ENABLING TETHERLESS CARE WITH CONTEXT-AWARENESS AND OPPORTUNISTIC COMMUNICATION

by

PJ Dillon

Bachelor's of Science, Clarkson University, 2004

Submitted to the Graduate Faculty of

the Kenneth P. Dietrich Graduate School of Arts and Sciences in

partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2014

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

UNIVERSITY OF PITTSBURGH

KENNETH P. DIETRICH GRADUATE SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

PJ Dillon

It was defended on

October 13th, 2014

and approved by

Dr. Taieb Znati, Department of Computer Science

Dr. Daniel Mossé, Department of Computer Science

Dr. S.K. Chang, Department of Computer Science

Dr. Robert Sclabassi, Computational Diagnostics, Inc.

Dissertation Director: Dr. Taieb Znati, Department of Computer Science

ENABLING TETHERLESS CARE WITH CONTEXT-AWARENESS AND OPPORTUNISTIC COMMUNICATION

PJ Dillon, PhD

University of Pittsburgh, 2014

Tetherless care is a novel healthcare delivery paradigm that enables an interaction between caregivers and patients beyond the confines of traditional points of care. This thesis presents a synthesis of recent advances in wearable, ubiquitous sensing; mobile computing; wireless networks; and health information technology into a cohesive framework that enables and supports the tetherless care concept. Tetherless care is formally defined and modeled in a higher order logical framework. The model distills three relations between several classes in the model's domain of discourse. A prototype implementation is developed and evaluated to capture and represent the logical classes of tetherless care and provide the development infrastructure upon which the relational logic outlined by the model can be implemented. An algorithm is presented and evaluated to support the delivery of traffic between mobile devices and servers despite intermittent connectivity given the changing urgency of the patient's situation. And an example tetherless care application is presented, developed for the framework, and compared with its deployment on a similar platform. Results show that contemporary mobile devices supply sufficient power to support 24 hours of operation and that, at least, some patient environments provide sufficient opportunities for connectivity to reliably meet the demands of some tetherless care applications, ultimately leading to a conclusion of proof-of-concept for tetherless care.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ACKNOWLEDGEMENTS

I'd like to thank my advisor, Dr. Taieb Znati, and my doctoral committee members, Dr. Daniel Mossé, Dr. Robert Sclabassi, and Dr. Shi-Kui Chang, for their help and guidance throughout my thesis; my parents for helping me cope with my status among the working poor; my sister, Rachel, for her help in proofreading; and Jorge Cham, the creator of Ph.D. comics, for saving my sanity along the way.

1.0 INTRODUCTION

The healthcare systems in the industrialized world have come under scrutiny in recent years, particularly due to a trend in the U.S. of both increasing costs and poorer outcomes, that exceed those of other countries [9]. These trends burden private individuals, corporations offering medical insurance, and public financing with an unsustainable status quo that, if left unaddressed, will ultimately leave care out of the reach of a large portion of the population.

Sources of this costly and poor performance are numerous and not only pervade many aspects of complex healthcare systems but also extend into other industries, such as industrialized food. A variety of solutions have been proposed or implemented that seek to address them. Among them is a movement towards a greater reliance on information technology (IT) to provide greater efficiency, share information across providers, and tailor care to the patient. Yet these solutions suffer a flaw insomuch that they fail to pervade into the patient's everyday life. By extending the capabilities of healthcare delivery into everyday life, the healthcare industry can remotely monitor patients, observe phenomena in the patient's real-world environment, reduce the duration of hospital stays, reduce rehospitalizations, and provide an mechanism by which the elderly can remain independent longer, all of which could contribute to a reduced cost of care. The work presented here offers a novel solution to support the remote care of healthcare patients given the coming move towards a larger reliance on IT.

The remainder of this chapter outlines this shift to a more IT-friendly healthcare system and the proposed solutions for extending the delivery of care into the everyday lives of patients. Section 1.1 discusses the sources of costs and poor performance and proposed solutions for cost reductions. Section 1.2 defines tetherless care and enumerates its requirements. Section 1.3 defines the problem addressed by this thesis formulated as a set of questions. Section 1.4 discusses the approach

taken to address this problem and the limitations of the results found. And section 1.5 outlines the structure of the remaining chapters of this thesis.

1.1 BACKGROUND AND MOTIVATION

1.1.1 Costs of Care

In the coming years, individuals in a demographic known as the *baby boomers* will reach retirement age. And, by 2019, the percentage of the U.S. population over the age of 65 will have increased from 13% to 16%. This demographic changes the population distribution such that the portion of individuals at advanced age outnumber younger demographics, which is unprecedented in human history. Furthermore, social services, such as Medicare and Social Security in the U.S., are designed for pyramidal distributions where larger, younger demographic groups contribute a small portion of their labor to support relatively few elderly. Coupled with an again unprecedented and lengthy life expectancy, this shift of demographics is poised to strain the social contract on which it's founded.

Moreover, the most common chronic conditions and diseases—heart disease, diabetes, hypertension, and obesity—correlate with advancing age. These diseases can produce numerous episodes that require hospitalization, surgeries, and other treatment. Also, they not only develop over a long period of poor lifestyle choices, dietary deficiencies, and bad habits, but also often require long term care, medication, and treatment beyond the bounds of traditional points of care. And, after an acute episode, many patients are often ill-equipped to properly manage or alter their behavior as indicated by the fact that ninety percent of rehospitalizations within 30 days of the first episode are unplanned (which accounted for $17.4 billion in medicare spending in the U.S. [42]).

Simultaneously, patients in need of care are not the classical discerning consumers of market economics. Acute episodes do not afford the time to examine and determine the least costly care option. Response infrastructure is further predicated on proximity, delivering a patient to the closest facility in the least amount of time. Furthermore, the incurred costs of care are often offloaded to insurance companies, which absolves the patient of a cost-benefit analysis with his or her doctor.

For these reasons, hospitals have no incentive to reduce prices to compete in a traditional market, yet the aging population continues to increase the demand for care. As such, administrators are afforded the luxury of arbitrary price inflation. [14]

Altogether, the effect of this evolving environment is a projected total spending on healthcare in the U.S. of 4.5 trillion dollars by 2019, almost one fifth of the U.S. GDP, with no indication of a trajectory change [4]. This situation demands new, innovative, and low-cost means of achieving healthcare delivery.

1.1.2 Proposed Solutions

Contemporary arguments characterize current healthcare practices as *reactive* and *disease-centric*. That is, the healthcare system, including insurers, generally waits for and responds to acute events–insulin shock, heart attacks, strokes, ruptured arteries, etc.—that result from long term chronic conditions. The response is limited to the symptoms and effects of the event, i.e. the disease that required hospitalization, and every patient experiencing the same event is treated in a similar manner. Furthermore, little effort is made to involve healthcare resources in the patient's everyday life in order to effect a healthier lifestyle prior to or after these events.

By contrast, a *patient-centric* healthcare paradigm tailors the response to each individual patient according to his or her history, capabilities, and needs. A context of prior care, medical history, and available resources follows the patient through a fluid transition from point-of-care to point-of-care such that each caregiver can exploit prior knowledge and adjust treatment specifically to the patient. Furthermore, care extends into the patient's home and everyday life, becoming an ongoing dialog between her and one or more caregivers.

It is believed that information technology can enable this shift to a patient-centric paradigm. Specifically, cloud-based storage systems, termed Personal Health Records (PHRs), could store the entirety of each patient's medical history and be universally and securely accessible at each point of care with which the patient engages. Caregivers could readily access prior test results, current prescriptions, and instructions or notes from other caregivers. The system could contain logic to determine common issues, such as known drug interactions. Finally, a web-based interface provides the patient with remote access to log activity, request advice, and receive instructions.

Simultaneously, other work has explored what are known as body area networks (BANs): a group of body-worn sensing devices tasked with monitoring vital signs and environmental signals while the patient goes about everyday life. Each device is specialized for one or more vital signs. The devices form a small collaborative wireless network to collect, process, and upload information to other entities, such as caregivers.

1.2 TETHERLESS CARE

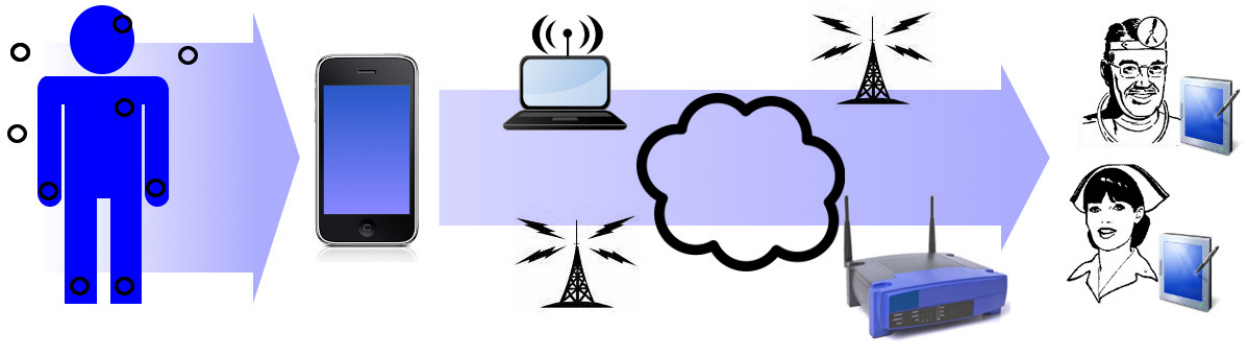


Figure 1: Tetherless Care

Sensing devices on or around the patient collect and transmit sensor information to a smartphone or device with similar capabilities where intelligent analysis of the patient's state can take place. Important information is transmitted to family and healthcare professionals using opportunistic connections to any available networks in the patient's environment.

This thesis envisions a novel paradigm, called *tetherless care*: a patient under observation beyond the confines of traditional points of care and free to go about his or her everyday life. The advances in health information technology in support of the patient-centric paradigm coupled with the advances in body area networks and home health monitoring have created opportunities to enable ambulatory care and to strengthen the ongoing dialog between patient and caregiver with tangible, unbiased medical data.

Currently, acute events requiring hospitalization often involve a period of observation where the patient is equipped with several bedside sensing devices that monitor vital signs and alert hospital staff if the condition of the patient deteriorates. While otherwise stable, hospital staff is

free to attend to other matters, but the patient remains hospitalized to exploit their proximity in the event of complication.

While not amenable to all conditions, a large and potentially growing class of conditions could be similarly under observation in an out-patient manner by employing a system of sensors and wireless networks as illustrated in figure 1. The sensors are attached to the patient or embedded in his or her environment. Data is collected and analyzed on his or her smartphone. And important information or alerts are uploaded to caregivers or servers through one or more available networks. By releasing the patient from the facility, hospital resources are freed for other purposes and the resulting shortened hospital stay translates into cost savings. Moreover, the patient returns to work and his or her everyday life more quickly, providing greater dignity and less burden to employers.

1.2.1 Landscape

Tetherless care will likely have a wide range of applications in the foreseeable future. As such, it offers a space of interesting investigation along several dimensions. One end of the spectrum encompasses emerging self-contained commercial fitness devices that provide personal lifestyle feedback such as caloric expenditures, step counting, sleep quality, and activity detection. These are voluntarily adopted by otherwise healthy individuals to quantify and optimize their daily activities for health improvement. The applications they support have little criticality and need not communicate much information to servers or caregivers with any degree of urgency.

The other end of the spectrum remains an open question. There likely exists some health condition with a maximal risk that still allows the free mobility of the patient given the monitoring and response capabilities of a novel, cutting edge system targeted at the condition. Perhaps a recent heart attack victim can be released from the hospital provided that (1) the likelihood of a subsequent episode is low and (2) he or she avoids stressful situations. With these types of applications, a deterioration in the patient's state necessitates a significant response from the healthcare community such that the accuracy in assessing that state is paramount. Similarly, minor changes in the patient's state could be worrisome and important. And caregivers would likely require regular updates to ensure the patient is adhering to instructions and the system is functioning.

Other applications that likely fall somewhere between the two include the management of long-term chronic conditions such as hypertension, diabetes, obesity, and heart disease, which may present a lower risk of developing into an acute, critical episode yet demand careful attention to the patient's state and activity. These applications serve two distinct purposes: (1) like the commercial heath improvement devices, they monitor the daily, routine activity of the patient so that either the system or a caregiver can nudge the patient towards a lifestyle that no longer contributes to the condition, and (2) they must also detect acute, critical events that require attention, e.g. rising blood glucose such that an insulin shot is needed. From patient to patient, the degree of criticality may change greatly depending on his or her needs, the distance from care, and the specifics of his or her condition.

In all cases, a context-aware infrastructure of sensors and wireless networks enable and support these applications, and, as such, they are all applicable to the work of this thesis. As these systems pervade into everyday life, they present a novel computing platform that will require infrastructures that mitigate the challenges of the system, ease the application development process, create and support industry standards, and integrate various system components. As both the sophistication of this infrastructure and the sensing capabilities increase, the boundary of tetherless care, i.e. the capacity for delivering remote care, may grow to encompass a larger and larger set of conditions and diseases.

1.2.2 Challenges

The applications described above present a diverse set of challenges. At a fundamental level, they must interpret data from one or more sensors within a context of their user's current environment and are therefore limited to (1) those aspects of the user and his or her environment that can be sensed and (2) the areas of the environment into which a sensor can be embedded for an extended period of time.

Algorithms must then extrapolate activities and context from the sampled data in real-time. The precision and accuracy of these algorithms varies across deployments. A calorie counting algorithm can include a certain amount of error in its calculation with little consequence. But, where assessing some aspect of the patient's state results in a significant and costly response from

the healthcare community, the algorithms cannot initiate such a response as a result of a false positive. Moreover, where initial attempts at tetherless care applications with naive algorithms may only detect and respond to acute, critical episodes at the moment they occur, the real potential of these algorithms is the ability to detect an oncoming event and initiate responses earlier than as a result of traditional care. This predictive ability coupled with the need for certainty presents a significant challenge.

Furthermore, where the goal of tetherless care is the full mobility of the user, the system must be able to interpret the activity and context of the user in a wide range of environments, each containing a set of stimuli that affect the physiological state of the patient and thus likely affect the data collected by system sensors. For instance, a heart rate of 140 bpm while at the gym could be considered normal but while at home on the couch may indicate a serious condition.

Also, these environments make no guarantee of safety or available infrastructure. While the system observing a user freely moving about at home can make use of embedded home sensors, assistive technologies, well provisioned wireless infrastructure, and readily available Internet services, once the user leaves this environment, the scarcity of computing infrastructure likely limits the system to the set of devices carried by the patient, and it must make use of opportunistic connections to whatever wireless networks the user happens to encounter.

As such, there likely exists environments ill-equipped to support some operation of the system, either due to limited power, storage, or computing resources or due to deteriorating patient state. Where possible, these systems must predict the likely future availability of resources along with the likely future condition of the patient, evaluate the risk, and allocate resources appropriately. The projection of future computing environments allows the system to escalate its action over time not only in response to the patient's state but also in response to its own ability to assess that state. In doing so, it may then be able to exploit computing resources while they're available rather than finding situations where the only option is to elicit help from the patient or bystanders via audible alerts. Conversely, this projection may also allow the system, in more predictable scenarios, to reduce its monitoring effort, tolerate greater risk, and conserve resources.

The ability to make these predictions requires knowledge of the environmental dynamics with respect to the rate at which a given patient's state can deteriorate, the repetitive and typical behavior of the patient, and the ways in which computing resources can fluctuate over time.

1.2.3 Requirements

This section formally enumerates a set of requirements for tetherless care systems. These are divided into requirements imposed on tetherless care applications and quality of service requirements imposed on data transferred between patients and caregivers.

1.2.3.1 Application

Data Rates

Table 1 lists the typical sampling rates of various sensors, which range from less than 1 Hz to 1 kHz. Assuming 16-bit digital samples with no data compression techniques, a combination of sensors can generate raw data rates on the order of 10s to 100s $kbps$. A network of sensors around the patient must properly manage and sustain these rates, and the application must process them in real-time. Furthermore, where longer range wireless networks may not be available, applications cannot fully rely on this communication for critical analysis and assessment of the patient's state.

Table 1: Typical Sampling Rates [68, 69, 63]

Vital Sign	Sampling Rate (Hz)
Pulse	.5–4
SpO2	50
EKG	120
ECG	500
EMG	1000
Accelerometer	100
Gyroscope	100

Pervasive and Ubiquitous Operation

The form factor, placement, and number of sensors or supporting devices must be designed for

long term use and patient comfort. Distractions must be minimized. However, the patient's state must remain under constant observation subject to environmental constraints.

Respect for the Caregiver's Time

The effect of the tetherless care paradigm on healthcare infrastructure is not yet known in terms of where and how personnel and other resources will be dedicated to tetherless patients. A reasonable assumption, however, is that one caregiver will likely be responsible for a number of tetherless patients similar to current observation wings in hospitals. Thus, while it may not be possible to limit or discard information delivered to caregivers, tetherless care systems must incorporate mechanisms to intelligently monitor the patient's state such that the caregivers are only alerted when necessary.

Context Awareness

The system can only infer meaning from and respond to the signals it can observe, and the scope of observation is confined to the available sensing devices. Any given signal, e.g. a ECG signal of the patient's heart, can fluctuate due to a large number of possible causes, and it becomes necessary for the system to include other sensors that provide additional signals capable of differentiating the important causes of a given fluctuation from the benign ones. The *context* of the patient is defined as the collection of these signals, or their digital representations rather, that are necessary to algorithmically identify and respond to important events.

This definition implies a relationship between the algorithms employed for a given application and the sensing devices needed to support them. Different algorithms can depend on a disjoint set of sensing devices to achieve the same monitoring functionality. And improved sensing technology can inform new algorithms.

This definition also includes a dependency on the history of one or more signals. In other words, to make sense of a signal at time, t, an algorithm may require access to the signal at time, $t-k$, for some arbitrary k.

1.2.3.2 Quality of Service Requirements

Reliability

Information transferred between the patient and caregivers may have a range of reliability requirements. Alerts and supporting data that indicate a critical condition must be delivered

without failure despite poor link quality or periods of disconnection. However, under scenarios where it's necessary to stream raw vital sign samples to caregivers, several lost samples may still provide enough fidelity to deliver care.

Context-Dependent Delay

Data and alerts must be delivered to caregivers within a reasonable deadline, but the interval between the creation of the data or alert and the time at which it must be delivered varies from situation to situation. Critical situations often carry short deadlines whereas more benign situations can tolerate longer deadlines. The system must be capable of managing and responding to a variety of delay requirements.

Security and Privacy

The 1996 Health Insurance Portability and Accountability Act (HIPAA) lays out a number of security and privacy mandates. It defines four broad security requirements for Health Information Systems: Access Controls, Audit Controls, Integrity Controls, and Transmission Security. Access Controls must identify the systems users (healthcare professionals) and grant access to patient information based on the role the user is currently fulfilling. Audit Controls require a system to log access grants and other activities for a period of six years. Integrity Controls require a system to put in place mechanisms ensuring health information is not altered or destroyed. Transmission Security requires steps to ensure data is not accessed in transit over a network.

1.3 PROBLEM STATEMENT

The operating environment of tetherless care presents a novel set of challenges. Sensors can potentially produce a large volume of data with critical requirements. Mobile devices have limited processing and power resources. Wireless networks are intermittently available. The combination of these demands and limitations requires the system to carefully allocate resources in proportion to the criticality of the current situation. The analysis of the current situation is further complicated by a couple factors: (1) that knowledge must be and can only be extrapolated from the stream of

sensor data and (2) that the circumstances surrounding a patient may influence some sensor data so as to deceptively mimic an important event.

In-hospital care avoids these complications by managing and maintaining a regular, predictable environment around the patient, confining or preventing the patient's movement to a bed or ward, and provisioning ample power and computing resources to the devices monitoring the patient. As such, the range of influences on the given vital sign sensor's sample is reduced, allowing monitoring systems to assume a given change in sensor data is directly induced by an important event.

By removing these assumptions from the context around the patient, endowing the patient with full mobility, and deploying a tetherless care system of sensors and wireless networks around him or her, the fundamental question arises of *whether it's possible to provide tetherless care that is commensurate with traditional, in-hospital care*.

Here, *commensurate* is defined as viewed from the perspective of a caregiver observing multiple patients. Within an in-hospital scenario, patients under observation are assigned a hospital bed, are attached to one or more bedside monitoring devices, and are generally left alone provided caregivers are only a few steps away to respond when one of the devices indicates an issue. If this proximity and rapid response are unnecessary, meaning the caregiver could still fulfill his or her responsibilities with a greater distance and a delayed response, the care he or she provides with tetherless care is then considered commensurate with the case of in-hospital care.

This thesis argues the affirmative to the aforementioned fundamental question by addressing three sub-problems formulated as the following questions:

Question 1. *Is it possible to develop an abstraction to capture the context and situations of tetherless patients that can be efficiently implemented on a resource constrained device?*

Question 2. *Is it possible to exchange information between tetherless patients and caregivers in a reliable and timely fashion given the mobility of the tetherless patients, the intermittence of connectivity, constrained resources, and the situation of the tetherless patient?*

Question 3. *Can the disparate pieces of the tetherless care abstraction, intermittently connected environments, and data with quality of service requirements be integrated to provide an infrastructure to support the needs of a tetherless care application?*

1.4 APPROACH AND LIMITATIONS

The investigation follows the design, modeling, implementation, and evaluation of a proof-of-concept prototype system for tetherless care. The scope of the prototype and its evaluation is limited to a framework intended to support and simplify the development of tetherless care applications deployed on a mobile device, such as a smartphone. A set of abstractions are defined to codify the components of the patient's context, the knowledge extracted from it, and the responses the system must perform. For a given application, a developer can create a set of these components or select from those available from third parties such that each component manages the necessary signal processing, feature extraction, decision making, or response to a specific, limited domain of the overall context. Then, several algorithms are presented that operate on these components to aid in analyzing the risk of the patient's situation.

A specific focus is taken on applying these algorithms to assess and appropriately respond to the risk of the patient's network environment to effect the timely yet energy-efficient delivery of network traffic from the patient's on-body device(s) to servers and caregivers. This environment consists of multiple, intermittently connected networks of varying technologies, yet tetherless care applications have stringent, context-dependent quality of service demands on the data transferred to servers and caregivers. Given the complexity of managing this relationship between the network environment, which itself can be part of the context, and the demands of the application, a middleware is presented that abstracts the use of these algorithms such that developers are free to focus on the medical aspects of the patient's state.

Since observation beyond traditional points of care is unprecedented, policies for assigning or reassigning responsibility, assessing compensation, responding to patients, and administering tetherless care systems are not yet known. Where tetherless care system architectures and healthcare infrastructures will likely mutually adapt to one another, it's difficult to model and examine the impact of caregiver mobility and the network traffic across healthcare infrastructure.

Specifically, contributions of this thesis include the following :

- A model of tetherless care described in formal language, which encodes the concepts of the patient's context and situation, and provides a set of abstractions that facilitates reasoning about and developing context-aware tetherless care applications;

- An algorithm for predicting the likely future state(s) of the patient given his or her current state,
- An algorithm for managing network traffic with quality of service requirements within an environment of multiple intermittently connected wireless networks;
- An architecture that materializes the tetherless care model, facilitates the creation of tetherless care applications, enables the execution of these applications on resource constrained devices, and supports the delivery of tetherless care network information;
- An analysis of the network traffic management algorithm and its sensitivity to several of its inputs given the mobility and connectivity profile of generally well-connected, urban patients;
- A comparison of several implementations of an example tetherless care application, some utilizing the tetherless care architecture and some not.

The primary platform of the investigation is a set of Motorola Droid Pro running the Android 2.2, and later Android 2.3.4, operating system, which imposed several usage restrictions that limited the realism that the prototype could capture. Primarily, cellular data plans were not associated with these devices, limiting the devices to WiFi connectivity, and this prevented the investigation from determining the effect the additional cellular connectivity would have on the system. Results are then limited to the behavior of the WiFi environment encountered during the investigation. Moreover, when compared to later versions of Android, particularly later than Android 4.0, and possibly other manufacturers, these devices did not seek out and associate with WiFi access points as actively. They generally only did so as the result of user interaction with the device, which complicated investigations into their pervasive and ubiquitous operation in the background of users' lives.

Also, Android 2.2 and 2.3.4 prevented the use of the accelerometer and camera sensors while the screen was off, which are sensors that support the PandaCare application described in chapter 5. While they were emulated for the study, the use of these sensors impacts the energy usage of the device.

The overall impact of these limitations reduces the obtained results to an upper bound on the energy usage of the system given the battery technology circa 2010.

1.5 THESIS STRUCTURE

The remainder of this document describes the development and evaluation of a proof-of-concept prototype for tetherless care.

1.5.1 Literature Review

Chapter 2 reviews the state of the art in mobile health monitoring, intermittently connected networks, and utilizing multiple network technologies. A large body of work has demonstrated the ability to assemble sensor networks around users both in specific, well known environments and in more general ubiquitous environments. The literature presents protocols for managing and reducing the stream of data samples. And it supports the architectural design of the hardware components adopted for the tetherless care prototype. As a result, the evaluation here can assume a system of sensors as a black box that effectively and reliably delivers sensor data to a smartphone in a low power manner.

1.5.2 The Tetherless Care System

Chapter 3 expounds upon the concept of tetherless care. A formal definition of a tetherless care system is defined and includes definitions of the the patient's context, the patient's situation, and a tetherless care application. This model forms the basis upon which an algorithm is presented for assessing the risk of the patient's state in an application specific manner and with which the case is made for a middleware that abstracts the management of network communication given the intermittently connected environment. The chapter outlines the hardware components employed by tetherless care systems. A software architecture is then presented to operate upon these, manage the system components, and support tetherless care applications. This architecture focuses on the patient's smartphone, which manages the system of sensors around the patient and collects sensor data from them. An evaluation of this architecture demonstrates its feasibility on a contemporary mobile device and illustrates the need for the intelligent use of the network technologies.

1.5.3 Information Delivery

Chapter 4 presents a study of the intermittently connected network environment in greater detail and a novel set of algorithms for the management of real-time data within this environment. A four month study indicates that, while WiFi networks are intermittently available, significant predictability exists in patient movement behavior such that data can be algorithmically scheduled for upload in future network opportunities with probabilistic guarantees. Results show that, with greater than 80% probability, the system is able to limit and delay network connectivity in favor of power savings yet still meet the demands of data with quality of service constraints.

1.5.4 A Case Study: e-Button and PandaCare

Chapter 5 examines a comparison of several implementations of an example tetherless care application for monitoring dementia care patients. One implementation mimics an application deployed to a device called the e-Button, which contains minimal logic on the device and requires a long lasting, always-on connection with a server to maintain system operation. Other implementations utilize the tetherless care framework. Results of the comparison demonstrate that the original e-Button implementation is unable to meet the reliability, pervasive and ubiquitous, and mobility requirements whereas the other implementation achieves the desired operability. However, the tetherless care implementations were less able to meet the context-dependent delay requirements of the application's data utilizing only WiFi-based connectivity, which was an expected result given the results collected from chapter 4.

1.5.5 Conclusion

Finally, chapter 6 concludes the work and provides several avenues of future research. It reviews the novelty of the tetherless care framework and the simplicity and flexibility with which a tetherless care application can be designed to achieve context-awareness and long lasting operation on a resource constrained device. And, where the investigation into the performance of this framework was limited to an older device, an outdated operating system, and the WiFi connectivity around

users in an urban setting, the chapter discusses future work needed to examine a better performing operating system, other cellular network technologies, and a range of connectivity environments.

2.0 LITERATURE REVIEW

A number of applications and systems have been proposed towards enabling tetherless care: a survey of some of these applications can be found in [46]. The proposed systems can be categorized by their scope and architectural features. Some systems intend only to be deployed in the patient's home. Others confine themselves to on-body sensing with a system of wireless sensors, and among these are systems that assume a central data collection point on the body or that place this collection point off the body on some server. Surveys of these architectures can be found in [23, 52].

The majority of this prior work demonstrates the feasibility of collecting sensor data to effectively monitor the patient's condition in real-time, but, as the survey in [23] outlines, the communication technologies supporting this monitoring are lacking. The work of this thesis addresses this shortcoming by borrowing ideas from the delay-tolerant community and adopting a system architecture that assumes a central data collection point on the patient's body to overcome the uncertainty inherent in the network environment due to the patient's mobility.

The remainder of this chapter presents the state of the art in these systems, outlining several different avenues of research and categorizing approaches. Section 2.1 reviews prior work that targets a particular venue, such as the patient's home, a nursing home facility, or hospital. Section 2.2 reviews prior work in systems that organize a network of sensors around a central node with more processing and storage capacity, such as a smartphone or PDA. Section 2.4 discusses several areas of prior work that considered the use of multiple network technologies, tolerating long delays in the network, and studying smartphone usage, which illustrates how often modern smartphones are awake and looking for network opportunities. And section 2.5 concludes the chapter.

