6-9-2016

The New Grid

Vincente Ciancio
Santa Clara University

Andrew Dobbins
Santa Clara University

Steven Sanz
Santa Clara University

SANTA CLARA UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING

Date: June 8, 2016

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Vincente Ciancio
Andrew Dobbins
Steven Sanz

ENTITLED

The New Grid

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE
in
COMPUTER SCIENCE AND ENGINEERING

[illegible] June 8th 2016
Thesis Advisor date

N. Ling 6/9/16
Department Chair date

The New Grid

by

Vincente Ciancio
Andrew Dobbins
Steven Sanz

Submitted in partial fulfillment of the requirements
for the degree of

Bachelor of Science in Computer Science and Engineering
Santa Clara University

Santa Clara, California
June 8, 2016

The New Grid

Vincente Ciancio
Andrew Dobbins
Steven Sanz

The New Grid seeks to provide mobile users with an additional method for off-grid communication, or communication without connection to Internet infrastructure. The motivation for this project was to find another alternative to Internet-dependent communication. Current Internet infrastructure is antiquated; it is expensive to maintain and expand, it has numerous vulnerabilities and high-impact points of failure, and can be rendered unusable for lengthy periods of time by natural disasters or other catastrophes. This current grid will eventually need to be replaced by a more modern, scalable, and adaptive infrastructure. The results of the projects research showed that implementing a library to allow for the creation of mobile peer-to-peer mesh networks could serve as a starting point for a transition from current Internet infrastructure to a more scalable, adaptive, and reliable Internet-independent network grid.

Development of The New Grid largely followed the Rational Unified Process, in which the development process is split into four phases: requirements gathering, system design, implementation, and testing. Most of fall quarter was spent outlining functional requirements for the system, designing possible methods of implementation, and researching similar solutions that seek to transition mass mobile communication to a newer, more modern network grid. The New Grid differs from similar solutions because it has been implemented as a modular library. Current systems that allow for off-grid mobile connection exist as independent applications with a defined context and predetermined usability scope.

We, the design team, found that implementing the system in the form of a modular library has multiple benefits. Primarily, this implementation would allow The New Grid to be deployed as widely as possible. Developers can both write applications around our library as well as include specific modules into existing applications without impacting other modules or introducing additional overhead into a system. Another benefit of deploying the system as a modular library is adaptability. The current, initial stable build of The New Grid uses Bluetooth Low Energy as its backbone for facilitating communication within large networks of mobile devices; however, this library could use any existing or future communication protocol to facilitate connection as long as a hook is written to allow The New Grid to interface with that protocol. Thus, The New Grid is not limited by which connection protocols currently exist, a property that other similar systems do not possess.

The New Grid can be used in any application that requires connection between users. The most common applications would likely be messaging, file sharing, or social networking. While developers may find a variety of uses for The New Grid, its primary purpose is to facilitate reliable connection and secure data transfer in an environment with a large user base. Achieving this goal was proven feasible through research and testing the library with a small cluster of Android devices communicating solely with Bluetooth Low Energy. Expanding this group of a few phones to a larger mesh network of hundreds of devices was shown to be feasible through testing the librarys algorithms and protocols on a large network of virtual devices. As long as developers seek to create applications that allow users to communicate independent of Internet infrastructure, The New Grid will allow smartphone users to communicate off-grid and hopefully spur a switch from infrastructure-dependent mobile communication to user-centric, adaptive, and flexible connection.

Computer Science and Engineering
Santa Clara University
June 8, 2016

Table of Contents

List of Figures

List of Tables

app has since been discontinued [4]. Moreover, current libraries and frameworks that attempt to facilitate peer-to-peer data sharing are not widely adopted, and include many restrictions on the types of communication protocols that can be used and the types of devices that are supported. Some peer-to-peer routing protocols require users to have extensive technical knowledge of their devices' abstracted functionality; for example, the Better Approach to Mobile Ad-Hoc Networking (B.A.T.M.A.N.) peer-to-peer routing protocol requires users to "root" their Android phones and re-write their operating system's kernel code [5].

1.3 Our Solution

We are proposing a mobile communication and data sharing solution that will allow Android users to remotely connect with each other without the need for Wi-Fi or internet infrastructure. This solution will be a library of peer-to-peer connectivity protocols that provides users multiple options for data transfer without an internet or cellular connection. The solution will also include a basic user interface as a proof-of-concept model that will allow non-developers to set up mobile peer-to-peer networks in the form of a messaging app. Users will be able to choose between sending messages to specific mobile users in a close range or broadcasting to all users within connection range. A user will thus be able to send personal data in a simple and efficient manner. By including options for controlling which users can receive data, we are also providing users increased data privacy when they desire it.

The solution we propose will be scalable to different data and network sizes. Additionally, the library can work with any communication protocol, even ones that have not been created yet. Our solution's hook interface will allow developers to write interfaces between our solution and any communication protocol, allowing the app to expand and include various methods of communication. Our solution will also include functionality for securely partitioning, hashing, and storing on any given network, creating numerous changes in the way users store and share data; rather than relying on a central data repository to store and provide information, users will store and share data themselves, reducing the amount of infrastructure required to maintain a global mobile network. In summary, our implementation will provide a consistent and expansive interface in an area of connectivity where no universal, convenient standard currently exists. With a peer-to-peer connectivity standard in place, mass mobile communication could see a transition from the current Internet infrastructure to a more scalable, adaptive, and reliable Internet-independent network grid.

Chapter 2

Design Strategy

Our development process focused mainly on setting and meeting specific goals. By keeping track of these goals and factoring them into each stage of the development, we made sure that our finished product satisfied the goals and requirements set for our system. Many of the decisions made were based on demonstrating feasibility of the system's functionality; many of the aspects of the system are not optimized, but were included or created to show that certain features were reasonable to include, or that certain functionality was feasible within the bounds of our library. We feel that it is important to highlight some of the major design decisions that we made and why these choices were important in both completing our system and delivering an effective product.

2.1 Requirements

The beginning stages of development on our system were spent detailing the requirements that our system's design must include. Requirements are divided into two categories: functional requirements, which entail the actual behavior of our system, and non-functional requirements, which attempt to quantify the nature of the system's functionality. These requirements helped us to keep a realistic scope for our project while also setting specific goals to meet for the final implementation of our system.

2.1.1 Functional Requirements

- The system will provide a method of transferring data between mobile device users independent of an Internet connection or cellular network.
- The system will provide data security, ensuring that users' data is protected if they desire.

- The system will encourage abstraction; other protocols should run using our libraries as an underlying infrastructure.
- The system will include a basic interface for non-developers to utilize our functionality.
- The system will automatically allocate and handle data storage on the users' end.

2.1.2 Non-Functional Requirements

- The system will be scalable and adaptable for mobile device networks of varying size and topology.
- The system's interface will be user-friendly for developers and non-developers.

2.1.3 Design Constraints

- Varying mobile device specifications and operating systems may hinder cross-platform compatibility.
- Signal strength and broadcast range will vary between different types of devices, which will affect connectivity and efficiency of file transfer.

2.2 Design Rationale

Throughout the course of development, we were forced to make many decisions that greatly shaped the process of development and the system itself.

2.2.1 Ease of development

Certain design decisions were focused on making development easier for potential developers who may be creating applications with library functions.

- **Library**: This library is meant to be a middleman between the application layer and the raw protocol layer. Instead of creating our system solely as a native app, where developers would be restricted by the features and functionality of the app itself, a developer can use the functionality of our library to drive the operation of their own applications. On the other hand, another enterprising developer can easily tweak or expand the existing functionality to suit their own specific needs without having an effect on existing applications.

- **Hooks**: The hooks give us the flexibility of not having to code this library for every single device and port it to every different platform. Instead, a developer can simply create a hook for a specific platform or communication protocol to make them compatible with our library.
- **C++**: Since our library is written in C++, which is a language supported widely on many different platforms, it can be incorporated into a wide range of existing applications and development frameworks, allowing the library to easily scale horizontally to extremely large applications. Additionally, C++ is a language with which many developers are familiar. Developers should have an easy time incorporating a C++ library into their applications.

2.2.2 Impact on final product

Other decisions had a greater impact on the system itself and how users could potentially experience our system. Here are some of the technologies that we felt had a major impact on the final implementation of our system.

