Enabling Ubiquitous Applications using Existing Infrastructure

Student Name: Digvijay Singh

IIIT-D M.Tech in Computer Science Engineering

Indraprastha Institute of Information Technology
New Delhi

Thesis Committee
Pushpendra Singh (Chair)

Submitted in partial fulfillment of the requirements
for the Degree of M.Tech. in Computer Science

Keywords: Ubiquitous Applications, System Design, SNMP, WiFi Traces,

Certificate

This is to certify that the thesis titled **Enabling Ubiquitous Applications using Existing Infrastructure** submitted by **Digvijay Singh** for the partial fulfilment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by him under my guidance and supervision at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Pushpendra Singh
Indraprastha Institute of Information Technology, New Delhi

Abstract

A key factor limiting the wide adoption of ubiquitous computing enabling systems is the need for deployment of expensive sensors and hardware to provide contextual information. These sensors deployments are not commonly used. Moreover extra effort and costs in maintenance is required in these deployments. In this work we explore use of techniques through which to obtain contextual information using existing infrastructure. WiFi networks and hubs have become common in public places, shopping malls, campuses, corporate offices and other industries. In this work, we try to exploit the presence of these networks to provide location information of a WiFi device. This enables many opportunities and applications. We first present our system, which collects, stores and exposes this information in real-time. Next we present deployed ubiquitous applications built using this system and also explore potential ubiquitous applications which may be enabled using this system.

Acknowledgments

I would like to thank my advisor, who has guided me in making this system a reality.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the widespread use of tablets, mobile phones and laptops, Mark Weisser's vision of the 21st century computer is becoming more of a reality day by day. Ubiquitous or pervasive computing - where everything is seamlessly connected and made to work around human beings, has formed the umbrella of multiple research fields: distributed computing, mobile computing, localization, context-aware computing, sensor infrastructures, networks, artificial intelligence. With the increasing number of smart phones and tablets, technology has become an integral part of human life. At the same time from needing to sit in front of a desktop computer around 20 years ago, to having wearable technology with the same computational prowess, if not more, has led to many more possibilities.

One of the key requirements highlighted by Mark Weisser [1] of reaching that future was the ability to localize a user. Localization helps in various scenarios - to know where in a burning building are people trapped or which part of the museum you are in, or in knowing the path to nearest grocery shop. As and how technology has become more readily available, new techniques have been used and researched upon to identify the location of a person [2] [3] [4]. Added to this the boom in the number of mobile phones, and other devices carried by a person, have further paved the way for research in localization. This includes the recent flurry of wearable devices being introduced into the market, which paves the way for new research. Localization has been carried out through devices such as Global Positioning Satellites(GPS), cell tower triangulation, beacons, cameras, ultrasonics, proximity detecting sensors, RFIDs and many other technologies.

Outdoors localization has reached very high accuracies using GPS , which has a resolution in centimetres [5] (In these cases, military satellites are used to localize an object or oneself. Publicly available satellites provide co-ordinates with error bounds usually starting from 5 me-

tres). GPS cannot be used in the case of indoor localization because of the weak signal received from the GPS satellites inside most buildings [6].

Indoor localization is still an open problem and is being actively researched. Indoor localization helps in scenarios like finding one's way in a new or unknown building, or for providing location aware services in indoor environments. These can also help in space management, navigation, give new information on user mobility patterns, emergency response plans and much more. Medical Care, social networking, environment modelling, museums and other public places, opportunistic motion capturing, etc. have been identified by Mautz [4] as areas which require or benefit from indoor localization. However, it may be noted that different type of applications require different types of locations information and to a different degree of precision. The locations can thus be classified into physical location, symbolic location, absolute location and relative location [2] [3]. Physical locations, refer to co-ordinates, as represented in degrees/minutes/seconds. Symbolic locations refer to those which can be classified using labels such as room, building name and so on. Absolute locations are given in a shared map, or a reference grid. Relative locations are location with reference to an object, or device or way point.

For the purpose of indoor localization various researches suggest use of expensive sensor deployment [2] [3]. These vary from $10 to several thousand dollars, and more in the cost involved in maintaining these sensors deployments. These include RFID, Ultrasonics, Infrared and Cameras, touch sensitive floors. Thus as of now, there is little widespread use of these technologies [4]. WiFi LANs prove to be a good candidate for general localization. This is due to the wide-spread use of WiFi enabled, mobile phones, laptops, tablets, etc. and the presence of WiFi networks in campuses, public spots, and office spaces. This can be leveraged to localize people or objects, without the need for setting up new hardware. And also no extra costs are involved for the maintenance of the network, which is already done.

The focus of this work, is to enable readily available symbolic locations, from existing infrastructure. And then to explore, what applications can be set up using them. The applications in consideration should allow for use in real world problems. Only a few attempts have been made for real world applications using symbolic localization. These include navigating through a building [7], enable basic occupancy information [8] and HVAC control [9]. Also these locations are used by restaurants, malls, and other commercial places, to study mobility patterns of their customers. With the presence of WiFi LANs in education institutions, Corporate Offices, along with a floor map/location of the access points, this symbolic location is readily available. Moreover, without any additional infrastructure costs and leveraging the existing infrastructure

software, and little overhead in maintenance, we can localize a user to a building, floor, and/or room level. The information available from this coarse-grained data can be utilized in many location aware applications, for occupancy information, space management, games, file sharing. In an Academic/University setting, there can be further use cases, such as attendance, course content transfer, etc.

As part of our work, we deploy a system which enables us to obtain this symbolic location information of all users. This enables knowing not only current location of users, but also those nearby, a key component of context-aware computing. This is accompanied by a thorough system description. Subsequently we explore it's uses by creating and deploying applications and also perform simulations for useful applications. Thus using existing infrastructure, (and little overhead costs) we present the system and applications of symbolic localization.

Chapter 2

Preliminaries and Infrastructure used

Our work makes use of existing protocols and infrastructure. In the subsequent sections, we discuss the Simple Network Management Protocol(SNMP), which can be used to perform data acquisition in enterprise networks. This forms the base of collecting real-time data. The protocol, with it's relevant features are discussed. Later we present how to use the protocol to collect real-time data and get symbolic location information of user devices.

Another key component of our work, is the presence of WiFi Local Area Networks(LAN) in offices, public and commercial spaces, campuses and industries. These places are characterized by the presence of robust Enterprise networks. We look at some of the key nodes, which are used in setting up the system.

2.1 SNMP

SNMP is a widely supported by network hardware manufacturers, used in network management systems to monitor network attached devices, such as routers, access points, network peripherals, etc. SNMP, was created in an effort to help easily manage network nodes.

As of today, the SNMP protocol is used widely to manage networks, but also perform simulations, evaluations of different network protocols, monitor usage of networks, etc. Moreover, it can be used as a foundation to intelligently (and autonomously) improve network manage-

ment [10].

The first draft of the Simple Network Management Protocol [11] was released as an RFC in IETF in 1990. It's primary aim was to manage networking hardware and nodes in the internet. Since it's first deployment, it has been widely used by research and commercial communities [11]. Since then it has gone multiple revisions and as of now the latest version of the protocol is version 3.

Management Information Base(MIB) for TCP/IP were defined which described "managed objects". These "managed objects" are not part of SNMP, but can be configured using SNMP. These managed objects are also extensible. They can be parameters, which a hardware unit allows for, or control commands it can support. Example: An object might be a threshold value, to turn off an overheated access point. And another object might be the current embedded chip temperature in an access point. This allows for various types of hardware to define their own managed objects, which can then be read/modified using the SNMP.

MIBs are defined hierarchically, with unique object identifiers(OID). The objected referenced through these OIDs can thus be read or set. Hardware manufacturers are free to write their own MIBS and register OIDs for their own use, and also reserve these OIDs.

The protocol enables communication via get-set paradigm. It also describes asynchronous traps, being sent on certain events. This alleviates the need to actively monitor the WiFi network to gather data. Instead asynchronous traps can be used to determine network information. Moreover, this helps to reduce the network overhead required to enable this system.

SNMP managed networks comprise of managers, running on dedicated servers. These managers are responsible for managing a group of agents on a network. Agents and the manager communicate via SNMP. And the data which can be configured or read by managers is as defined in MIBs which are supported by the managed devices.
SNMP managed networks consist of three key agents:

1. Managed Devices: These are network nodes which implement the SNMP interface and thus can be used to be read from and written to. They can be configured to enable or disable a variety of information, depending upon what is supported. The managed devices include access points, routers, bridges, etc.

2. SNMP Agent: This is the software running on the managed device. This has knowledge of the management parameters and translated to and from a SNMP specific form.