2.1 FIXED VENUE MONITORING

Several systems have sought methods of assisting the patient in managing chronic conditions in his or her home. Testbeds of "smart living spaces", such as The Placelab [39], The Aware Home Project [2, 43], and The Smart Medical Home [3], have installed hundreds of sensing devices throughout a living space to monitor the behavior of inhabitants, provide feedback, and interact with third parties such as healthcare professionals. Other projects have also sought to develop an architecture to abstract the details of in-home sensor networking and interface these systems with Internet Protocol (IP) networks, e.g. [1, 31, 77, 78]. These endeavors demonstrate the feasibility of deploying systems to well known, more predictable environments.

CodeBlue [54, 67, 68], MEDiSN [45], and MASN [32] target hospitals, nursing homes, and emergency management scenarios. Each employs a multi-hop wireless sensor network consisting of small, battery-powered wireless sensors worn by multiple patients. The sensors collect raw vital sign samples from each patient and form a single, collaborative sensor network that streams the samples from each patient to a server.

2.2 SMARTPHONE/PDA-SUPPORTED MONITORING

Other systems have targeted more general environments, yet still fail to address the issue of intermittent connectivity [15, 48, 55, 60, 62]. The architectures of these works assume a more tiered structure, utilizing a smartphone, PDA, or similar device to manage a smaller sensor network around a single patient and exclusive to the patient. Raw data samples are collected through this sensor network to the smartphone, which then uses IP-based networks to forward them to servers or caregivers. Where many assume WiFi-based networks, MobiHealth [44, 47, 73] relies exclusively on wireless wide area networks (WWANs) for communication with fixed infrastructure and dedicated healthcare professionals.

These works assume a constant, always available wireless network for forwarding sensor information from the patient's smartphone or PDA to a central server, and this assumption encompasses

the real-time nature of the transferred data. A well provisioned, always available network will likely always deliver information in a timely fashion.

One of the main defining characteristics of tetherless care is the patient's ability to roam, leaving behind the curated monitoring environments of their home and the broadcast radius of wireless access points, crippling most proposed systems. Even in WWANs, dead zones exist, for which the system must account given the critical nature of these applications.

In considering this novel paradigm of intermittent connectivity, such environments are likely less capable of supporting a consistent stream of sensor data to the server, which may reduce the set of applications amenable to tetherless care and engenders a question as to the range of applications it can support. Moreover, it invites an investigation into the interplay between the dynamics of the environmental resources and the range of demands placed on the system by typical tetherless care applications. The chapters that follow demonstrate that this space is feasible but does not explore its bounds.

2.3 BODY AREA NETWORKS

The on-body components of the monitoring systems in prior work, small wireless networks of wearable sensors, termed *Body Area Networks* (BANs), have gained special attention due to the communication challenges introduced by the human body. Natarajan et al. [21, 57, 58, 59] studied the airspace around the human body and concluded that an adaptive, multi-hop architecture for a body area network achieved the greatest performance and energy savings. Other work, such as [11, 12, 19, 20, 22, 29, 40, 56, 61] proposed techniques for reducing the volume of traffic across the BAN by profiling vital sign signals and predicting future values. Then, Patel et al. [63] achieved event detection and determined the real-time patient state for parkinson's patients by analyzing sensor data collected from a BAN.

Spadini et al. [69] and Varshney [74] considered the addition of auxiliary sensor data to achieve context-aware monitoring, placing the physiological response of the patient into a broader context of the patient's current situation.

2.4 THE NETWORK ENVIRONMENT

This section reviews the state of the art from several different research areas that contribute to the approach taken to achieve the tetherless care framework and prototype. Most of this work originates from the Delay Tolerant Networking Community [8] that centers around an infrastructureless networking paradigm where mobile devices employ a store-carry-and-forward paradigm to physically move data across a geographic space in hopes of getting it closer to a given destination.

2.4.1 Multiple Networks

Modern smartphones are or will soon be equipped with several hardware radios, e.g. 802.11, GPRS, UTMS, Bluetooth, and near field communications. These new networking paradigms give us an opportunity to achieve greater communication reliability because the ability to transmit information is not solely dependent upon the performance and availability of a single network.

In the space of tetherless care systems, to our knowledge, only Varshney [74, 75, 76] argues for the use of multiple network interfaces as a means of increasing reliability in data transmissions but only under the assumption of constant, real-time streaming of raw sensor data to fixed infrastructure components. He recognizes the need of an algorithm for selecting an appropriate network based upon the respective characteristics of each, but leaves an analytical model for future work.

Recently, Balasubramanian et al. [13] and Ra et al. [64] studied the energy costs and performance of mobile 3G and WiFi networks. Their findings confirm the intuition that 3G network interfaces provide more sustained throughput, are more available, but consume more power. WiFi, however, has the potential to achieve faster data rates at lower costs. Huang et al. [33] found that 4G/LTE networks, in general, provide greater throughput than WiFi but with an energy cost exceeding both 3G and WiFi. Also, Ra et al. [64], presents an algorithm that solves what they term the "link-selection problem" of choosing among currently available network technologies in order to minimize power, but this algorithm does not consider real-time data or deadline guarantees.

Lee et al. [49] examined mobility patterns of 100 iPhone users in an urban environment. Their results show that urban mobile device users experience 70% WiFi availability, and 65% of mobile data is currently uploaded over WiFi networks. Furthermore, by introducing a delay on the order

of tens of minutes, this percentage increases to over 75% of mobile data uploads, and, with a delay of two to six hours, it can increase to around 80%.

The result of this prior work informs a heuristic by which the tetherless care framework makes its own link selection, favoring Bluetooth over WiFi, WiFi over 3G, and 3G over 4G. However, in practice, modern operating systems do not provide a mechanism to step down from 4G or LTE to 3G for power consumption purposes. The framework requires applications to declare explicit delivery constraints for network data. With these, an algorithm can determine if uploading can wait for more favorable, low-power network opportunities, such as WiFi.

2.4.2 Host Availability

This section presents an overview of investigations into smartphone usage and movement behavior in the literature.

Falaki et al. [24] studied the usage behavior of smartphone users, and found a range of usage patterns from short, infrequent interactions of a few minutes to prolonged usage periods of several hundred minutes. In terms of inter-device interactions, data may only be traded and routed in those times when two devices are co-located and in use. Otherwise, one device is likely to be in a sleep state. Thus, the findings of this work reveal greater limitations on mobile networks dependent upon human-carried mobile devices. The study presented in chapter 4 results from very infrequent device interactions, and more work is needed to assess the environment across a range of usage behavior.

A number of studies have examined human mobility from a networking perspective, e.g. those in [16, 34, 35, 38, 65]. They characterize the heavy tail power-law distribution of times between node contacts, finding that some pairs of nodes contact each other frequently while others extremely infrequently, and they attribute this difference to the social ties among the humans carrying the devices.

More recently, an argument has been made that the growing ubiquity of WWANs has or will obviate infrastructure-less mobile networks like these. Addressing this, Lindgren et al. [50] and Hui et al. [37] studied the impact of additional infrastructure on mobile delay tolerant networks, ultimately finding that the two paradigms compliment each other.

2.4.3 Delay Tolerant Networking

The concept of providing an abstraction above several different network technologies has been incorporated into, if not originating from, the concept of Delay Tolerant Networking (DTN) [25, 26] (also see the Haggle Project [66]), which argues for an abstraction that unifies the interaction with each network technology and allows for the implementation of a protocol to intelligently select among them for communication.

However, as the name suggests, the general DTN paradigm is not well suited for the quality-of-service requirements of tetherless care. DTN is designed for extreme networking environments where no fixed infrastructure is available, end-to-end paths can't be guaranteed, and end-to-end delays are extremely long and variable. Mitigating these environments requires participating nodes to adopt routing capabilities and a store-carry-forward approach, buffering each message until the next connection becomes available and recomputing paths as newer information is received. Given that the delays are subject to the movements and chance encounters of mobile devices, a consideration for the end-to-end delay of the network is sacrificed, i.e. extreme delays are tolerable.

Routing in DTN acquired a significant amount of attention because traditional routing algorithms failed in the face of disconnections and the lack of available routing information or the ability to quickly update it presented significant challenges. As such, many of the early protocols adopted a random forwarding strategy [28, 30, 70, 72]. Others, recognizing the heavy-tail distribution of time between successive encounters, sought to identify and prefer transmissions across links formed by hosts that encountered each other more frequently, some by observing the past encounter frequency [51, 71] and others by profiling the social ties among the humans carrying the mobile devices [27, 36, 79].

The work in this thesis adopts the concept of tolerating delays and utilizing multiple network technologies but presents a novel solution for mitigating these aspects of the networking environment given the quality of service requirements of the data and that, in essence, considers environments consisting of only a single DTN hop.

2.5 CONCLUSION

The scope of prior work demonstrates the feasibility of several facets of tetherless care and provides the basis of a synergy presented in later chapters. In particular, hardware platforms exist with form factors capable of (1) achieving the needed proximity to vital signs and other environmental signals and (2) continuously monitoring these signals in a pervasive and ubiquitous fashion. Collocated on these platforms are low-power wireless transceivers with protocols to organize wireless sensor networks tasked with collecting sampled data. Compression and profiling techniques ensure the aggregate sensor data rate will not overload the capacity of the sensor network. And algorithms can operate on this data to achieve a level of context awareness within a given application. Altogether, this prior work addresses the data rate, pervasive and ubiquitous operation, and context awareness requirements of tetherless care.

The work in the remaining chapters presents a prototype tetherless care system and confirms it's capable of meeting these requirements as well. Then, it presents an algorithm that, borrowing concepts from DTN to utilize multiple networking technologies and mitigate intervals of disconnection, works to meet the delivery deadlines of network data despite the intermittent connectivity. In doing so, it meets the delay and reliability requirements of tetherless care.

3.0 THE TETHERLESS CARE SYSTEM

Tetherless care operates in a challenging environment and must balance the tradeoff between limited resources and application demands with life threatening criticality. Furthermore, as the context awareness requirement outlined in section 1.2.3 states, a tetherless care system must be capable of responding to changes in the available resources and in the patient's state such that the patient's safety is never compromised. For example, the system could tolerate a patient with stable vital signs and a lower risk medical condition traveling into an environment with more intermittent connectivity and at a greater distance from a power source. On the other hand, with greater risk in the patient's condition, the system may act to alert the patient or caregivers when the patient moves into such an environment in an attempt to prevent such movement.

This evaluation of the patient's situation depends on several factors including the nature of the patient's condition, the capabilities of the patient, the history of the patient, and the characteristics of the environment. Much of this evaluation is beyond the scope of this work. For example, an algorithm for using accelerometer samples to classify or characterize an elderly patient's frailty within some range of physical fitness is left for future work. Furthermore, exploring the bounds of the range of possible levels of physical fitness is beyond the scope of this work. Moreso, there are numerous other dimensions along which patient movement and behavior could vary in ways that impact an application's response.

Nevertheless, this chapter seeks to push the state of the art towards exploring and addressing these variations in patient behavior. It first formally defines and models tetherless care. The observable environment of the patient, i.e. the context of the patient, along with the patient's state derived from this context are partitioned into respective sets of abstractions and each algorithm involved in the evaluation of the patient's state is encapsulated in its own abstraction. Algorithms are presented for predicting possible future states of the patient and computing the risk of his or her

current state. This chapter then formally constructs a space in which these algorithms can be used to mitigate the delivery of network data with context-dependent quality-of-service requirements given the intermittently connected network environment of the patient. It defines a reference implementation that renders the formal definition into an architecture to support the development of tetherless care applications and a middleware that encapsulates the risk analysis and network traffic management algorithms. This reference implementation serves as the basis for evaluation in later chapters.

By partitioning the context and state of the patient into sets of abstractions, the framework and its implementation are flexible enough to evolve with the state of the art as sensor technology improves, more becomes known about the variation in patient behavior, and more sophisticated algorithms are developed for computing aspects of the patient's state. For example, if an inconspicuous sensor could be developed to capture the patient's neural activity, i.e. an inconspicuous EEG sensor, an application need only augment the context and state with a set of abstractions that represent derivations from the new sensor data. Moreover, these sets are free to vary from application to application such that no resources or computation need be wasted.

The remainder of this chapter discusses the progression from model to architecture and its evaluation on a resource-constrained mobile device in greater detail. Section 3.1 describes an assumed set of hardware devices that will support tetherless care systems for the foreseeable future. Section 3.2 formally models the basic functionality that tetherless care systems must accomplish, which includes formal definitions of the patient's context and situation. Specifically, any tetherless care system must perform three basic functions: it must access vital sign and environment signals, it must assess the patient's state, and it must communicate information to caregivers. An example system is discussed in section 3.3 in terms of the tetherless care model. Section 3.4 defines a derivation from the model for the system's middleware functionality that encapsulates the network related considerations of the patient's context. Section 3.5 defines an algorithm for learning patient behavior and assessing the risk of the patient's state. Sections 3.6 and 3.7 renders the model into a software architecture as a hypothesis for the first question stated in section 1.3. A preliminary evaluation of this architecture, presented in section 3.8, demonstrates the feasibility of deploying this architecture to a contemporary mobile device. And section 3.9 concludes the chapter with several ideas that have driven the research of subsequent chapters.

3.1 HARDWARE ARCHITECTURE

3.1.1 Sensors

The ability to access and understand vital sign and environmental signals is accomplished with a set of sensing devices. These are typically small, battery-powered devices equipped with analog-to-digital sampling hardware, a microprocessor, and a low-power wireless radio. Some may implement communication protocols to organize into wireless sensor networks—termed *body area networks* (BANs) when solely worn by the patient—for the purpose of collaboratively collecting data samples at a specified gateway node. Typically, each device is specialized for one or a few vital signs or other signals, thus distributing the sensing work across the network of devices.

Examples of these BANs include existing commercial devices that form a single hop, wireless link with a user's smartphone or PC [5, 7, 6]. Also, several projects have developed intelligent living spaces equipped with hundreds of sensors and supporting infrastructure [39, 3, 2, 43]. These devote greater power and processing resources to the sensing devices but restrict mobility. Nevertheless, they each employ a similar low-level architecture to collect data samples at a gateway node.

3.1.2 Coordinator

The gateway node mentioned above is designated the *coordinator* and is generally considered the patient's smartphone but could also be his or her home PC or any device capable of effectively performing an assessment of the patient's state. It has greater battery capacity and processing resources than sensors. It also may have a range of communication technologies available, such as Ethernet, EVDO, UMTS, HSPA, WiFi, ZigBee, Ant, or Bluetooth.

The fact that this device can be comfortably carried on the person for long periods of time such that it accompanies the patient into disconnected environments uniquely positions it to support the pervasive and ubiquitous requirement of tetherless care. Its greater processing, storage, and communication capacities reduce the required capabilities, and thus the monetary cost, of individual sensors. It, thus, acts as an intelligent gateway between sensors and the outside world. It centrally collects sampled data, executes the assessment of the patient's state, mitigates periods of

disconnection, minimizes data transmitted to caregivers, and adjusts operation or communication strategies to suit the context. Given this central role, this device is the primary focus of this work and the intended target for the much of the tetherless care system architecture.

3.1.3 Sensor Network Protocols

The tetherless care architecture presented below is agnostic to the protocols and technologies that support the wireless network between sensors and the coordinator. A large body of work has already demonstrated the feasibly of constructing such low-power networks. Furthermore, protocols exist for profiling or aggregating sensor samples when transferring them from individual sensors to the coordinator. For example, the data of an ECG sensor exhibits a periodic, consistent pattern of quick peaks and valleys in an otherwise flat signal, which corresponds to the patient's heartbeat and is known as the QRS complex. In communicating the ECG data samples from an ECG sensor to the coordinator, the sensor need only communicate the start time of the QRS complex instead of the series of samples that compose the entire waveform. Other types of sensors, such as a body temperature sensor, may need only transmit a value to the coordinator when the value changes.

The architecture below encapsulates these protocols on the coordinator within an abstraction called a Virtual Sensor, which is responsible for acquiring sensor data and extracting information that is important to the application. In order to incorporate sensors into the tetherless care system, then, a developer need only provide an implementation of that abstraction.

3.1.4 Cloud Services

The use of the term *cloud service* or *server* is used interchangeably and refers to a central server that performs several functions. It stores or accesses the Personal Health Record (PHR) for multiple patients to log information reported by their respective coordinators. In conjunction with the coordinator, it executes algorithms to manage the mobility of the patient, assist in extracting knowledge of the patient's state from the context, learning patient behavior, and optimizing coordinator performance. Finally, it assists with or manages the communication with caregiver devices.

The terms are also used to refer to web services by which patients and caregivers may access information stored on the server.

Server hardware can range from a single, self-contained machine to a distributed set of machines that manage a locality of patients and engage a set of protocols to coordinate their efforts. While not specifically required, storage is generally considered to be a structured data store ranging from a single machine implementation, such as MySQL, to a distributed structured data store, such as BigTable [17] or Cassandra.

3.1.5 Caregiver Infrastructure

Throughout this thesis, we use the terms *caregivers*, *clinicians*, and *healthcare professionals* interchangeably. These refer to primary care physicians, nurses, emergency responders, doctors, surgeons, technicians, and administrators who share responsibility in administering care to a given patient. Each can securely interact with a tetherless care system through a number of interfaces, such as stationary workstations, tablets, or smartphones, to receive alerts or notifications, to review collected patient data, and to provide feedback or instruction to the patient.

As stated in section 1.2.3, it is not yet known how the healthcare system will evolve to incorporate tetherless care, making it difficult to specifically determine the hardware devices that will support their interaction with tetherless care and the requirements of those devices. Nevertheless, at a minimum, these will likely be devices capable of authenticating the caregiver to the system, managing data privacy, and delivering alerts about significant changes in patient state to a given caregiver. Other functionality, such as the caregiver's response to such an alert, can potentially occur through another medium.

3.2 MODEL FOR TETHERLESS CARE

This section presents a formal logical model of tetherless care. Thus far, tetherless care systems have been characterized as consisting of three basic functions: (1) accessing the vital sign and environmental signals of and around the patient, (2) extracting the patient's state from the context, and (3) taking appropriate actions to respond to the state of the patient, e.g. communicating information to caregivers. The first of these functions is taken as an assumption here as a result of

prior work in sensing hardware and sensor networking. What follows is a logical model for the context-aware assessment of patient state and the responses to it.

The model consists of a series of transformations, starting with a collection of raw data samples arriving at the coordinator from the sensor network and ending with a set of actions that the system must perform to properly respond to the patient's state. The first transformation involves some computation that produces a set of features from the stream of raw data samples in the context. e.g. a QRS complex is a feature of a stream of ECG data samples. The second transformation relates this set of features to specific attributes of the patient's state, where each attribute is referred to as a *state*, e.g. a state could be defined to encapsulate the patient's pulse in beats-per-minute and could be computed from the QRS complex features. In this way, the patient's overall state is described by a set of *state* objects. The third transformation triggers one or more actions in response to the set of *state* objects describing the patient's state. For example, a highly elevated pulse might be part of a set of states describing a panic attack, which triggers an alert for the family members and caregivers of the patient.

This series of transformations is rendered by a set of *virtual sensor* and *state descriptor* objects that collectively form a tetherless care application and that communicate by exchanging samples, features, states, and actions. A set of formal relations assign a *type* to each sample, feature, and state. Another set describes the declared interest, i.e. subscription, of each virtual sensor or state descriptor in one or more types. And a third set of relations describes the publish relationship between each virtual sensor or state descriptor and one or more types so as to formally define each end of the data paths between them. Using these relations, a tetherless care application is then formally modeled as a graph where each virtual sensor or state descriptor is a node and the data paths between them define the edges.

Below presents and discusses a set of formal sentences that express the tetherless care concept, TC. The discussion and interpretation of these sentences will draw from the following formal structure:

$$(\mathscr{P}, \mathscr{V}, \mathscr{C}, \mathscr{F}, \mathscr{I}, \mathscr{A}, \mathscr{T}, \mathscr{L}, \mathbb{N}), \text{ where}$$

- $\mathscr{P}$ is the domain of all patients under consideration;
- $\mathscr{V}$ is the domain of all digital samples of vital sign and environmental signals;
- $\mathscr{C}$ is the domain of all sets of sets of digital samples, i.e. the power set of the power set of $\mathscr{V}$;

- $\mathscr{F}$ is the domain of all features;
- $\mathscr{I}$ is the domain of all patient states;
- $\mathscr{A}$ is the domain of all actions;
- $\mathscr{T} \subset \mathscr{V}$ is the time domain;
- $\mathscr{L} \subset \mathscr{V}$ is the domain of all locations; and
- $\mathbb{N}$, the natural numbers.

TC can then be expressed as the sentence:

$$\forall_P(\exists_{R_P}\exists_{Q_P}\exists_{M_P}[(R_P(C,F) \rightarrow Q_P(F,I)) \wedge (Q_P(F,I) \rightarrow M_P(I,A))]) \tag{3.1}$$

This presumes the existence of three relations, R_P, Q_P, and M_P, where:

- $R_P : \mathscr{C} \rightarrow \mathscr{F}^i$ is a relation from the domain class, $\mathscr{C}$, to a subset of the domain class, $\mathscr{F}$, of size $i \in \mathbb{N}$;
- $Q_P : \mathscr{F}^j \rightarrow \mathscr{I}^k$ is a relation from a subset of the domain class, $\mathscr{F}$, of size $j \in \mathbb{N}$ to a subset of the domain class, $\mathscr{I}$, of size $k \in \mathbb{N}$;
- $M_P : \mathscr{I}^m \rightarrow \mathscr{A}^n$ is a relation from a subset of the domain class, $\mathscr{I}$, of some size $m \in \mathbb{N}$ to a subset of the domain class, $\mathscr{A}$, of some size $n \in \mathbb{N}$;

Specifically, the concept of tetherless care is modeled by an assignment from the described classes to the variables C, F, I, and A such that, for each patient under consideration, P, there exists relations, R_P, Q_P, and M_P, where R_P maps a set of sets of digital samples, C, to a set of features, F; Q_P maps F to a set of patient states, I; and M_P maps I to a set of actions, A. For simplicity, R_P, Q_P, and M_P will be referred to as R, Q, and M in the remainder of this document.

The notion of *satisfaction*, Ψ, is modeled in the following sentence:

$$\forall_{q \in Q}\forall_{I \in q}\exists_{A \subseteq \mathscr{A}}\exists_{A' \subseteq \mathscr{A}}[\{I,A\} \in \Psi \wedge \{I,A'\} \in M \wedge A \cap A' \neq \oslash], \text{ where} \tag{3.2}$$

- $\Psi : \mathscr{I}^m \rightarrow \mathscr{A}^n$ is a relation from a subset of the domain class, $\mathscr{I}$, to a subset of domain class, $\mathscr{A}$.

In colloquial terms, it states that, for every set of states that is a target of the relation, Q, there exists some set of actions that, if executed, could satisfy the patient's state represented by I; that, by performing one or more of these actions, the system has satisfactorily achieved the mission of tetherless care. Thus, the relation, M, must map from I to a set of actions that is strictly not disjoint from these actions.

Where referenced elsewhere in this document, the *context* of patient, P, is formally defined as the set of sets of digital samples, C, which is generally considered to have an internal structure that, in the least, conforms to the property described by the sentence:

$$\forall_{V \in C} \exists_{t \in V} \exists_{\ell \in V} [t \in \mathscr{T} \wedge \ell \in \mathscr{L}] \tag{3.3}$$

In other words, each set in the context is generally considered to contain a value from the time and location domains. This value of $t \in V$ along with all sample values, $v \in V$, constitute the following relation by fact of their co-membership in V:

$$\exists_{t_v} \exists_{t \in V} (\forall_{v \in V} [\{v, t\} \in t_v]) \tag{3.4}$$

This definition assigns the value of t as the timestamp property for each sample, $v \in V$.

Similarly, this value of t serves as the timestamp property for features and states derived from the samples in V, which is formally defined by the following two relations and assigned in an application-dependent manner:

$$\exists_{t_f} \forall_{f \in F} \exists_{V \in C} \exists_{t \in V} [\{f, t\} \in t_f] \tag{3.5}$$

$$\exists_{t_i} \forall_{i \in I} \exists_{V \in C} \exists_{t \in V} [\{i, t\} \in t_i] \tag{3.6}$$

- $t_f : \mathscr{F} \to \mathscr{T}$ assigns a timestamp property to every feature, and
- $t_i : \mathscr{I} \to \mathscr{T}$ assigns a timestamp property to every state.

The *situation* of patient, P, where referenced elsewhere in this document, is formally defined as the assignment, s, from values of the model structure to the variables C, F, I, and A such that the tetherless care theory holds. In other words, it refers to the context, set of features, set of states, and set of actions that describe the operation of a tetherless care system for a given patient.

3.2.1 Tetherless Care Application

A tetherless care application is described by the formal sentences below. The discussion of each sentence will draw upon the existing structure described above and add the following domains:

$$(\mathscr{K}, \mathscr{N}, \mathscr{S}, \mathscr{D}, \mathscr{E})$$

where:

- $\mathscr{K}$ is the domain of all types,
- $\mathscr{N}$ is the domain of all nodes,
- $\mathscr{S} \subset \mathscr{N}$ is the domain of all Virtual Sensor objects,
- $\mathscr{D} \subset \mathscr{N}$ is the domain of all State Descriptor objects, and
- $\mathscr{E}$ is a subset of $\mathscr{N} \times \mathscr{N}$.

First, a set of relations define a property, k, of each sample, v, feature, f, and state, i, referred to as its *type* and is as follows:

$$\exists_{h_v} \forall_{v \in \mathscr{V}} \exists_{k \in \mathscr{K}} [\{v,k\} \in h_v] \tag{3.7}$$

$$\exists_{h_f} \forall_{f \in \mathscr{F}} \exists_{k \in \mathscr{K}} [\{f,k\} \in h_f] \tag{3.8}$$

$$\exists_{h_i} \forall_{i \in \mathscr{I}} \exists_{k \in \mathscr{K}} [\{i,k\} \in h_i] \tag{3.9}$$

where:

- h_v is a relation on $\mathscr{V} \times \mathscr{K}$,
- h_f is a relation on $\mathscr{F} \times \mathscr{K}$, and
- h_i is a relation on $\mathscr{I} \times \mathscr{K}$.

Similarly, the following two relations define a set of interests for each virtual sensor, s, and each state descriptor, d, by relating each to a set of types from the domain, $\mathscr{K}$:

$$\exists_{g_s} \forall_{s \in \mathscr{S}} \exists_{K \subset \mathscr{K}} [\{s,K\} \in g_s] \tag{3.10}$$

$$\exists_{g_d} \forall_{d \in \mathscr{D}} \exists_{K \subset \mathscr{K}} [\{d,K\} \in g_d] \tag{3.11}$$

where:

- g_s is a relation from $\mathscr{S}$ to the power set of $\mathscr{K}$, and

- g_d is a relation from $\mathscr{D}$ to the power set of $\mathscr{K}$.

Also, a set of relations define a producer relationship between virtual sensors and features, state descriptors and states, and state descriptors and actions, as follows:

$$\exists_{p_s} \forall_{r \in R} \forall_{F \in r} \forall_{f \in F} \exists_s [\{s, f\} \in p_s] \tag{3.12}$$

$$\exists_{p_d} \forall_{q \in Q} \forall_{I \in q} \forall_{i \in I} \exists_d [\{d, i\} \in p_d] \tag{3.13}$$

$$\exists_{p_d} \forall_{m \in M} \forall_{A \in m} \forall_{a \in A} \exists_d [\{d, a\} \in p_d] \tag{3.14}$$

where

- p_s is a relation from $\mathscr{S}$ to the power set of $\mathscr{F}$, and
- p_d is a relation from $\mathscr{D}$ to (the power set of $\mathscr{I}$)$\cup$(the power set of $\mathscr{A}$).

A restriction is placed on the relation, p_d, in that no two state descriptors produce the same state, described by the formal sentence:

$$\forall_{d \in \mathscr{D}} \forall_{d' \in \mathscr{D}} \exists_i [\{d, i\} \in p_d \wedge \{d', i\} \in p_d \rightarrow d = d'] \tag{3.15}$$

or, in other words, for any two state descriptors, if both produce the same state, then they are the same state descriptor.

Finally, the above definitions allow for a formal description of a tetherless care application, modeled as a graph, $G = \{N, E\}$, where virtual sensors and state descriptors comprised the set of nodes, N, and the edges, E, are defined as follows:

$$\forall_{r \in R} \forall_{F \in r} \forall_{f \in F} \exists_s \exists_d \exists_k \exists_{K_d} [\{s, f\} \in p_s \wedge \{f, k\} \in h_f \wedge k \in K_d \wedge \{d, K_d\} \in g_d \rightarrow \langle s, d \rangle \in E] \tag{3.16}$$

$$\forall_{q \in Q} \forall_{I \in q} \forall_{i \in I} \exists_{d'} \exists_d \exists_k \exists_{K_d} [\{d', i\} \in p_d \wedge \{i, k\} \in h_f \wedge k \in K_d \wedge \{d, K_d\} \in g_d \rightarrow \langle d', d \rangle \in E] \tag{3.17}$$

In colloquial terms, 3.16 states that, for every feature, f, and its type, k, found in the relation, R, if some virtual sensor, s, produces the feature and some state descriptor includes k among its interest set, then there is an edge from s to d in G. And 3.17 states that for every state, i, and its type, k, found in the relation, Q, if some state descriptor, d', produces the state and some other state descriptor, d, includes k among its interest set, then $\langle d', d \rangle$ is a member of the set, E, as well.