- **BLE**: Bluetooth Low Energy (BLE) is the communication protocol that our system depends on. As mentioned in the problem statement, we are trying to address the issue of communication without Wi-Fi or a cellular connection. Bluetooth is available on virtually all modern phones which allows us to potentially reach an extremely vast user base [6]. In addition, BLE's low power consumption is ideal for emergency situations where energy may be at a premium and for potential deployment on older or less powerful mobile devices.
- **C++**: We chose to use C++ as our programming language because of its compatibility with Android's Native Development Kit, Android Studio, and iOS development libraries. This will allow us to pursue a cross platform implementation of our software protocol which to this point has not been fully realized.
- **Demo app**: Android NDK and Android Studio allowed us to create a mobile application that is fully compatible with the Android mobile operating system. This will give users a greater incentive to adopt our protocol because they will not have to have background knowledge of Android to install our protocol on their system.

- **Github**: Github will allow us to keep track of each stage of development, and revert to older implementations if need be. This allowed us to focus on individual development of modules and create a more fully realized implementation.

2.2.3 Why Create a Library?

Constructing the system in the form of a library, in which functionality is split up into independent sections, allows the system to be deployed much more widely than a complete application with a predetermined user context and scope [7]. This is achieved through:

- **Modularity**: Separate aspects of functionality are contained in different modules. These modules all work together, but can work independently of one another if necessary.
- **Flexibility**: Developers can write a new application around The New Grid or incorporate parts or all of the library into an existing application. Developers do not need to include all parts of The New Grid in their apps; if a developer is concerned about adding additional overhead to an app, that developer can choose which parts of The New Grid to include in an app.
- **Adaptability**: The New Grid can work with any communication protocol so long as a developer writes a hook to allow The New Grid to interface with that communication protocol. Once a hook is written between for a specific protocol, any developer can use The New Grid with that communication protocol.
- **Abstraction**: End users do not need to possess any technical knowledge of their device or computer networking in general to use the functionality of The New Grid. All technical functionality, such as the routing algorithm, network structure, and security measures, is not visible to the user.

2.2.4 Why Did We Decide to Develop for Android?

One of the largest design decisions that was made was the decision to design solely for Android phones. We determined that cross-platform development would be too difficult due to the Bluetooth hardware in iOS devices working unreliably with the Bluetooth hardware in Android devices. We were effectively forced to choose between developing for Android an developing for iOS; we chose

Android because we could develop more easily in C++, a language with which we were all familiar. Additionally, Android development tools are more

2.2.5 How Do We Find Nodes With A Routing Algorithm?

The decision to create a routing protocol was born out of an inability to find an existing one that suited the unique needs of an environment based around mobile communication. In chapter 3, we will go into specifics about the details and implementation of our custom routing algorithm.

2.2.6 Is A Demo Application Necessary?

Our decision to include a demo application as part of our system is important because it allows non-technical users to use our protocol for both personal and practical purposes. This greatly expands the potential range of users for our system while simultaneously serving as a proof of concept for the efficacy of our library.

Chapter 3

Library Implementation

This chapter outlines the overall architecture of The New Grid and describes the implementation of each aspect of the system. The structure and implementation of each module in our library is also discussed at length in this chapter.

3.1 System Architecture

The system architecture, shown in Figure 3.1, revolves around the philosophy that a modular design creates the most flexible and adaptable system. A modular design allows us to treat our library as a switch or router not only for one protocol, but for any protocol. The modules that support new communication protocols, known as "hooks," are the basis as to what will be the multi-protocol mesh network.

In addition to a module for different communication protocols, the system also includes modules for implementing secure connections and finding a reliable path through a network to a specified node.

3.2 Library Hooks

Hooks act as an interface between The New Grid Library and an individual communication protocol. The New Grid can support a new communication protocol with each new hook that is written for the library. The workflow between hooks and The New Grid can be seen in 3.2.

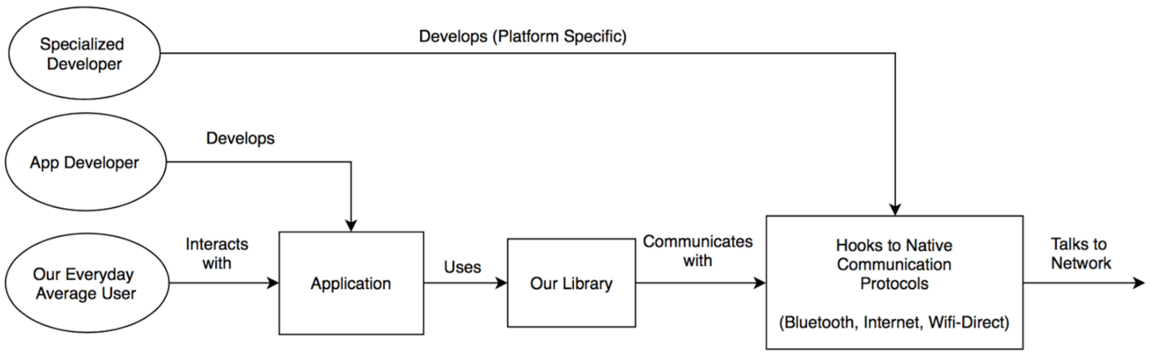

Specialized Developer
Develops (Platform Specific)
App Developer
Develops
Our Everyday Average User
Interacts with
Application
Uses
Our Library
Communicates with
Hooks to Native Communication Protocols
(Bluetooth, Internet, Wifi-Direct)
Talks to Network

Figure 3.1: Overview of The New Grid system structure and workflow

3.3 Android Demo System Flow

Functionality of The New Grid was demonstrated using three Android Nexus 5 devices running a demo application called T.N.G. Messenger. This app was a simple messaging app that utilized the routing algorithm and Bluetooth Low Energy hook to send messages from one device to another, using the third device as a relay.

3.3.1 Hooks for Android

When designing the Android application, we wanted to allow for the possibility of multiple hooks. Designing the hooks to include the library would lead to a lot of duplication of files, and was thus seen as undesirable. Because each of us was working on different parts of the project, we created an interface between the library and the hooks through Android's native Broadcast framework. When a hook receives a message, it sends a broadcast to the system telling the application that a message was received. The application will then either save the message itself, or send it to our library if the message is meant to be forwarded.

3.3.2 Library

Applications using our library can handle database management in multiple ways. We decided to achieve this with our demo app by keeping track of addresses; however, since BLE addresses change regularly, we had to implement UUIDs to uniquely identify each device on the network.

1. Broadcasts: The application handles all of the database transactions; the library and hook

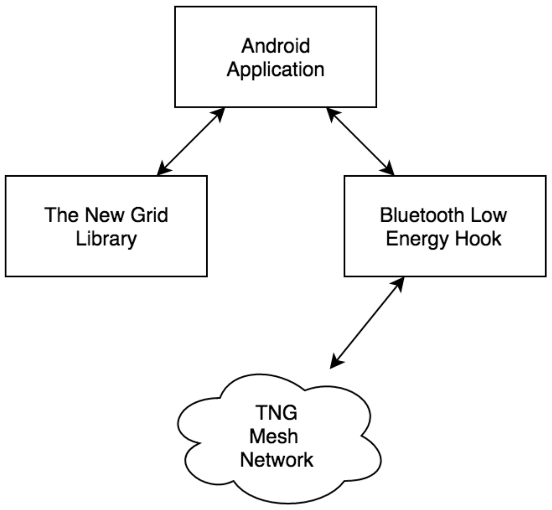


Figure 3.2: Android Hook/Application/Library Interface

simply notify the application of a change in database information through the Broadcast framework. The application stores database information and, based on the contents of the new information, forwards it to the specified module in the library. This was the method used in the demo application

2. Separate SQLite Databases: The library would hold the routing information about the device in its own database, and then the Application's db can access it and then communicate based off of the UUID.

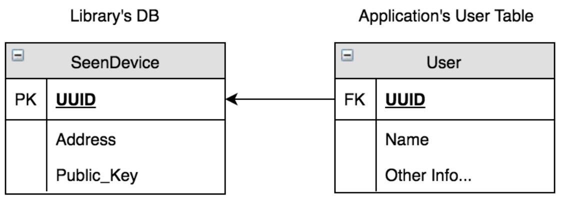


Figure 3.3: Android Database paired with The New Grid Library's separate Database

3.4 Security using TrusNet

The TrusNet security library, developed by Santa Clara University students Jonathan and Adrian Bedard, was included to provide security measures for The New Grid [8]. Only the modules of TrusNet that implement secure connections between individual devices were used. These modules generate public, private, and secret keys that are used by each device or each individual session.