3. SNMP Manager: This the software running on a network node, which is used to configure, monitor and even act, on the managed devices. It can be used to log, and collect data received from the SNMP agents.

Security and other issues

Starting SNMP version 3, (SNMPv3), cryptographic security was incorporated into the protocol. SNMPv3 defines a secure version of the SNMP protocol. Without these additions, the data sent from access points, to the configured manager, is unencrypted and thus prone to network sniffing attacks.

Secondly, SNMP is usually run over UDP, and thus prone to loss of information. However, research has shown that TCP protocols require a certain critical threshold of network reliability after which their performance deteriorates(this is primarily due to TCP's congestion control). UDP however, performs optimally even after the network crosses the TCP threshold [12].
If needed, most SNMP softwares support being run using UDP as well as TCP, and given certain guarantees on network congestion and reliability, TCP may also be used to collect the data.

The widespread use of the protocol, enables the system to be set up quickly, without the need of additional software or hardware to collect sensor data. Since most Enterprise networks, use solutions which are built on top of SNMP, only minimal setting up of access points is required. This is, as opposed to, in deployment of other sensors, which have to be configured to gather this data, and new software/hardware developed. Also extra effort needs to be taken in setting up this new software/hardware.

2.2 Enterprise Networks

Modern day Enterprise networks consists of user nodes, devices used by clients, access points through which the users connect, routers, switches, hubs to route or transmit network packets, network controllers which are used to configure internal nodes such as routers, firewalls, etc.

Network controllers can be used to configure the access points to enable SNMP traps from the access points. Network controllers can be in hardware or software and are generally used

to manage the networks. In the absence of these, the network administrators use custom solutions to make changes to the network settings. In the latter case, configuring the access points to enable SNMP traps would require scripts or simple programs which can interface over the network. This comprises of communicating via SNMP. However there are tools available which can be set up and used to configure the SNMP agents [13].
Otherwise network controllers and device specific tools can usually be bought from the vendors themselves.

Chapter 3

System Description and Design

The system comprises of three core functionalities. Firstly, acquiring data, from clients connected to the WiFi LAN. We leverage existing protocols and components present in existing Enterprise Networks to gather this data. Secondly, parsing/cleaning and storing the data. And finally, releasing useful information through REST APIs, for it's use.

In this section we give details of the architecture and system design - which is discussed in detail. The description includes the constraints and challenges faced. Important security and privacy challenges and solutions are also discussed. The system uses various technologies present in enterprise networks: SNMP, Network Controllers. We overview the system design and it's end goals. These goals are set according to the need of the final users of the system, and how best the information available may be dispersed, without violating the privacy of any of the WiFi users. We then look at the internal workings of the system. This includes working of core components of the system, problems faced, and their solutions, along with other plausible solutions.

The system was deployed in the campus of IIIT Delhi. It is a campus spread over 25 acres of land, with separate Academic building and buildings for the library, a students centre, residential building, hostels, etc. All of these buildings are WiFi equipped and serve approximately 1000 - 2000 users of the network. The network comprises of Airespace Access Points, and Cisco routers and hardware network controllers, also purchased from Cisco. A dedicated IT team manages, operates and looks after the functioning of the campus LAN and the placement of access points.

3.1 System Design and Architecture

We now give the system design in full. To enable building ubiquitous applications, we first outline the requirements from such a system. This is done by establishing the goals of such a system. Then we look at the architecture of the developed system and it's core components. The architecture consists of multiple layers. We discuss the architecture layer by layer, describing functionalities of modules present in each layer. We discuss security and privacy modules wherever they have been added in the system.

3.1.1 System Design Goals

The primary goal of the system is to enable applications to get connection information of user devices moving in the WiFi enabled campus.

The goals of the system are thus:

- Enable applications to get connection information of user devices across the campus, with symbolic locations, in a timely, and robust manner.
- Protect privacy of user information collected and securely store sensitive information.
- The system should have high availability (in order to store and expose all user connection information).

If a system meets these goals, then previously mentioned applications can be made using this system.

3.1.2 System Architecture

The system architecture with it's layers and modules is given in figure 3.1. Here, each layer provides an abstraction to the layer above. In total there are 5 layers in the system - the user device layer, the access point layer, core layer, web service layer and application layer. We discuss each layer along with it's components in subsequent sections.

Before discussing each layer in detail, we give a high level view of the architecture. To better understand the system architecture, here we describe the flow of data in the system. Data generation starts at the bottommost User Device Layer. This layer comprises of all WiFi devices

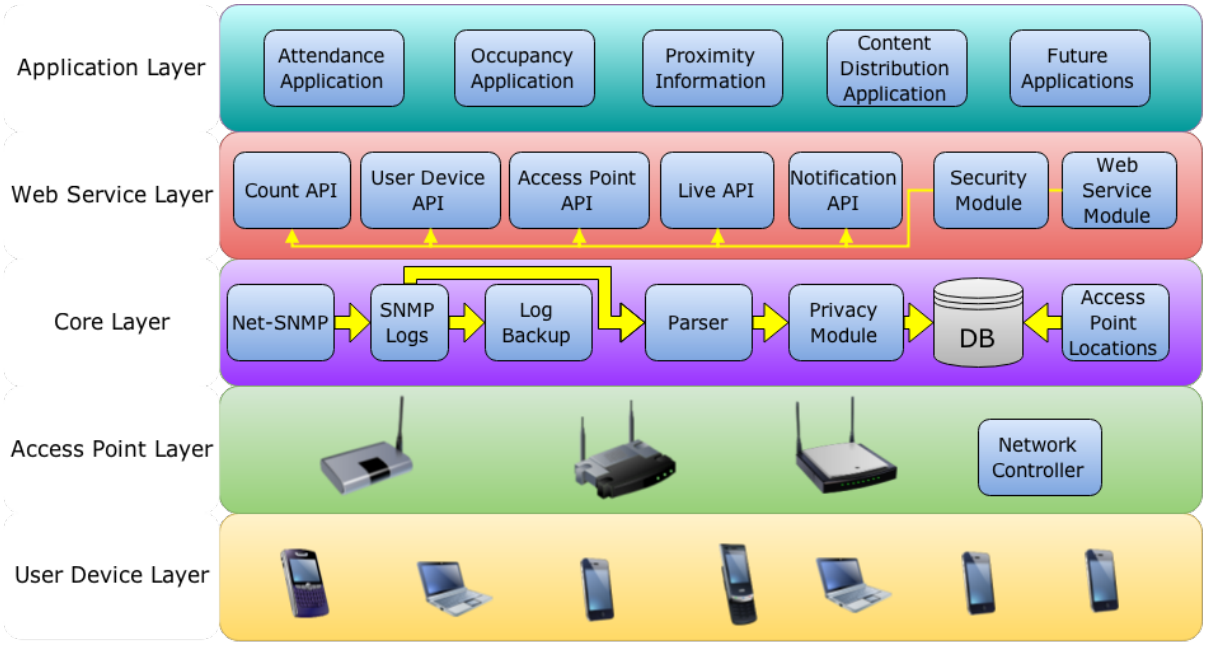


Figure 3.1: System Architecture

present in the network - including mobile phones, tablets, laptops, and other such devices. The devices get connected to an access point. The access points form the next abstracted layer. These access points are configured using the network controller to generate traps, whenever a user device is connected or disconnected. This is done through SNMP. The configuration sets up the access points to send the required traps to a specified server. This server runs a software which receives and logs these traps. This software is the first module in the Core Layer. The software dumps the output into a log, which is subsequently parsed by the parser. This parsed data, goes through a privacy module and then put into the database. Additional information of access point locations is fed into the database directly. This forms the core layer, which collects real-time data. The data then gets exposed through web services. This is done through the Web Service Layer, which includes a security module, to provide access control to the services. Finally the Application Layer, comprises of applications which use the web services provided by modules in the Web Service Layer.

We now follow the bottom-up approach to understand working of key components in each layer.

3.1.3 User Device Layer

User devices such as phones, tablets, laptops and e-book readers, are devices which a user can use over WiFi. A key advantage in using such WiFi based symoblic localization, is that unlike localization using GPS, or Cell tower triangulation, the device itself, does not need to perform any computation or task itself. In fact, the device does not need to use localization techinques such as SLAM or Markov Localization to know it's symbolic location. This is especially useful in case of robots. In fact only a battery and a WiFi network card/adapter is enough to localize up to a resolution of building, floor, wing and room. This leads to the possibility of simple WiFi tags. These tags can connect to an access point periodically. And the system can be used to localize the tag.