To summarize, a tetherless care application is defined as the graph, $G = \{N, E\}$, where the nodes of the graph are virtual sensors and state descriptors. Each virtual sensor has an interest

in one or more types of samples and produces one or more features. Each state descriptor has an interest in one or more features or states and produces one or more states or actions. The edges of the graph are defined by the publish-subscribe relationships among the nodes, i.e. a virtual sensor produces a feature in which a state descriptor is interested. In this way, the virtual sensors, collectively, are responsible for computing the relation, R, using samples as input and producing features as output. The exact mechanism by which this is done is left to a specific implementation.

Similarly, the state descriptors, collectively, are responsible for computing both the relations, Q and M, using features as input and producing states and actions as output. Again, the mechanism by which this is done is subject to a given application.

3.3 EXAMPLE SYSTEM

Consider an elderly patient suspected of exhibiting the early stages of dementia. Her doctor and family, concerned with her ability to seek help on her own, outfit her with a system of sensors, a coordinator device, and a tetherless care application that regularly updates a central server with the state of the patient and promises to alert them to risky behavior or environments. Specifically, in addition to monitoring the general health of the patient, the family is concerned about the patient falling over and not recovering herself or wandering well beyond known, trusted locations and becoming lost.

The set of sensors attached to the patient include a body temperature sensor, a pulse sensor, a GPS positioning sensor, several triaxial accelerometers, and several triaxial gyroscopes. Additionally, to effectively communicate with family, healthcare professionals, and the server, the system must monitor the available networking technologies, which can be treated as an environmental signal itself that also assists in physically locating the patient.

For the purpose of this example, the discussion below focuses on the detection of and response to the patient suffering a fall. Prior to a fall, with the patient at rest, accelerometers register values equivalent to 9.8 m/s in the direction of gravity. When a fall starts and the patient begins to descend, the magnitude of this value drops significantly. Similarly, the gyroscope likely measures the angular momentum curving down towards the floor around some axis that might be the patient's

foot. And, upon contact with the floor, the impact registers a brief, large spike in the accelerometers value.

The moments after the fall determine the systems response. If the patient starts getting back up and continuing with everyday activities, the event only needs to be recorded in the medical history of the patient. However, if the fall results in an injury, the patient may remain on the ground in a prone position, and a measurable change in her pulse and body temperature may present itself. As such, over the course of the next several minutes after the fall, the system must either escalate its response to garner attention from emergency services, healthcare professionals, and family members or return to a more neutral posture.

Each type of sensor produces samples from a subset of $\mathscr{V}$. A sample is an instantaneous representation of an environmental phenomenon encoded as a digital number within a range of possible values. This example considers the following samples:

- A character string corresponding to the type of a sensor;
- An n-bit unsigned integer that correlates to the magnitude of the acceleration of the sensor along an arbitrarily defined x-, y-, or z-direction;
- An n-bit unsigned integer that correlates to the magnitude of the angular velocity of the sensor around an arbitrarily defined x-, y-, or z-axis;
- An n-bit unsigned integer that correlates to the temperature around the body sensor temperature;
- An n-bit unsigned integer that correlates to the beats per minute of the patient's heart;
- A 6-byte MAC address of a WiFi access point;
- An n-bit unsigned integer that corresponds to the received signal strength from a remote wireless entity; and
- An n-bit unsigned integer whose value indicates the availability of a given network technology.

Note that, for discussion purposes, specific sample values are described in the most convenient, human-readable units.

Features represent knowledge about the patient's environment that can be derived from the set of available samples. A simple example is a transformation from the individual x-, y-, and z-direction accelerometer samples to a single acceleration-vector feature with equivalent respec-

tive component values. For the discussion that follows, the set of potential features includes the following:

- A vector of accelerometer samples along three axes;
- A vector of gyroscope samples around three axes;
- The accelerometer exhibiting free fall;
- The gyroscope exhibiting free fall;
- An impact with a hard surface, e.g. the patient impacts the ground;
- Pulse transitions above a threshold;
- Pulse transitions below a threshold;
- Body temperature transitions above a threshold;
- Body temperature transitions below a threshold;
- A list of MAC addresses of nearby access points with their signal strengths;
- A WiFi connection became available;
- An LTE connection became available;
- A 3G connection became available;
- A WiFi connection was lost;
- An LTE connection was lost; and
- A 3G connection was lost.

States describe aspects of the patient's condition, e.g. the patient is in a seated position. Also, where a feature indicates a condition derived from one or more signals, a state represents the attribution of or derivation from this condition to the patient. For example, an accelerometer may exhibit a free-fall feature, but a patient-in-free-fall state may only result from multiple free-fall features derived from separate sensors. The set of potential states under consideration in this example include the following:

- States describing heart rate descriptions, e.g. elevated, normal, too-low, etc.;
- States describing body temperatures, e.g. feverish, high, normal, low, hypothermia, etc.;
- States describing body orientation, e.g. seated, prone, standing, walking, running, fetal position, in free fall, fallen, fallen-heightened, fallen-severe;
- The state of the WiFi network, e.g. connected to MAC address X;

- The state of the WWAN network, e.g. disconnected from LTE;

The set of potential actions includes the following:

- A prompt to the user asking if she is ok;
- An alert indicating the patient has fallen and remains in an abnormal state, e.g. remains in a prone position with an elevated heart rate;
- An instruction to increase the sampling frequency of the accelerometer;
- An instruction to increase the sampling frequency of the gyroscope;
- An instruction to adjust the threshold value for an elevated pulse.

3.3.1 The Relation R

Without loss of generality, consider gravity applied in the negative z-direction while standing and the negative x-direction while prone. The corresponding samples include:

- $x_{rest} = 0$,
- $x_{free} = 0$,
- $x_{impact} = 200m/s^2$,
- $x_{prone} = -9.8m/s^2$,
- $y_{rest} = 0$
- $y_{free} = 0$
- $y_{impact} = 200m/s^2$,
- $y_{prone} = 0$,
- $z_{rest} = -9.8m/s2$,
- $z_{free} = -5.0m/s2$,
- $z_{impact} = 200m/s2$,
- $z_{prone} = 0$,
- $t_1, t_2, t_3, t_4, t_5, t_6$,
- $ui_1 =$ *patient-ui-prompt-displayed*,
- $ui_2 =$ *patient-ui-responds-ok*,
- $ui_3 =$ *patient-ui-responds-injured*,
- $ui_4 =$ *patient-ui-no-response*,

- $n_1 =$ *wifi-disconnected*,
- $n_2 =$ *wifi-connected*, and
- $n_3 =$ *AP-nearby*

The domain of sample sets, V, then includes:

- $V_1 = \{t_1, x_{rest}, y_{rest}, z_{rest}, \ldots, n_1, n_3\}$,
- $V_2 = \{t_2, x_{free}, y_{free}, z_{free}, \ldots, n_1, n_3\}$,
- $V_3 = \{t_3, x_{impact}, y_{impact}, z_{impact}, \ldots, n_1, n_3\}$,
- $V_4 = \{t_4, x_{prone}, y_{prone}, z_{prone}, \ldots, n_1, n_3\}$,
- $V_5 = \{t_5, ui_1, \ldots, n_2, n_3\}$, and
- $V_6 = \{t_6, ui_3, \ldots, n_2, n_3\}$.

The context domain, $\mathscr{C}$, is defined as all sets of sets of samples, which includes:

- $C1 = \{V_1, \ldots\}$.
- $C2 = \{V_2, \ldots, V_1, \ldots\}$,
- $C3 = \{V_3, \ldots, V_2, \ldots, V_1, \ldots\}$,
- $C4 = \{V_4, \ldots, V_3, \ldots, V_2, \ldots V_1, \ldots\}$,
- $C5 = \{V_5, \ldots, V_4, \ldots, V_3, \ldots, V_2, \ldots V_1, \ldots\}$, and
- $C6 = \{V_6, \ldots, V_5, \ldots, V_4, \ldots, V_3, \ldots, V_2, \ldots V_1, \ldots\}$

and the domain of all features includes:

- $f_1 =$ *accelerometer-at-rest-standing*,
- $f_2 =$ *accelerometer-free-fall*,
- $f_3 =$ *accelerometer-impact*,
- $f_4 =$ *accelerometer-at-rest-prone*,
- $f_5 =$ *accelerometer-impact-X-seconds-ago*,
- $f_6 =$ *ui-patient-prompted*,
- $f_7 =$ *ui-patient-response-injured*,
- $f_8 =$ *ui-patient-response-ok*,
- $f_9 =$ *wifi-session-start-MAC*, and
- $f_{10} =$ *wifi-session-end*.

The relation, R, then includes the following:

- $\{C_1, \{f_1, \ldots\}\}$,
- $\{C_2, \{f_2, \ldots\}\}$,
- $\{C_3, \{f_3, \ldots\}\}$,
- $\{C_4, \{f_4, \ldots\}\}$,
- $\{C_5, \{f_5, f_6, f_9, \ldots\}\}$, and
- $\{C_6, \{f_7, f_9, \ldots\}\}$.

In words, R, in this example, associates the individual directional accelerometer samples that correspond to resting and standing positions to a feature that indicates as much, i.e. x_{rest}, y_{rest}, and z_{rest} are related to f_1. Subsequent contexts can be thought of as occurring in subsequent moments in time. And, as this context includes a set of samples indicating free-fall, R associates this with the *accelerometer-free-fall* sample. Similarly, the user interface can be modeled as another aspect to the physical environment that produces signals to which the system must respond. Thus, once the interface changes to produce a prompt for the user, an aspect of the physical environment has changed, which is modeled here as the sample *patient-ui-prompt-displayed*. The relation, R, associates this sample to an equivalent feature. And, as this sample moves into the past along with an accompanying timestamp sample, t_x, it models a timer functionality by which the system can determine that the patient may be incapacitated and unable to respond to the prompt.

3.3.2 The Relation Q

Continuing the example of a patient suffering a fall, (1) if the following features represent those associated with the samples from some accelerometer, a:

- $f_{a1} =$ *accelerometer-a-at-rest-standing*,
- $f_{a2} =$ *accelerometer-a-free-fall*,
- $f_{a3} =$ *accelerometer-a-impact*,
- $f_{a4} =$ *accelerometer-a-at-rest-prone*, and
- $f_{a5} =$ *accelerometer-impact-X-seconds-ago*;

(2) if the following features represent those associated with the samples from some accelerometer, b:

- f_{b1} = *accelerometer-b-at-rest-standing*,
- f_{b2} = *accelerometer-b-free-fall*,
- f_{b3} = *accelerometer-b-impact*,
- f_{b4} = *accelerometer-b-at-rest-prone*, and
- f_{b5} = *accelerometer-impact-X-seconds-ago*;

(3) if the following features represent those associated with the networking hardware:

- f_{n1} = *wifi-session-start-MAC* and
- f_{n2} = *wifi-session-end*;

and (4) if the domain of all states, $\mathscr{I}$, includes:

- i_1 = *patient-standing*,
- i_2 = *patient-in-free-fall*,
- i_3 = *patient-impacted*,
- i_4 = *patient-fallen*,
- i_5 = *possible-patient-injury*,
- i_6 = *patient-injured-by-fall*,
- i_{n1} = *wifi-disconnected*, and
- i_{n2} = *wifi-connected-MAC*;

then the relation, Q, includes the following:

- $\{\{f_{a1}, f_{b1}, \ldots\}, \{i_1, \ldots, i_{n1}, \ldots\}\}$,
- $\{\{f_{a2}, f_{b2}, \ldots\}, \{i_2, \ldots, i_{n1}, \ldots\}\}$,
- $\{\{f_{a3}, f_{b3}, \ldots\}, \{i_3, \ldots, i_{n1}, \ldots\}\}$,
- $\{\{f_{a4}, f_{b4}, \ldots\}, \{i_4, \ldots, i_{n1}, \ldots\}\}$,
- $\{\{f_{a5}, f_{a4}, f_{b5}, f_{b4}, \ldots, f_{n1}, \ldots\}, \{i_5, i_4, \ldots, i_{n2}, \ldots\}\}$,
- $\{\{f_{u1}, f_{a5}, f_{a4}, f_{b5}, f_{b4}, \ldots, f_{n1}, \ldots\}, \{i_5, i_4, \ldots, i_{n2}, \ldots\}\}$, and
- $\{\{f_{u2}, f_{a5}, f_{a4}, f_{b5}, f_{b4}, \ldots, f_{n1}, \ldots\}, \{i_6, i_4, \ldots, i_{n2}, \ldots\}\}$.

In words, the relation, Q, attributes the accelerometer features indicating a standing position, f_{a1} and f_{b1}, to the patient being in a standing position, i_1. Similarly, the free fall, impact, and prone body position features are attributed to the patient's condition accordingly. Once a fall is detected,

the set of features indicating a fall (in the recent past) and the patient remaining prone on the floor (in the present) are associated with the state of a possible patient injury. As the system interacts with the patient after the fall, the response, or lack thereof, supplied by the patient is encoded in additional features. As such, the response (in this example), which declares some injury, in combination with the fallen and prone position features of the accelerometers are associated with the *patient-injured-by-fall* state.

3.3.3 The Relation M

Consider the following states in the domain of all states, $\mathscr{I}$:

- $i_1 =$ *patient-standing*,
- $i_2 =$ *patient-in-free-fall*,
- $i_3 =$ *patient-impacted*,
- $i_4 =$ *patient-fallen*,
- $i_5 =$ *possible-patient-injury*,
- $i_6 =$ *patient-injured-by-fall*, and
- $i_{n2} =$ *wifi-connected-MAC*,

and the following actions:

- $a_1 =$ *increase-accelerometer-sampling-frequency*,
- $a_2 =$ *increase-gyroscope-sampling-frequency*,
- $a_3 =$ *initiate-wifi-connection*,
- $a_4 =$ *prompt-is-patient-ok*, and
- $a_5 =$ *alert-others-injury-due-to-fall*,

then the relation, M, includes:

1. $\{\{i_1,\ldots\},\{\ldots\}\}$,
2. $\{\{i_2,\ldots\},\{a_1,a_2,\ldots\}\}$,
3. $\{\{i_3,\ldots\},\{\ldots\}\}$,
4. $\{\{i_4,\ldots\},\{a_3,\ldots\}\}$,
5. $\{\{i_4,i_5,\ldots,i_{n2},\ldots\},\{a_4,\ldots\}\}$, and

6. $\{\{i_4, i_6, \ldots, i_{n2}, \ldots\}, \{a_5, \ldots\}\}$.

In words, the pairs listed above and included in the relation, M, encode the following logic, respectively: (1) when the patient is standing at rest, no action needs to be taken in this example; (2) When the patient is in free fall, the accelerometer and gyroscope sensors need to increase their sampling rate in order to be able to detect a possible impact with the ground; (3) Upon an impact, no action needs to be taken until the patient and her physiology respond to the impact; (4) In anticipation of needing to communicate with the outside world, once a fall has been concluded, the system should initiate a WiFi connection (or other lower power communication technology); (5) Since it has been determined that the patient has a potential injury, the system should display a prompt on the devices user interface asking the patient if she is ok; (6) Since it has been determined that the patient is most likely injured (resulting from a confirmation received from the patient through the UI), the system must generate an alert for emergency services, other healthcare professionals, and the patient's family to bring their immediate attention to the matter.

This example belies the cases where a network connection is not available. Environments where this is the case result in a context that contains a different set of samples, which should ultimately map to a different action other than alerting others using the WiFi connection. Replacement actions could utilize a 3G, 4G, or LTE connection; could dial 911 on behalf of the patient if it's available; could alert the patient that the system needs assistance finding a connection; could attempt communication with nearby devices to relay information; or, as a last resort, start making noise to get the attention and assistance of a passerby. A complete tetherless care application must be able to predict and differentiate the contexts that necessitate these disparate actions.

3.3.4 Fall Application

This example can be modeled with the following set of virtual sensors:

- accelerometer-sensor-a,
- accelerometer-sensor-b,
- ui-sensor, and
- network-sensor

and state descriptors:

- FallDetectionState and
- NetworkState.

Each directional accelerometer sample corresponding to *at-rest*, *free-fall*, *impact*, and *prone* for the first accelerometer, along with the timestamp samples, all map to the type, *accel-sample-a*. Each directional accelerometer sample corresponding to *at-rest*, *free-fall*, *impact*, and *prone* for the second accelerometer, along with the timestamp samples, map to the type, *accel-sample-b*. The user interface samples and the timestamp samples map to the type, *ui-sample*. And the WiFi related samples along with the timestamp samples map to the type, *network-sample*.

A similar mapping can be extrapolated to the features and states of the above example, which is indicated by the nomenclature used for each.

The interest sets of the virtual sensors and state descriptors are then:

- accelerometer-sensor-a = $\{$*accelerometer-sample-a*$\}$,
- accelerometer-sensor-b = $\{$*accelerometer-sample-b*$\}$,
- ui-sensor = $\{$*ui-sample*$\}$,
- network-sensor = $\{$*network-sample*$\}$,
- FallDetectionState = $\{$*accelerometer-feature*, *ui-feature*$\}$, and
- NetworkState = $\{$*network-feature*, *fall-dectection-state*$\}$

Also, the relations, p_s and p_d, can be defined as follows:

- $\{$accelerometer-sensor-a$\} \times \{f_{a1}, f_{a2}, f_{a3}, f_{a4}, f_{a5}\}$,
- $\{$accelerometer-sensor-b$\} \times \{f_{b1}, f_{b2}, f_{b3}, f_{b4}, f_{b5}\}$,
- $\{$ui-sensor$\} \times \{f_{u1}, f_{u2}, f_{u3}\}$,
- $\{$network-sensor$\} \times \{f_{n1}, f_{n2}\}$,
- $\{$FallDetectionState$\} \times \{i_1, i_2, i_3, i_4, i_5, i_6, a_1, a_2, a_4, a_5\}$, and
- $\{$NetworkState$\} \times \{i_{n1}, i_{n2}, a_3\}$

From the sentences, 3.16 and 3.17, in the application model, the edges in the graph, G, are then:

- $\langle$accelerometer-sensor-a, FallDectectionState$\rangle$,
- $\langle$accelerometer-sensor-b, FallDectectionState$\rangle$,

- $\langle$ui-sensor, FallDectectionState$\rangle$,
- $\langle$network-sensor, NetworkState$\rangle$, and
- $\langle$FallDectectionState, NetworkState$\rangle$

Within this framework, each of the accelerometer virtual sensors encapsulates an algorithm that maps individual accelerometer values to features describing a patient suffering a fall. A simple algorithm to do so might apply a threshold to the magnitude of the gravitational vector calculated from the accelerometer samples, which is $9.8 m/s^2$ when at rest, drops below some threshold in free fall, and rises above some threshold upon impact. The vector also changes direction significantly with the patient lying prone.

The FallDetectionState state descriptor encapsulates an algorithm that maps these features, which occur successively in time, to their corresponding states, which is a trivial one-to-one correspondence in this example. It must also map these states to the appropriate actions to respond to the states. An algorithm to do so could be a simple finite automaton that triggers actions upon a state transition, e.g. the transition from i_3 (impact) to i_4 (fallen) triggers a_3 (prompt if ok).

3.4 DELAYED ACTIONS AND QUEUING INFRASTRUCTURE

From above, note the relation M only associates the *alert-others-injury-due-to-fall* action to the set of states because included in the set of states was the *wifi-connected* state. Another environment may be absent of any network connection, which would result in a context, a set of features, and a set of states all indicating as much. Yet, the need to alert a healthcare professional or family member may still result from the computation.

For this example application, the desirable behavior when a critical event occurs, such as a fall where the patient gets injured, would be for the system to first attempt communicating with third parties using the lowest power option, such as WiFi, then fall back to more power intensive options, such as 3G, 4G, or LTE, then fall back to dialing 911 if possible, and finally fall back to alerting the patient or nearby passersby to request assistance or to notify them of the need to get the system within some technology's communication range.

To model this, assume the following definitions:

- $i_4 =$ *patient-fallen* state,
- $i_6 =$ *patient-injured-by-fall* state,
- $i_{n2} =$ *wifi-connected-MAC*,
- $i_{n3} =$ *3G-connected*,
- $i_{n4} =$ *4G-connected*,
- $i_{n5} =$ *LTE-connected*,
- $a_5 =$ *alert-others-injury-due-to-fall-wifi*,
- $a_6 =$ *alert-others-injury-due-to-fall-3G-4G-LTE*,
- $a_7 =$ *alert-others-injury-due-to-fall-911*,
- $a_8 =$ *alert-others-injury-due-to-fall-ui-panic*,
- $t_e = t_i(i_6)$, the time at which the patient is determined to be injured,
- $t_e + 1$ is the first time instance where action $a5$ no longer satisfies I,
- $t_e + 2$ is the first time instance where action $a6$ no longer satisfies I,
- $t_e + 3$ is the time at which i_{n5} is produced by the NetworkState state descriptor (see above), i.e. $i_{n5} \in I_{t_e+3}$,
- $t_e + 4$ is the first time instance where action a_7 no longer satisfies I,
- $I_{t_e} = I_{t_e+1} = I_{t_e+2} = \{i_4, i_6, \dots\}$, and
- $\{I_{t_e}, \{a_5, a_6, a_7, a_8\}\} \in \Psi$

In other words, assume the system determines that the patient is injured at time, t_e, which is expressed in the set of states, I_{t_e}, and which can be satisfied by the actions a_5, a_6, a_7, or a_8.

At time, $t_e + 3$, an LTE connection becomes available, at which time a_6 must result from the computation of the relation M, i.e.:

$$\{I_{t_e+3}, \{\dots, a_6, \dots\}\} \in M$$

If, instead, no network connection became available at $t_e + 3$, then, at time $t_e + 4$, action a_8 would result from the relation M.

Up to this point, this discussion relies on the fact that all memory of the recent past is modeled in the context, C, in a time-ordered series of sets of samples. Those samples map to features, and those features similarly map to states. In this way, with access to C, the system is able to reapply

the computation to a set of sampl es, V, at a time , $t-k$, for some arbitrary k. Once k exceeds some delay value, it's then able to initiate an appropriate time-delayed action.

In a practical implementation, though, maintaining an indefinitely large collection of samples would be the least efficient, if not impractical, use of storage on a resource constrained device. Furthermore, nothing prevents virtual sensors and state descriptors from transforming and storing samples, features, or states in another form. The storage of samples also makes the development of tetherless care applications more cumbersome and less intuitive.

For this purpose, the architecture presented next and discussed in subsequent chapters outlines additional infrastructure for storing the results of tetherless care computation such that it need not be reapplied elsewhere in time. Specifically, the contract between a tetherless care application and an intelligent queuing system is formally defined. This contract allows for the delayed execution of tetherless care actions that necessitate network communication, codifies the notion of a fall back action, and ensures the patient's condition is satisfied by some action for all time. As such, an application can transform samples to features, features to states, and states to actions in real time and then leave the infrastructure to execute the network-related actions in the most suitable environment and subject to the satisfaction constraints of the actions.

3.4.1 Network Actions

Consider the following definitions:

- Let C_t be the context for some patient at time t,
- Let s_n be the network virtual sensor,
- Let d_n be the network state descriptor, and
- Let $\mathscr{I}_n$ be the set of connected network states, $\{i_{wifi}, i_{3G}, i_{4G}, \dots\}$ where $i_{wifi} = \textit{wifi-connected}$, $i_{3G} = \textit{3G-connected}$, etc.

From these definitions, the following variables can be computed:

- $F_t \leftarrow R(C_t)$ is the set of features related to the context, C_t,
- $I_t \leftarrow Q(F_t)$ is the set of states related to F_t,
- $A_t \leftarrow M(I_t)$ is the set of actions related to I_t,
- $A_t\Psi \leftarrow \Psi(I_t)$ is the set of all actions that satisfy I_t,

- $F_{nt} \leftarrow F_t \cap p_s(s_n)$ be the subset of features produced by s_n,
- $I_{nt} \leftarrow I_t \cap p_d(d_n)$ be the subset of states produced by d_n,
- $A'_t \leftarrow M(I_t - I_{nt})$ is the set of actions produced absent of any network connections, and
- $A_{nt} \leftarrow A_t - A'_t$ is the set of network dependent actions that could be produced if a network was available.

The intelligent queuing system seeks to relax the system's constraints on the relation, M, such that:

$$M(I_t) = M((I_t - I_{nt}) \cup \mathscr{I}_n) = A_e$$

In other words, the set of actions, A_e, associated by M with the current set of states, I_t, is equivalent to the set that would be associated as if I_t included all possible network connections.

The addition of the network queuing system introduces a promise of satisfying the patient condition, I_t, by executing at least one action, a, from the subset of these that are network actions, $A_e - A'_t \subseteq A_t\Psi$, at some time, $t+k$, such that $a \in A_{(t+k)\Psi}$ still holds.

For instance, in the above example, the patient is injured at time, t_e, but M does not associate the action, a_6, with this condition until time, $t_e + 3$, because no network connection is available before then. The queuing system infrastructure allows M to associate the set of actions, $\{a_5, a_6, a_7, a_8\}$, with I_{the} and promises to execute one action from the set at some point in time after t_e provided the action still satisfies the condition at that future time.

3.4.1.1 Messages To simplify reasoning about network actions, let a message, m, be logically equivalent to a set of actions, A, where each action, $a \in A$, is either a network action over some network technology, such as *alert-other-injury-due-to-fall-wifi* or *alert-others-injury-due-to-fall-3G*, or a fall back action, such as *alert-others-injury-due-to-fall-ui-panic*.

A tetherless care application creates messages in response to the patient state and passes them to the queuing infrastructure to indicate a desire to have one of it's logically equivalent actions executed to satisfy the patient's state. The convention adopted by the queuing system assumes the message has one or more properties that describes the constraints of satisfying the patient state, i.e. that the satisfaction relation is encoded in these properties. In chapter 4, the properties utilized for messages of the reference implementation are described in more detail.

3.5 RISK ANALYSIS

Analyzing the risk of the patient's current environment is ultimately an application-level task requiring knowledge of the patient's capabilities, the nature of his or her condition, the speed at which critical events onset, the speed and precision with which they can be detected, and the amount of time caregivers have to respond. Nevertheless, using the above model, a set of algorithms can provide an infrastructure for risk analysis based on the prediction of future states using an $O(1)$ Markov predictor[1]. The system projects likely future sets of states and allows a tetherless care application to assign an application-specific and opaque risk value to each possible future set of states.

For each state descriptor, d, and for times, $t-k$, $t-k+1$, ..., $t-1$, and t, let I_{t-i} denote the set of states produced by d at time $t-i$ for $i = 0 \ldots k$. The probability of d producing some set of states, I_{t+1}, at the next time, $t+1$, given it produced the set, I_t, at time, t, is defined as follows:

$$P(I_t, I_{t+1}) = P(X_{t+1} = I_{t+1} | X_t = I_t) = \frac{N(I_t I_{(t+1)}, \{I_{t-k}, \ldots, I_t\})}{N(I_t, \{I_{t-k}, \ldots, I_t\})} \tag{3.18}$$

where X_t is a set of states and $N(q,H)$ is the number of times the sequence, q, occurs in the recent history, H, produced by the state descriptor, d.

Algorithm 1 uses this probability calculation, a threshold probability, Φ, and a application-specific function for assigning risk to a set of states, Θ, to recursively construct a tree of possible future sets of states. The tree is built along the descent into the recursion, and the risk is assigned upon the return. Thus, the function, Θ, can assume and depend on the existence of child nodes in the tree, each of which has a risk value assigned. In this way, the risk of the current set of states, I, can depend on what is predicted to occur in the future and how risky it is predicted to be.

The middleware described in the architecture below, and discussed in detail in chapter 4, uses algorithm 1 to assess the risk of delaying transmission of a set of messages currently held in the queuing infrastructure, i.e. delaying the execution of the actions equivalent to each message. It operates on the sets of network connectivity states produced by a network state descriptor, which are more conveniently thought of, described as, and stored as *network sessions*.

[1] A more general predictor of any order is left for future work

Algorithm 1: Future State Prediction and Risk Analysis

Given a root tree node, r; the current set of states, I_t, produced by some state descriptor, d; a history of recent sets of states, $H = \{I_{t-1}, I_{t-2}, \ldots, I_{t-k}\}$; a threshold probability, Φ; and a function, Θ, for assigning a risk value to a potential set of states, this algorithm recursively constructs a tree of future scenarios where each node, n, represents a possible set of states, I, that could be produced by d after the set represented by the parent tree node. Each edge in the tree is weighted by the probability of following the path through the tree from the root to the node, n. And, along the return path, the function, Θ, is used to assign a risk value to each node in the tree. Thus, the final risk value assigned to the root of the tree is the risk of the current patient state.

