language characteristics.
- the subsequent rows of the table are various instances of languages.

TABLE 4
Technologies: ROUS Software and System Host

Programming Language	Compatibility(code source)	Integration with	Portability (Docker)	cost
C	compatible, close to the hardware	python	yes	No cost
Python	runs on a variety of systems	C, Java	yes	No Cost/ open source
Java	Needs JVM to run on some systems	C++, Python	No	may need a license

3.4.3 ROUS Programming Language of choice

Using information on the Table above, we will use C and Python programming languages to implement the ROUS system solution because among other requirements, the system will be:

- hosted on Docker container environment (requirement from the client)
- implemented on hardware device like Raspberry pi or alternatives ASUS Tinker Board,NanoPC-T3, ODroid Xu4 . . . (hardware developed by another member of the team)
- run on Linux OS.

REFERENCES

[1] Steven Haines JavaWorld,"Open source Java projects: Docker",NOV 3, 2015 3:42 PM PT.
https://www.javaworld.com/article/3000781/development-tools/open-source-java-projects-docker.html
[2] wikipedia,"Wireless sensor network", 6 November 2017, at 17:21,[last Update].
https://en.wikipedia.org/wiki/Wireless_sensor_network