The keys generated by TrusNet facilitate secure transfer from one device all the way to a specified recipient, regardless of how many intermediate nodes a file must pass through to ultimately reach the final recipient. When a file is sent, its contents are encrypted, and can only be decrypted by the final recipient. If a file is broadcasted to all devices on a network, the contents are not encrypted.

3.5 Custom Routing Algorithm

The routing algorithm used in The New Grid does not make any distinctions between nodes; all devices on the network are given equal permissions and equal importance. While this approach may not be as efficient as one in which certain nodes are given ”super-peer” status [9], this approach eliminates the risk of introducing high-impact single points of failure.

The algorithm focuses on recognizing a path to a specified end node through previously visited devices. It uses the diagram in Figure 3.4 to determine when and to whom packets should be sent.

3.5.1 Keeping Track of ”Neighbors”

Initially, the routing algorithm required devices to carry indices and keep track of network geography, a strategy that would allow messages to be delivered through as short a path as possible [10]. However, we soon discovered that because paths will change constantly in a mobile peer-to-peer environment, such information would become obsolete within mere minutes of being gathered.

Instead, the routing algorithm requires each device on the network to keep track of other devices that have recently been contacted. These devices are tracked by their UUID and kept in a list of ”neighbors.” Each device keeps track of a set amount of neighbors; this number is set by the creator of an application, and can be set to be either a constant value, or a value that changes as the network size changes. While a constant value of neighbors will reduce the amount of storage required by each device to maintain a good network map, a larger number will be necessary when a network expands and more devices communicate with each other more frequently.

Neighbors are stored in a list, with the most recent neighbor at the head of the list and the least

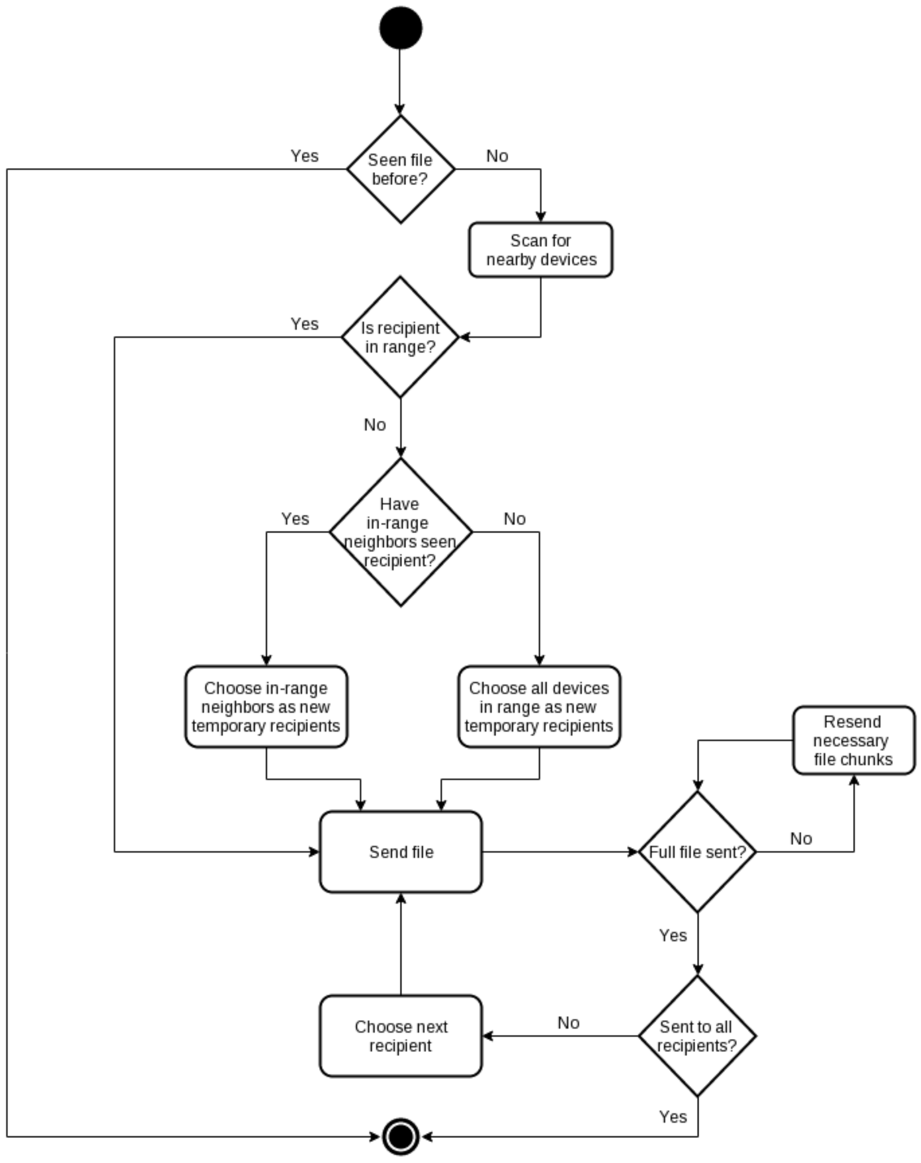

Seen file before?
Yes
No
Scan for nearby devices
Is recipient in range?
Yes
No
Have in-range neighbors seen recipient?
Yes
No
Choose in-range neighbors as new temporary recipients
Choose all devices in range as new temporary recipients
Resend necessary file chunks
Send file
Full file sent?
No
Yes
Choose next recipient
No
Sent to all recipients?
Yes

Figure 3.4: Routing Algorithm: Path Discovery and File Transfer

recent neighbor at the tail. Any time two devices communicate, each device updates the other as its most recent neighbor. If the list of neighbors is full, and a new neighbor must be added, the neighbor at the tail of the list is removed.

3.5.2 Determining Network Path

When a device needs to send a file, it follows the same steps regardless of whether it is the original sender or it is forwarding a received file:

1. Check to see if this file, marked by a sequence number, has already been received from this original sender. This step is important to minimizing message duplication, on of the largest sources of inefficiency in peer-to-peer networks [11].
2. Check to see if the recipient of the file is in range. If so, send all necessary files.
3. If the recipient has not come in range within a given amount of time, the device asks all neighbors in range if the recipient is one of their neighbors.
4. All neighbors in range are asked if the recipient is a neighbor. If this is true, the file is sent to the original device's neighbor, who starts this process from the beginning.
5. If no neighbors are in range, the device will flood the network with the data packet.

When a large message or file is sent, the message must be broken up into smaller chunks, each with its own header and unique sequence number. The first chunk for a large file will contain information regarding the number of chunks that need to be sent. Upon receiving all of these chunks, the recipient checks to see if the entire file has been received. If so, the file can be pieced together; if not, the recipient sends information back to the sender informing the sender which chunks need to be sent again.

The routing algorithm also contains procedures for new devices that enter the network. When a device connects to the network for the first time, it floods the network with its public key. This allows the new device to establish neighbors on the network, and notifies all other devices on the network of its presence.

3.5.3 Sequence Number-Controlled Flooding

If the initial steps to discover a path through the network fail, the routing algorithm resorts to sequence number-controlled flooding. Each device must keep track of recent messages seen from its neighbors so that when a new message is received, the device can check to see if it has already processed this message. To save space on each device, only a specified number of messages from each neighbor are saved.

3.6 File Management With The New Grid

The New Grid includes a simple file chunking and distribution mechanism that is native to the library. Files can be broken up, distributed, and stored throughout the network, and they can be just as easily downloaded and joined back together on any device that requests a download.

The key aspects of The New Grid's file management structure are upload and download requests. Figure 3.5 shows the process through which the library goes when a user uploads a file to a network. Users can upload files to the network that may be over the suggested minimum file size; if a file is over the limit, then the library divides the file up into a number of chunks that are the minimum file size or smaller and computes a hash value based on the file name and the chunk number. This hash value is used to determine which device will receive a given chunk of the original file.

As Figure 3.6 shows, users have the option to request a file that is potentially available on the network. Because the locations of the files are noted when the file is uploaded, locating the files on a network at any time is easy. Whenever a download is requested, a network report is generated, which returns the location of all chunks belonging to that file on the network. If the entire file is found to be on the network, then a user can get the individual pieces from the users that have them. Once all of the individual pieces have been downloaded by the requesting user the files can be pieced together into the full, original file.