3.1.4 Access Point Layer

The access points along with the network controller, form the access point layer. All devices connect to one of the the access points in the campus network. Each access point runs the SNMP agent. A SNMP manager running on the network controller is used to configure the access points. The access points across campus are configured to generate asynchronous traps to be sent to a dedicated server. The traps that we configure the access points to generate are [14]:

1. AIRESPACE-WIRELESS-MI::bsnDot11StationAssociate
   "The associate notification shall be sent when any of the watchlisted clients(present on at least one watch list) associates with an AP. The value of the notification shall include the MAC address and the Slot ID of the radio to which the station Associated."

2. AIRESPACE-WIRELESS-MIB::bsnDot11StationDeauthenticate
   "The deauthenticate notification shall be sent when the Station sends a Deauthentication frame. The value of the notification shall include the MAC address of the MAC to which the Deauthentication frame was sent and the reason for the deauthentication."

3. AIRESPACE-WIRELESS-MI::bsnDot11StationDisassociate
   "The disassociate notification shall be sent when the Station sends a Disassociation frame. The value of the notification shall include the MAC address of the MAC to which the Disassociation frame was sent and the reason for the disassociation"

4. CISCO-LWAPP-RRM-MIB::ciscoLwappDot11ClientCoverageHolePreAlarm
   "This trap can be used to test if a user is about to leave the physical area in which the

router can serve it. If it tests positive against RSSI threshold, then this trap is generated."

The IEEE 802.11 protocol uses management frames to allow for maintenance of communication over WiFi. The protocol begins with a client device sending an authentication frame. Followed by an association request frame. If there is successful association, then the access point generates a trap, and sends it to the configured server.

Similarly, when a client is getting disconnected, it first sends a deauthentication frame, and then possibly a disassociation frame. The first three traps being monitored capture these steps of the protocol, and thus provide information regarding when a user device is connected or disconnects from the campus WiFi network.
The last trap being monitored, is generated when a user device's signal strength goes below a certain threshold value.

3.1.5 Core Layer

This layer contains module which perform key tasks in the system. As a whole this layer is responsible for collecting all the information of connections and disconnections across campus. This layer comprises of modules for, logging the traps received, parsing them, taking regular backups, maintaining privacy of users, the data store and the access point location information.

Net-SNMP

Net-SNMP is used to receive the asynchronous traps generated by the access points. These traps are logged, along with their time stamp. Net-SNMP is configured to log these traps into a text file on the server. Net-SNMP had to be configured to use AIRESPACE-WIRELESS-MIB and CISCO-LWAPP-RRM-MIB (along with their dependencies). Once that is configured Net-SNMP is able to successfully receive the traps and read them.

It is necessary to ensure all traps are stored. This requires the Net-SNMP service to be run throughout, while being able to perform system essential tasks such as parsing of the log file in real-time and administrative tasks such as taking backups. In the Institute where the system was deployed we found on an average 300 to 500 MBs of log file is generated on a weekday, during a semester, and 150 to 250 MBs on a weekend. This is in proportion to the activity in the campus network. The logs contained other traps which were enabled.

Parser

In order to read the file, we made an in-house parser which reads the logs from the standard input, parses the data and pushes it in to the data store. The data is sent over TCP, using a simple protocol. To parse the file in real time, we read the file the file using the linux utility tail, with the -f parameter, pipe it's output to the parser.
A critical parameter to observe is the rate of generation of traps. This should be lesser than the rate at which the database can be updated. This was found to be lesser than the rate at which the data is parsed and pushed in to the database, even under periods of high activity.

Log Backup

In order to avoid dealing with reading huge log files, we created a small script which works similar to the the Linux utility logrotate. This is run every 24 hours, and the logs file stored with it's date. This helps in data recovery or verification in case of system downtimes. A key step, is to send a SIGHUP signal to Net-SNMP which forces it to close and then re-open all of it's connections including file handles. This enables us to copy the log file, and truncate it, without having the need to stop or restart Net-SNMP.

Privacy Module

The requirement for any applications built on top is to be able to access device connection information. At the same time, at no point the privacy of a user or other sensitive information be available, such as MAC addresses. Thus, in order to protect user privacy, we strip the parsed trap of the MAC address, hash them using SHA256 scheme and then assign a Universal ID (UID) to each hash. All subsequent queries can be made through these UIDs (figure 3.2). Or through hashes of the MAC address, if in case an application has the MAC address available.

Since we store only the hash and UIDs, and not the MAC address, privacy of a user is preserved. Thus the data may be used for research purposes in it's raw format without any violation of privacy of a user. This also helps to secure against malicious users wanting to access user specific data. A key advantage of using UIDs is that this allows for unique identification of user devices. And thus lets the applications previously mentioned make full use of the system, in applications such as finding user mobility patterns, occupancy information, etc.

Access Point Locations

The system assumes mapping of the access point to physical locations is available. In the Institute the system was deployed, this was available with the IT Department. The locations are organized into building, floor, wing and room(if possible). Consideration need to be made in these locations, as due to various factors such as wall thickness, material, other obstacles, etc. devices might connect to other nearby access points, rather than the closest one.

For high physical availability in a WiFi LAN, coverage area of access points overlaps. Thus labels have to be as specific as possible without giving incorrect information. In case an access point in room A, is kept so that users in an adjacent room B are also able to connect to it, then the next level in the hierarchy, such as wing or floor should only be mentioned and not the room name.

The access points are fed directly into the database, and updated as when required - when a new access point is installed or location of an access point changed.

3.1.6 Web Service Layer

Information can be accessed from the system using a RESTful API. This layer comprises of a web server to handle the API requests, and modules to serve the API requests. These API modules use the datastore to access the information. Web services and APIs are widely used in distributed applications. Libraries in Python (urllib, etc.), C++ (curl, etc.) and Java(java.net.HttpUrlConnection, etc.) to handle web calls are thus mature and offer wide variety of features for different types of applications. In the applications built using these services, the prototyping became much easier through the use of these APIs.

Web Service Module

The web service module forms the backbone of all the APIs offered. APIs can be added or removed from this module as and when required. The module is written using libmicrohttpd - a web server library written in C. Libmircohttpd is a light-weight library, which provides simple functions to start a web server and handle different HTTP requests. Also it can be configured to be used by polling or through select. It supports multiple threads, a key requirement for scaling web services.

The web services offered try to meet the following requirements established based on the system design goals:

- Applications should be able to query location information of a particular device.
- Access to real time data as well as past data.
- Apart from those of specific user devices, similar queries should be available for different access points.
- Aggregated data should be available, for applications which do not need user device specific data.

The points are listed in descending order of personalization of applications possible. Thus they are also in descending order of information required, along with needing higher access levels.

Security and Access Control Module

Authentication: In order to authenticate different applications, a token based authentication system is incorporated into the system. This helps to keep check on who can use the API. Basic authentication is required to control access to the API. HTTP Basic access authentication may be used, provided it is over SSL.

As part of the token-based authentication, the applications are assigned unique usernames and passwords. Applications can use these to generate a token, which is generated by hashing the username, current time and a salt, to generate the token. This token is returned through the API. Subsequent requests can be made through this token, up till the sessions ends. After which a new token needs to be generated.

Token based authentication is carried out using the **/auth** API. This is used to generate the token. The following GET parameters need to be passed: username, password. This is the only query which is used without needing a token. All subsequent queries need to pass a GET parameter "token", along with the token generated.

Secure Communication: All communication through the API is done over HTTPS. Thus only encrypted queries are sent over the network. Sensitive information such as username, password, and the information returned by the service, all are thus protected from network sniffing attacks.

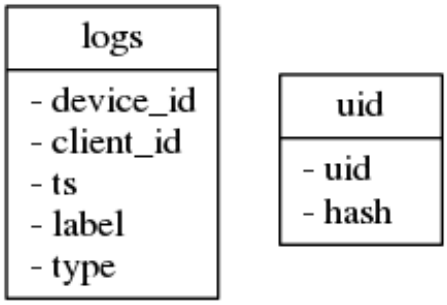


Figure 3.2: Logs and UID table

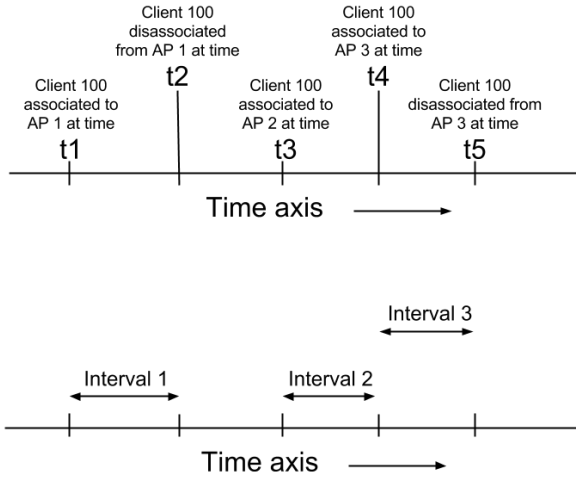


Figure 3.3: Incoming traps and associated intervals

Privacy: In order to protect user privacy, and not allow applications to know their location without their consent, privacy levels for each device has been set up. All API return information according to the set privacy levels of the users. According to the privacy level set, the */client* API returns no information, or information in the hierarchy: building, floor, wing, room, according the access point they are connected to. By default, no information for any device is returned.
The */ap* API returns only the UIDs of connected clients, with no location information.