Require: root tree node r, I_t, history H, N, min probability Φ, risk function Θ

function BUILD_TREE($n, I_t, H, N, p_s, \Phi, \Theta$)

$total \leftarrow N(I_t, H)$

for all $I \subseteq \mathscr{I}$ **do**

$count \leftarrow N(\langle I_t, I \rangle, H)$

$p_I \leftarrow count \div total \cdot p_s$

if $p_I > \Phi$ **then**

$c \leftarrow TreeNode()$

$c.edgeWeight \leftarrow p_I$

$c.I \leftarrow I$

$addChild(n, c)$

BUILD_TREE($(c, I, H, N, p_I, \Phi, \Theta)$)

end if

end for

$n.risk \leftarrow \Theta(n, I))$

end function

3.6 SOFTWARE ARCHITECTURE

The following software system architecture, illustrated in figure 2, defines major component abstractions that mimic the classes of objects in the above model so as to allow an application developer to more easily design, construct, and reason about the implementation of the relations, R, Q, and M, to satisfy tetherless care.

The coordinator is the platform of focus for this architecture given its central role in gathering sensor data and mitigating the disconnections due to the patient's mobility.

3.6.1 Virtual Sensors

Each sensor, either physically attached to the coordinator or wirelessly accessible from it, is associated with a *Virtual Sensor*. The Virtual Sensor is responsible for encapsulating the details of communicating with and acquiring samples from its associated sensor. It also contains sufficient logic to compute a *feature* of the sample data stream. For example, an ECG virtual sensor might be tasked with determining the set of samples that comprise the QRS complex of the patient's heartbeat.

Features, when identified, are made available in the system through a software-defined bus, termed the *backplane*, that is local and exclusive to the software running on the coordinator. Each feature is associated with a *type* and other application-defined meta data.

3.6.2 State Descriptors

A set of *State Descriptors* encapsulate the logic of (1) computing the patient's state from the features broadcast by virtual sensors and (2) mapping the patient's state to a set of actions, i.e. both the relations Q and M in the above model. Each state descriptor is responsible for computing one aspect of the patient's state. For example, a *pulse* state descriptor may listen for the QRS complex features from an ECG virtual sensor to compute a beats-per-minute value. When the computed state changes, the state descriptor may emit either an *action*, a *state* object, or both. State objects are similar to features and broadcast on the same medium with an associated *type* value.

Each state descriptor declares its interest in one or more *type*s, and the system ensures that all features or states of interest are delivered to the given state descriptor.

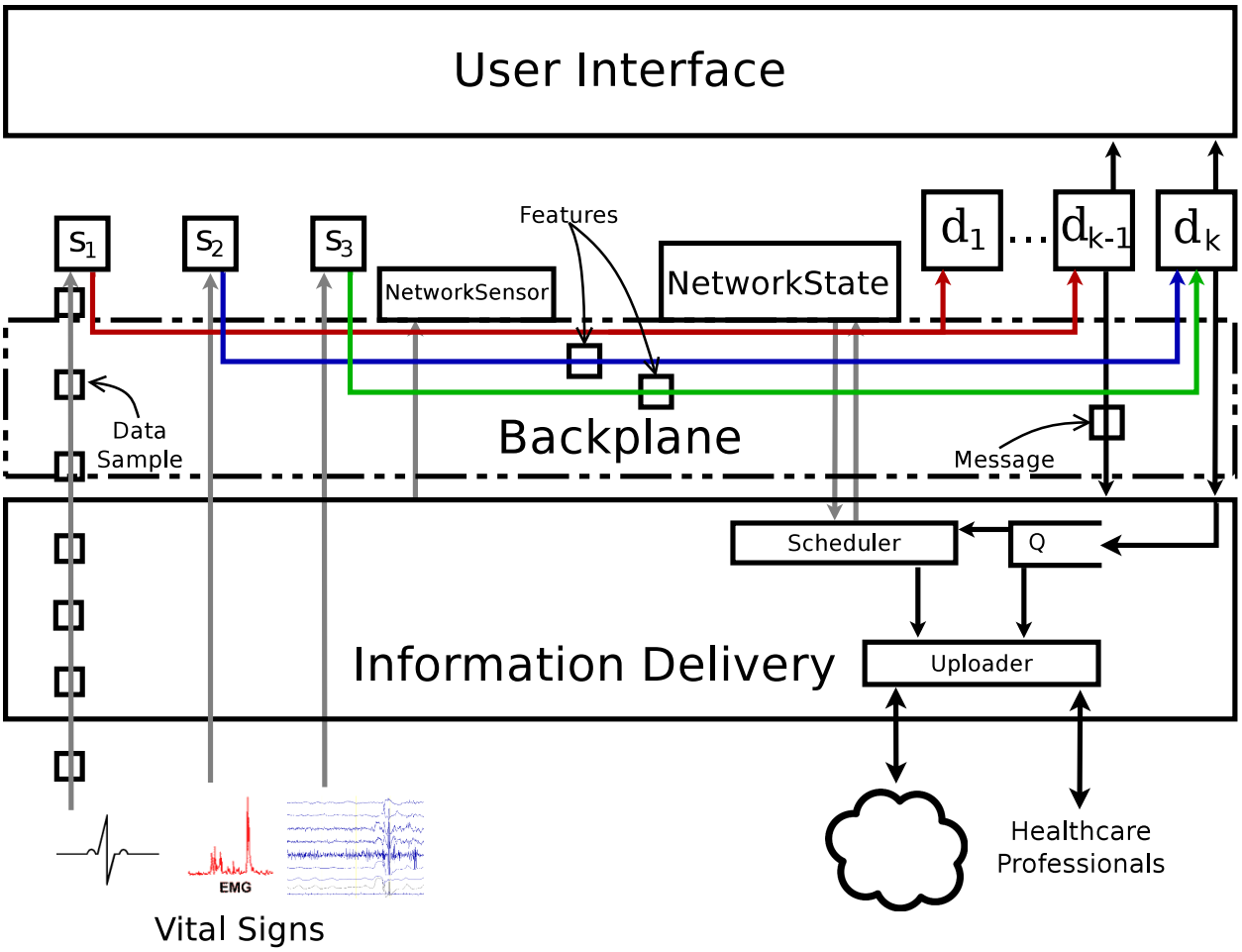


Figure 2: Tetherless Care Software Architecture for the Coordinator

Vital signs are sampled by the sensor network and transmitted to the coordinator where each is delivered to its respective virtual sensor. The virtual sensors compute and emit features of the stream of samples, which are then delivered to interested state descriptors. The state descriptors compute state objects, trigger actions, and send messages to the Information Delivery component for transmission to other entities in the system.

3.6.3 Actions

Actions are units of execution triggered by state descriptors as needed. However, unlike virtual sensors and state descriptors, they are not realized as high level abstractions in the reference implementation. This was done to allow the application developer greater ease and flexibility in devel-

oping state descriptors and because the underlying frameworks in contemporary mobile operating systems provide a comparable and widely understood abstraction. Instead, the system provides a mechanism by which a state descriptor can queue an action for execution either immediately or after a defined interval of delay.

3.6.4 Messages

Messages bundle a set of data to be transferred to the cloud or caregiver device and are the mechanism by which applications interact with the information delivery component to request network operations by a declared deadline. As discussed above, each message is equivalent to a set of actions that include those of (1) uploading the message's data subject to its constraints over each of the network technologies available and (2) fallback actions alerting the patient when no network connection is available that can meet the required constraints. Each message then gives the information delivery component the freedom to select an optimal transmission opportunity subject to its constraints.

3.6.5 Backplane

The backplane handles the exchange of features, states, actions, and network messages between virtual sensors, state descriptors, the information delivery component, and the user interfaces component. Most modern mobile operating systems provide some message exchange mechanism around which the tetherless care framework is able to provide a wrapper API.

3.6.6 Information Delivery

The *Information Delivery* component is an abstraction that manages the exchange of *messages* between state descriptors or virtual sensors and remote entities, such as cloud servers or caregiver devices. It houses an implementation of algorithm 1 to project likely future transmission opportunities and assess the risk of delaying the transmission of its message load. Where each message provides a degree of freedom to select among the network technologies available now or in the future for its transmission, the system assumes a certain amount of risk waiting for an opportunity that may not materialize. Nevertheless, energy savings can be realized by introducing a delay

before the transmission of a set of messages. The information delivery component balances the tradeoffs of the criticality of the patient's state as expressed in message constraints, the desire for low power communications, and the risk in the network environment. These tradeoffs are explored in greater detail in chapter 4.

3.7 A TETHERLESS CARE APPLICATION

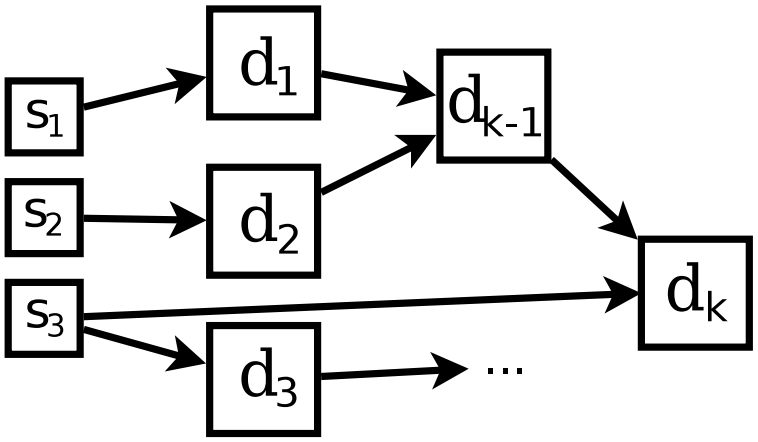


Figure 3: An Example Tetherless Care Application

From a software engineer's perspective, a Tetherless Care Application can be thought of as a graph where virtual sensors and state descriptors comprise the nodes and their declared interests define the edges. His or her goal is to provide an implementation of a set of virtual sensors and state descriptors that properly declare their interests and produce features, states, and actions to effect the relations R, Q, and M.

As shown in figure 3, the software engineer's task in the creation of a *tetherless care application* is tantamount to the development of a set of virtual sensors and state descriptors, which are arranged as a graph. The virtual sensors and state descriptors (1) collectively house an implementation of the relations, R, Q, and M; (2) implicitly define the edges of the graph by declaring their respective interests in the objects produced by other virtual sensors or state descriptors; and (3) produce features, states, actions, and messages.

This architecture allows the software engineer to build abstractions and reason upon the context—the raw samples arriving from the system's sensors—in a focused manner. Each component, be it a virtual sensor or state descriptor, can be designed to make a single transformation of its input

into another domain, i.e. one measurement of electric potential in a ECG sensor is combined with contemporaneous others and defined as a heartbeat.

The implicit definition of the edges between virtual sensors and state descriptors can also allow the graph to change over time in that, as a result of the patient's context, a given state descriptor can cease listening for some type of object and start listening for others. Similarly, virtual sensors and state descriptors can be removed from or added to a running system. This is left for future work.

A second consequence of this architecture allows the intermixing of multiple applications, i.e. the joining of multiple graphs, such that the features or states produced by one virtual sensor or state descriptor are available to those provided by another application. For example, the work of chapter 4 provides a network virtual sensor and network state descriptor for use in the applications that follow in this work. A single instance of these components could potentially service the virtual sensors and state descriptors of multiple tetherless care applications.

3.8 EVALUATION

This section presents an initial investigation that sought to ensure contemporary mobile devices had sufficient battery capacity to sustain the long term operation of the tetherless care system. At a minimum, operation must last for a day, allowing for a nightly recharge of the coordinator while the patient is sleeping. And, if contemporary mobile devices fail to support the continuous sampling and analysis required by tetherless care, an alternate system design and architecture would be necessary.

Two principle factors affect the lifetime of the battery: the amount of processing performed and the amount of networking performed, which can be defined in proportion to the number of instructions executed and the number of bytes transmitted, respectively.

Also, a given tetherless care application requires a certain number of sensors on the patient or embedded in his environment, each of which imposes a minimum sampling rate on the system to effectively interpret its respective signal. This, in turn, results in a proportional rate of feature gen-

eration in the architecture, which the system must respond to with a certain amount of processing and networking, as dictated by the application.

3.8.1 Methodology

To evaluate the architecture, a prototype implementation was developed for Android-based devices, and several tetherless care applications were developed containing a set of virtual sensors and state descriptors that simulated the operation of a tetherless care application. Among them was a battery virtual sensor that monitors the state and remaining charge of the device's battery and an experiment state descriptor that computes the time interval between the point of time when the battery switches from external power to battery power to the point when the remaining battery capacity drops to 50% charge. The experiments defined in each of the applications use these components to measure the system's lifetime given the respective conditions of each experiment. At a minimum, system operation should consume no more than 50% of the charge in 12 hours in order for the full charge to last 24 hours.

The system and applications were deployed to a set of Motorola Droid Pros from Verizon with 1GB of internal storage and running Android 2.2 and, later, 2.3.4. The devices were configured to minimize the number of running background processes such that they remained consistent across trials and that the device's resources were dedicated to the tetherless care application under consideration. Also, the device's other physical sensors, such as the GPS sensor, remained dormant for these experiments.

The first experiment defined a mock virtual sensor that generated small features at a fixed interval and broadcasted them to a mock state descriptor, where they were then discarded. Each trial varied the generation interval and measured the time until the battery capacity dropped to 50% charge. This demonstrated the maximum lifetime that can be expected from any tetherless care application that generates no network traffic.

The experiment was repeated where the mock state descriptor generated a message for a server after the reception of each feature. The information delivery component opened and maintained a TCP connection for the duration of each trial. And, as each message was generated by the mock state descriptor, it was transmitted to the server immediately.

Both of these experiments were again repeated, but where a Bluetooth connection was created between the smartphone and a laptop. Data samples were generated on the laptop periodically and transferred to the smartphone so as to simulate the demands of a secondary sensor device on the patient's body that wirelessly transmitted information to the coordinator. With each received sample, the virtual sensor generated a feature. From there, the operation is the same as the other experiments with a mock state descriptor that either discarded the feature or generated a message for the server.

A prolonged, open TCP connection was chosen over a repeated set up and tear down of it after an initial experiment demonstrated the idle time between two messages was likely to be shorter than the set up and tear down times. While this prolonged connection violates the required mobility of tetherless care, the focus here is on the power requirements of constant network use for a single tetherless care application.

3.8.2 Results

The results obtained are shown in figure 4. Without requiring a network connection, the system can last well beyond the minimum lifetime while the feature rate remains below roughly 130 features per second. This demonstrates the feasibility of the architectural design for some possible tetherless care application with matching processing requirements.

A persistent TCP connection proved to be a poorer implementation of the information delivery component, but it demonstrated feasibility for feature rates around ten features per second or lower. This may be amenable to certain tetherless care applications with minimal processing requirements, but such applications would likely require a dedicated device with this implementation. The user's interaction with other apps on their smartphone would likely consume the excess energy and push the system below the minimum lifetime. Moreover, the user may be more inclined to disable the system to save energy for these other applications.

The addition of a persistent Bluetooth connection from a simulated sensor added a significant additional energy expenditure in both cases. Surprisingly, though, the configuration involving a persistent Bluetooth and persistent TCP connection significantly outlasted the configuration with a TCP connection alone at 25 features per second. With the timer functionality that generated

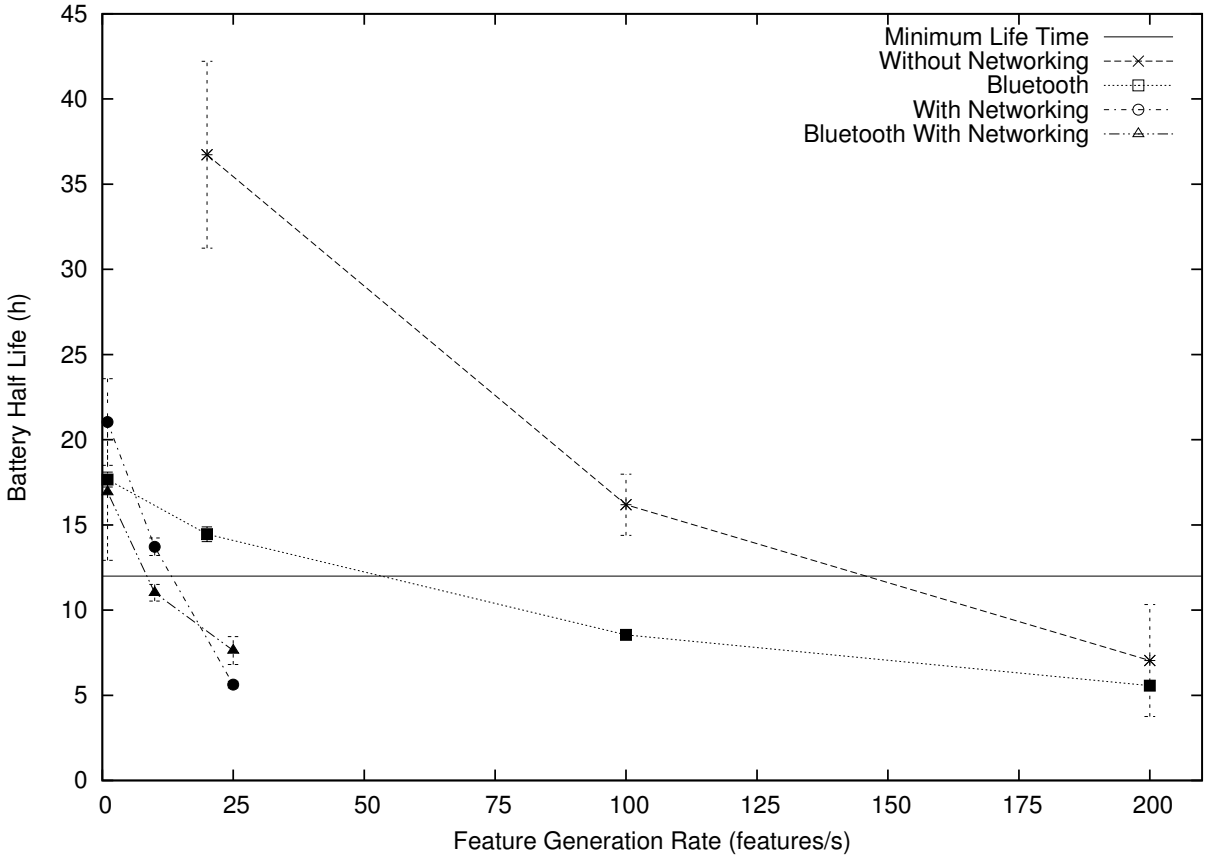


Figure 4: Effects of Feature Generation Rate on Battery Half Life

mock data relegated off the Android device, the hardware was more IO-bound, blocked waiting for data from the Bluetooth connection, and able to put the processor in a low-power state with this configuration. Where the device was responsible for generating the mock data itself, it required its own processor to compute the timer functionality, which likely impacted its lifetime.

Still unknown is the aggregate feature rate produced by tetherless care applications and where such an application would fall within the space outlined by this figure. For example, an ECG virtual sensor is likely to produce and broadcast a feature for each heartbeat, and each heartbeat consists of a P wave, QRS complex, and T wave, all which occurs over about .62 seconds. Thus, at 16-bit samples and given the sampling rate from Table 1, each heartbeat feature contains about 650 bytes and is generated once per second with a maximum of four times per second. Where these features demonstrate a normal heartbeat, no transmission over the network may be necessary. As such, a heart monitoring application may likely fall within the feasible bounds.

Also, in the four years since the release of the Motorola Droid Pro, considerable advances have been made in battery capacity, the ARM architecture, and the Android operating system. It is likely that, on more contemporary hardware, the bounds of feasibility are modestly wider.

3.9 CONCLUSION

These results outline the expected operating bounds of tetherless care applications, illustrating the maximum processing that the platform can currently sustain and demonstrating that intelligent use of communication channels are necessary. In practice, tetherless care applications may never approach the feature generation rate limit outlined here nor need to transmit each feature to the server. What is clear from this initial experimentation, however, is that the use of network hardware must be minimized in order to meet the mobility requirement of tetherless care. With an information delivery component that assiduously manages the network environment and the demands of tetherless care application data, these results argue the affirmative for the fundamental question of providing tetherless care equivalent to traditional, in-hospital care.

4.0 INFORMATION DELIVERY

In chapter 1, the fundamental question was posed of whether it's possible to provide care in a tetherless manner that is commensurate with traditional tethered care. In the previous chapter, results demonstrated that a persistent connection from the coordinator to the server, while a violation of the tetherless care assumption in that it confines the user to the broadcast radius of the associated access point, also dissipates the battery too quickly to provide the longevity the system requires.

As such, the coordinator must disconnect from all remote entities for some time in order to save enough power to survive a day's mobility. But, moreover, the patient's mobility may bring the coordinator into environments that force a disconnection due to the lack of any available communication network. These challenges—the limited power availability and necessary intermittence of connectivity—present two of several challenges that also include the risk inherent in such environments and the evolving criticality of the patient's state.

A number of situations occur for which the system must respond with some communication. These situations range in their severity. But, overall, the desired operation of the system would respond to more severe situations more quickly using greater power and less severe situations with greater convenience and lower power. This notion of severity, however, requires combined knowledge about the patient's situation from several disciplines, at least one of which is specific to the application monitoring the patient and one of which understands the network technologies available to the patient in a given moment. The model and architecture presented in chapter 3 allows each of these disparate knowledge domains to be encapsulated in separate virtual sensors or state descriptors. But, where the middleware[1] assumes some responsibility in managing the

[1] the terms *queuing infrastructure*, *middleware*, and *information delivery component* are used interchangeably within this chapter

network actions generated by a given application, a method of communicating this severity to the queuing infrastructure is necessary.

Chapter 3 modeled this method as the notion of *satisfaction*, Ψ, which, at a given time, t, associates a set of actions, A, with the state of the patient, I_t, defined as a set of state objects, such that any one of the actions, if executed, could satisfy the mission of the application with respect to the patient's condition. The constraints that dictate inclusion of $\{A, I_t\}$ in Ψ are application-dependent and are expected to vary with the severity of I_t. Thus, given the separation between the application and the middleware in the system architecture, each component must compute its respective constraints, a representation of them must be communicated from application to middleware when network-related actions are necessary, and the middleware must make a final determination of which action in A to execute. A full exploration of possible representations of these constraints is left for future work, yet the implementation discussed in this chapter expresses them as a single deadline value. For situations with higher severity, such as the fall event discussed in section 3.3, the application expresses a shorter deadline, and, for less severe situations, the application expresses a longer deadline.

The notion of a *message* is also discussed in chapter 3 as being logically equivalent to a set of actions, A, where a message is a response to the patient's state, I, generated by the application in order to express the need to execute one of the actions in A subject to the constraints of Ψ. These actions can be thought of as a standard set of actions that include the subset modeling the upload of the message's payload over each of the available network technologies (e.g., *upload-WiFi*, *upload-3G*, etc.), along with a subset that includes fallback actions when no connection is available subject to the message's constraints, such as directing the user to find an available connection, dialing 911, or making noise to attract help from nearby people. And thus, to each message, the application must attach its representation of the constraints for Ψ.

The task of the information delivery component is then to evaluate the constraints expressed in the messages arriving from the application within the context of the patient's network environment (i.e., his or her network state), the exploration of which is the subject of this chapter. To accomplish this, the implementation defines a special virtual sensor and a special state descriptor, referred to as the *NetworkSensor* and *NetworkState* hereafter, that observe the moment to moment changes in the available network technologies and compute a series of network *state*s that indicate the formation

and loss of connection opportunities. Algorithm 1, presented in chapter 3, is employed over the series of states produced by the NetworkState to assess a component of the risk in the patient's environment (i.e., the severity of the patient's environment from a networking perspective). It provides an implementation of the function, Θ, which is an input to algorithm 1, that maps the constraints expressed in each message on to the predicted network environment, and it ultimately determines which of the actions in A to execute for each message.

Section 4.1 defines this task in formal language and discusses a transformation of the series of states produced by the NetworkState that allows for a smaller storage and more intuitive reasoning. Section 4.2 formally describes the use of algorithm 1 and defines the Θ function in use. Section 4.3 discusses the evaluation the performance of the middleware. Section 4.4 concludes the chapter.

4.1 PROBLEM DEFINITION

The middleware must attend to several phenomena: the data to transfer, the state of the patient expressed by the application, the risk associated with that state, the time-varying state of each available network technology, and the available power on the coordinator.

The data to transfer is defined as a set of messages, Q, received from the application. Each message is annotated with the constraints to which the middleware must adhere in acting upon the message. Thus, each message is a tuple:

$$\langle id, size, payload, \kappa \rangle$$

where *id* is a the message identifier, *size* is the size of the payload, and κ is the representation of the constraints of Ψ.

Let $\mathscr{I}_{NS} \subset \mathscr{I}$ denote the domain of states produced by the NetworkState. A network session can be derived from the following property of the states produced by the NetworkState:

$$\forall_{i \in \mathscr{I}_{NS}} \exists_{t \in \mathscr{T}} \exists_{t+d \in \mathscr{T}} \forall_{t' \in \mathscr{T}} [t' \geq t \wedge t' < t+d \wedge i \notin I_{t-1} \wedge i \in I_{t'} \wedge i \notin I_{t+d}] \tag{4.1}$$

where i is a state object and I_t denotes the set of states produced by the NetworkState at time t.

This sentence states that there exists some time, t, at which state, i, is first added to the set of states produced by NetworkState, that i remains a member of the set for some contiguous time interval, d, and i is removed from the set at time $t+d$.

Assuming that each i differentiates a network opportunity to a given remote entity over a given network technology (i.e. it identities the remote entity, in terms of either a identifier or venue, and the type of technology used to access it), a *network session* is then defined from the phenomenon described above as a tuple:

$$\langle id, type, s, d, t, v \rangle,$$

where id is an identifier of the remote entity to which a connection is made as defined by i; $type$ identifies the underlying technology of the session, e.g. 802.11g/n, EVDO, LTE, bluetooth, also defined by i; s is the session start timestamp; d is the duration of the session; t is the measured throughput achieved during the session; and v is an optional venue or location identifier.

Finally, with respect to the available power, each message with its constraints, κ, constitute a directive from the application to expend energy in service of the message, a directive which the middleware cannot ignore. Nevertheless, the energy required differs amongst the set of actions the middleware could execute in service of a given message: the energy consumption of 4G/LTE networks exceeds that of 3G networks which, in turn, often exceeds that of WiFi networks. On the other hand, WiFi connectivity typically experiences less availability than WWAN technologies, like 3G and 4G. This indicates that the technology chosen to upload a given message affects the power consumption of the system. [13, 33]

Thus, the problem is stated as:

> Given a set of messages, Q, and the history of recent network sessions, H, produced by the NetworkState, for each message, m, the middleware must select a time, now or in the future, at which to transfer the message subject to its constraints and the goal of reducing the power consumed.

The remainder of this chapter presents a solution to this problem utilizing algorithm 1 over the history of network sessions, which essentially assesses the risk of the current network environment, applying a Θ function that incorporates the constraints of each message in Q into the risk. If the risk is low enough, the middleware can delay the upload of Q until some future opportunity. Otherwise, it must initiate communication immediately.

Algorithm 1 uses an $O(1)$ Markov process over the history, H, the current network session, s_t, and a probability threshold, Φ, to project future possible network sessions that could occur after s_t. The history defines a sequence of sessions, each with an identifier of the remote entity providing the network opportunity, and an order in which the patient visited them. Section 4.2 describes an algorithm for extracting every temporal pair of sessions—a current session and its successor—and storing this relationship in a form that allows for easier lookup (tantamount to memoizing algorithm 1). The frequency with which a given pair appears in H is assumed to increase the probability that, if the patient initiates the starting session of the pair, he or she will later move to the subsequent session of the pair, which is described by equation 3.18.

Ultimately, algorithm 1 constructs a tree of potential paths the patient could take from the current session to subsequent ones. The root of the tree corresponds to s_t, and each child is one probable session to be seen in the future. Given the current session, s_t, it finds all the pairs from the history, H, where the starting session equals s_t (i.e. where the coordinator was connected to the same remote entity) and collects the set of unique subsequent sessions from those pairs. For each of those potential subsequent sessions, b, if the probability of moving to b next is above Φ, it adds a branch to a tree from the root node to a new node corresponding to the session, b. It can then recursively build the next level of the tree using each b as the "current" session. The expected interval of time between the start of any current session and its subsequent one along with the expected duration of each session is also stored in the tree to obtain an estimate of the delivery delay along each path.

Upon the return path up the recursion, the algorithm then applies the supplied Θ function, which was described to assign a risk value to each node in the tree using the risk already computed for each of its children. The Θ function implemented for the information delivery component is discussed in section 4.2.3. This function determines if the child sessions of the current node are capable of uploading the messages in Q, each subject to its constraints, or if its necessary to initiate an upload during the session that corresponds to the current node. A delay is only possible if each child provides sufficient bandwidth to service the entire message load and meets the constraints of each message in Q. If any child fails this condition, the risk of a delay is concluded to be too great, and an upload must be initiated during the current session.