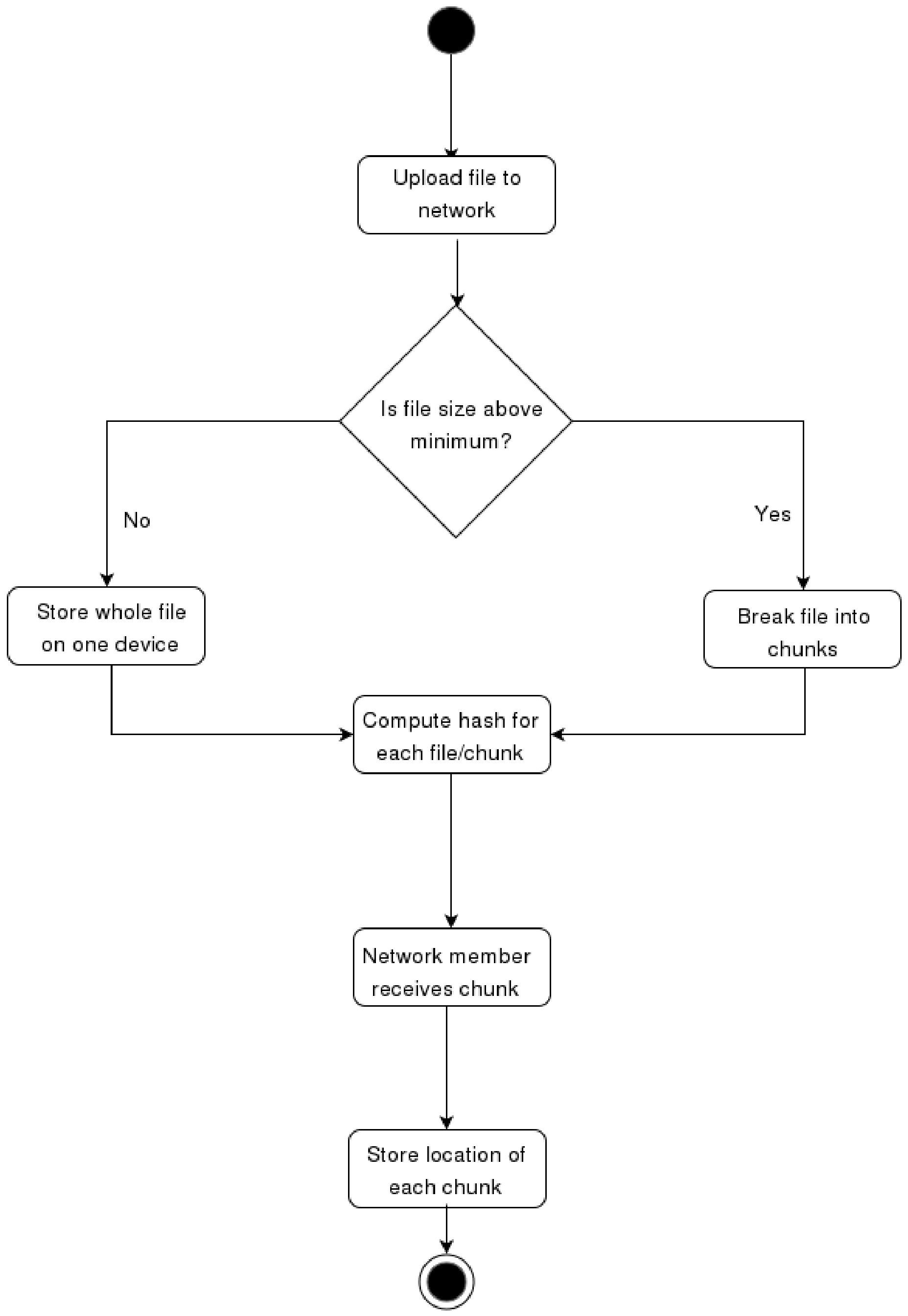

Upload file to network
Is file size above minimum?
No
Yes
Store whole file on one device
Break file into chunks
Compute hash for each file/chunk
Network member receives chunk
Store location of each chunk

Figure 3.5: How Upload Requests are Processed

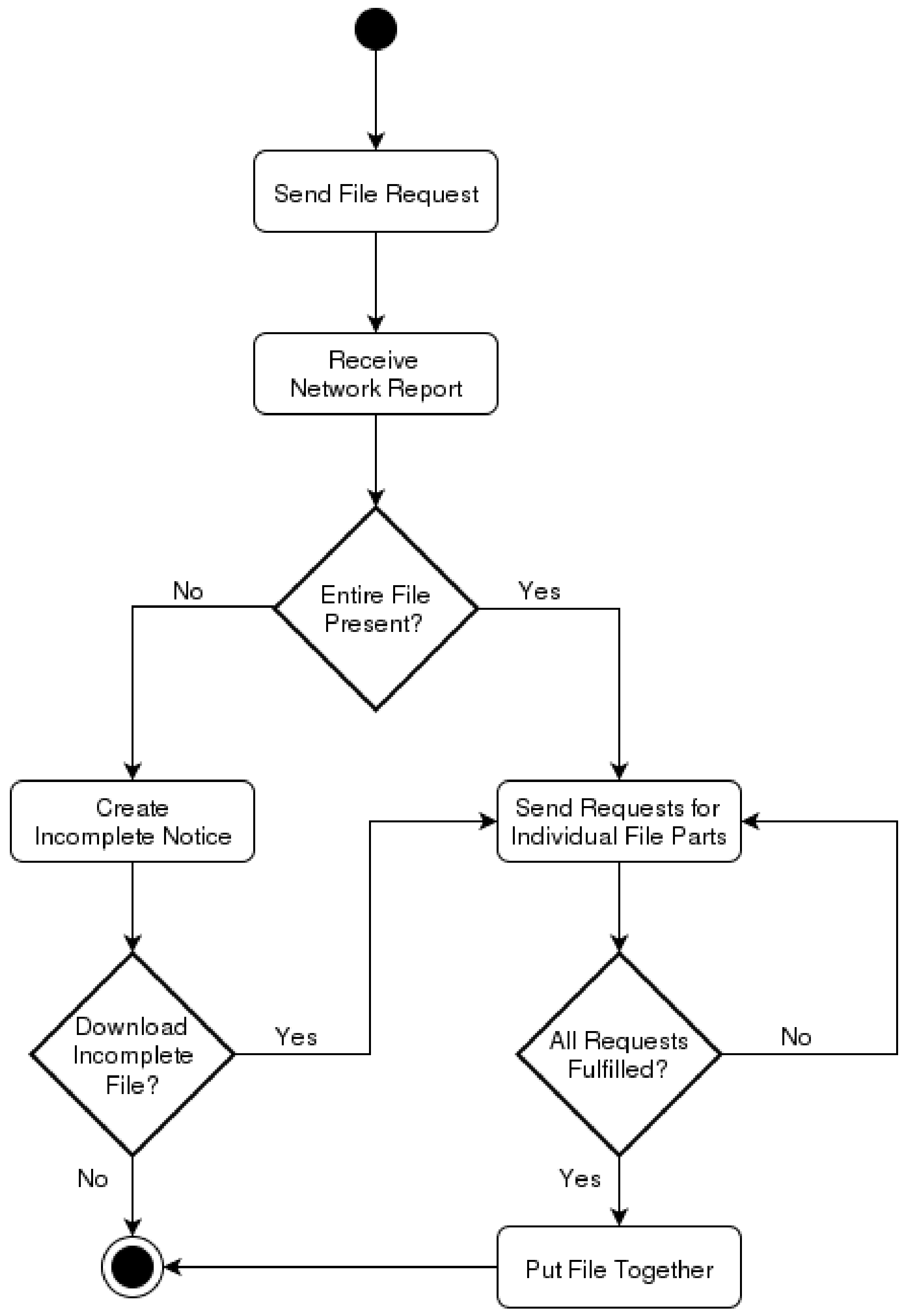


Figure 3.6: Downloading a File from a Network

Chapter 4

Demo Application

To prove that The New Grid functions properly, we needed to write a simple application that successfully used our library. We wrote a simple messaging app called T.N.G. Messenger to demonstrate the functionality of The New Grid during our Senior Design Conference session.

4.1 Integration with The New Grid

The core of T.N.G. Messenger is written in Java, like any native Android application. A hook for Bluetooth Low Energy was written so that this communication protocol, included on the Android Nexus 5 test devices, could interface with the app and The New Grid.