Access Point and Client Connection APIs (*/ap and /client*)

The module consists of the following APIs:

- */uid*: This returns the system generated UID for a specific client device or access point MAC. GET argument needed: mac.
- */ap or /client*: Getting access point or user devices related connection ‘ information. The GET arguments are used to specify the timestamps between which the information needs to be returned, along with the UID of the device.
  This API returns connection information for the access point or client. It specifies which access point (/client) or which client(/ap) got connected from what time, to what time. Thus the API returns a set of intervals.

Connection intervals can be obtained from this data for a client or an access point. For a client we iterate over the access point connected to in the order of time stamp and use the type of the trap to generate the intervals. Figure 3.3, shows such a trace. Here we have five traps, each at unique time stamps. The first two traps can be translated to interval 1. There is an association of client 100 to access point 1 at time t1. And then it disassociates at time t2. The next interval however is of importance. Client 100 associates to access point 2 at time t3. And then associates to access point 3 at time t4. Such ordering of received traps is encountered frequently. Disassociation or deauthentication frames, allow for clean closing of connections, however, are not always sent by devices. Devices, such as mobile phones, which aggressively conserve battery, try to keep overhead communication to a minimum. Thus upto t4, client device 100, was indeed connected to AP 2. After which, it got connected to AP 3, upto t5 when it finally a disassociation trap is generated.

Finding access points a client was connected to with given timestamps is simpler than finding all clients which connected to an access point along with their correct duration. This is because of disassociation/deauthentication traps are not always generated. If the user device, does not send the disassociation or deauthentication management frames, then the trap is not generated. Therefore if a client was connected to an access point and switched to another, with disassociating itself from the previous one then a trivial approach might show the client connected to two access points simultaneously. Thus in returning access point information, the state of all connections needs to be maintained.

All the data available from SNMP logs is stored in a table named logs (Figure 3.2). This stores the access point UID, the client device UID, timestamp associated with the trap, a predefined label with the access point, which is stored for logging purposes and a type column to denote the type of trap received.

Using the above APIs, apart from occupancy information, simulations of routing techniques, social information can be done. Moreover, applications which require routing can be built using only the only the API as the back end, and thus significantly simplify prototyping applications which provide peer to peer services.

Count and Live API (*/count* and */live*)

The Count API returns connections which were active at a particular time. This is aggregated according to buildings, floor, wing and room. Also there is an additional option to view the

UIDs which are connected similar to the the */ap* API. Live API is similar to the count API, except it always presents the connected user devices, in the current moment.

- */count*: This API presents aggregated location based data, grouped according to applications requirement. It returns the number of devices in any area, at a specified time instant. The aggregation can be done on the basis of a building, floors, wings and rooms or any combination of the two.
- */live*: This API presents users who are connected to the college WiFi Network currently.

Applications which require such aggregate data include: applications for proximity information and occupancy information.

Real-time notifications API

An experimental API to send push notifications was made. Through this a mobile can be receive real-time notifications of the access point or location it is connected to. This API is used in conjunction with a mobile device application, which receives the notifications on the mobile device.

3.2 Summary

The first 4 layers form the core part of the system which enables several application. The final layer includes applications. These applications, can use real-time localization information of user devices, group of user devices, or access points. Also the layers discussed so far enable applications to get information regarding user device location history. Moreover, the APIs allow for applications to use the system itself as the back-end and need to develop only the front end.

We look at some applications made using this in the next chapter.

Chapter 4

Applications

Using the information exposed by the presented system, we developed prototypes for real-world applications and deployed them. The attendance system and device occupancy information were deployed in IIIT. In addition to these, we also explored further applications, which can be be enabled using this system. These additional applications, can be incorporated into the core API, or developed as standalone systems themselves. Offloading course content using mobile ad-hoc networks or proximity and social graphs, constitute the latter.

Firstly we look at a campus attendance system. This was developed for IIIT, to meet the need of a basic attendance for Teaching Assistants (TAs) at IIIT. TAs can use their WiFi devices to mark their attendance by entering a building during working hours. This system was first deployed in August 2014 in beta stage. Since then it has matured in to a robust attendance managing system. Since January 2015, it has been used to officially take attendance of the TAs.

Mobile Ad-hoc Networks (MANETS), is another area, which can leverage the information provided by the system, for various routing techniques such as opportunistic routing and social routing. We explore such a routing technique in an academic setting. Since instructors provide course material through web portals, FTP or course management software, multiple requests are sent over the campus network to download these resources. We simulate sending of files ubiquitously to a target group of user devices. We simulate Epidemic Routing [15] through MANETS for content transfer and provide promising result for prototype development.

Device Occupancy information, provides the number of devices connected to the campus network at a particular date and time. Many ubiquitous applications may be developed using

this data. Applications for space management for WiFi LANs, where there is a need to maximize throughput, Areas where WiFi is less used can do with fewer overlapping access points, whereas areas where more than the supported capacity of users are connecting, more access points may be installed. Alternatively an application which suggest a user to move to a certain area can also be made. This can be also be used for energy management. HVACs consume large amounts of power, and can be switched off, if the number of devices in area fall below a threshold.

Proximity and social information is useful in a variety of areas, including context aware services, applications which leverage social or proximity information, opportunistic routing, social networks, etc. Connection information provided by the system, can be used to gather proximity information between users. We model this information in the form of a graph.

4.1 Attendance of Teaching Assistants

Teaching assistants(TAs) in institutions, are considered to be part-time employees. The Institute, the system was set up in, required a basic taking of attendance of TAs. The requirements were as follows: If a TA enters any of the work buildings - Library, Academic Block, or Labs, during working hours, then he or she should be marked present for that day. This data needs to be recorded for each working day of the month, after which monthly attendance would considered.

An ubiquitous solution given to problem was that whenever any TA entered any of the above said places and uses his or her mobile phone, or laptop, during the set timings, then he be marked present for that day. This way TAs don't have to go to a particular place or stand in a queue for biometrics or attendance register signing, to mark their attendance. Since most students at the Institute carry a WiFi enabled phone or laptop with them, their mobile or laptop would connect to the campus WiFi, on getting enough signal strength to any access point. Thus just by coming to the building, their attendance is marked seamlessly, provided they are carrying their mobile or device. In case they forget to do so, then a portal was set up, through the academic officer could mark them present for the same day.

The portal was further developed so that, the academic officer is able to download attendance for the TAs, view the current months attendance for the TAs, register devices, and perform other administrative tasks.Similarly a portal was created for the TAs, through which they can view their own attendance, for the current and previous months.