4.2 OPERATION OF THE INFORMATION DELIVERY COMPONENT

The information delivery component can reduce its power consumption with a heuristic that universally favors networking technologies in order of their assumed respective power consumption: Bluetooth, WiFi, 2G, 3G, 4G, LTE, respectively. The most economical choice may often be to wait for a future Bluetooth- or WiFi-based session before uploading information, introducing delay to save power. However, this delay is limited by the constraints of the messages to be uploaded, which could prevent the system from delaying indefinitely and relying solely on some future WiFi connection.

4.2.1 Memoizing Session Pairs

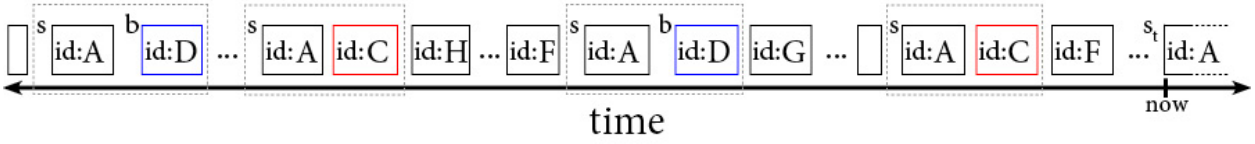


Figure 5: Log Analysis

For a given ongoing session, s_t, find all previous sessions s and b where $s = s_t$ and b is the temporally subsequent session after s.

The first step of algorithm 1 must compute the probability of transitioning to all possible subsequent sets of states given the current set of states, which, for the purposes of this chapter, computes the probability of all possible subsequent network sessions given the current session. As defined by the probability definition in equation 3.18, this amounts to a scan through the history, H, in search of all sessions, s, and its subsequent session, b, where s matches the current session, s_t. Figure 5 shows an example where the current session, s_t, has an identifier of *'A'*, and algorithm 1 finds the session pairs, $\langle A,D\rangle$, $\langle A,C\rangle$, $\langle A,D\rangle$, and $\langle A,C\rangle$; that is, two instances of the pair, $\langle A,D\rangle$, and two of the pair, $\langle A,C\rangle$. It is the set of these subsequent sessions, i.e. the set containing *'D'* and *'C'* in this example, to which equation 3.18 assigns a nonzero probability, or in other words, it is one of these sessions that is more likely to occur after s_t.

In recursively building a tree of possible future sessions, along with the probability of transition to each session, the middleware will estimate the duration of the current session under considera-

tion as well as the time interval from the start of the current session until the start of each possible subsequent session.

Algorithm 2 constructs a three-dimensional mapping, M, that memoizes part of this work. The first two dimensions each index along all the unique sessions observed in the history, where the first index represents the current session, s, identified by its id, and the second index represents its subsequent session, b, identified by its id. The third dimension is a list of tuples, each of which store information about one instance of the sequence $\langle s, b \rangle$ found in H. Specific to the middleware, each tuple is $\langle s.s, s.d, b.s - s.s \rangle$, i.e. the start time of s, the duration of s, and the interval of time between the start of s and the start of b. Considering s_t then, its duration can be estimated as a function or aggregate of the set of $s.d$ values stored in the list, and a similar estimation can be made for the start of a possible subsequent session.

Algorithm 2: Memoizing session transitions

Iterate over each session, s, in history, H, find the subsequent session, b, and store in M the the start time of s, the duration of s, and the interval of time between the start of s and the start of b, each of which is necessary to make a prediction about a future transition from session s to session b.

Require: session log L, map M
  for all sessions s in L **do**
    $b \leftarrow findSubsequentSessionOfType(s)$
    $push(M[s.id][b.id], \langle s.t, s.d, b.t - s.t \rangle)$
  end for

While not implemented in the system prototype and left for future work, this mapping can easily be implemented on cloud storage technologies such as BigTable [17], which offers a multidimensional key(s)/value storage abstraction for structured data over a distributed set of nodes. They offer an interface such that the developer can reason about the storage system as one, very large, sparsely populated table. A *row key* identifies a row, which can consist of a dynamic number of columns that each belong to a predefined number of *column families*. At each row and column, the developer is not only able to store or retrieve an object but also access previous, older versions of the object. Upon a *put* operation, each object is timestamped, such that the timestamp creates an ordering of the versions of the object.

For the tetherless care middleware, the row key is a concatenation of a user identifier, the current session identifier, $s.id$, and any venue identifier in $s.v$. A column family would be defined

for subsequent sessions where each column in the family would be identified by $b.id$. The start timestamp of the current session would be used as the timestamp for the table entry, which would store the tuple, and the versioning history of the storage would compose the list.

4.2.2 Session Prediction Tree

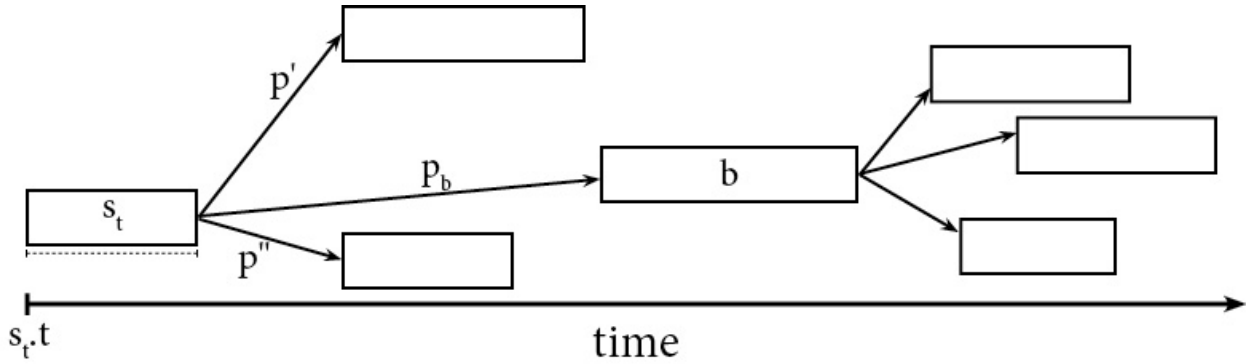


Figure 6: Tree of Likely Future Sessions

An illustration of the tree constructed by algorithm 1. The size and spacing of the nodes, defined by the expected session duration, $E[s.d]$, and expected start time interval, $E[b.s - s.s]$, respectively, for sessions s and b, are assigned by the Θ function.

As mentioned above, the middleware seeks to delay transmission of the set of messages in Q in order to save power while still adhering to the constraints defined in each message. The approach taken characterizes the problem as an assessment of the environmental risk from a network perspective, such that algorithm 1 can be applied. This approach allows tetherless care applications to encode the risk assessed from other perspectives into the constraints of a given message where it can then be evaluated further in a networking context within the middleware. This section discusses the application of algorithm 1 to M, Q, and the domain of network sessions.

Figure 6 illustrates the tree of likely future sessions ultimately constructed by algorithm 1 and aided by the mapping, M, constructed by algorithm 2. Each node corresponds to a session, s. For any given node, n, and its corresponding session, s, the map, M, stores a row at $s.id$ where each entry in the row corresponds to some session b that occurred after a previous instance of s, and the cell at $M[s.id][b.id]$ stores a collection of tuples that counts the number of times a transition occurred from s to b in the past. Thus, the *transition probability* of a future transition from s to b,

as defined by equation 3.18, is the number of items in the collection at $M[s.id][b.id]$ divided by the sum of the number of items in every list across the row $M[s.id]$. Note that, for this computation, only the quantity of objects in the collections of M are needed and not their contents.

Starting with a root node that corresponds to the current, ongoing session, s_t, the algorithm adds a child node for each, b, in $M[s.id]$ whose transition probability, p_b, is above the threshold probability, Φ. The expected start time and expected duration of the node, or the session represented by the node, rather, are computed from the contents of the collection at $M[s.id][b.id]$. And the process is then repeated for each child node.

4.2.3 Middleware Theta Function

Once the above tree is constructed, algorithm 1 applies a Θ function to each node as the recursion returns up the tree that assigns a *risk* to each node. The middleware utilizes an implementation of this function, shown in algorithm 3. The computation applied to a given node depends on the risk computed for its children, and the final, resulting risk object assigned to the root of the tree is used to ultimately decide whether to initiate an upload immediately or to wait for some future opportunity, thereby saving power.

The risk assigned to each node in a given schedule tree is defined as a tuple, $\langle wait, miss, freeTimeNow, freeTimeFuture \rangle$, where *wait* is a boolean set true if the sessions corresponding to all child nodes in a particular subtree can service the queue; *miss* is a boolean set true if some deadline would be missed in the session at this location in the tree; $freeTimeNow$ is the minimum number of free, unscheduled milliseconds available in this session, which can be negative; and $freeTimeFuture$ is the minimum number of free, unscheduled milliseconds available in some future session.

Algorithm 3 assumes the existence of several functions, namely *parent*, *duration*, and *nextStartInterval*. The function, *parent*, returns the parent node and corresponding session of a given node. The function, $duration$, is a function of the duration values stored in the collection at a given cell in M (e.g., it computes the average duration from the duration in each tuple of the collection). The function,$nextStartInterval$, is a similar function over the start time interval in each tuple of the collection, where start time interval refers to the interval stored in the tuple between the start of

Algorithm 3: Middleware Theta Function

Assigns a computed risk object to each node in the tree of likely future sessions. It decides if an upload should be conducted during the session, s. An upload can only be delayed if each child session could service the message queue, Q, subject to each message's constraints. Messages are slotted into the window defined by the session's duration, at which point their constraints are evaluated. The remaining free time, which can be negative, and a boolean indicating if the system would fail to meet the constraints of one or more messages during this session are returned in a tuple, which is assigned to the *risk* of the node, n. The remaining free time is then added to the parent's window in future executions of the function so as to concatenate network opportunities.

Require: map M, message queue Q, function *parent*, function $duration$, function *nextStartInterval*
  function $\Theta(n, s)$
    $p, s_p \leftarrow parent(n)$
    $s.d \leftarrow duration(M[s.id])$
    $s.s \leftarrow s_p.s + nextStartInterval(M[s_p.id][s.id])$
    $\langle d, m \rangle \leftarrow$ SCHEDULEMESSAGES$(s.s, s.s + s.d, Q, s.t)$
    if $size(children(n)) = 0$ **then**
      return $\langle false, m, d, 0 \rangle$
    end if
    $min \leftarrow \langle false, true, \infty, 0 \rangle$
    for all node $c \in$ CHILDREN(n) **do**
      $u \leftarrow$ c.risk
      if $s.d + u.freeTimeNow < min.freeTimeNow$ **then**
        $min \leftarrow \langle u.freeTimeNow > 0$ and not $u.miss, m, s.d + u.freeTimeNow, u.freeTimeNow \rangle$
      end if
    end for
    return *min*
  end function

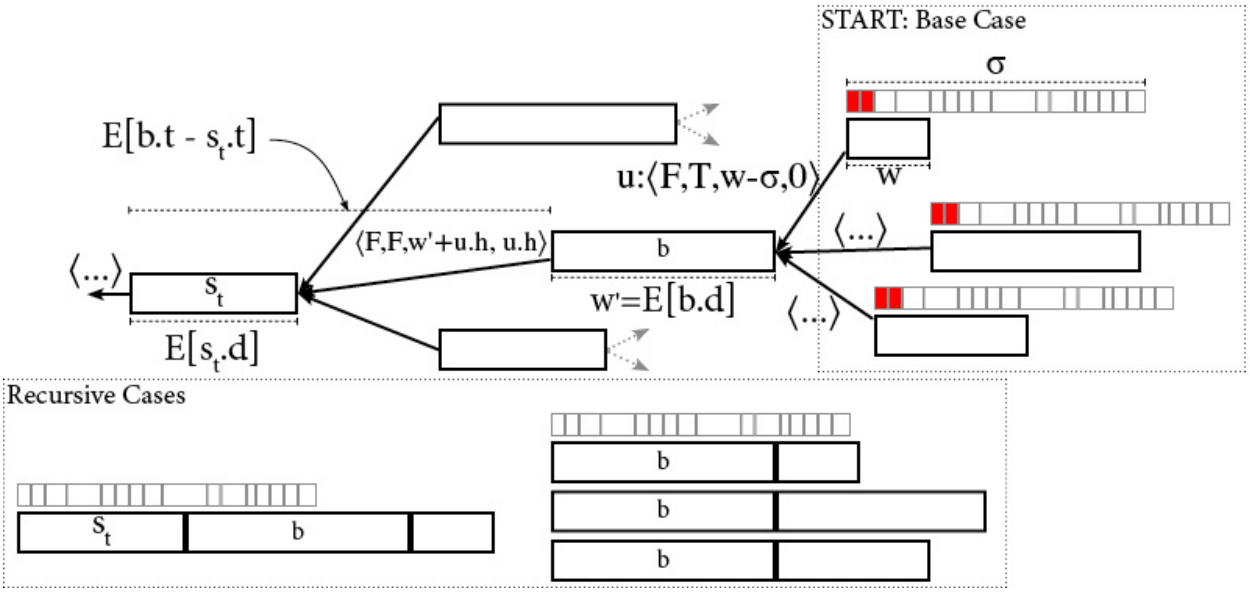


Figure 7: Scheduling Illustration

An illustration of applying algorithms 3 and 4 to the schedule tree. At any node, the algorithms must decide if a transfer can be delayed to a future session, which is only true when each child node provides sufficient bandwidth to service the message queue while meeting each messages constraints. For the leaves, this is always false, but the amount of unused bandwidth is needed as the recursion returns up the tree, which is denoted as $w - \sigma$ here. An interior node can wait only when each child reports that all message constraints would be met by that child and that the unused bandwidth is positive or zero. Then, an interior node returns amongst its values the sum of its available bandwidth and the minimum value amongst its children.

session s and session b at cell $M[s.id][b.id]$ (e.g., it computes the median of the start time interval). These functions are fairly trivial and, therefore, not shown.

Algorithm 4: Message Scheduling

Slots each message in Q into the time window marked by starting time, t_s, and ending time, t_e and evaluates the constraints of each message to determine if any would be unmet in this time window, where $m.\kappa()$ is a application defined function for evaluating these constraints.

```
function SCHEDULEMESSAGES(t_s, t_e, Q, r)
    miss ← false
    for all message m ∈ Q do
        t_m ← m.size ÷ r
        if m.κ(t_s, t_m) then
            miss ← true
        end if
        t_s ← t_s + t_m
    end for
    return ⟨t_e − t_s, miss⟩
end function
```

For each node on which the Θ function is applied, algorithm 3 uses these functions to assign an expected duration, $E[s.d]$, and expected start time, $E[s.s]$ to the corresponding session, s. These values are then inputs for algorithm 4, which reserves a slot for each message along the time window defined by the session's duration. The size of each slot is computed as the ratio of the size of the corresponding message and the session's observed average data rate. Algorithm 4 then returns a tuple of two values, $free$ and $miss$: the number of remaining, free milliseconds in which no message is scheduled and a boolean value indicating that the system would fail to meet the constraints of one or more messages if they were uploaded during this predicted opportunity.

For the leaves of tree, when no subsequent sessions elicit a probability greater than the parameter, Φ, to algorithm 1, the system would be unable to wait for some subsequent session, and the Θ function would return a tuple where $wait$ is false, $miss$ and $freeTimeNow$ are set to the result of algorithm 4, and $freeTimeFuture$ is zero.

For non-leaf nodes, the Θ function has already been applied to each of the children, and a transfer can deferred into the future from the current node under two conditions: (1) each child node must be capable of meeting each of the message constraints and (2) each child node must have sufficient bandwidth to service the entire message queue. The first condition is evaluated

from the *miss* value associated with each of the child nodes: if each is set false, it may be possible to defer transfer. For the second condition, algorithm 3 selects from the tuples assigned to each child node the tuple, u, with the minimum value of $u.freeTimeNow$. If this value is positive, a deferred transfer may also be possible. It then returns a tuple where *wait* is set to true if both conditions hold, *miss* is set true if the result of algorithm 4 indicates that some constraint would be missed in the corresponding session, $freeTimeNow$ is set to the sum of $u.freeTimeNow$ and $s.d$, and $freeTimeFuture$ is set to $u.freeTimeNow$.

Figure 7 illustrates a schedule tree where the width of each node depicts the time window of each corresponding session, their horizontal arrangement depicts the expected start time relative to the root of the tree, and an array of message slots are shown above several of the nodes. The illustration follows the recursion from the children of session, b, up to the root of the tree at session, s_t. The most relevant child is that which returns the smallest value of $freeTimeNow$, which is the top node in the illustration and which returns a value of $w - \sigma$. Since at least one of these children cannot meet the message constraints, identified as the filled in message slots, the node corresponding to session, b, cannot defer a transfer into the future, setting its *wait* value to false. It, however, is able to meet the constraints of each message. And it essentially returns a value of $E[b.d] + w - \sigma$ for $freeTimeNow$. Assuming the other children of the root node return similar values, algorithm 3 can then conclude that a transfer can be deferred into the future.

Algorithm 5: Session Selection

Require: schedule tree r, current session s_t, message queue Q

BUILD_TREE($(r, s_t, M, 1.0, \Phi, \Theta)$)

tuple $u \leftarrow$ r.risk

if not $u.wait$ or $u.miss$ **then**

    STARTUPLOAD( )

end if

By selecting the minimum $freeTimeNow$ value amongst a set of child nodes, the algorithm selects the path through the tree with the minimum available bandwidth to determine if communication can be initiated at some point in the future. If this path provides sufficient bandwidth to do

so while meeting the constraints of each message, which is indicated by the *wait* and *miss* values returned in Algorithm 4, the system can assume, with reasonable certainty, that a transfer can be delayed. Otherwise, for safety, a transfer should be initiated in the current session, which is shown in Algorithm 5.

4.3 EVALUATION

4.3.1 Methodology

The algorithms above are designed to execute in both a cloud computing environment and on a mobile device. However, the mobile device must be capable of executing the algorithm in a stand-alone fashion in a limited capacity when cloud computing services are not available for network availability or power reasons. As such, a proof-of-concept implementation of each of the algorithms was developed to support the tetherless care architecture, running exclusively on the mobile device.

A tetherless care application was developed to execute within the tetherless care architecture outlined in section 3.6. It generated messages for the system containing the results of instrumenting the NetworkSensor and NetworkState. Also, the application implemented a location monitoring algorithm based on three sources of location information from the Android OS [18] and generated messages regarding the device's location periodically. The application defined a sole constraint for each message to be a *deadline* value, where a severe situation would result in a short deadline and normal situations would result in longer deadlines. However, the severity of the situation in this study did not fluctuate for this application, and all deadlines were set at a fixed 24 hours to represent low priority information and not risk frustrating volunteers of the study. The implementation was confined to monitor, predict, and utilize WiFi-based sessions only to exploit their intermittent availability and prevent impacting the cellular data usage of volunteers in the study.

The middleware was similarly instrumented to generate messages as if they were generated by the application as well, also with a fixed 24 hour deadline constraint for each message. The Θ function assumed every session had the same data rate of $400kbps$, or 50 bytes per millisecond,

which is well below the advertised throughput of $22Mbps$ for 802.11g. The middleware was also implemented to execute algorithms 1 and 3 in response to two events: (1) a new session began and (2) a new message was sent to the middleware whose deadline was less than six minutes, a value chosen that would ensure enough time was given to the middleware to setup a connection for an urgent transfer.

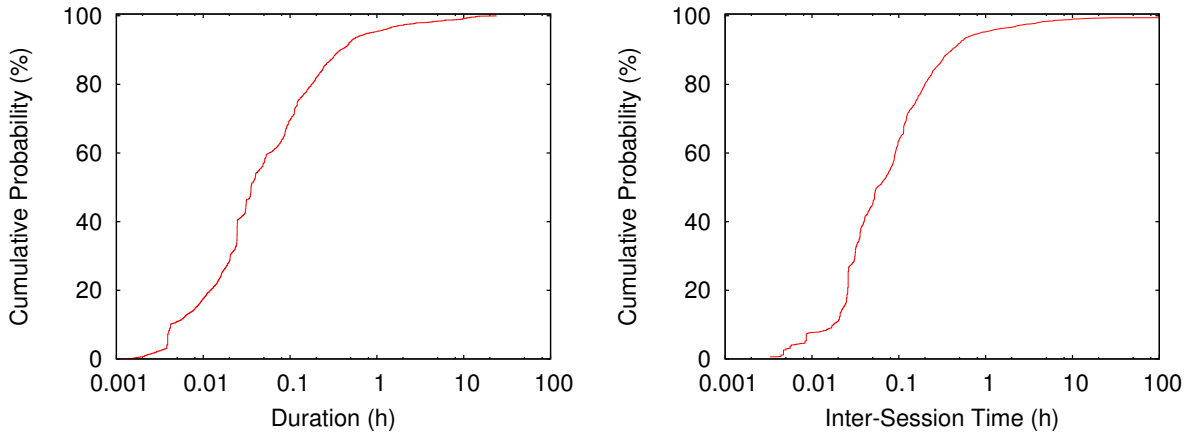
The architecture and application were deployed to a set of Motorola Droid Pros running Android 2.2 from Verizon with 1GB of internal storage, which was carried by the author for a period of two months. In addition, volunteers were petitioned to install and run the software on their personal Android devices of unknown and varying capabilities. However, only two of the volunteers maintained the running software, and neither participated for the full two months.

A log of the WiFi network sessions was collected for a period of two months, stored on each device, and used as the basis for the algorithms' computation. Each outgoing edge of a given node in the schedule tree stored an expected duration of the session and an expected start time of its potential subsequent session. As each session ended and if the schedule existed, a comparison was made between the actual duration and the expected in each child node. Similarly, when a session began, the start time was compared to both the expected start time and the median start time stored in the associated child node, if it existed. The number of times the system delayed a transmission until a future session was counted along with the free unscheduled bandwidth as defined by algorithm 3, the number of missed deadlines, and the actual measured speed of each transmission.

After the two months, a set of simulations was executed using the collected session log from one of the devices in the study. Each session start and end event was replayed to the algorithms in the application and system, and the same set of information was collected from instrumentation in the application and middleware. Each trial of the simulation was executed with a different fixed deadline value ranging between 15 seconds and 24 hours and value of Φ selected from the set, $\{0.005, 0.01, 0.05, 0.2, 0.4\}$.

4.3.2 Results

First presented are the results obtained from the real-world trace collected over two months with a message deadline of 24 hours and Φ value of 0.2. Figures 8a and 8b illustrate characteristics of the network environment found around the collection of volunteers in the study. These are independent of the application and system and are likely to change from patient to patient. Figures 9a and 9b illustrate the ability of algorithm 1 to observe and predict this environment. And figures 10a and 10b illustrate characteristics of the transfers performed by the system, which demonstrate the load the test application placed upon the system.



(a) Variation in Duration

(b) Variation in Inter-Session Interval

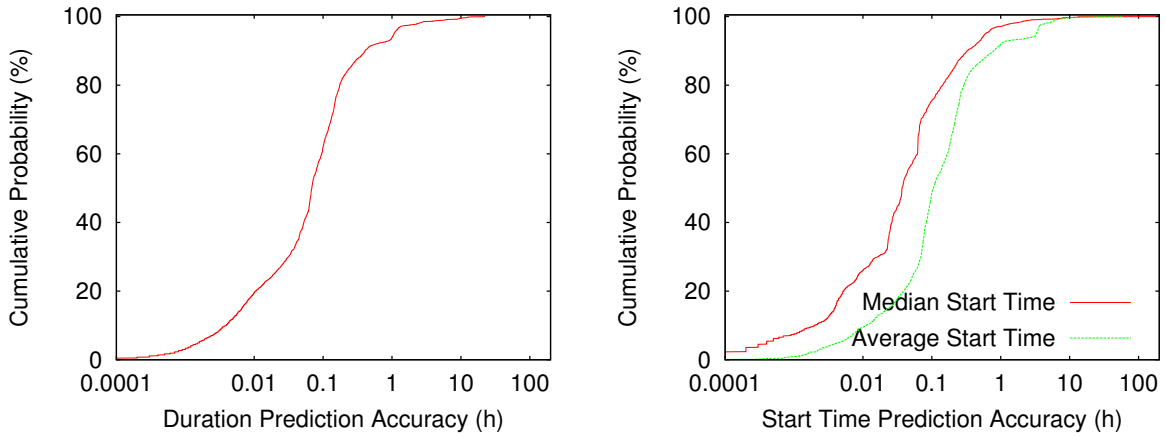
Figure 8: Session Environment

The distributions of the length of each WiFi session and the interval between subsequent sessions, shown here plotted on a log scale, characterize the network environment found by volunteers in the study. These are independent of the application, middleware, and algorithms deployed and likely change from patient to patient. Thus, they provide a context in which future work may be compared with these results.

Figures 8a and 8b plot the cumulative probabilities of the measured session durations and inter-session time intervals—that is, the time interval between the end of one session and the start of the next—respectively, collected from 5711 sessions initiated with 231 different access points, 93 of which were visited repeatedly, over the course of the study. The inter-session interval distribution is a common measure in delay tolerant networks because it illustrates the largest component of the delay experienced by data traversing the network. We observed that 80% of sessions were eleven minutes or less, ranging from several seconds to several hours, and 93% of the delays were less

than an hour. Also, for this study, the 24 hour deadline was lax enough to encompass 99.6% of delays. As a result, with each new message that came into the system, there was almost always some predicted future session capable of handling the message. And, given the majority of delays were on the order of tens of minutes, this environment suggests that the intermittently connected WiFi environment experienced by the volunteers in the study was capable of deadlines also on the order of tens of minutes and would only need to fall back on other network technologies for shorter deadlines or uncharacteristically long delays. This is corroborated in the deadline sensitivity results below.

These characteristics of the network environment are likely to change from patient to patient. As such, they provide a context in which to place the subsequent results below and to which future studies can be compared. Future work must also seek out patients with variations in these distributions and seek to explore the bounds of these variations.



(a) Accuracy of Duration Prediction

(b) Accuracy of Session Start Prediction

Figure 9: Accuracy of Session Predictions

Plots the distribution of the absolute value of the difference between the actual duration or start time and the predicted duration or start time contained in the schedule tree.

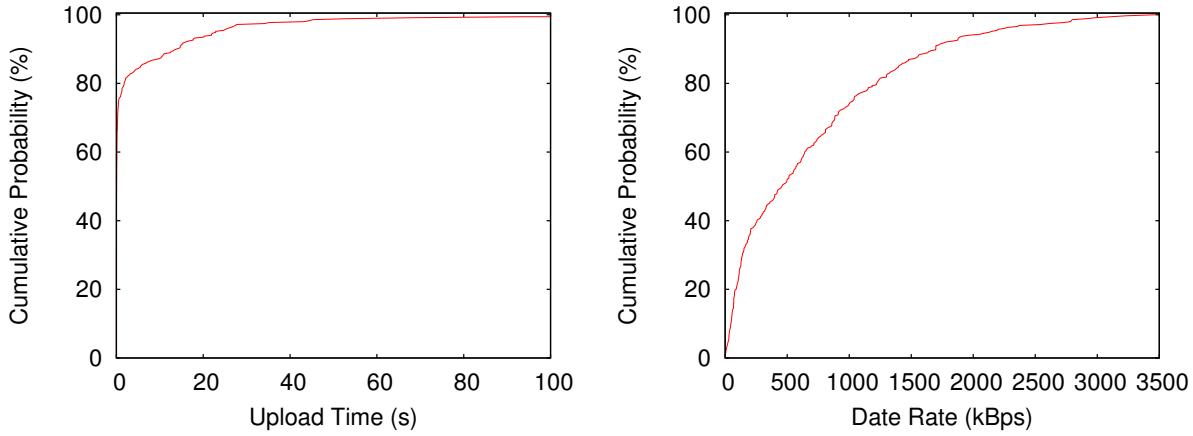
Figures 9a and 9b plot cumulative probabilities of the accuracy of the predicted duration and the predicted start time values, respectively, e.g. the distribution of the absolute value of the difference between the actual duration of a given session and the algorithm's predicted duration. More specifically, figure 9a plots the most accurate prediction from the set of predictions contained in

the child nodes of the current session in the schedule tree. We observed that 80% of the predicted durations were within 4.8 minutes of the actual. Similarly, 80% of the average predicted start times were within 6.6 minutes of the actual start time, and 80% of the median predicted start times were within 3 minutes of the actual start time.

These plots demonstrate that the predicted schedule matches the observed schedule, but most sessions lasted a few minutes and 60% of the predicted durations were off by a few minutes, an order of magnitude of error that prevents an accurate estimate of the amount of data that an ongoing or future session can transfer. This is mitigated in two ways: (1) recomputing the schedule at the beginning of each session so as to avoid compounded error deeper in a schedule tree and (2) initiating transfers upon completion of the scheduling algorithm at the beginning of a session, which avoids an attempt to upload just before the session is expected to conclude. Moreover, while algorithms 1 and 3 were intended to model the likely future environment, they needed, at a minimum, only to differentiate the safer, predictable environments from risky, new, and less predictable ones. And the delivery metrics presented below demonstrate the degree to which they succeeded for this test application and these volunteers. Several avenues of future work can (1) examine the impact of an increased traffic load in these delivery metrics given the limited accuracy of the prediction and (2) examine the inclusion of other session identifiers that more accurately target the patient's movement as it relates to session duration and start time (e.g. the time of day at which a patient connects to a certain access point may better indicate how long the patient remains connected).