4.2 Conceptual Model

We decided early in development that creating a messaging app would be the simplest way to prove functionality of The New Grid. Initially, the application contained additional functionality, such as creating and joining networks of connected devices; this was cut once we decided to limit all applications to having only one active network. However, this functionality may be added for a future release. A conceptual model for adding and joining networks is shown in figure Figure 4.1.

4.3 Initial Build

The current build of T.N.G. Messenger contains enough functionality to support sending messages to specific devices on a network. The application allows users to scan for nearby devices, remember devices that were previously contacted, and send messages to previously contacted devices on the

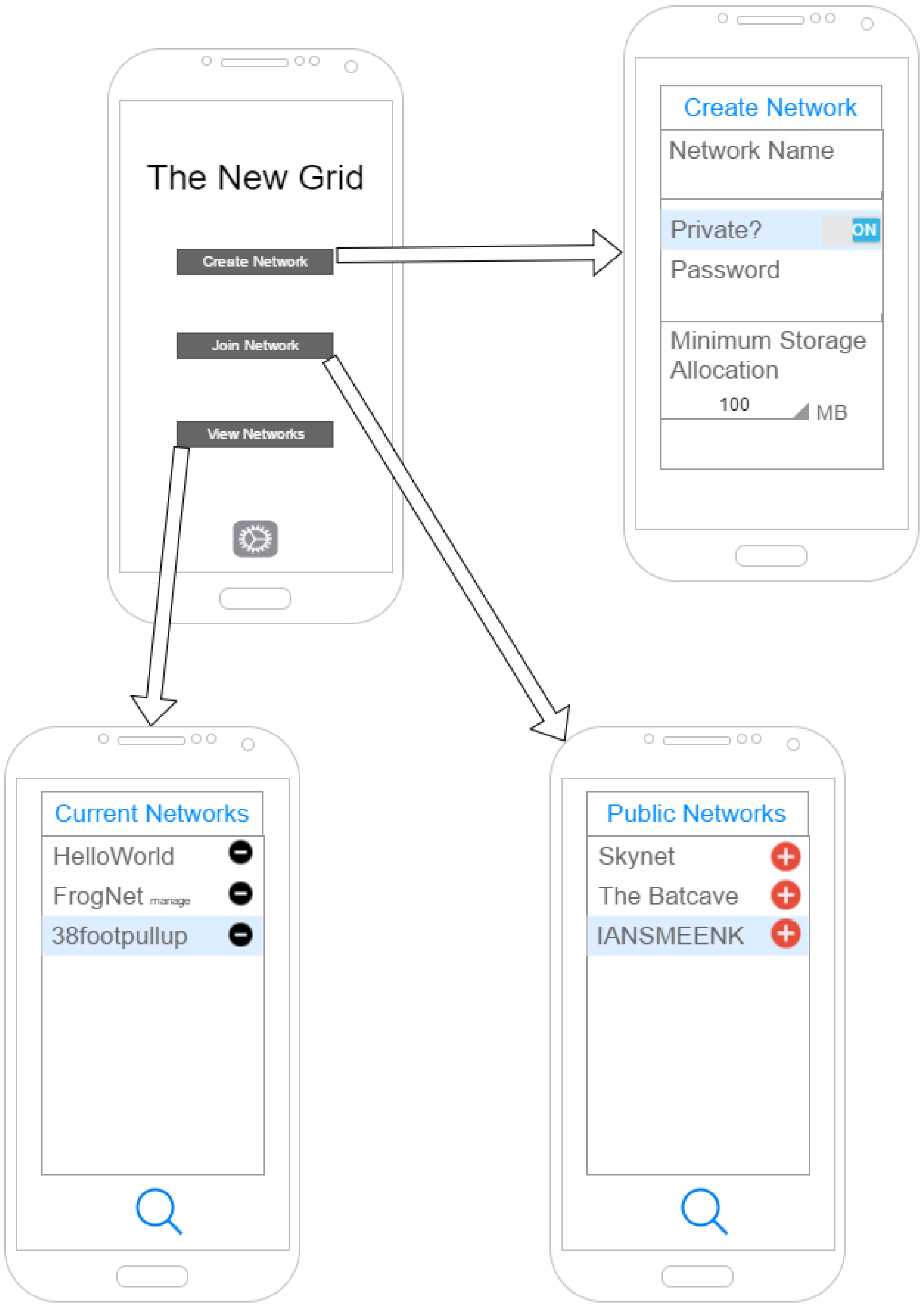
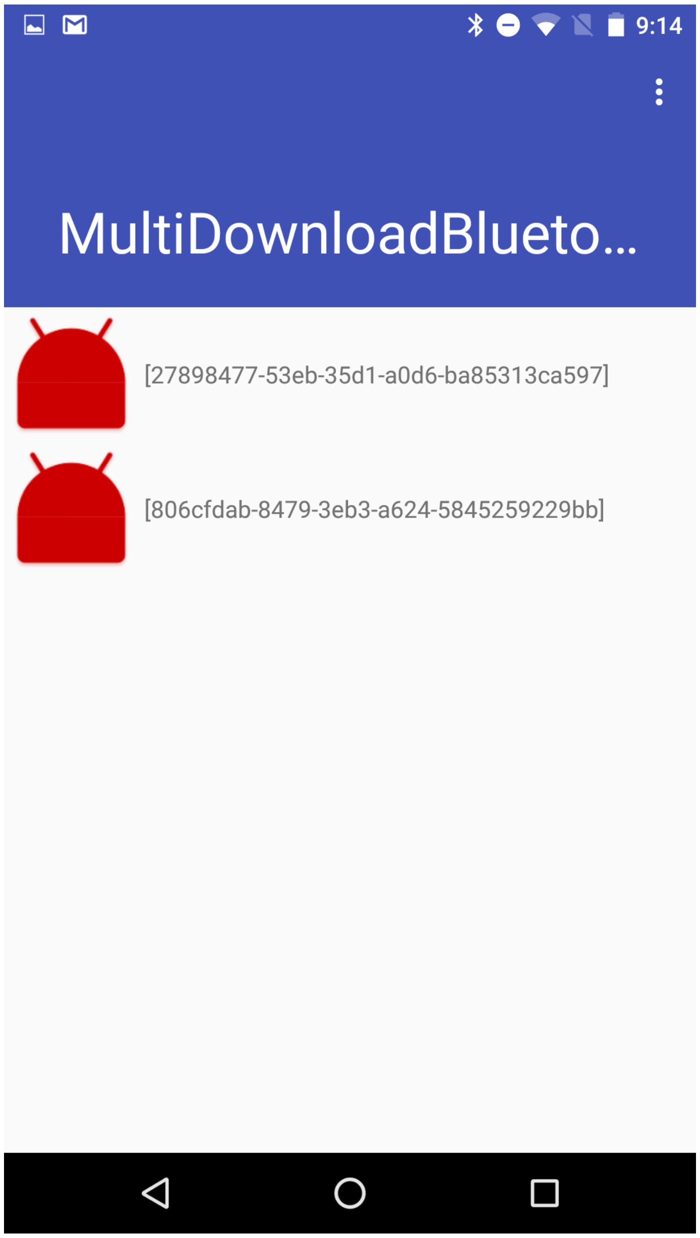


Figure 4.1: Original conceptual model for creating and joining networks

network or any device in range.

Before the user can send messages, the application will scan for users in range. The screen in

Figure 4.2 will appear, and the user can choose to send a message to any of the devices found.