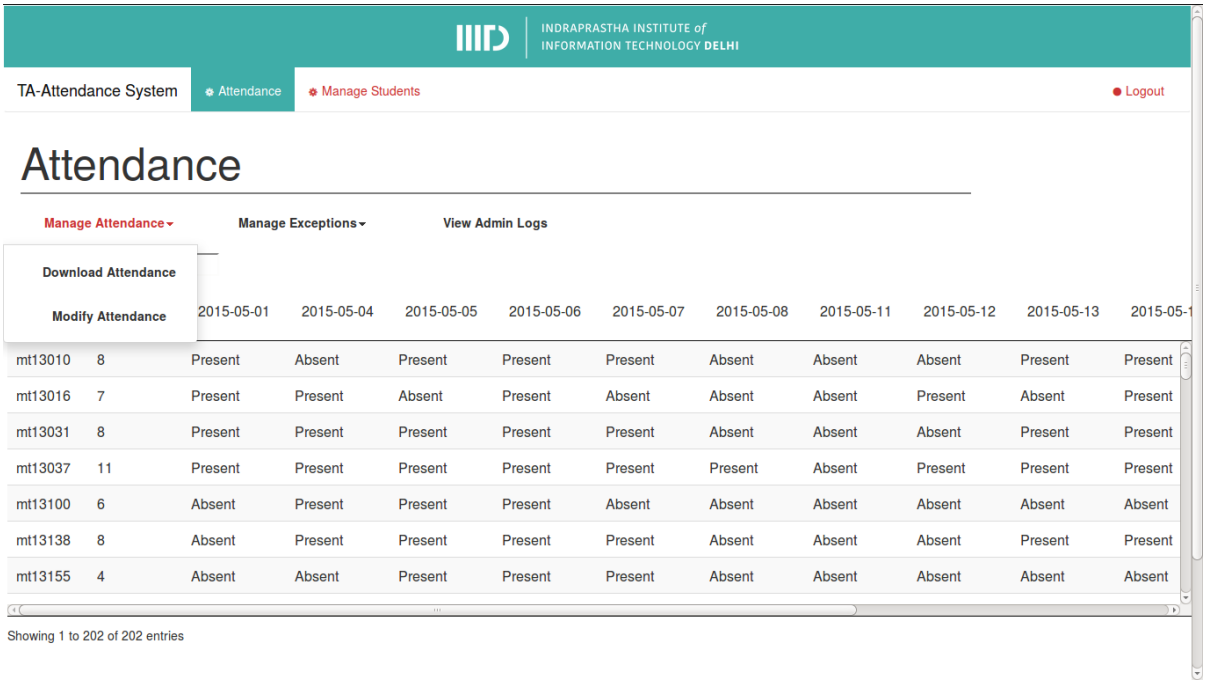


Figure 4.1: Attendance Admin Portal

For the purpose of attendance, a TA needs to supply the MAC address of devices that he or she use. Up to three devices were allowed by the Institute policy.

To find out how really ubiquitous the application could become we took a user survey. For such applications such as attendance to become ubiquitous, the application should become as pervasive as possible. We wanted to test this on the basis of 3 parameters: trust, ease of use, and pervasiveness. All three form important factors for ubiquitous computing. Trust on the system enables a person to not worry about the application failing. Ease of use lets oneself involve less with a terminal such as laptop, tablet, mobile phone, etc. Finally pervasiveness, which is measure of how seamless the application became overall.

To quantify these 3 parameters, we asked the TAs who used the system in the Winter Semester 2015 at IIIT to fill a brief survey. We received a total of 53 responses. These include PhD scholars and M.Tech students. First we needed to identify how easy they found using such an automated system, compared to conventional methods of biometrics and register signing. To do this we asked them to rate each of the methods based on convenience on a scale of 1 to 4, 4 begin "convenient". Subsequently we calculated a weighted score for each. An automated

system such as the WiFi attendance system scored far higher than the rest, with a score of 194 of a total possible score of 212. Biometrics came in next with a score of 120 and register signing last with a score of 105. There is only a marginal difference in convenience, as perceived by the TAs, between biometrics and register signing. Next to establish whether the system itself was easy to use, 92.4% users rated it 4 or more on a scale of 5, 5 begin easy. Next, to determine their trust we asked two questions. One, directly asked them to rate how much they trust the system to mark their attendance correctly. Secondly the times they explicitly went to a location to mark their attendance might indicate lack of trust in the application. They are asked to rate the former on a scale of 5, 5 being trust completely. The weighted score percentage is 75.09%. While more than 50% of the TAs explicitly went to mark their attendance less than 25% of the time. And 31.5% went there more than half the time. We found no co-relation between trust and them having to explicitly go to a location to mark their attendance, with a Pearson Bivariate correlation of -0.09 and one side significance of 0.27. A subset of the TAs responded they go to explicitly mark their attendance because they prefer to work from their home or hostel rooms.

The pervasiveness of the application depends upon how often the TAs go the academic building. More than 50% of the TAs had their attendance marked seamlessly atleast 75% of the time. This is a good indicator of the ubiquity of the application. Also, we found a moderate negative co-relation ($r = -0.34$) with a 1-tailed significance of 0.01. As more often the TA goes to the work buildings, less he or she will have to explicitly go to mark their attendance. Greater than 80% of the TAs go to work in Academic building or a lab 4 or more times in week. Thus the TAs who more often work in the labs and Academic building experienced grater levels of pervasiveness.

The application delivered on ubiquity for at least half the TAs. It came close to being pervasive, as they did not need to explicitly go to mark attendance. This is a key result, and further improvements can be made based on the feedback of the teaching assistants. Since the TAs need to be present in the work places, only during tutorials, lab sessions etc. if attendance needs to be taken only during this time, then the need to explicitly go to mark their attendance would further reduce. And increase the ubiquity of the attendance application.

Limitations

The system is not fool-proof. False negatives are easily possible. This is because of the scenarios where the WiFi of user devices is turned off, or in the case that the user enters the building only for a brief period, the mobile never associates with an access point. Also, approximately 30% of

the TAs still marked attendance conventionally, by going the areas designated for attendance. There may also be proxy attendance, where a TA might ask someone to take his device to a designated location to get his attendance marked.

However, for a majority of the TAs, who go often to work in the labs or Academic building, attendance was marked seamlessly. Moreover, more the frequently the TAs go to work in these places, the seamless the application becomes.

4.2 Offloading Content to Mobile Ad-hoc Networks

Often among a group of people, a file needs to be transferred from one user to all the other member of the group. One example of such a scenario is that of a course. In a course, the instructor usually needs to share course updates, lecture notes, and other resources to the rest of the students. Modern day course management systems follow a client-server architecture, where the student goes online to download content from a portal. Even outside an academic setting, attachments or cloud services such as Google Drive or Dropbox are used to share file amongst peers are used. Use of such services, requires the presence of a network, over which the transfer happens. Files transfers consume bandwidth of the network. However in the presence of aware routing techniques, and mobile ad-hoc networks, these file transfers can be offloaded, freeing up bandwidth utilization of the network.

However, for using MANETs, the mobile devices need to regularly scan their environment for possible devices to deliver a file too. This consumes a lot of power in mobile devices, reducing their battery life. Thus limiting usability. Bluetooth scanning takes greater than 140mW when scanning the environment [16]. Also previous research [17] suggests an 8 seconds for two bluetooth devices to discover each other (mean discovery time) Assuming a device need to regularly scan the environment at 10 second gaps for 8 seconds intervals for 24 hours in a day, this leads to a power usage of 5376J in 1 day. Compare this against the battery of an iPhone 5, which holds around 5 watt hours = 15000J. That is, one third of the battery is spent in discovery of near by nodes.

Instead of regular scanning, the SNMP system can be used to know whether there is a near by node. Since the mobile devices are already connected to the WiFi network, queries/notifications can be sent which let the device know of a nearby device. Subsequently, the mobile can proceed to link up with the node for the transfer of such files.

We try to establish the time constraints in university set up, where there is a need to transfer such files. We show that such an application can be used in a campus environment while conserving battery life of a mobile.

We take a random group of people, and set one member as the source of the file. Through their WiFi traces, we determine when two users of the group are close by, that is they are connected to the same access point. Next we simulate a file transfer by transferring the file only if they remain connected for a pre-determined amount of time. We look at the time taken for each person to receive the file. We use epidemic routing to simulate delivery of the file. In epidemic routing when ever a device having the file comes into the range of any another device not having the file, the file is transferred.

An example of one such simulation can be seen in figure 4.2. Each node is labelled with the UID assigned to the device. A directed edge denotes the transfer of file. Nodes which received the file within 12 hours are coloured red, within 24 hours are coloured orange, within 36 hours - green and within 48 hours - blue. Others are coloured grey. In this particular run, 46/51 students receive the file within 26 hours of the start. Also, the time duration for which they are considered in range is 10 seconds.

Bluetooth 2.0 allows for a maximum data rate of 2178 Kbits/second and the time taken to establish a bluetooth connection would be roughly less than 3 seconds [18]. We then proceed to simulate file transfer of different sizes. A text notification can be conveyed within 10KBs. Course updates or notifications can fit this size. Time required for transferring such a notification is $(10 * 8)/2178 = 0.0367$ seconds. Add to this the time to set up a connection , total time to pass the notification is 3.0367. Table 4.1 lists the time calculated to transfer files of different sizes. In the experiments, the target device group size was chosen as 50. The devices were chosen randomly from a set of all devices active in the campus from during the Winter Semester 2015. We simulated transferring a file once every month of the semester, for each of the time intervals identified in the table.