Figure 10a plots the cumulative distribution function of the upload time during each transfer, which could last for a few minutes at times but typically was on the order of a few milliseconds. Thus, this illustrates the light load imposed on the system during this study. As such, the needed transfer time was unlikely to overlap and be negatively impacted by an inaccurate prediction in the session duration. Also, the average depth of the schedule tree was 1.0103 with a max of 3, meaning that, most often, the algorithm was only able to predict one additional session into the future given the threshold value of $\Phi = 0.2$, which often prevented a compounding effect of the prediction error on the scheduling algorithm.

Of the more than 5700 executions of the scheduling algorithm, 89.10% of them determined that it would be possible to delay transfer for some future session. Moreover, 94.21% of the



(a) Transfer Time Distribution

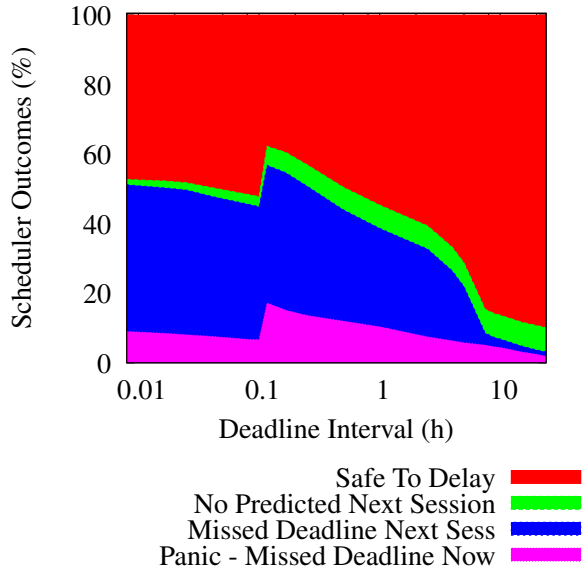
(b) Transfer Rate Distribution

Figure 10: Transfer Characteristics

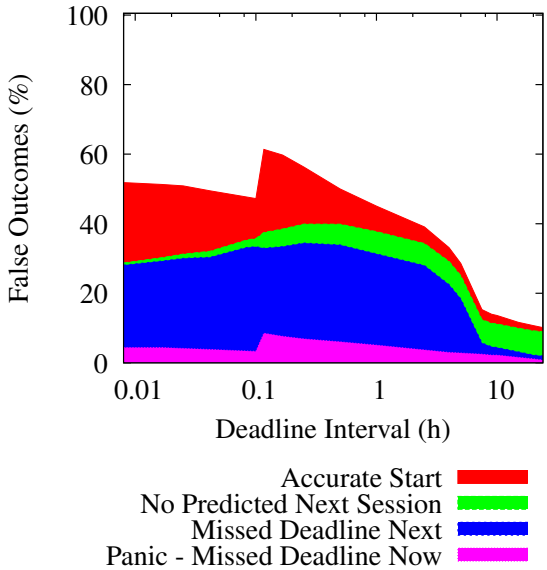
schedules produced were expected to provide enough bandwidth to service an existing data load. The remaining 5.72% of computed schedules occurred either when messages had already missed their deadline at the time of the scheduler execution or when a new access point was encountered such that no prediction could be made regarding its duration or its successive sessions. The former is corroborated by the fact that 8.42% of the messages missed their deadlines, which were often received by the server in bursts. This phenomena was likely due to periods of time over the course of the study where volunteers were not running the prototype in the background yet had a set of messages waiting in persistent storage, e.g. a user forgetting to restart the app for some time after restarting his or her device, which occasionally occurred with the author's devices.

Figure 10b plots the cumulative probability distribution of the observed transfer rate for the 10.9% of the time when the scheduler determined a transfer was necessary, which spans three orders of magnitude from several tens of bytes per second to several megabytes per second. This demonstrates that the chosen transfer rate of $50kBps$ (50 bytes per millisecond) was an appropriately conservative estimate, larger than only 10.8% of observed transfer rates.

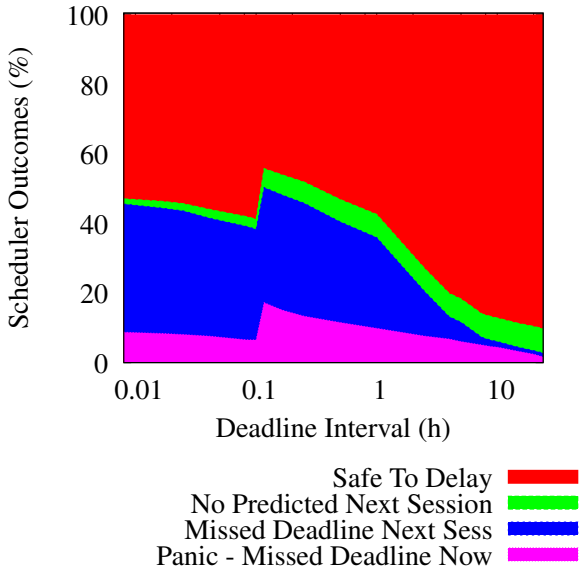
(a) $\Phi = 0.005$



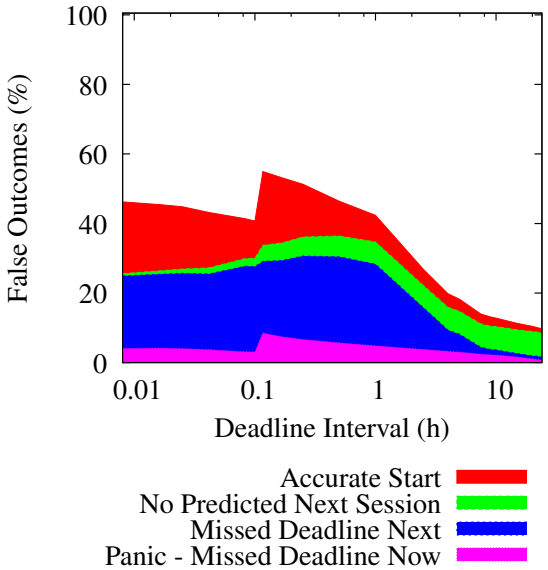
(b) $\Phi = 0.005$



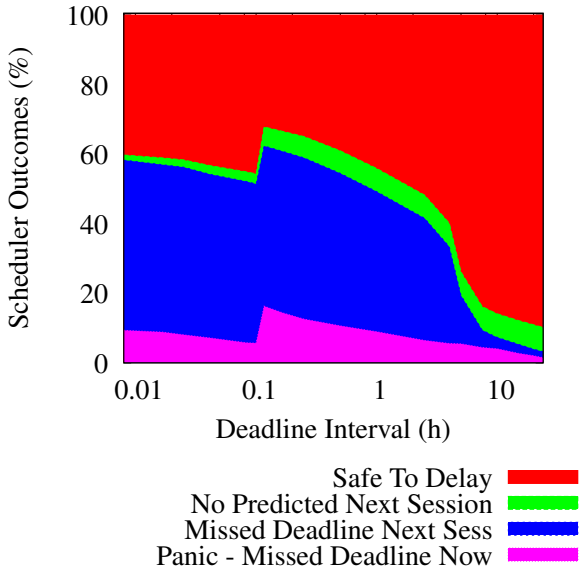
(c) $\Phi = 0.01$



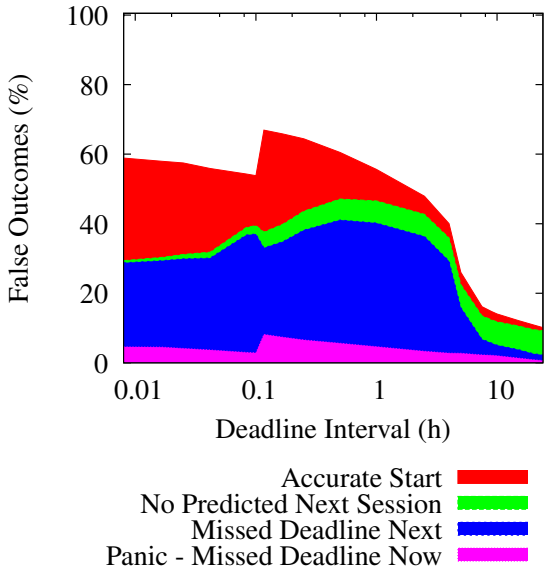
(d) $\Phi = 0.01$



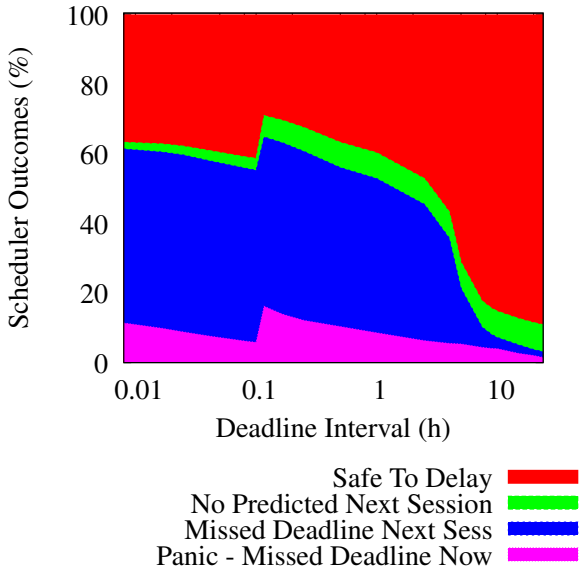
(e) $\Phi = 0.05$



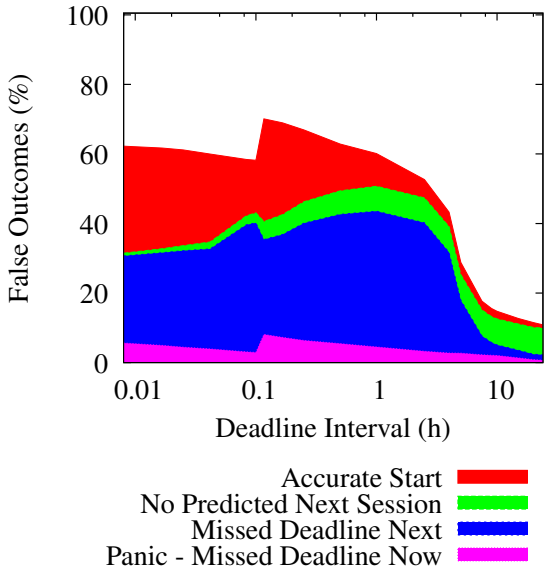
(f) $\Phi = 0.05$



(g) $\Phi = 0.2$



(h) $\Phi = 0.2$



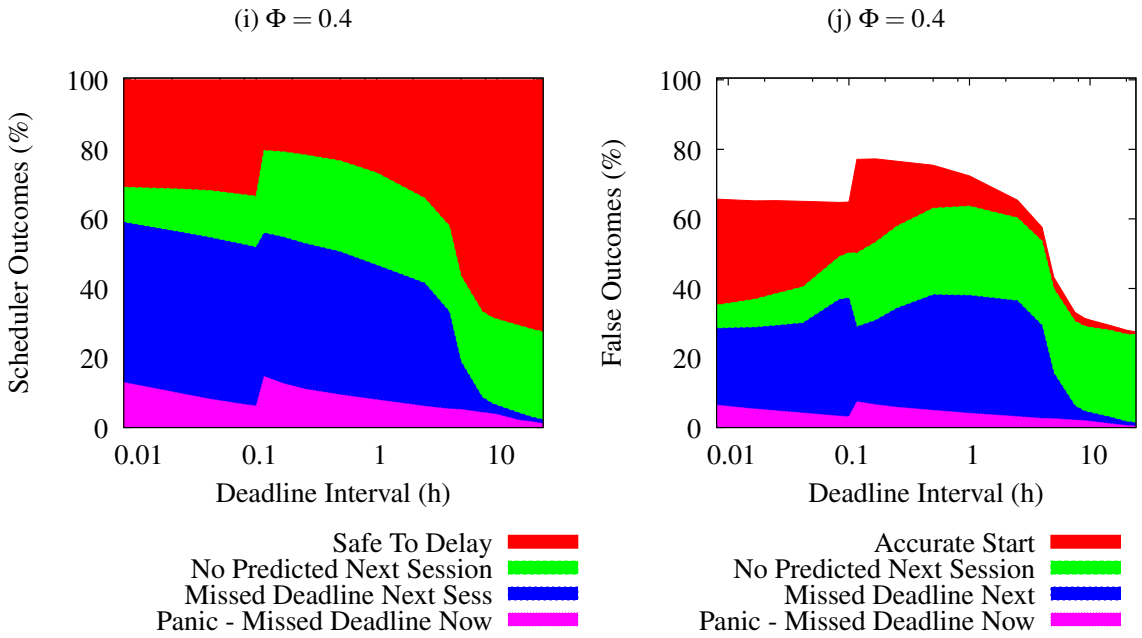


Figure 11: Scheduler Outcomes

Figures on the left plot the conditions that resulted in a delayed versus an immediate upload for deadlines ranging from 15 seconds to 24 hours and values of Φ at 0.005, 0.01, 0.05, 0.2, and 0.4. The drop shown at 6 minutes is a reflection of the implementation of the algorithm that chose to rerun the scheduler whenever a message was received whose deadline was less than 6 minutes. Figures on the right plot a best attempt at evaluating the predictions that caused an immediate upload in the past. If not an "Accurate Start," the corresponding condition was determined not to have come into fruition. But this determination lacked the full state of all the relevant past executions and the corresponding message queue, which was untenable to maintain in simulation.

4.3.2.1 Deadline and Φ Sensitivity The scheduling algorithm is sensitive to the behavior of its user; the demands of its application(s), which are expressed as message deadlines; and the parameter Φ. A number of factors cause the scheduler to delay an upload: (1) the prediction tree must have at least one child node, (2) there must be enough expected bandwidth along all paths in the tree to service the queue, (3) no future session can be expected to fail a constraint of any message, and (4) the current session cannot be failing a constraint of some message. Figure 11 plots, on a log scale, the percentage of scheduler executions that resulted in a delayed upload versus the conditions that caused an immediate upload across deadlines ranging from 15 seconds to 24 hours and for values of Φ in the set, $\{0.005, 0.01, 0.05, 0.2, 0.4\}$. In other words, it illustrates the five possible outcomes of the scheduler: the case where a delay is possible plus the cases where one of the above four conditions failed such that an upload was initiated. Note, however, that the second case described above—a lack of sufficient bandwidth to service the queue in an existing schedule—never occurred. Insufficient bandwidth was always due to the inability to make a prediction of the future.

A second figure accompanies each of these plots that illustrates a best attempt at accounting the conditions causing an immediate upload once a subsequent execution of the scheduler is started in the future. For example, at some session, A, the scheduler makes a prediction about a future session, B, that results in an upload during session A. At the time that B begins, the instrumentation compares the past prediction made during session A with the environment of session B. If the past prediction turns out to be true at the time session B begins, the past prediction is counted as *accurate*. Otherwise, a *false start* was counted for the respective past prediction. Each accompanying plot segments the percentage of occurrences where the condition of the past prediction resulted in an accurate or false start.

The cases where the scheduler ran multiple times within a given session (i.e. multiple times during session A) caused some difficulty in extracting this information due to a location event or a new message coming into the middleware. In these cases, the instrumentation would compare the current time with the deadline expected to be missed and wrongly conclude the previously transferred message could have been uploaded at the time of this execution. For example, the scheduler executes at the beginning of session, A, and initiates an upload. It then executes twice more during session, A, referred to as A' and A'', respectively. It is difficult to ascertain what the

scheduler would have predicted during the subsequent executions given that the message queue was emptied during the first execution. Furthermore, it may be the case that execution, A'', would have initiated an upload for a different reason that execution A. Thus, the instrumentation morphed from an evaluation of one past prediction in the current environment to the maintenance of a set of past contingencies of arbitrary, yet hopefully small, size upon which the conditions of the current environment are iteratively reevaluated.

The trace of a single user was used in these simulations, characterized by the session duration and inter-session time distributions shown above. These results juxtapose this user behavior with application demands of increasing severity. A given constraint, i.e. deadline, is onerous only when the environment is unable to support it. Thus, the results shown in figure 11 demonstrate what a specific deadline value means for the system in this environment. For all values of Φ, the system begins responding more aggressively once the deadlines drop below 9 hours, but it exhibits a linear degradation as larger deadline values decrease.

The value of Φ affects the size of the prediction trees. A larger value reduces the maximum number of children each node can have as well as the maximum depth of the tree; a smaller value allows for greater fanout and deeper trees. As such, a smaller tree offers fewer opportunities to push work into the future but provides greater certainty in predicted outcomes. A larger tree provides more opportunities to offload work into the future but with less certainty, and it's more susceptible to the accuracy distribution exhibited in figures 9a and 9b. Ultimately, the desire is to achieve a tree that most closely resembles the actual environment of the user. The figures demonstrate that larger values of Φ fail to include a sufficient number of predicted sessions where more executions are triggered due to a lack of a future predicted session with a high enough probability. However, the best performing value for this user was found not with the smallest value of Φ, which was 0.005, but with a value of 0.01, as shown in figures 11c and 11d. This was mostly likely due to extra paths included in the tree with a value of 0.005 that prevent the scheduler from delaying more often. For example, if the extra fanout allowed algorithm 1 to attach a node to the root of the tree where the corresponding session started well into the future such that some message deadline would be missed, the scheduler would be forced to initiate an upload immediately. By pruning those paths with a slightly larger value, the scheduler was able to delay more often and in a manner consistent with the user's environment.

The relatively low number of immediate upload decisions with deadlines above 9 hours illustrates the system's ability to capture the most repetitive, daily behavior of the user. It is reasonable to assume a high certainty that urban users come in contact with a WiFi access point at least once during waking hours. As the deadline reduced below this standard behavior, the scheduler decisions reflect a greater amount uncertainty that may or may not be inherent in user behavior but that result in less delayed uploads. In cases where this performance is desirable, more work is needed to target certainty that can be gleaned from specific attributes or indicators of the user's behavior.

Also, the abrupt change in performance at a deadline of six minutes demonstrates how profusely the system responds to a proactive reevaluation of the environment every time that a new message is received. This change is due to the artifact in the system implementation that chose to reevaluate algorithms 1 and 3 when a new message was received whose deadline was less than six minutes. In these trials of the simulation, the deadline of every message passed to the middleware initiated reevaluation of the schedule. Often these short deadlines expired prior to the start of a subsequent session. Without some decision to initiate an upload, the corresponding messages would be stored until some time after they likely expired. As such, a better implementation would make this choice of reevaluation a function of the minimum start time of any subsequent session and the amount of free, unscheduled bandwidth in the subsequent session along the minimum path in the schedule tree (see section 4.2.3).

Figure 12 illustrates the percentage of traffic uploaded where one or more messages in the queue missed a deadline. While this occurred infrequently, illustrated by the "Panic - Missed Deadline Now" sections of figure 11, figure 12 shows that between 20-60% of the traffic was uploaded under these circumstances. This occurred because a missed deadline was often preceded by a long period of disconnection, which could be due to less routine behaviors, such as long distance travel or periods when volunteers forgot to restart the test software when their device restarted. During occasions where the software was running, the system would amass a large number of messages. These were then uploaded in a burst once a new connection was formed. Had the implementation had access to network technologies other than WiFi, these results would have changed in two ways: (1) the additional connection opportunities may have allowed the system to mitigate the missed deadlines better, and (2) having the offending message deadline removed from the queue earlier would likely affect the total percentage of traffic that would ultimately

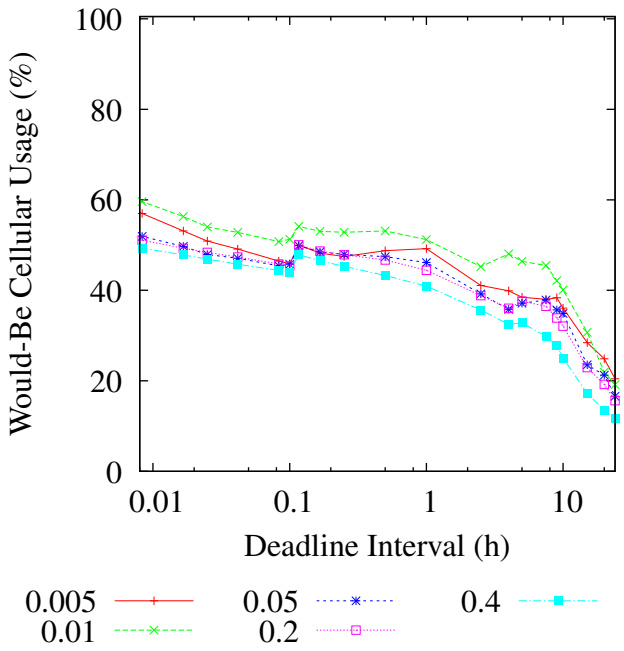


Figure 12: Percentage of Failed Constraints with WiFi

The percentage of traffic uploaded with failed constraints for each value of Φ, i.e. the messages transferred on the occasions when the scheduler determined a panic (that some message deadline was missed). This is the percentage of traffic uploaded during the "Panic - Missed Deadline Now" cases as illustrated in figure 11. Had the system had communications technologies available other than WiFi, such as 4G/LTE, some of this traffic would have been uploaded using connection opportunities that they would have provided, and, having serviced the messages whose deadlines would have been missed, a larger portion of the traffic may have been able to be uploaded using less costly means.

necessitate connection opportunities provided by 3G/4G/LTE technologies. And note that in the event of a deadline still being missed, the system would fall back to alerting the user or passersby for assistance.

4.4 CONCLUSION

This chapter presented a solution for the execution of network related actions for the tetherless care system. Chapter 3 outlined a space whereby the application developer can transfer responsibility for execution of such actions to the information delivery component of the system using a *message* abstraction to represent a set of actions to be executed in response to the situation that generated the message. A representation of the constraints for the execution of these actions accompanies each message, which was encoded as a deadline value within the implementation used in this chapter, where highly critical events would express shorter deadlines and less critical events longer ones.

The work of this chapter demonstrated the ability to execute these actions, i.e. transfer each message subject to its deadline, despite limited power availability and an intermittently connected network environment. Using the O(1) Markov risk analysis algorithm from chapter 3 to construct a tree of the likely future network opportunities, the system is able to accurately predict the user's regular behavior from a network perspective with an 80% accuracy to within tens of minutes. It then selects one from these predicted sessions over which to transfer a set of messages such that the constraints of each message are met. As such, the proposed solution is able to better manage the limited power and network opportunities available to the patient's smartphone by purposefully introducing delivery delay in pursuit of the lowest power opportunity to transfer data to its destination.

This work demonstrates the feasibility of the solution given a relatively light load on the system and relatively long deadlines on the order of several hours. More work is needed to stress the bounds of the presented algorithms and examine the session behavior variation from user to user. The observed predictions introduce a significant error into the schedule of future sessions. As the load on the system increases, this error is likely to impact the system's ability to deliver messages on time. Furthermore, the load imposed by a representative tetherless care application is needed,

if not a range of applications, to determine its likelihood of reaching the limits of the presented algorithm. Finally, where this study was limited to students in an urban environment with numerous WiFi access points available, an exploration of the algorithm's response to a diverse range of environments is needed.

5.0 A CASE STUDY: E-BUTTON AND PANDACARE

A shortcoming in the work of the previous chapter is the test application that did not identify and react to changes in its user's environment. Instead, multiple trials were simulated with varying fixed deadlines. A more realistic application would create messages whose constraints reflect the variation in important aspects of the environment. The work of this chapter sought to examine the system's performance on such an application.

The chosen application, called *PandaCare*, described in this chapter is originally developed for a device called the *eButton*, which is a specialized device under active research developed essentially, though not specifically, to fill the coordinator role of tetherless care. The similarity between the eButton and the coordinator provides a research avenue to compare the device with a contemporary smartphone both with and without the use of the tetherless care framework for one or more given applications. The work of this chapter focuses its interest on the target application and specifically on examining the network performance of several implementations of PandaCare on a smartphone. One implementation approximates PandaCare as it behaves on the eButton, and other implementations utilize the tetherless care framework. Utilizing the smartphone platform allows a comparison between the results of this chapter and those of the previous two chapters. And, by remaining consistent with the eButton's implementation, the results of this chapter should then be comparable to future work that utilizes the eButton platform with and without the tetherless care framework.

PandaCare assists healthcare professionals and family members with detecting behavior or health conditions that jeopardize the safety of dementia care patients. Since many dementia care patients are elderly, this includes general health abnormalities and risky events, such as falls. It also includes confusion-induced wandering that may take the patient beyond the confines of a home, care facility, or other well known venue.

As such, PandaCare must, in real-time, monitor the basic vital signs, body movements, and location of one or more patients who is under the care of a staff of one or more healthcare professionals or family members. A set of algorithms must detect abnormal vital signs, movements or rapid body orientation changes that indicate a fall, and unexpected exits from some set of venues. And, when a significant event occurs, i.e. a fall or unexpected exit, staff or family must be alerted to the event within a matter of seconds to effect a quick response.

In the remainder of this chapter, section 5.1 describes the necessary sensing hardware for PandaCare. Section 5.2 outlines PandaCare in terms of the features, states, actions, and messages required to carry out its operation. Where the implementation running on the eButton does not formally or architecturally define these objects, it must still perform the computation of them such that the discussion remains an accurate model of the eButton software. Section 5.3 first describes the intended implementation of PandaCare on the eButton followed by a discussion of the three other implementations that approximate its functionality. Section 5.4 describes the methodology for evaluating the three implementations. Section 5.5 presents the results of the evaluation. And Section 5.6 concludes the chapter.

5.1 PANDACARE SENSORS

Three classes of sensors are needed to support PandaCare: one that encompasses all the possible sensors that could inform and support the vital sign monitoring, one that encompasses the sensors that could inform the fall detection, and one that defines the sensors that could support location monitoring.

Given that both the author and the eButton team are not medical professionals, the optimal choice of vital sign sensors and the algorithms that identify nuances in the patient state via them is left for future work. The implementations below utilize pulse and skin temperature sensors.

The pulse sensor produces an analog voltage corresponding to the amount of light reflected into the sensor. An observable spike in the voltage corresponds to each heartbeat of the patient. The sensor periodically creates a 16-bit sample of this voltage where the value of the sample, which

ranges of 0-65,535, is proportional to the voltage value in the range of possible voltages, typically zero to five volts.

The skin temperature sensor produces a similar analog voltage signal where the voltage varies with the temperature of and around the sensing hardware, which, when in contact with the skin, is the skin temperature. A 16-bit digital sample is produced from this voltage value in a similar manner where the proportions of the voltage and the sample within their respective ranges are equivalent.

Fall detection is most commonly supported by a set of accelerometers and gyroscopes. For the eButton implementation, a single triaxial accelerometer and a single triaxial gyroscope are used. The accelerometer measures the magnitude of the force applied to the device in each of three specific directions, and the gyroscope measures the angular velocity of the device about each of three specific axes. Each of the three directions or axes are oriented in a perpendicular fashion to measure the three dimensional component values of their respective signals. Extra hardware computes a 32-bit floating point value from the digital samples in m/s^2 or m/s, respectively.

Location monitoring can be achieved in a number of ways utilizing several different sensors or hardware devices. A GPS sensor observes the broadcasted signal from geosynchronous satellites while outdoors. Its accompanying hardware ultimately produces three values: a latitude, a longitude, and an altitude that correspond to the device's location. Databases on the Internet, such as SkyHook Wireless [10], store latitude and longitude values of a large number of WiFi access points. A given device can use the MAC address of one or more nearby access points, perform a lookup in one of these databases, and conclude the device must be within a certain radius of the queried location. By a similar notion, cell towers provide information about the tower's location to currently connected communication devices, which could also be used to approximate the device's location.

A barometric pressure sensor measures the air pressure around the sensor. It produces an analog voltage value within some range, again usually zero to five volts, where the produced voltage is proportional to the measured air pressure relative to a range of values to which the sensor is sensitive, i.e. 300-1100 hPa. Because this measurement correlates with it, a barometric pressure sensor can be used to determine the device's altitude.

A camera is sensitive to the light energy in front of its lens, producing a two dimensional array of digital samples representing a projection of the three-dimensional space in front of the camera sensor. A number of algorithms, such as the scale invariant feature transformation algorithm (SIFT [53]), can be used to identify objects or markers in a given image.

5.2 TETHERLESS PANDACARE

The work below outlines PandaCare in terms of the samples, features, states, actions, and messages that are required to carry out the mission of the application given the sensors described above.