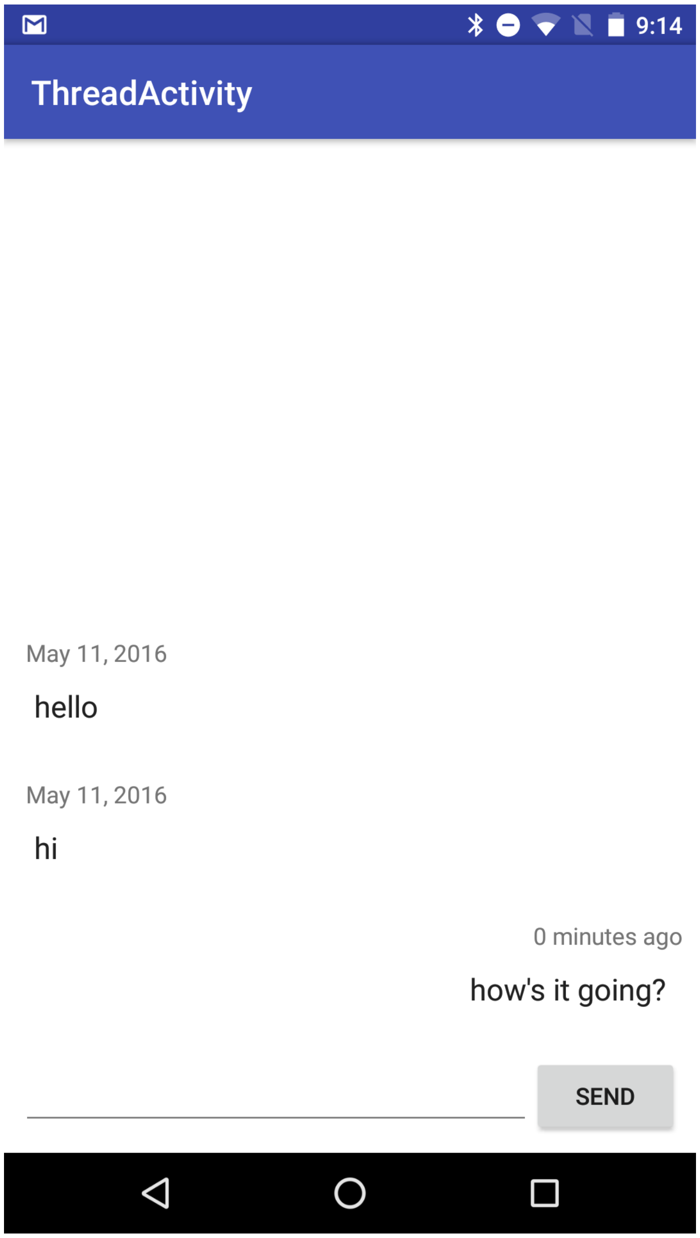


[27898477-53eb-35d1-a0d6-ba85313ca597]

[806cfdab-8479-3eb3-a624-5845259229bb]

Figure 4.2: T.N.G. Messenger displaying devices in range

The screen for conducting, sending, and displaying received messages is shown in Figure 4.3.



May 11, 2016

hello

May 11, 2016

hi

0 minutes ago

how's it going?

Figure 4.3: T.N.G. Messenger's messaging screen

Chapter 5

Development Process

The development of The New Grid roughly followed the Rational Unified Process, a process by which development is split up into four discernible phases: Requirements Engineering, Design, Implementation, and Testing.

5.1 Development Timeline

The current build of The New Grid was developed between May 2015 and June 2016. After June, development will be continued by Vincente and passed on to a new team of Santa Clara University Computer Science and Engineering students.

5.1.1 Spring and Summer 2015

Development of The New Grid began in the late spring of 2015. At this point, the development team consisted only of Stevie and Andrew, who began outlining requirements for a system that would facilitate connection between large networks of mobile devices without Internet connection. Development was mostly halted during the summer due to the team not being in close proximity for much of the summer months. However, during this time, Stevie and Andrew worked independently to create loose diagrams of system architecture and rough drafts of requirements documents.

Member	Color
Andrew and Stevie	⬛
Andrew	🟩
Stevie	🟥
Vincente	🟫
All	🟦

Table 5.1: Color key for the project development timelines

5.1.2 Fall 2015

At the end of September, Vincente formally joined the development team. During this time, between September and December of 2015, the requirements for the system were finalized. Most of the important design decisions had been made, as well; we decided that a modular library would be best for the project, and we created a list of what modules needed to be included in the library.

A final draft of the design document for The New Grid was published in December 2015.

Task	Week									
	1	2	3	4	5	6	7	8	9	10
Initial Research	⬛	⬛	⬛	⬛	⬛					
Problem Statement	⬛	🟦	🟦							
Requirements Gathering				⬛	⬛	🟦	🟦	🟦	🟦	🟦
Design Document						🟦	🟦	🟦	🟦	🟦
System Architecture							🟩	🟩	🟥	🟥
Initial Testing									🟦	🟦

Table 5.2: Development timeline for Fall 2015

5.1.3 Winter 2016

At the beginning of January 2016, the project had been renamed to ”The New Grid.” The first task of 2016 was to deliver a presentation for a design review, at which our requirements, system architecture, and design decisions were evaluated by Santa Clara University faculty and industry professionals.

By the middle of January, the three members of the development team began taking on different, specialized tasks:

- Andrew was tasked with researching a viable routing algorithm and working with the TrusNet development team to incorporate parts of the TrusNet library into The New Grid.
- Stevie was put in charge of creating a module to partition files, then hash and store each file fragment on various devices throughout a network.
- Vincente was responsible for developing in Android NDK and SDK and ensuring that the library could interface with Android devices.

During this time, we also continued working together on various aspects of the system. All three of us contributed to the design of the demo app, and began work on the project's final thesis.

Towards the end of February, Vincente discovered through testing that the routing algorithm

Andrew had selected, the Better Approach to Mobile Ad-Hoc Networking (B.A.T.M.A.N), was incompatible with The New Grid. B.A.T.M.A.N. required its users to have rooted phones and custom Android kernel code, a requirement that violated The New Grid's nonfunctional requirements relating to ease of use, abstraction, and accessibility. After further research, Andrew was unable to find a routing protocol that would fit the needs of The New Grid, so he designed one from scratch.

After consulting Dr. Amer, we ordered three test devices during March to be used for testing and development during the spring. The devices were Android Nexus 5 phones, each with the same internal Bluetooth radio and slightly varying device specs.

Task	Week									
	1	2	3	4	5	6	7	8	9	10
Design Review	🟦	🟦								
Protocol Research				⬛	🟦	🟥	🟥	⬛		
File Partitioning		🟩					🟥	🟥	🟥	🟥
Device Testing							🟫	🟫	🟫	🟫
Routing protocol				⬛	⬛	🟩	🟩	🟩	🟩	🟩
Android Development				🟫	🟫	🟫	🟫	🟫	🟫	🟫

Table 5.3: Development timeline for Winter 2016

5.1.4 Spring 2016

In late March, work on the new routing protocol had started. The routing protocol underwent a number of revisions; the design was scrapped and restarted twice before a final design was finally selected and implemented.

Most of April was spent preparing for the Senior Design Conference, which took place on May 12. However, development on the routing protocol and demo application continued.

By the end of April, Stevie had finished developing the file partitioning module. Vincente was able to get the three test devices to communicate with each other using the new demo app. We had finalized our demo app in early May, in time to demonstrate the system at the Senior Design Conference.

After the Senior Design Conference, our focus shifted to writing our thesis. However, development on certain aspects of the system still had to be finalized. The routing algorithm was finalized towards the end of May, and additional connectivity and user experience testing will be carried out through the end of the academic year.

Task	Week									
	1	2	3	4	5	6	7	8	9	10
Design Conference	🟦	🟦	🟦	🟦	🟦	🟦	🟦			
Unit Testing	🟥	🟥	🟫	🟫	🟩	🟦	🟦	🟦	🟦	🟦
User Testing	🟦	🟦	🟦	🟩	🟩	🟫	🟫	🟩	🟩	🟥
Documentation	🟥	🟩	🟫	🟥	🟩	🟫	🟥	🟩	🟫	🟦
Senior Thesis	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

Table 5.4: Development timeline for Spring 2016

5.2 Risk Analysis

As with any major development project (software or otherwise), we had to consider specific risks that may hinder the development of our system. Table 5.2 details the primary risks that we face as well as the consequences and potential strategies that we devised to mitigate the potential impact of said risks. We included the probability of the risk occurring (on a scale between 0.0 and 1.0), the severity of the effect that the risk would have had in its worst case (between 0 and 10), and the overall impact the risk and its consideration had on our project (calculated as a product of the probability and severity). Risks are ordered from highest impact to lowest impact. In section 8.1 we discuss some of these risks in hindsight and the general conclusions that we took from dealing with these risks and their consequences.

5.3 Testing

Testing is an extremely important part of any project, especially one of a more complex scope like The New Grid. We formed a comprehensive plan to test our system constantly throughout the development process. By testing each module of our system, we guaranteed that individual modules worked independently of each other. By testing the whole system every few weeks, we guaranteed that all modules interacted properly. The goal of our test plan was to keep the development process moving as smoothly as possible, and we are satisfied with the results of our testing.