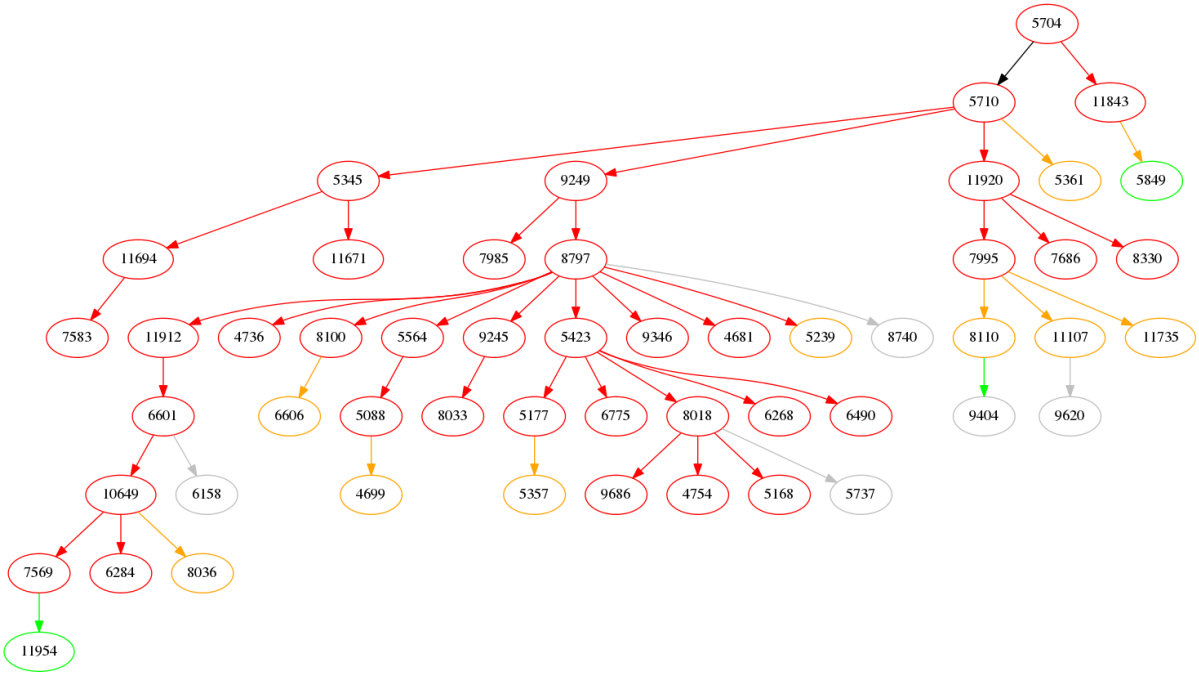
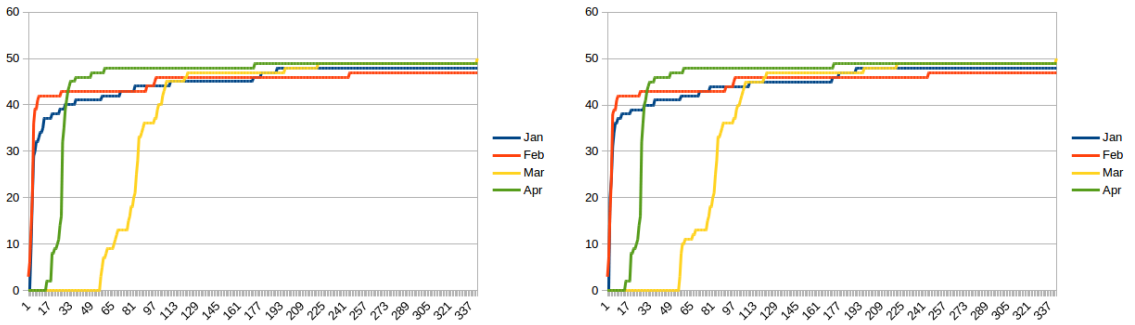


Figure 4.2: File transfer tree

File Size	Total Time to transfer	Uses
10KB	3.0367	Small text notifications, announcements, etc.
200KB	3.734	PDFs of lecture notes, all text lecture slides
1MB	6.76	Lecture Slides with images,etc.
10MB	40.61	Short Lecture Videos

Table 4.1: Time to transfer files using bluetooth

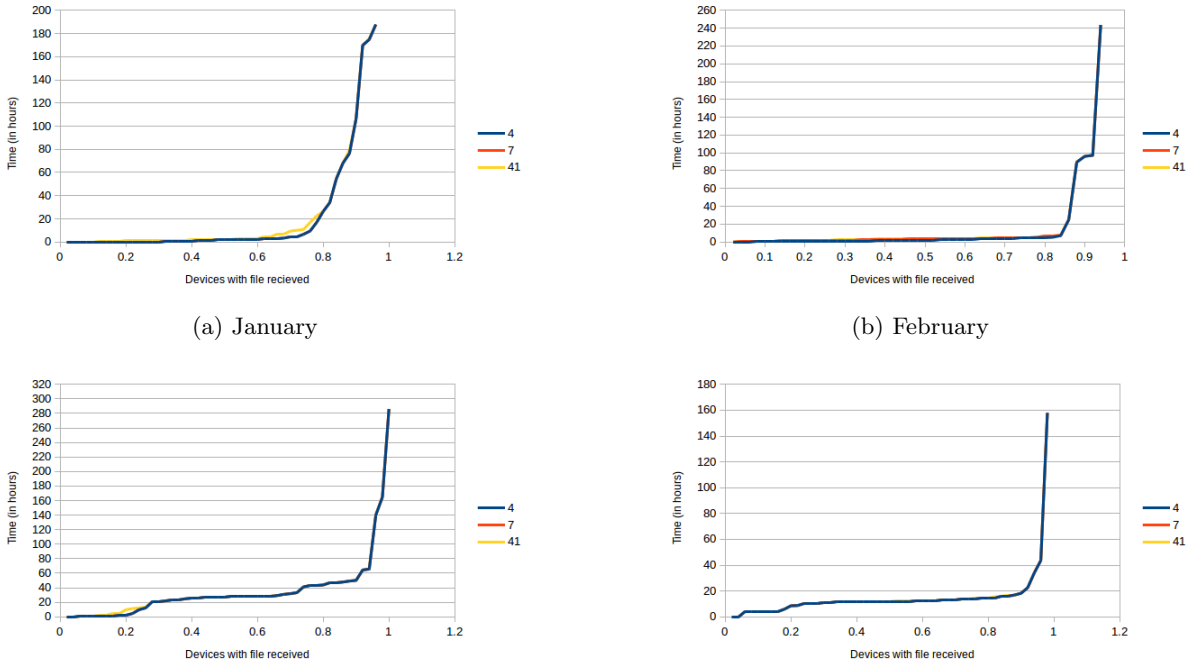
In figure 4.3, we plot the number of devices to receive the file against time taken (in hours). The interval for which the devices needed to be near each other was set as 41s in figure 4.3a and 7s in 4.3b. We also noted that the time taken for 90% of the devices to receive the file was same despite the different time intervals selected. Next we measured how quickly the file would disperse once the first delivery takes place. The time taken to deliver the file to target devices (normalized) for each month and time interval is plotted in Figure 4.4. We concluded that the time for files to be delivered, is more sensitive to external factors such as location of device sending the file, mobility patterns than the size of the file being transferred (up to the size simulations were carried for). This may be seen in the plots in figure 4.4, where the files of size 10 kBs and 10MBs are received by the same number of target devices around the same time. As seen in figure 4.3, for the month of March, there is a significant delay in first delivery.



(a) Time Transfer interval = 41s

(b) Time Transfer interval = 7s

Figure 4.3: No. of devices received file vs time



(a) January

(b) February

(c) March

(d) April

Figure 4.4: Time to deliver file (after first delivery) vs devices which received files

After which the spread of the file is similar to that of other months. In a course setting, since there exists an upper bound in which the target group of students come together (either for a lecture, tutorial or lab session), we are interested in finding an upper bound on the time taken for delivery after that. Plots in figure 4.4 capture this information. As seen there, in all the simulations the file would be transferred to 80% of the target group within 48 hours, since the first delivery. However the time in reaching greater than 90% devices increases exponentially fast.

In a course setting, much better results can be expected. In these experiments since the devices were randomly chosen, the devices to which the file was sent might not have any relation to the device sending the file. In an academic/course the time of delivery would have a lower upper bound. The users of the devices have common lectures, tutorial or labs where they regularly stay together for prolonged periods. Also due to social relationships amongst peers, they would meet more often. Thus the upper bounds would likely be further reduced.

Furthermore, the performance of such an application can be improved using location information presented by the developed system. Historical data can be used to choose between different routing techniques which would enable faster delivery. Also the application can query for devices, which may not be in signal range but 1 hop away from direct connection, and with co-operation from these devices, further improve results.

4.3 Other Applications

We now present other applications built using the system. These applications though themselves not ubiquitous, can be used to create or enable further ubiquitous applications.