Let the values in table 2 represent samples from the sensors described above generated at time, t. At each time step, the set $V \subseteq \{p_t, s_t, a_{t0x}, a_{t0y}, a_{t0z}, \ldots, a_{tKx}, a_{tKy}, a_{tKz}, g_{t0p}, g_{t0r}, g_{t0y}, \ldots, g_{tMp}, g_{tMr}, g_{tMy}, lat_t, lon_t, alt_t, b_t, c_t, mac_{t0}, \ldots, mac_{tN}, ss_{t0}, \ldots, ss_{tM}\}$ is added to the context, C, where K accelerometers are available to the system, M gyroscopes are available, and N WiFi access points are currently within range of the WiFi radio of the device.

Table 2: Types of PandaCare Samples

p_t	Pulse waveform sample
s_t	Skin temperature
a_{tix}	Accelerometer, i, in the x direction
a_{tiy}	Accelerometer, i, in the y direction
a_{tiz}	Accelerometer, i, in the z direction
g_{tip}	Gyroscope, i, pitch
g_{tir}	Gyroscope, i, roll
g_{tiy}	Gyroscope, i, yaw
lat_t	Latitude
lon_t	Longitude
alt_t	Altitude
b_t	Barometric pressure
c_t	Camera image
mac_{ti}	MAC address of WiFi access point, i
ss_{ti}	Signal strength of WiFi access point, i
mac_c	MAC address of currently connected access point
$ssid_c$	SSID of currently connected WiFi network
lac	Location Area Code from associated cell tower
n_3	state of the 3G radio
n_4	state of the 4G radio
n_L	state of the LTE radio

5.2.1 Features

Given the context, C, as described above, the relation, R, includes the mapping, $\{C, F\}$, where F is a set of features of the types listed in table 3. This section describes the set of algorithms needed to accomplish this mapping.

Algorithm 6: Heart Rate Computation

Computes a beats-per-minute value from the last n sets of samples added to the context, C.

Require: threshold Π, context C, current time t, time step count n
  $count \leftarrow 0$
  $p_{prior} \leftarrow -\infty$
  for all i in $0 \ldots n$ **do**
    Let $V_{t-i} \in C$
    Let $p_{t-i} \in V_{t-i}$
    Let $t' \in V_{t-1}$
    if $i = n-1$ **then**
      $t_{start} \leftarrow t'$
    end if
    if $p_{t-1} > \Pi$ and $p_{prior} < \Pi$ **then**
      $count \leftarrow count + 1$
    end if
    $p_{prior} \leftarrow p_{t-i}$
  end for
  $f_{pulse} \leftarrow count \div (t - t_{start})$

The peak detection algorithm shown in algorithm 6 examines the pulse waveform over a recent time interval, performs a simple peak detection algorithm that determines each time the waveform rises above a certain threshold, and computes a beats-per-minute value, which is assigned to the pulse feature.

The skin temperature sensor defines a minimum temperature, min_t, to which it's sensitive that corresponds to a zero voltage and zero digital sample value along with a maximum temperature, max_t, that corresponds to a maximum voltage and digital sample value. Assuming the minimum digital sample is zero, the skin temperature, f_{temp}, can then be computed from the reported digital sample value, s_t, with the following equation:

$$f_{temp} = \frac{max_t - min_t}{65535} s_t + min_t \tag{5.1}$$

Table 3: PandaCare Features

f_{pulse}	Pulse in beats per minute
f_{temp}	Skin temperature in Celsius or Fahrenheit
a_i	Acceleration vector for accelerometer i
$a_{stationary}$	Accelerometer, i, stationary
$a_{freefall}$	Accelerometer, i, free fall
a_{impact}	Accelerometer, i, impact
g_v	Angular velocity vector for Gyroscope, i
g_a	Gyroscope, i, angular acceleration
g_d	Gyroscope, i, angular displacement
ℓ_{gps}	Location vector from GPS
h_b	Altitude computed from barometric pressure
c_j	Camera image feature, j
$scan$	$\{\langle mac_t, ss_t\rangle_0, \ldots, \langle mac_t, ss_t\rangle_N\}$ vector of MAC address and signal strength for each access point
ℓ_{wifi}	Location vector provided by access point location database
f_{lac}	LAC from cell tower
w_{wifi}	SSID and MAC address of currently connected network
w_{3G}	3G network connected
w_{4G}	4G network connected
w_{LTE}	LTE network connected

The accelerometer measures the three component values of the acceleration applied to the device. Each of the instantaneous values are combined into the a_i acceleration vector feature. Remaining features of this data depend upon the magnitude of this vector defined as $sum = \sqrt{a_{tix}^2 + a_{tiy}^2 + a_{tiz}^2}$. At rest, this magnitude is expected to be $9.8m/s$, to which the accelerometer associates the $a_{stationary}$ feature. In free fall, the magnitude drops significantly such that a minimum threshold can indicate free fall whenever magnitude drops below it. Similarly, the magnitude crossing above some defined maximum threshold can determine the impact feature.

The gyroscope measures the three component values of the angular velocity applied to the device, i.e. the *pitch*, *roll*, and *yaw*, and features of this data can be computed in a similar manner to the accelerometer. The g_v feature assembles the component angular velocities into an angular velocity vector. Fall detection depends on three aspects of this vector: it's magnitude, it's rate of change, and the total angular displacement from sample to sample. The magnitude is computed with the pythagorean theorem as with the accelerometer. For a gyroscope, i, given two temporarily adjacent sets of samples, V_x and V_{x-1}, such that $\{g_{xip}, g_{xir}, g_{xiy}, t_x\} \subseteq V_x$ and $\{g_{(x-1)ip}, g_{(x-1)ir}, g_{(x-1)iy}, t_{x-1}\} \subseteq V_{x-1}$, the instantaneous angular acceleration can be computed as:

$$g_a = \left\langle \frac{g_{xip} - g_{(x-1)ip}}{t_x - t_{x-1}}, \frac{g_{xir} - g_{(x-1)ir}}{t_x - t_{x-1}}, \frac{g_{xiy} - g_{(x-1)iy}}{t_x - t_{x-1}} \right\rangle$$

and the instantaneous angular displacement can be computed as:

$$g_d = \left\langle \frac{g_{xip} + g_{(x-1)ip}}{2} \times (t_x - t_{x-1}), \frac{g_{xir} + g_{(x-1)ir}}{2} \times (t_x - t_{x-1}), \frac{g_{xiy} + g_{(x-1)iy}}{2} \times (t_x - t_{x-1}) \right\rangle$$

The altitude, h_b, can be computed from the barometric pressure using the following equation:

$$h_p = \frac{RT}{gM} log_e(p_0/b_t) \tag{5.2}$$

where R is the gas constant ($8.3144621\ \frac{J}{mol*K}$), T is the temperature, g is the acceleration due to gravity, M is the molar mass of air, p_0 is the atmospheric pressure at sea level, and b_t is the current atmospheric pressure. Many commercially available devices treat the air temperature, T, as a constant. By doing the same in PandaCare, the altitude feature, h_p, is a function of and solely dependent upon any given barometric pressure sample. [41]

The the feature, ℓ_{gps}, is a simple composition of the GPS latitude, longitude, and altitude values. Likewise, the *scan* feature is a simple composition of the available mac_{ti} and ss_{ti} samples.

Features of each camera image include the scale invariant features produced by various image feature detection and extraction algorithms, such as SIFT [53]. Each of these features is a collection of points that represent and uniquely identify an object in a given image. They are computed through a process of convolving the image with Gaussian blurs at different scales, performing a blob detection algorithm to extract a representative set of *keypoints* for the image's object, and filtering out the noisy keypoints. The exact implementation of this algorithm is left for future work given that it is a requirement of the eButton-specific implementation of PandaCare.

The WiFi hardware scans the wireless environment around the device and reports at least the SSID, MAC address, and signal strength of each wireless access point in the vicinity that broadcasts a beacon. By submitting this data to Internet-based lookup services, a latitude and longitude value can be retrieved that corresponds to the location of the device with some degree of accuracy, which are then properties of the ℓ_{wifi} feature.

5.2.2 States

Given the set of features, F, described above, the relation, Q, defines the mapping, $\{F, I\}$,where I is a set of states of types described in the following subsections, which each focus on one of the three monitoring functions of PandaCare.

5.2.2.1 Vital Sign Monitoring

The vital sign monitoring function relies on the following states:

$s_{p,h}$	Pulse High
$s_{p,0}$	Pulse Normal
$s_{p,l}$	Pulse Low
$s_{p,-1}$	Pulse Invalid
$s_{t,h}$	Skin Temperature High
$s_{t,0}$	Skin Temperature Normal
$s_{t,l}$	Skin Temperature Low
$s_{t,-1}$	Skin Temperature Invalid

A set of thresholds are defined for each patient that partition the space of possible pulse and skin temperature values into regions, where each region corresponds to one of the above states. Both features express a single 32-bit floating point value that can range from 1.18×10^{-38} to 3.4×10^{38}. The feasible values of the pulse range from 0-250 beats per minute, and the feasible values of the skin temperature range from roughly $-80°$F to 150°F. Thus, any floating point value beyond the bounds of these two ranges correspond to their respective *invalid* state. Given that the normal, resting pulse rates range from 30 to 100 beats per minute depending on the physical fitness of the individual, the bounds of the partition corresponding to the *normal* state is defined for each patient individually. For example, if a given patient's resting heart rate tends to be around 60 beats per minute, the normal range could be defined between 50 and 70 beats per minute. The *low* state would then correspond to the range below this, from 0 to 50 beats per minute, and the *high* state above it, from 80 to 250 beats per minute.

Also, note that, given that PandaCare is targeted at the elderly, any strenuous activity that raises their heart might be cause for getting the attention of caregivers, such that the application need not apply a partitioning in a context-dependent manner. Conversely, certain locations could be defined where a *normal* state corresponds to a wider range of pulse rates. This improvement, however, is left for future work.

5.2.2.2 Fall Detection

The fall detection functionality relies on the following states:

$s_{f,free}$	In Free Fall
$s_{f,down}$	Fallen
$s_{f,0}$	Normal

Algorithm 7 describes a state machine for computing the fall state from the series of feature sets related to the context by relation, R. A fall exhibits a large spike in the magnitude of the angular velocity such that its value crossing some defined maximum threshold can indicate the start of a potential fall. Conversely, the accelerometer expresses a drop in the magnitude of the acceleration vector, defined by the $a_{freefall}$ feature.

These features trigger a transition to the the $s_{f,free}$ state. For sets of features when in this state, the algorithm adds the angular displacement, g_d, to a total displacement and compares the angular

Algorithm 7: Fall State Computation

Describes a state machine that computes the fall detection states and actions to relate to each set of features.

Require: relation R, angular velocity fall threshold G_v, angular acceleration fall threshold G_a, maximum angular displacement fall threshold G_d, minimum angular displacement return threshold $G_{d'}$, max fall interval t_{max}

$fallState \leftarrow s_{f,0}$
$d \leftarrow 0$
$over_a \leftarrow 0$
$t_s \leftarrow 0$
for all $t \in \mathscr{T}$ **do**
  for all F_t in R **do**
    if $fallState = s_{f,0}$ **then**
      if $a_{freefall} \in F_t$ or $F_t.g_v \geq G_v$ **then**
        $d \leftarrow F_t.g_d$
        $fallState \leftarrow s_{f,free}$
        $t_s \leftarrow F_t.t$
        if $F_t.g_a \geq G_a$ **then**
          $over_a \leftarrow 1$
        end if
      end if
    else if $fallState = s_{f,free}$ **then**
      $d = d + F_t.g_d$
      if $F_t.g_a \geq G_a$ **then**
        $over_a \leftarrow 1$
      end if
      if $(a_{stationary} \in F_t$ and **not** $over_a$ **and** $d < G_d)$ **or** $F_t.t - t_s > t_{max}$ **then**
        $fallState \leftarrow s_{f,0}$
        $d \leftarrow 0$
      else if $a_{impact} \in F_t$ **or** $(over_a$ **and** $d \geq G_d)$ **then**
        $fallState \leftarrow s_{f,down}$
      end if
    else if $fallState = s_{f,down}$ **then**
      $d = d + F_t.g_d$
      if $displace \leq G_{d'}$ **then**
        $fallState \leftarrow s_{f,0}$
        $d \leftarrow 0$
        $over_a \leftarrow 0$
        $t_s \leftarrow 0$
      end if
    end if
  end for
end for

acceleration against a defined maximum threshold. Once the total displacement crosses above a maximum displacement threshold, G_d, and a spike in the angular acceleration was found in a prior set of features or the accelerometer expresses an a_{impact} feature, the algorithm transitions to the $s_{f,down}$ state.

As the patient gets back up, the total angular displacement reduces below some minimum threshold, $G_{d'}$, which transitions the state back to $s_{f,0}$.

5.2.2.3 Location Monitoring

Location monitoring relies on the following states:

v_t Venue

v_a Authorized Venue

v_u Unauthorized Venue

A *Venue* is a state with several properties, $\langle \ell_1, \ell_2, \ell_3, \ell_b, \ell_c \rangle$, where:

- ℓ_1 is an LAC,
- ℓ_2 is a *scan* feature, referred to below as the *wifi signature*,
- ℓ_3 is the tuple of the latitude, longitude, and altitude from the GPS,
- ℓ_b is an altitude value resulting from the barometric pressure computation, and
- ℓ_c is a set of camera image features.

The system maintains a database of known venues, into which a current venue is matched on a regular basis. For each patient, a predefined set of venues are specified as authorized for the patient, which correspond to the locations in which the patient is expected to be. The current venue matching one of these results in the *authorized venue* state. The current venue matching any other venue results in the *unauthorized venue* state.

The the properties of the current venue are set from the corresponding features produced by the relation, R. Not all the properties are necessary to uniquely identify a venue, and not all properties may be available at each venue. Also, each of the implementations examined in this chapter utilize different algorithms that acquire these properties with different priorities and each rely on a different subset of the properties.

Both the wifi signature and camera image features act as a signature of the venue. The system can identify the current venue by computing the Tanimoto coefficient between the current wifi signature and those of venues stored in the database, and, if the coefficient is greater than 0.7, the two venues are considered to be the same location [18]. Similarly, SIFT [53] computes the correlation between the current venue's camera image features and those of the venues stored in the database, where again, if the correlation is above a defined threshold, the venues are considered to be the same location.

An improvement, which is left for future work, is an algorithm for assessing and describing the motion of the patient so as to determine if the movement is indicative of a directed, conscious trip towards a likely destination or a confused, meandering wander. The former case may relax the constraint that the patient strictly remain within authorized locations, whereas the latter case may be cause to proactively alert caregivers or family members even as the patient remains within authorized locations. Accomplishing this would require the system to maintain a history of prior venues upon which analysis could operate. As such, algorithm 1 could likely be appropriated for this task.

5.2.3 Actions and Messages

The primary purpose of PandaCare is to alert family members and caregivers to important events that occur with the patient, of which three specific events have been defined: abnormal vital signs, falls, and unauthorized exits. As such, the following alert messages are defined to correspond with each:

a_h Health condition alert

a_f Fall alert

a_v Unauthorized exit

Prior to generating and sending an alert, the system may engage in a dialog with the patient to gather more information, further assess the situation, or provide feedback. For this purpose, the following actions are necessary:

a_{down} Prompt “Have you been injured?”

a_{calm} Prompt user to calm himself or herself

a_{ret} Prompt return to authorized locations

The events for which an alert could be associated have varying degrees of severity. Specifically, a fall is severe in proportion to the patient behavior after the event and the feedback he or she provides to the system. If the patient responds to a prompt positively and starts righting herself, an alert may not need to be generated. However, if the patient fails to respond to a prompt at all and remains lying down, the event then has a greater severity.

A pulse or skin temperature transitioning to their respective low or high states triggers the corresponding alert message. Both a fall and an unauthorized exit trigger a protocol for first interacting with the patient to get feedback about the situation or direct the patient's actions, such as prompting him or her to return to a safe area. If this initial interaction fails to alleviate the situation, a health condition or unauthorized exit alert is created. The exact algorithm for this protocol is left for future work.

Finally, for the tetherless care implementation, the following actions provide feedback to the virtual sensors involved in location monitoring:

a_{L1} Monitor the cell tower location only

a_{L2} Acquire one or more WiFi scans, query Internet location databases

a_{L3} Acquire GPS location

5.3 IMPLEMENTATIONS

The eButton and its implementation of PandaCare offer a comparable system with which to compare tetherless care. However, at the time of this study, the eButton is still under active hardware development such that its networking functionality is limited. As such, the choice was made to mimic the eButton and PandaCare on a smartphone as closely as possible. Thus, the competing network performance of the two software implementations could be compared on a uniform hardware platform.

Four implementations of PandaCare, referred to as the *eButton*, *original*, *tetherless*, and *naive* implementation, respectively, are described in the following subsections. The first, discussed in section 5.3.1, refers to the original design and function of PandaCare on the eButton system. The remaining three implementations describe efforts to port the eButton implementation to the Android Droid Pro running the Android OS, each designed to isolate and differentiate specific aspects of the system behavior.

5.3.1 eButton and PandaCare

Dementia care patients are outfitted with an Arduino-based wristband equipped with the skin temperature sensor, pulse sensor, and a ZigBee radio. The wristband form factor provides the needed contact with the skin to acquire pulse and temperature samples and is comfortable enough to be worn for extended periods of time. This platform complements the eButton, which is worn on the patient's chest and houses the GPS, accelerometer, gyroscope, barometer, camera, WiFi radio, ZigBee radio, and an ARM-based processor. Each patient is also issued a *patient-side terminal*, which is a PC dedicated to one patient residing in the patient's home or private room. Healthcare professionals interact with patient information through a desktop workstation and web browser that connects to a web server running on the patient-side terminal; family members are provided a similar mobile web display.

The wristband Arduino executes the peak detection algorithm to calculate the pulse in beats per minute along with the unit transformation computation for the skin temperature sensor. The eButton and wristband engage in a protocol to transmit the resulting values from the wristband to the eButton.

This action is driven by the eButton, which collects samples from the two wristband sensors, GPS, accelerometer, gyroscope, barometer, and WiFi every three seconds and from the camera every fifteen seconds.

The responsibility of computing R, Q, and M is divided between the eButton and the patient-side terminal. In general, the eButton computes the features specified above in subsection 5.2.1 in real-time and transmits them to the patient-side terminal every three or fifteen seconds.

The patient-side terminal maintains the database of venues indexed along several dimensions. Outdoor venues are determined by a GPS location vector feature, which is produced every three seconds. Since data from the GPS sensor is not available indoors, the venue is queried using the last received GPS location vector feature, the barometric altitude feature, the camera image feature, and the WiFi scan feature. The last received GPS location vector feature is used to look up a building into which the patient has gone. The barometric altitude feature indexes the floor on which the patient currently resides. The WiFi scan indexes a set of rooms on the floor from where a similar set of access points are in range. And, finally, the camera image features are compared with those extracted from images taken in each room.

Given the ease with which the GPS location vector can be compared against a set of authorized venues, this task is performed on the eButton. When the comparison indicates that the patient is outside any authorized venue, the eButton generates an alert for caregivers and family members. The indoor location monitoring algorithm is performed on the patient-side terminal in a similar manner. As such, the two platforms share responsibility for Q and M.

The eButton utilizes a simplified fall detection algorithm from that discussed above in subsection 5.2.2.2. The stationary, free fall, and impact features of the accelerometer are not computed. Instead, the eButton maintains a brief history of the acceleration vector, which indicates the downward direction in most instances. For each new acceleration vector feature, the eButton computes the angle between the new feature and the oldest in the history. If the change in angle in the downward direction is above a certain threshold, it concludes a fall has occurred, in which case an alert is generated for caregivers and family members.

With respect to vital sign monitoring, the state of the patient and the necessary response to it are computed as described above in subsection 5.2.2.1 on the patient-side terminal.

Two assumptions heavily influence this design and implementation. First, the wireless connection between the eButton and patient-side terminal is assumed to be constant, reliable, and low-latency. As such, only a brief amount of time is needed for the transmission of a sample or feature from the eButton to the patient-side terminal. Thus, it can be assumed this small delay will never impact the patient's safety. Second, the energy requirements of the sensors, wireless radios, and processor are assumed to be sufficient for the longevity that PandaCare requires. This

renders the transmission of every sample or feature from eButton to patient-side terminal trivial and inconsequential.

Furthermore, these assumptions combined ignore the need to define the severity of the patient's situation and determine the real-time deadlines that codify the interval of time in which a response must complete so as to ensure the patient's safety. If a critical event of any severity is always accompanied by a low-latency wireless connection to other fixed infrastructure and an infinite amount of energy, an alert, or the samples that would ultimately cause an alert, is always readily delivered to the appropriate caregivers.

5.3.2 Original PandaCare Implementation

At the time that this work was completed, the eButton and its wristband were under active hardware development, rendering it unsuitable for examining the performance of tetherless care compared to the implementation described above. As such, the above implementation of PandaCare was recreated on an Android Droid Pro smartphone as faithfully as possible. The focus in doing so is to accurately mimic the network traffic between the smartphone and the patient-side terminal.

The smartphone sensors were limited to an accelerometer, GPS, camera, and WiFi, requiring simulated data for the pulse, skin temperature, gyroscope, and barometer. Furthermore, the Android operating system only energized the accelerometer hardware while the device's screen was illuminated, preventing a continuous sampling of its user's motion. Also, capturing an image from the camera sensor required the user to interact with a view finder user interface element, which prevented any practical use of it for the purpose of this study.

To overcome these, the software simulated a mock sample every three seconds for the pulse, skin temperature, gyroscope, and barometer. The accelerometer was sampled during the moments the user turned on the screen, and, when sampling forcibly ended, the final sample obtained was repeatedly used as the current sample in the software. Also, a series of images were stored on the device, and the software loaded one, in turn, every fifteen seconds to act as a camera image sample.

No model of patient data or behavior was used to generate the mock samples, which is left for future work that depends on the completion of the eButton. As such, the values of the mock pulse,

skin temperature, barometer, gyroscope, and camera samples were chosen so as to prevent alert conditions in subsequent computation.

Given that the absence of the barometer and camera prevented a lookup of the floor and room in determining a venue, the set of authorized venues were defined as a set of square, two-dimensional geofences against which the GPS coordinates could be compared to determine if the patient was safe. An exit from this defined space then generated an alert event.

Despite these, albeit numerous, limitations, the original implementation is capable of monitoring its patient's location and creating a network traffic profile that accurately models that of PandaCare running on a fully developed eButton. The messages generated by the application, destined for the patient-side terminal, are summarized below:

Table 4: Types of Messages in the Original Implementation

Message Type	Period/Event
Fall Alert	Fall Event
Unauthorized Exit Alert	Exit Event
Barometer	$3s$
GPS	$3s$
Body Temperature	$3s$
Pulse	$3s$
Accelerometer	$3s$
Gyroscope	$3s$
Camera Image	$15s$

5.3.3 Tetherless PandaCare Implementation

The eButton and original implementations discussed above illustrate the typical assumptions and design decisions that violate the requirements of tetherless care. First, they assume an always available, reliable, and performant wireless network to support the communication between the eButton and the patient-side terminal. Second, they ignore the energy constraints of the device,

which is an assumption that there exists at least a reasonable amount of energy for the everyday operation of the device.

These two design decisions also illustrate the logical separation in the two goals of developing a tetherless care application: (1) the primary focus on monitoring patient health, and (2) the necessity to accommodate the device's computing and networking constraints. And it is this logical separation that contributed to the design of the tetherless care architecture presented in chapter 3.

The assumptions also complicated a comparison with the tetherless care implementation, which must conform to the constraints of tetherless care with respect to energy and intermittent connectivity. This required that the computation of Q and M be located on the smartphone instead of a patient-side terminal or other server. Furthermore, another location monitoring algorithm was needed to reduce its storage and computation demands so as to accommodate the capabilities of the Android Droid Pro. This change introduced another variable that affected the performance of the respective systems. This flaw was tolerated to accommodate a comparison with a completed eButton and its software in the future. The *naive* implementation described below provides a more direct comparison with the tetherless care implementation, varying only the information delivery component in use.

The tetherless care implementation is also implemented on the Android Droid Pro; absent of the wristband, it's accompanying sensors, the eButton's gyroscope, and the eButton's barometer; and with the limitations of the accelerometer and camera. The function of the patient-side terminal is divided among the smartphone and a central server that serves multiple patients in multiple homes or facilities.

The virtual sensors and state descriptors that comprise the application of the tetherless care implementation are listed in table 5 with the samples, features, states, and actions in which each is interested and which each produces.

The FallSensor virtual sensor and FallState state descriptor implement the simplified fall detection algorithm described above in section 5.3.1. The FallSensor receives the individual accelerometer and gyroscope samples and produces the corresponding accelerometer vector, gyroscope vector, stationary, free fall, and impact features. The FallState receives these features, maintains a history of the acceleration and gyroscope vectors, determines when the patient has fallen as previously

Table 5: Major Software Components of PandaCare

Class	Interests	Produces
Virtual Sensors		
FallSensor	$\forall_i a_{tix}, a_{tiy}, a_{tiz}$	a_i, $a_{stationary}$, $a_{freefall}$, a_{impact}
	$\forall_i g_{tip}, g_{tir}, g_{tiy}$	g_a, g_v, g_d, $g_{stationary}$, $g_{freefall}$
BarometerSensor	b_t	h_b
LocationSensor	lat_t, lon_t, alt_t	ℓ_{gps}
	lac	f_{lac}
	$\forall_i mac_{ti}, ss_{ti}$	ℓ_{wifi}
NetworkSensor	$\forall_i mac_{ti}, ss_{ti}$; mac_c, $ssid_c$, n_3, n_4, n_L	$scan$, w_{wifi}, w_{3G}, w_{4G}, w_{LTE}
BodyTempSensor	s_t	f_{temp}
PulseSensor	p_t	f_{pulse}
State Descriptors		
FallState	a_i, $a_{stationary}$, $a_{freefall}$, a_{impact}	$s_{f,free}$, a_f, a_{down}
	g_a, g_v, g_d, $g_{stationary}$, $g_{freefall}$	$s_{f,down}$, $s_{f,0}$
NetworkState	$scan$, v_t	Sessions
LocationState	ℓ_{gps}, ℓ_{wifi}, f_{lac}	v_t, a_{L1}, a_{L2}, a_{L3}
AuthVenueState	v_t	v_a, v_u, a_v, a_{ret}
VitalSignState	f_{temp}, f_{pulse}	$s_{p,h}$, $s_{p,0}$, $s_{p,l}$, $s_{p,-1}$, $s_{t,h}$, $s_{t,0}$, $s_{t,l}$, $s_{t,-1}$, a_h

The virtual sensors and state descriptors that comprise the PandaCare tetherless care application. The samples, features, and states in which each is respectively interested or which each respectively produces is also listed. For brevity, the *type*s associated with each sample, feature, and state are not specified.

described, produces the prompt action asking if the patient has been injured, and produces an alert message for caregivers and family members when a fall has occurred.

The NetworkSensor and NetworkState implement the algorithm described in section 4.2. The NetworkSensor receives the MAC addresses and signal strengths of nearby access points along with samples indicating the state of the available wireless transceivers, such as WiFi, 3G, 4G, and LTE. The NetworkState defines each session, maintains a recent history of sessions, and implements algorithm 1 over the history to predict likely future ones as described in chapter 4.

The LocationSensor virtual sensor and LocationState state descriptor implement the three tiered location sensing algorithm by Chon et al. [18]. The LocationSensor receives the lac, mac_{ti}, ss_{ti}, lat_t, lon_t, and alt_t samples, programmatically controls the sampling of each, and produces the f_{lac}, ℓ_{wifi}, and ℓ_{gps} features. From these, the LocationState assembles the current venue state, v_t. Once the current venue is determined, the LocationState produces the a_{L1} action to direct the LocationSensor to power down the WiFi and GPS hardware, and any future change in the f_{lac} feature causes the LocationState to produce the a_{L2} action to begin acquiring samples from the WiFi-based location sensing. Also, the LocationState periodically produces an a_{L2} action to acquire a more accurate location.

These actions are expected to be followed by an ℓ_{wifi} feature in the near future. Once received, the ℓ_{wifi} feature is compared with the current venue by computing their Tanimoto coefficient. If they match, the LocationState produces the a_{L1} action to once again put the system in a more dormant state. If they differ, the patient has moved to a new venue, and the feature is compared with a history of known venues, again by computing the Tanimoto coefficient with each venue. If a match is found, the current venue is updated to the matching previous venue and the a_{L1} action is produced.

When no ℓ_{wifi} feature is received within an interval of time after the a_{L2} action is produced or when no matching venue is found for a given ℓ_{wifi} feature, the LocationState produces the a_{L3} action to direct the LocationSensor to acquire a set of samples from the GPS hardware. An ℓ_{gps} feature is expected to follow this action within a given time interval. Once received, the LocationState updates the current venue with the f_{lac}, ℓ_{wifi}, and ℓ_{gps} features and stores it in the history.