Risk	Consequence	Probability	Severity	Impact	Mitigation Strategy
Testing fails to capture all errors/problems	Project may be released with errors that will hinder the user experience	0.9	5	4.5	Make sure requirements are always met; formulate exhaustive test cases that cover both unit and acceptance testing
Running out of time	Deadlines approach and our system must be released in an incomplete state	0.2	10	2.0	Stick to development timeline while remaining aware of impending deadlines and new developments, prioritizing features and focusing on those that are necessary for our system's operation and success
Illness or other personal tragedy	Other team member(s) may have to pick up more work for incapacitated team member(s) during a specific time frame	0.2	5	1.0	Divide up work evenly, being sure to not put a significant portion of our system on one team member
Failing to meet requirements	System must be released in an incomplete form and will not behave as advertised	0.05	10	0.5	Remain aware of changing requirements, make sure that requirements are addressed first and foremost at every stage of development

Table 5.5: Risk Analysis table for system

Chapter 6

User Manual

This user manual corresponds to the initial alpha release (0.1.0) of The New Grid, created by Vincente Ciancio, Andrew Dobbins, and Stevie Sanz.

6.1 Introduction and Purpose

Like every other library, The New Grid provides a guide to downloading, including, and using the library. However, with The New Grid, the instructions for including the library are somewhat different from most libraries. The library is based in C++; therefore, applications on different platforms will have to include the library in separate ways.

6.1.1 Download and Installation

To begin, download The New Grid Library from GitHub (https://github.com/ajdobbins/TheNewGrid) and unzip the files into your project directory. Then, find a hook for your specific operating system and protocol, and include that in your project. From there, you'll have to set up the interface between your application's native language and C++. Refer to the interface's user manual for setup instructions; each hook will have unique instructions. For example, Android uses the NDK (Native Developer Kit) to interface its Java Application code with C++ native libraries. Refer to the Android NDK documentation for help setting up C++ libraries with Java applications.

6.1.2 Setting up the Hooks

Each operating system will have its own toolkit for handling broadcasts and dealing with inter-module communication. Refer to each hook's manual, as each hook will have different setup requirements. Hooks will most likely be developed in the application's native language, so you will likely not have to handle setting up an interface between the hook language and the native app language.

6.1.3 Creating / Joining an Existing Network

Creating a network is simple and easy. Before we start creating networks, though, we should clearly define what a "network" is: a collection of all devices communicating through an application. These networks are defined in practice by devices advertising an Application UUID through their communication protocol hook. This UUID is defined by the developer-written application, not the library. This UUID must be unique, and must remain constant for all users of the application.

Because each application has one unique UUID, and any device using the app can communicate with any other device using the same app, only one network can be created per application. This does not prohibit or restrict the ability for developers to include support for creating "subnetworks," or smaller groups of devices using an app, within an application's network.

To join a network, a device must simply establish communication with another device on the network. This connection is established by all new devices automatically flooding the network with their public key and UUID when the application is first installed and run on the device.

6.1.4 Additional Documentation

Organized, sorted documentation for each module can be found in both the source code and the included documentation pages, found in the "docs" folder. These documents can be viewed with any text editor.

6.1.5 Legal Information

This software is provided open-source, free of charge on GitHub. For questions, please contact Andrew Dobbins at ajdobbins94@gmail.com.

Chapter 7

Ethical Analysis

Because our solution is being deployed as a library, the possibilities for its implementation are virtually infinite. Although this project was started with the intent to increase mobile connectivity and decrease dependability on large scale infrastructure, we know that not all developers that incorporate The New Grid into their applications will use it for that purpose.

7.1 Copyright Infringement

Developers can take advantage of The New Grid's file storage module to implement mobile networks dedicated to storing large amounts of illegally obtained data on various users' phones. Developers that create these types of apps can easily create mobile versions of large-scale torrenting websites. These websites serve as hubs for users to share and store large quantities of pirated digital media [13] [14]. These websites operate by implementing file sharing techniques similar to what The New Grid offers: large, often multi-gigabyte files, such as entire HD movies or full seasons of TV shows, are partitioned into millions of small "chunks." Each of these "chunks" is stored on a different user's personal computer or other dedicated storage device. When a user wants to download a file, each small file fragment is downloaded from a different location. The file is pieced together once at least one copy of each fragment has been downloaded.

We have put measures in place to combat the creation of such networks. The New Grid's file storage module limits the file size and total space a phone can dedicate to storing network files. Additionally, natural constraints imposed by the mass mobile environment make large-scale file storage difficult to implement. Most phones do not have high-capacity solid-state disk drives; most devices can only hold up to fifty gigabytes of user data, and thus would not have enough space to hold large amounts of file fragments. Because The New Grid allows for communication independent of Internet infrastructure, the lack of a common, constant, public means of communication will exponentially increase the amount of time required for all file fragments to be located on and sent through a large mesh network. Download speed would be dependent on how often different users come into range of each other, a phenomenon that is difficult to predict.

7.2 Data Privacy

As mentioned in Chapter 2, all data sent over any network using The New Grid must be encrypted and secured. We made the decision early in development to include data security as a functional requirement, opting to make all communication with The New Grid secure rather than putting responsibility for security on individual developers. Because information sent from one user to another will likely travel through other users' devices, all data sent must be encrypted. If a user was to send unprotected data through a mesh network, every device that relays that data on its way to the final recipient would be able to read, copy, and use that data.

We decided that, as the creators of the source software of The New Grid, it is our responsibility to include security measures with our routing algorithm. We were generously given permission to use the parts of the TrusNet library that we needed in order to make communication between mobile devices using The New Grid protected. We have explored adding additional security measures, such as incorporating responses to malicious activity, as seen in systems like Indra[15], which incorporate "watchers" to monitor suspicious activity on a network and "reporters" to notify access controllers of such activity.

Implementing tight security measures on peer-to-peer networks is especially critical due to the high presence of malicious users present on such networks. Due to the lack of central authority in peer-to-peer networks, and the relative ease of joining such networks, data thieves tend to target peer-to-peer networks more often than other types of networks [16]. Protecting The New Grid users against such security vulnerabilities becomes an important task in a network environment with a high presence of malicious activity.

7.3 Disaster Response

At the beginning of development, we came up with a list of situations in which an application using The New Grid would be helpful in a social situation. An example that we used during the Senior Design Conference was a hypothetical situation in which an earthquake of large magnitude strikes the San Francisco Bay Area, likely rendering cell towers and other Internet infrastructure temporarily unusable.

In a situation like this, Internet may still be available in the form of Wi-Fi hotspots. However, the relays that are still operational in the aftermath of an earthquake would be overwhelmed with millions of people attempting to communicate with loved ones, friends, and acquaintances, as well as first responders trying to coordinate efforts to areas that need immediate relief.

The New Grid would allow anyone in an affected area to immediately act as a relay, switch, and router on a large-scale peer-to-peer network. Rather than waiting for technicians to repair damaged infrastructure, or waiting for temporary connectivity solutions to be deployed by outside agencies, The New Grid users can immediately begin communicating through peer-to-peer networks that have not been damaged by the natural disaster. Thus, The New Grid acts as a temporary disaster relief connectivity solution that can be deployed immediately from within an affected area.

7.4 Promoting Connectivity

The New Grid can be helpful for cities in areas of the world that do not have up-to-date, well-maintained infrastructure. As noted in the Introduction, the expansion of Internet infrastructure has been fast outpaced by increasing global access to smartphones and Internet connection [17]. This is illustrated in Figure Figure 7.1, which shows the number of secure Internet servers per thousand people in each country.

Attempting to rapidly construct Internet infrastructure in such regions would be irresponsible an impractical. Not only would this process be expensive and time-consuming, it may introduce opportunities for leaders of developing nations to exploit free or cheap labor. Additionally, maintenance of Internet infrastructure requires the presence of trained technicians. Deploying technicians to cell towers in rural areas would be extremely difficult and expensive.

The New Grid provides smartphone users across the world an alternative to communicating without Internet infrastructure. Rather than relying on Internet servers that may be located thousands of miles away to facilitate communication, smartphone users can use their own personal devices to facilitate communication, eliminating dependence on large networks of physical infrastructure.