The Device Occupancy Information application, can be used to visualize number of people in an area, and in the past. Applications along similar lines, can be used to help people find less crowded areas, or ask them to move to an adjacent area where their device might face lesser WiFi congestion.

We analyzed connections made across campus of different user devices, to find proximity relations between them. This leads to interesting file transfer routing techniques. Also, proximity is a key factor in determining social relations, and thus has further applications in finding social networks. We present the results of our analysis, along with statistics of the graph which models the information.

4.3.1 Device Occupancy Information

We believe that having information about occupancy can be used for identifying different avenues of energy/network management as well as overall space management.

In earlier work, as part of the author's B.Tech Project, we measured overall usage of access points across the campus. Next, we analyzed data for 9 days. Next we found average number of associations being made per hour, and then clustered them on the basis of hours. We set a threshold value of 5.0 connections in an hour to indicate less usage of an access point. We had noted that 20% of the total working hours of all APs, less than 5 devices were connected. This informaiton may be used to improve space management, as well as energy conservation. On finishing such regular patterns, for those hours the access points can be turned off to save energy. Similarly we identified other usage of occupancy information in finding out how crowded an area which can be used for better space management.

This led us to make a prototype application, which can help to easily capture the above information. In order to enable relevant authorities to be able to easily visualize such data, we created a web application based on the system. The application can be used to identify when to switch off power consuming HVAC systems in areas which are less occupied. Also, lighting, fans, computers, and other electrical devices, may be switched off in such areas, to further conserve energy. Figure 4.5 shows the web application. The web application allows a person to view, occupancy snapshot at any given point in time. Along with it, the previous 7 days of usage is also displayed.

The application also offers view of building usage over a period of 30 days, which can be used to track overall building usage, figure 4.6.

Limitations

Device occupancy information, and the actual number of people in area might differ. Thus, in applications where the information required is, the number of people, the above application should be used keeping that in mind. Further work, can try to relate occupancy information with the actual number of people in an area.

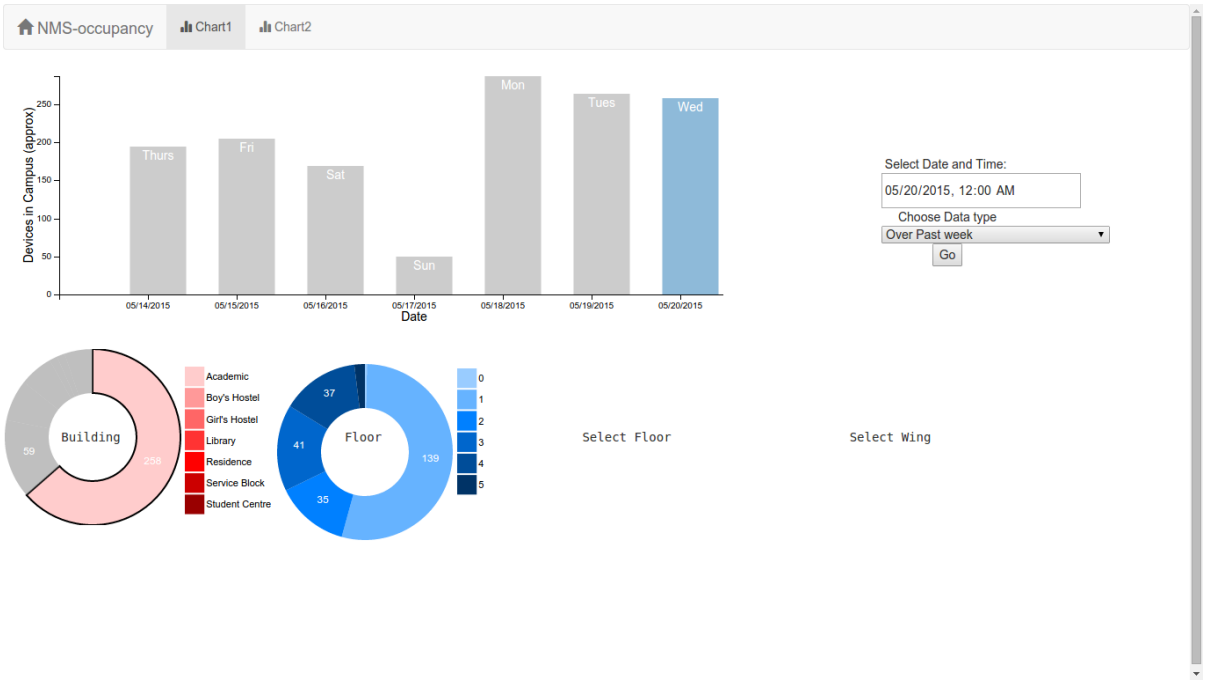


Figure 4.5: Occupancy Information Application: Main View

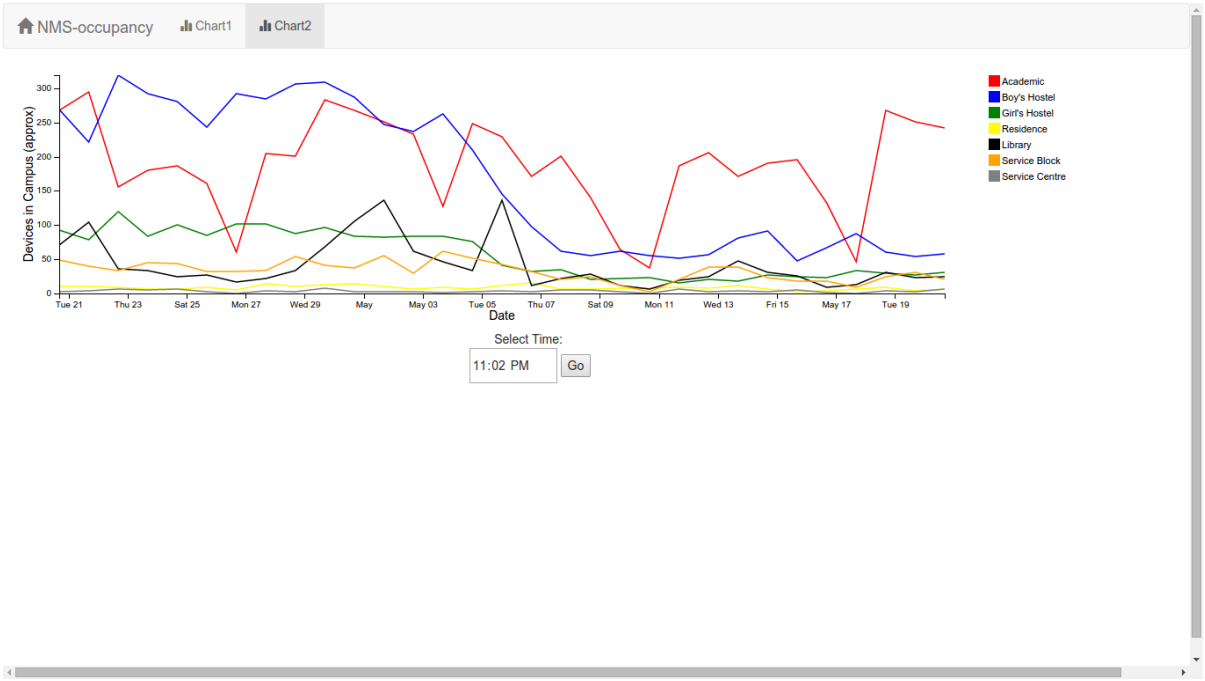


Figure 4.6: Occupancy Information Application: Building View

4.3.2 Proximity Information

In any social group, proximity between different member of the group forms contextual information. Parameters like, how often they come together, how long they stay together, etc. is useful in the case of ubiquitous computing.

The proximity information can be used in Mobile social network(MSN). MSN is an upcoming field of research which combines two previous fields: Social Networks (a field in Social Science) and Mobile Networks [19]. Social networks are being explored, to know the context of information, to provide efficient data exchange, sharing and delivery [20].

Proximity information also enables a user to understand social interactions and movement patterns of users. This can enable algorithms to find interdependencies amongst mobile users, which can be studied and used to further improve upon data dissemination and sharing [21].

We use the system to gather proximity information. We collected and analyzed connection information of users at IIIT. The parameter analyzed, is the frequency of proximity. That is, how frequently two user devices are close to on another.

Proximity Graphs

In order to easily comprehend proximity information of WiFi users at IIIT, we model it using weighted undirected graphs. Here a node represents a user device, and an edge represents a relation between them.