The New Grid can not only help people communicate, it can also help devices communicate. As the Internet of Things expands to include a diverse range of devices in a wide range of locations, communication between individual devices will increase. Rather than implementing more common data repositories through which these devices will communicate, devices can communicate directly

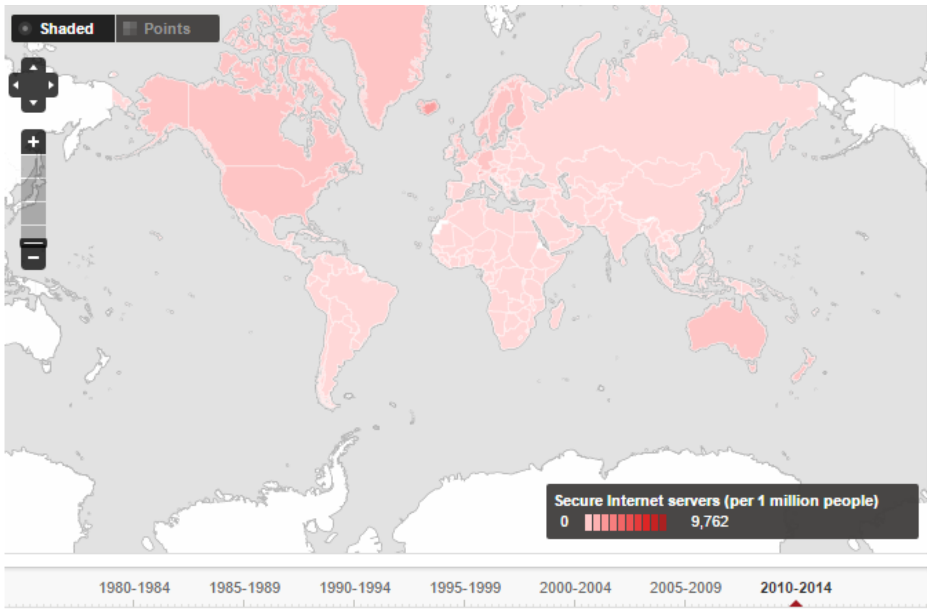


Figure 7.1: Secure Internet servers per thousand people [18]

with each other using peer-to-peer network technologies [19]. We hope that The New Grid can help facilitate communication between devices on the Internet of Things. Storing all network information in one location introduces two key vulnerabilities: (1) if the repository goes down, no device on the network can communicate with another; and (2) if the repository is compromised, an intruder can gain access to any device's information by infiltrating a single point.

Chapter 8

Conclusion

8.1 Lessons Learned

The development of a system such as The New Grid is extremely complex and can be impacted by numerous outside factors. In Table 5.2 we detailed some of the specific risks that we were forced to consider during the development process.

Because we are working on a project with a fixed deadline, we were forced to prioritize certain features that would be included in the final release. The concept of ”running out of time” is not necessarily meant to be taken literally, but is meant to keep us aware of deadlines, staying within the scope of the project, and finishing the necessary tasks for creating a finished product. Likewise, the risk of ”failing to meet requirements” forced us to ensure that the development of our system was focused around the requirements set forth at the beginning of development. Because these requirements stayed constant throughout development, we were able to gauge our progress and make sure that our design was fulfilling these requirements. Thankfully, illnesses were not severe at any point and no personal tragedies befell any of us during development; we were able to focus fairly consistently on development throughout the year.

One setback that occurred had to do with the alteration of our design itself. During February 2016, we decided that the existing routing protocol and library around which we planned to build our system violated our requirement that the system be user friendly, as it required the user to root their Android device (as mentioned earlier in subsection 5.1.3). This setback negated weeks of testing, reading documentation, and general purpose research on routing protocols and mobile communication. In addition, the decision to synthesize our own protocol created additional work for the team. While we were aware of the consequences of such a mistake with regards to lost time, we could have definitely spent some more time researching the nature of the original routing protocol's compatibility with Android devices.

Testing is always tricky, as we can never be completely sure if our code and therefore our system is bug-free. We can only test for the presence of bugs, not their absence; so to do our best to address the risk of buggy code, we started off testing early and often, forming exhaustive test cases that covered as many potential situations that we could think of. This started primarily with testing of the lower level code we were producing for the library itself, the communication protocols we planned to utilize, the Android phones with which we were using to test communication, and our demo application. For a more thorough discussion on our testing strategy and methodology, see **??**.

When we reached the actual implementation phase of our design, we adopted more independent roles in which certain members focused on specific aspects of the system. Given the fact that our project included writing a library and abstraction between the library functions, hook interface, and demo application, this was an extremely effective strategy that allowed us to complete the various parts of our library in parallel rather than waiting until development on one part had finished to start developing another part. By extension, we were able to start more thorough testing for higher level modules at an earlier stage with less worry about compatibility issues that may have broken the system. This also, in a way, approximated the way in which developers would approach The New Grid. A developer who is attempting to use our library may not need to use all modules; for example, a developer may only be trying to send a message between two phones and may already have storage and routing mechanisms. This manner of development increased modularity in our end system, as the various aspects of our system needed to be compatible with but not necessarily rely on one another.

8.2 Future Plans

The development team has set out a number of tasks to be completed in the future, including:

- Enhancing security to include incentives for preventing free-riders on file sharing networks [20]
- Refining the existing routing algorithm
- Implementing support for iOS devices
- Implementing native support for creating, joining, and managing multiple subnetworks within an application network
- Writing hooks for existing communication protocols, such as Wi-Fi Direct

Bibliography

[1] Kaustubh S. Phanse and Johan Nykvist, *Opportunistic wireless access networks*, Proceedings of the 1st international conference on Access networks, p.11-es, September 04-06, 2006, Athens, Greece.

[2] Conti, M. and Giordano, S., ”Mobile ad hoc networking: milestones, challenges, and new research directions,” in *Communications Magazine*, IEEE , vol.52, no.1, pp.85-96, January 2014

[3] How to use AirDrop with your iPhone, iPad, or iPod Touch - Apple Support. https://support.apple.com/en-us/HT204144

[4] Bump Blog. http://blog.bu.mp/

[5] Tweaking - batman-adv - Open Mesh. https://www.open-mesh.org/projects/batman-adv/wiki/Tweaking

[6] ”The Story Behind Bluetooth Technology.” *Bluetooth.* Bluetooth SIG, Inc., 2015. Web. 17 Oct. 2015.

[7] Meyer, Bertrand (1997). Chapter 8: The run-time structure: Objects. *Object-Oriented Software Construction, second edition.* Prentice Hall.

[8] GitHub - JonWBedard/WifiRC_Builder - Contains scripts necessary to build WifiRC programs and libraries. https://github.com/JonWBedard/WifiRC_Builder

[9] Wolfgang Nejdl, Martin Wolpers, Wolf Siberski, Christoph Schmitz, Mario Schlosser, Ingo Brunkhorst, and Alexander Lser. 2003. Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks. In *Proceedings of the 12th international conference on World Wide Web* (WWW ’03). ACM, New York, NY, USA, 536-543.

[10] A. Crespo and H. Garcia-Molina, ”Routing indices for peer-to-peer systems,” *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 23-32.

[11] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. 2002. Search and replication in unstructured peer-to-peer networks. In *Proceedings of the 16th international conference on Supercomputing* (ICS '02). ACM, New York, NY, USA, 84-95

[12] Gorski, Christopher, Charles A. Greiner, and Arthur R. Hair. Method and System for Establishing a Trusted and Decentralized Peer-to-Peer Network. DMT Licensing, L.L.C., assignee. Patent US20140304501 A1. 29 Oct. 2014. Web.

[13] Alexander, P. J. (2002). Peer-to-peer file sharing: the case of the music recording industry.' *Review of Industrial Organization*, 20, 15161.

[14] Cunningham, Brendan M., Peter J. Alexander, and Nodir Adilov (2001) The Napster Music Community, Working paper.

[15] R. Janakiraman, M. Waldvogel and Qi Zhang, "Indra: a peer-to-peer approach to network intrusion detection and prevention," *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*, 2003, pp. 226-231.

[16] Tahta, Ugur E., Ahmet B. Can, and Sevil Sen. "Evolving a Trust Model for Peer-to-Peer Networks Using Genetic Programming." *Lecture Notes in Computer Science* 8602 (n.d.): 3-14. textitSpringer.com. Springer Berlin Heidelberg, 29 Nov. 2014. Web. 18 Nov. 2015.

[17] Poushter, Jacob. "Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies." Pew Research Center, February 2016.

[18] Secure Internet Servers (per 1 Million People) — Data — Map. World Bank Group.

[19] Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions." *ScienceDirect* 29.7 (2013): 1645-660. Web.

[20] Michal Feldman, Kevin Lai, Ion Stoica, and John Chuang. 2004. Robust incentive techniques for peer-to-peer networks. In *Proceedings of the 5th ACM conference on Electronic commerce* (EC '04). ACM, New York, NY, USA, 102-111.