In order to generate proximity graphs, we query the */ap* API to gather data for all the access points over a given period of time. Proximity information can be captured using different weight functions defined on the edge. The number of seconds for which two devices remain connected to the same access point may be considered to absolute proximity information. Also the frequency with which two devices can also be similarly modelled, where the weight, is the number of times two devices, come close to one another. Here, we make the assumption that two user devices connected to the same access point, can be considered to be "near" to each other.

After generating the graph, we try to find communities between the users, who frequently come into contact with one another. For this purpose, we use Gephi - a social network analysis tool, which can be used to easily visualize social data [22].

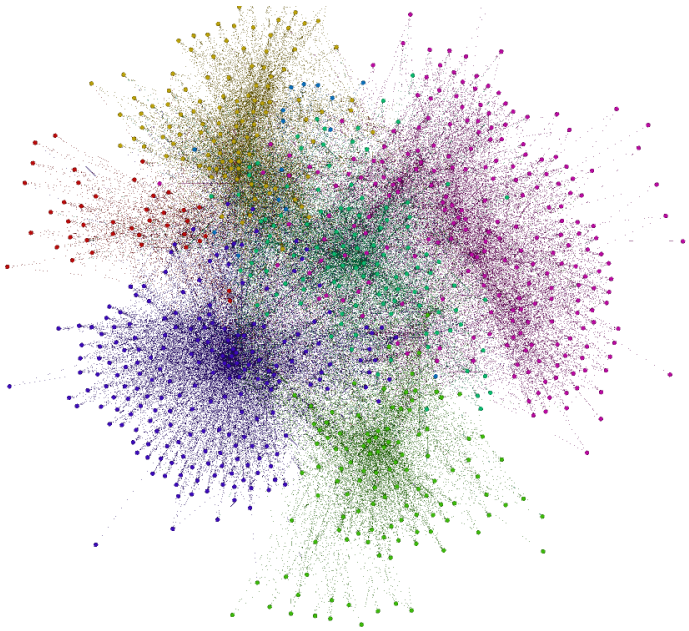


Figure 4.7: Frequency Proximity Graph

We look at the frequency of proximity 4.7. This graph was made using data from the Monsoon Semester of IIIT, 2014. The communities were established by the algorithm given by Blondel *et al.* [23]. 7 different communities are formed, with one community central to the graph. The graph contain around 2000 nodes. In each community a subset of nodes, forms the nucleus of the graph, those users who are active within that community and frequently come into contact with other members of the community. The central community, however are those nodes, which frequently come into contact with nodes from other communities. Each community shares the characteristic of frequently coming close to other members in the community. Frequency of proximity, is crucial in identifying user devices, which frequently come near users of the same community as well those of other communities. In the communities, each community has a concentrated nucleus comprising of closely coupled user devices. The nucleus are also those devices which lie closer to other communities. This can be seen in the graph, for example, towards the top right of the blue community. Finally we observe, that two communities (located in the center) are close to all other communities, and comprises of users who frequently come into contact with one another, as well as most frequently with the nuclei of other communities.

We summarize with a table of statistics found for the graph. The statistics of diameter, ra-

dius and average path length will be the same as the one formed for the proximity graph for time spent together. Average path length was found to be less than 3. This indicates that on an average two nodes are at most 3 hops away. Also, the upper bound on the path between any two nodes was found to be 5.

Table 4.2: Statistics of proximity graph

Statistic	Value	Summary
Diameter	5	Max. distance between any two nodes.
Radius	3	Min. eccentricity of any node.
Avg. Path Length	2.33	Average length between any two nodes.
Avg. Clustering Co-efficient	0.69	Average of probabilities that any two randomly chosen neighbors of a node are connected to each other.

Chapter 5

Future Work and Conclusion

Using existing infrastructure we were able to offer ubiquitous applications in a privacy-safe and secure manner. We were able to make applications where users could seamlessly mark their attendance. We also provided promising results of simulations where a file could be transferred to a group of people, seamlessly and without using campus network resources, while conserving battery

Through the use of the mobile devices themselves to provide further information, we can improve upon our work. This includes data available from the mobile device itself, such as accelerometer and gyro data. This would enable to enhance the quality of service being offered, and also provide more accurate localization. Such and more systems need to always be sensitive to privacy and security issues, which is key in adoption of such systems. The system can be made to incorporate further personalization. This includes allowing the user to set how much information each application receives. This allows the user to set a higher access level for a better trusted application, and less or no access of information to a non-trusted one.

Prototype for actual mobile application, which use the system as the backbone, and transfer file using Epidemic routing can be made to validate the simulations. Such seamless file transfer, forms a key part of Ubiquitous computing, where students do not need to open a site to download course content, but get it delivered without using campus bandwidth resources. Device occupancy information can be used to provide a slew of ubiquitous services to users. A user connected to a congested WiFi Access Point, can be advised to move to a neighbouring less congested one, for better bandwidth and latency. Also, automated energy management systems can be developed, which can control power to electricity consuming devices, to conserve energy.

As part of the work, we were able to establish a preliminary ubiquitous environment for applications. This was achieved without deployment of any extra sensors. Ubiquitous environments in order to be well adopted need to be easily and less expensively deployable. WiFi LANs are an infrastructure which are already widely deployed. Thus, further applications using these and other existing infrastructure need to be explored. This will enable wide adoption of such systems and lead towards true ubiquitous computing.

Bibliography

[1] M. Weiser, "The computer for the 21st century," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 3, pp. 3–11, July 1999.

[2] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer*, vol. 34, pp. 57–66, Aug. 2001.

[3] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *Trans. Sys. Man Cyber Part C*, vol. 37, pp. 1067–1080, Nov. 2007.

[4] D. R. Mautz, "Indoor positioning technologies," 2012.

[5] S. Pace, "The global positioning system: Assessing national policies," 1996.

[6] A. M. Ladd, K. E. Bekris, A. Rudys, D. S. Wallach, and L. E. Kavraki, "On the feasibility of using wireless ethernet for indoor localization.," vol. 20, pp. 555–559, 2004.

[7] K. Gubi, "Roughmaps: Indoor positioning using existing infrastructure and symbolic maps," 2010.

[8] Y.-C. Chen, Y.-J. Chan, and C.-W. She, "Enabling location-based services in wireless lan hotspots," vol. 15, (New York, NY, USA), pp. 163–175, John Wiley & Sons, Inc., May 2005.

[9] A. N. Bharathan Balaji, Jian Xu, "Sentinel: Occupancy based hvac actuation using existing wifi infrastructure within commercial buildings," in *SenSys, Conference on Embedded Networked Sensor Systems*, 2013.

[10] S. F. Francesco Colace, Massimo De Santo, "Snmp-si: A network management tool based on slow intelligence system approach," *International Conference, FGCN 2010*, vol. 120, Communications in Computer and Information Science, 2010.

[11] "Rfc 1157 https://tools.ietf.org/html/rfc1157," 1990.

[12] M. F. Luca Schenato, Bruno Sinopoli, "Foundations of control and estimation over lossy networks," *Special Issue on Networked Control Systems*, vol. 95, Issue 1, 2007.

[13] "Net-snmp http://www.net-snmp.org/."

[14] "Cisco snmp object navigator tool, http://tools.cisco.com/support/snmp/do/browseoid.do?local=en."

[15] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," tech. rep., 2000.

[16] R. Friedman, A. Kogan, and Y. Krivolapov, "On power and throughput tradeoffs of wifi and bluetooth in smartphones," *IEEE Transactions on Mobile Computing*, vol. 12, pp. 1363–1376, July 2013.

[17] D. Bohman, M. Frank, P. Martini, and C. Scholz, "Performance of symmetric neighbor discovery in bluetooth ad hoc networks," in *In German Workshop on Mobile Ad-hoc Networking (WMAN04*, 2004.

[18] U. Lee, S. Jung, D.-K. Cho, A. Chang, J. Choi, and M. Gerla, "P2p content distribution to mobile bluetooth users," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 1, pp. 356–367, 2010.

[19] P. W. Nipendra Kayastha, Dusit Niyato, "Applications, architectures,and protocol design issues for mobile social networks:a survey," *Proceedings of IEEE*, vol. 99, Issue 12, 2011.

[20] A. S. F. Nazir, J. Ma, "Time critical content delivery using predictable patterns in mobile social networks," in *International Conference on Computation Science and Engineering*, 2009.

[21] P. Dhakan and R. Menezes, "The role of social structures in mobile ad-hoc networks," in *Proceedings of the 43rd Annual Southeast Regional Conference - Volume 2*, ACM-SE 43, (New York, NY, USA), pp. 59–64, ACM, 2005.

[22] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," 2009.

[23] V. D. Blondel, J. loup Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," 2008.