In addition to alert messages, the tetherless care implementation generates several messages as a result of its instrumentation. The full list of messages created by this implementation is below:

Table 6: Types of Messages in the Tetherless PandaCare Implementation

Message Type	Event
Fall Alert	Fall Event
Unauthorized Exit Alert	Exit Event
Session End Report	End of Session
Scheduling Report	Start of WiFi Session
Connection Prediction Report	Start of WiFi Session
Disconnection Prediction Report	End of WiFi Session
Transfer Report	End of Transfer
New Venue Report	Stationary at new location
Enter Venue Report	Stationary at location
Exit Venue Report	Leaves stationary location
New Access Point	After WiFi Scan

5.3.4 Naive Implementation

The naive implementation replicates the virtual sensors and state descriptors of the tetherless care implementation, which produce the same features, states, and actions. It utilizes the same location monitoring algorithm. And it generates the same network traffic, differing only as a result of the patient behavior.

However, it utilizes a middleware component with a simplified algorithm. Rather than maintaining a history of prior sessions, utilizing algorithm 1 to predict likely future sessions, or scheduling messages within them, it employs a duty cycling algorithm. At the end of each fixed, regular interval, the middleware attempts to upload the entirety of the message queue. If no connection is available at that time, it retries at the next interval. When each new message is received, its constraints are evaluated in comparison to the next upload time. If the next upload time is unable

to meet the message's constraints, an unscheduled upload is initiated immediately, and the entire queue is uploaded.

Compared to the middleware discussed in chapter 4 and utilized in the tetherless care implementation, several scenarios can be conceived where the naive solution fails to exploit the available resources. For example, consider five minute intervals and a scenario where a connection becomes available two minutes into a given interval, lasts for one minute, and is the only opportunity available for the next 25 minutes. This approach could fail to meet the constraints of messages whose constraints would otherwise dictate they be uploaded during the first opportunity. However, it may be the case that these scenarios are rare enough and that, in practice, this scheme outperforms the more intelligent prediction.

Also, note that this scheme provides no contribution to the system's risk analysis.

5.4 METHODOLOGY

The original, tetherless, and naive implementations were deployed to a Motorola Droid Pro running Android 3.2.4 from Verizon with 1GB of internal storage, accelerometer, light sensor, camera, GPS, and 802.11n 2.4GHz radio. No WWAN data plans were associated with the devices such that the system was confined, once again, to WiFi-based sessions.

As with the prior chapter, the scheduling algorithm of the tetherless care implementation was configured with a threshold value of $\Phi = 0.2$ and an assumed fixed upload rate of 50 bytes per millisecond.

The interval between successive uploads of the duty cycling middleware in the naive implementation was set to ten minutes for one set of trials and one hour for another set of trials. This was done to determine the effect of the interval on the system lifetime. A more complete investigation is left for future work.

Alert messages were given a deadline of ten seconds in all three implementations. Regular data messages were given a deadline of 30 seconds in the original implementation and two hours in the naive and tetherless care implementations. The data rate of the original implementation dictated a shorter deadline. Two hours was chosen to account for the delays exhibited in the WiFi

environment from figure 8b yet would accommodate regular updates to a staff member who might be working an eight hour shift.

Each device was fully charged before being unplugged simultaneously and kept with the author until their respective battery capacities dropped to 50% power. After the original implementation discharged its respective battery, it was recharged, unplugged, and again kept with the device running the tetherless care implementation. Trials involving the naive implementation were conducted separately. Little attempt was made to ensure a WiFi access point was available to service the constant network needs of the original implementation, and where the author's mobility prevented a connection to the server, data were lost. Both the naive and tetherless care implementations would store messages until a connection with the server was established, at which point they would be uploaded even if their deadlines had expired.

5.5 RESULTS

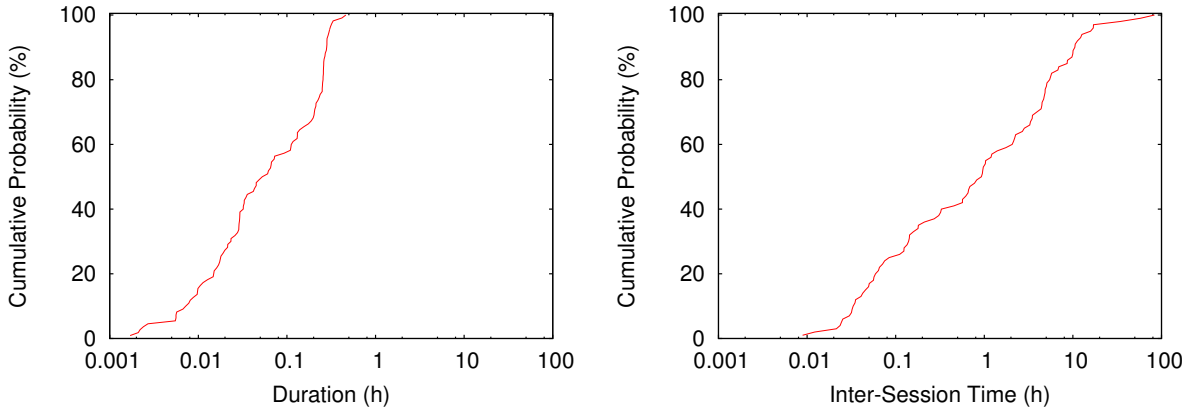
A total of 8 trials of the original implementation were collected. Table 7 summarizes the collected performance metrics with 95% confidence intervals. All but one of the trials included at least one trip by the author that required the Droid Pro to disconnect from the available WiFi in his apartment and attempt to exploit an opportunity along his path during the trip. Given the results shown in figure 4, the feature rate of the original implementation, combined with a persistent network connection, would suggest the battery of the Droid Pro could have supported a full day's mobility with the original implementation. However, this implementation falls significantly short of the feasibility limit of 12 hours for half of the battery, which is attributed to the added use of the GPS sensor every three seconds and file I/O needed to process the images moving through the system.

The small percentage of missed deadlines was most likely due to messages added to the queue immediately before a connection was broken. The system did not drop already queued messages once it became disconnected. Instead, the messages persisted in memory until a connection was formed again, at which time they would then be delivered, most likely after their deadlines had expired. Messages created after the system moved into a disconnected state were dropped altogether.

Table 7: Peformance of Original Implementation of PandaCare

Original Implementation		
Lifetime (h)	6.14	±0.871
Total Messages	63306	±9827
Feature Rate (Hz)	2.87	±0.14
Total Alerts	525	±1181
Total Data	57.11 MB	±9.65
Total Missed Deadlines	748	±1980
Percent of Messages Missed	1.18%	
Missed Alerts	8	±26
Percent of Alerts Missed	1.57%	
Dropped Messages	27707	±37419
Percent of Messages Dropped	43.77%	
Dropped Alerts	350	±486
Percent of Alerts Dropped	66.63%	

Given the message generation period of three seconds, the results of chapter 3 would suggest the original implementation could survive a full day's mobility. The lifetime found here, though, contradicts those findings, which is attributable to the extra work in utilizing the GPS, the storage hardware, and the more sizable network traffic. Also, note that confidence intervals are listed as a single number for readability, but all have a minimum lower bound of zero.



(a) Duration Distribution

(b) Inter-session Time Distribution

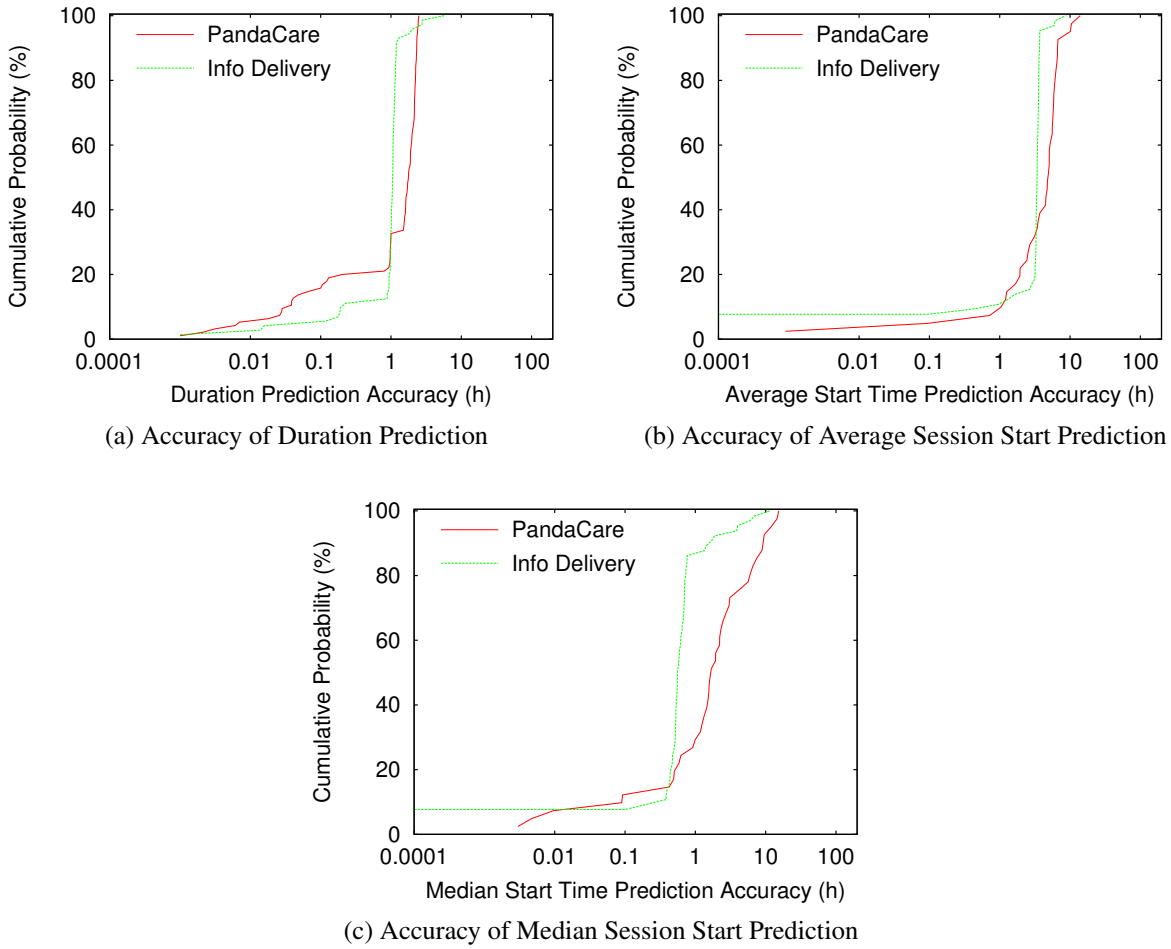
Figure 13: PandaCare Session Environment

The duration and inter-session time distributions, plotted on a log scale, of the 296 sessions formed during the 5 executions of the tetherless care implementation. Given that each trial was limited to the time needed to discharge half the battery, the distributions lost its long tail with respect to the results of chapter 4.

As expected, this implementation failed to meet the requirements of both PandaCare and tetherless care. The assumed always-available network connection violates the patient's need for mobility. The short system lifetime violates the pervasive and ubiquitous requirement. And the significant number of dropped messages violates the reliability requirement. Most notable are the dropped alert messages, which indicate the system would have failed to garner a response from a caregiver to the critical situation of the user. The lack of intelligent on-body computing failed to adapt to the changing environment, to conserve the limited resources around the patient, and to ensure the user's safety.

There were 296 sessions formed during the 5 executions of the tetherless care implementation. The Droid Pro encountered 14 access points and visited 10 of them repeatedly. Figure 13a and 13b plot the duration and inter-session distributions, respectively, on the same scale as the previous chapter for comparison purposes. Here, the duration distribution has lost its long tail with a max duration of 27.7 minutes. Durations averaged 7.02 minutes with a confidence interval of 14.00 minutes. This is due to the fact that longer sessions on the Droid Pro usually occurred while the device was charging, i.e. connected to an external power source. For this study, the app was not executing during charging sessions. Also, the inter-session time distribution exhibits longer delays

between successive sessions than that of the previous chapter. The two-hour regular message deadline encompassed only 60% of the delays, and the ten-second alert message deadline encompassed less than 1% of them.



(a) Accuracy of Duration Prediction

(b) Accuracy of Average Session Start Prediction

(c) Accuracy of Median Session Start Prediction

Figure 14: PandaCare Session Prediction Accuracy

The distribution of the difference between the predicted duration or start time and its respective actual value plotted on a log scale. Given that these results are limited to only the Droid Pro, the corresponding results from chapter 4 for that device are also plotted for comparison. This demonstrates the poorer performance is a result of the device, which performed poorly universally compared to other devices from the previous chapter.

Figures 14a, 14b, and 14c plot the cumulative probability distribution of differences between the observed session duration and start time and the corresponding predicted values for the Droid Pro running the tetherless care implementation. Given that the results of the previous chapter demonstrate significantly improved performance, the distribution exclusive to the Droid Pro from

the previous chapter's results is also plotted. As such, the poor performance here seems to be attributable to the device and its older Android operating system. Overall, the prediction accuracy was off on the order of a few hours, which is the same timescale as the two-hour deadline set for regular messages of this study. Furthermore, the accuracy diminished since the previous study, and this could be attributable to an overall change in the movement behavior of the author or, given the upgrade that occurred between experiments, to a change in the OS behavior between Android 2.2 and Android 2.3.4. In either case, the history collected on the device may not have accurately reflected the new environment. More work is needed to examine fluctuations in user behavior over the long term.

Table 8: Performance of Naive and Tetherless Care Implementations of PandaCare

	Naive		Tetherless Care	
Lifetime (h)	11.17	±2.681	33.12	±10.785
Total Messages	394	±263	439	±608
Feature Rate (Hz)	0.0097	±0.0060	0.0037	±0.0047
Total Alerts	203	±355	212	±452
Total Data	26.76 kB	±17.84	31.55 kB	±41.37
Total Missed Deadlines	271	±333	300	±545
Percent of Messages Missed	68.75%		68.40%	
Missed Alerts	206	±366	176	±472
Percent of Alerts Missed	101.57%		83.11%	

Unlike the original implementation, the network traffic for the tetherless and naive implementations was dominated by alert messages with short deadlines, which more often required unilateral action by both implementations and masked the effects of their power saving approaches. Also, note that no statistical difference was found between the naive trials with ten-minute duty cycle intervals and those with one-hour intervals such that the one-hour interval is displayed here.

A total of 5 trials of the naive implementation using a one hour upload interval and 5 trials of the tetherless care implementation were collected. Table 8 summarizes the performance metrics averaged over the trials of each respective implementation with 95% confidence intervals. Not shown are the number of dropped messages and alerts, which were zero in both cases, unlike

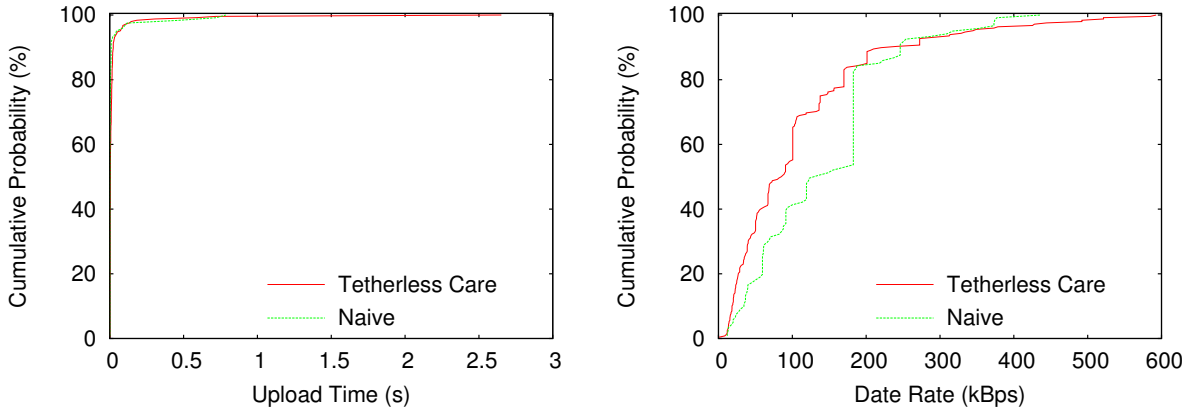
the original implementation. No statistical difference was found between naive trials with a ten-minute duty cycle and those with a one-hour duty cycle, such that the comparison is confined to the one-hour duty cycle here. The similarity found between the two is due to two factors: (1) the network traffic was dominated by alert messages that required unilateral action on the part of the middleware component and (2) the WiFi network was often unavailable at the time of a scheduled upload.

In both implementations, the local processing on the device reduced the needed network traffic by an order of magnitude from megabytes down to kilobytes. As mentioned, this local processing monitors the patient's state so as to only communicate essential information to external entities. The result of this skews the network traffic to a higher percentage of alerts, which impacts the ability of either middleware component to delay transmission in order to save power. Most of the alerts resulted from the author leaving the predefined geofences as well, which likely occurred while the device was disconnected and increased the likelihood of missing the alert's deadline.

Also, note that all of the alerts resulting from fall detection were found to be false positives due to changes in the device orientation, but not actual falls. The author never fell nor dropped a device during any of the trials, which would have resulted in a true fall alert. Nevertheless, the fall detection algorithm was consistent across implementations to provide a similar traffic profile for each implementation to manage.

The most notable difference between the two implementations is the average system lifetime. The naive implementation drained the power more quickly than the tetherless implementation yet supported a full day's mobility depending on the user's activity. The discrepancy between the two is attributed to the timer mechanism that scheduled the regular periodic uploads, which required the processor to stay active. Conversely, the tetherless care implementation operated in a passive manner reacting to system events before taking action. However, it is believed that a newer operating system on a more contemporary device would overcome this limitation and support system operation for a longer period of time given the naive implementation.

Compared with the work of the previous chapter, the data load offered by the PandaCare application was composed of shorter deadlines. The inter-session time distribution demonstrated that any disconnection would likely cause an alert to miss its deadline, which is confirmed by the large number of missed alerts for both the naive and tetherless care implementations. The impact of



(a) PandaCare Transfer Time Distribution

(b) PandaCare Transfer Rate Distribution

Figure 15: PandaCare Data Transfer Profile

Shorter deadlines and a smaller number of delayed upload opportunities resulted in smaller transfer sizes.

this relationship between the deadlines and the environmental delays allowed for only 3.04% of the scheduler executions in the tetherless care implementation to delay transmission. Both implementations were subject to a network environment incapable of supporting the demands of the application. As figure 15a shows, the lack of delayed transfers resulted in smaller transfer sizes that typically consumed only a few milliseconds of the session despite the somewhat slower transfer rates depicted in figure 15b.

Also, given that the alert deadlines were an order of magnitude smaller than the typical inter-session times, only 63.51% of the computed WiFi-only schedules of the tetherless care implementation were safe, which corroborates the conclusion of the previous chapter that alternate communication technologies would be needed for deadlines less amenable to the WiFi session behavior.

5.6 CONCLUSION

PandaCare is an application targeted at long term patient monitoring for dementia care. It offers the ability of one or more staff or family members to remotely care for a set of patients at varying distances with respect to three main functions: location monitoring, fall detection, and vital sign

monitoring. The fundamental contract made between the application and staff or family members is that important changes to the patient's state within these three areas will be communicated to the staff or family member in a timely fashion. As such, while the patient's state is otherwise normal, the staff or family are able to focus their attention elsewhere.

The application was originally conceived and developed for a system consisting of the eButton, an auxiliary wristband, and a patient-specific server. The software deployed for this application relegated the majority of the monitoring functionality to the server, to which raw data samples were constantly streamed. Furthermore, the implementation employed algorithms that absolutely required this constant connection to provide monitoring functionality, i.e. monitoring ceased while the device was disconnected.

As shown, this implementation fails in the face the patient's mobility and the constraints of a battery powered device, but the tetherless care infrastructure combined with algorithms that operate while disconnected overcomes most of these challenges. The original implementation expends energy too quickly by streaming data to a server, drops vital information while disconnected, and requires server computation to fix a room-level location. Both implementations utilizing the tetherless care framework employ greater intelligence on the device to reduce network traffic such that a constant server connection is unnecessary. In doing so, it allocates energy resources more efficiently and provides the patient with full mobility. Moreover, these implementations allocate resources in a manner commensurate with the state of the patient, reacting to important alerts with greater energy expenditures. Similarly, the reduced traffic allows the system to persist data to disk while disconnected, offering extremely high reliability in a manner transparent to the application developer.

This argument, specifically that the tetherless care concept is feasible, is limited by a combination of the Android OS used in the implementations here and the network environment observed by this investigation. Results show that a WiFi-only implementation cannot meet the delay requirement of tetherless care or of the version of PandaCare implemented. And more work is needed to examine the effect of incorporating cellular communication technology in meeting message constraints and the impact the added communication would have on battery consumption. Similarly, work is needed to examine system operation on newer versions of the Android OS, which seem to better utilize WiFi network hardware.

Also, a more complete implementation of PandaCare with more intelligent algorithms would not only likely augment the traffic profile in the system but also better utilize the intelligence of the framework. For instance, a full implementation of algorithm 7 would first detect the free-fall of the patient and the subsequent impact with the ground in conjunction with a change in body orientation. It would then attribute signs of extreme stress to any pain or injury caused by the impact. It would determine if the patient is moving or getting back up and ultimately adjust the criticality of the situation accordingly. A more context-aware location monitoring algorithm might differentiate trips that have purposeful motion and a likely well-known destination from random, confusion-induced wanderings. As these algorithms get more sophisticated, a degree of urgency may emerge that, in one way, could be expressed in a range of deadlines, which would impact the overall system performance in turn. These algorithms, though, were not the focus of this work.

However, this work demonstrates a feasible space exists for the development of some set of tetherless care applications upon this framework. This set is, at the moment, confined to those applications, i.e. staff or family members, that can tolerate a delay on the order of tens of minutes or hours, the minimum of which is still unknown.

6.0 CONCLUSION

The growing costs of healthcare stem from numerous sources ranging from an unhealthy, aging population that will increase the demand for care to hospital executives and suppliers looking to capitalize on a large, captive market. Where public financing subsidizes these costs, this trend has occupied public debate and policy in recent years as it consumes a greater portion of public funds. This debate has demonstrated that the complexity of the issues that culminate in the aggregate cost of care requires new, disruptive innovations in the delivery of care to overcome entrenched interests.

One such disruptive idea is the shift from a *disease-centric* to *patient-centric* care paradigm. Rather than execute a scripted response to any given disease when it manifests itself in a visit to a single point of care, an infrastructure of information technology manages a history of patient information that is universally accessible as the patient moves from point-of-care to point-of-care through the healthcare system, so as to allow each caregiver to exploit prior knowledge of the patient and tailor a custom care response to him or her.

What this patient-centric vision lacks, however, is a continued ongoing relationship between a patient and his or her caregivers. Despite a web portal by which the patient can log information and interact with caregivers in a limited fashion that's predicated on the patient's initiative, it provides no source of continued, unbiased information on the patient's behavior and state.

Because of this gap, this thesis introduced the concept of *tetherless care* whereby a patient is monitored beyond the confines of traditional points of care through a system of low-power sensors and wireless networks. Building on the work of body area networks, delay tolerant networks, and recent advances in mobile computing, it is now conceivable to deliver care to patients without confining them to facilities or prescribed geographic areas.

The fundamental challenge in achieving this vision of tetherless care is the ability to provide care that is commensurate with traditional in-hospital care for those conditions and diseases amenable to the tetherless care paradigm. The results of this work demonstrate a logical model of tetherless care and a proof-of-concept system implementation that overcomes this challenge along with the limitations imposed by contemporary, battery-operated mobile technology.

A summary of the argument affirming the fundamental question of the feasibility of providing tetherless care is presented in section 6.1. Remaining questions and new research opportunities are presented in section 6.2. Section 6.3 concludes the chapter.

6.1 THESIS CONTRIBUTIONS

The requirements for tetherless care applications and the systems that support them, outlined in section 1.2.3, are named as follows: data sampling rate, pervasive and ubiquitous operation, respect for caregiver's time, context awareness, context-dependent delay, reliability, and security.

Prior work has demonstrated the ability to meet the data sampling rate requirement using wireless, body-worn sensing devices; specialized, low-power wireless networks; and data compression techniques to efficiently collect sampled data at a single point of analysis. Similarly, prior work has shown the ability to achieve context awareness on mobile hardware in a ubiquitous, real-time fashion for specific applications. Also, the form factor of these devices have been shown to allow for all day comfort with minimal inconvenience.

Given that the context awareness requirement can only be fully addressed in an application-dependent manner, the argument of chapter 3 first models the operation of assessing the patient's context as a set of three relations, R, Q, and M, that respectively extract knowledge from collected samples through a series of relations from samples to features, features to states, and states to actions. Tetherless care applications are modeled as a graph of virtual sensors and state descriptors that declare interests in samples, features, and states and produce features, states, or actions. Based on this model, a formal construction separates from the patient's context the samples, features, states, and actions related to the observation and use of the network environment, such that these elements could be abstracted into a middleware component of a possible system implementation.

In doing so, the middleware relieves the burden from the application developer of attending to the technical aspects of the network environment. Also with this model, an algorithm is proposed for assessing the risk of the patient's state in an application dependent manner. Chapter 3 goes on to render and evaluate an architecture based on the tetherless model in order to demonstrate feasibility on a resource constrained device.

Chapter 4 evaluates the feasibility of the separation of the network-related context into an intelligent middleware utilizing the risk analysis algorithm to estimate the risk of delayed communication. It defines a *message* as a set of actions and that encodes a set of constraints for their execution. These constraints are evaluated against the expected network environment to determine if the defined network operation can be deferred into the future. By delaying communication, the system can save power and prolong its operation to better achieve the goal of tetherless care.

In chapter 5, the PandaCare application was implemented on the tetherless care architecture, which demonstrated the ability to support the feasible context-aware operation of tetherless care applications in two ways. First, a set of virtual sensors and state descriptors and the relationships among them are defined as a tetherless application for PandaCare, which compute a view of the patient's state and respond to changes in it. Second, by receiving an indication of the changing context encoded through message deadlines, the architecture responds to the context in a manner appropriate for the expected delays observed in the network environment.

The primary remaining limitation in achieving both the mobility and the pervasive and ubiquitous requirements is the longevity of the system's battery. Chapter 3 shows that contemporary mobile devices provide sufficient power for sensor communication and data analysis over a period of time long enough to at least require a nightly recharge of the device. However, a continuous, on-going communication channel with caregivers is untenable with current battery technology and the patient's mobility. This was corroborated with the original implementation of the PandaCare application in chapter 5.

This limitation of the battery requires that (1) the device be disconnected from caregivers or servers during operation in order to save power and (2) that applications be designed to tolerate this disconnection. As such, this limitation puts the pervasive and ubiquitous requirement at odds with the delay requirement. The logical model of the middleware in chapter 3 defines the space to manage the trade-off between these two objectives. Results from both chapters 4 and 5 show

that the implemented middleware can provide the needed system longevity to meet the pervasive and ubiquitous requirement and support some minimum deadline between tens of minutes and 24 hours by passively exploiting the WiFi access points encountered as the user ambulates.

The reliability requirement pervades all aspects of system operation. The patient state must be under a minimal amount of observation at all times. The implications of this extend to application design and implementation as well as the reliability of the hardware and operating system on which the application and tetherless care framework are running. Similarly, data traversing the network must be delivered in a reliable manner, which was the focus of this work. And, in mitigating the periods of disconnection, the middleware caches all network data to persistent storage until it encounters a suitable opportunity to transfer it to the server. Data are not removed from the cache until it is successfully transferred. Thus, assuming a reliable hardware platform and a stable and reliable tetherless care application, the system meets its reliability needs.

Left for future work are the security and respect for caregiver's time requirements. It is believed the security requirement is achievable using existing encryption techniques. Respect for caregiver's time is heavily dependent upon both the application and the mechanisms by which the healthcare system dedicates personnel to service tetherless patients.

6.2 FUTURE RESEARCH DIRECTIONS

As mentioned previously, it is unknown how the healthcare infrastructure will evolve to accommodate tetherless care. Personnel could be co-located with existing facilities, could be aggregated into a smaller number of central facilities, or could be entirely distributed and mobile. Each possibility affects the volume of alerts, the network traffic patterns, the jurisdiction of caregivers, and liability requirements. It is difficult to design and study the tetherless care infrastructure needed to reliably deliver the appropriate data to caregivers in a timely fashion and to ensure they are not inundated with an excess of information. More work is needed to enumerate and evaluate each possible design, which should likely involve a number of healthcare professionals.

Also, this work sought to examine the system performance while passively exploiting network opportunities, but an improved design would actively search for and initiate WiFi sessions when

needed. The passive design consumes resources when the user interacts with the device such that the battery depletion more closely mimics the device's usage pattern, which provides a perception of less resource consumption and was more amenable to volunteers of the investigation. A more active middleware could better react to more urgent message constraints, and this work provides a basis for comparison with the improvement.

There exists a relationship between the message constraints defined by the application and the expected delays in the network environment around a given user. Work is needed to explore the variability of this relationship. The nature by which the session distribution and intersession distribution change from user to user informs the number and types of needed message constraints. And the optimal performance of the middleware within this space of user behavior crossed by possible constraints is unknown.

The potential contributions of the server to tetherless care operation is another avenue of future work. The server can exploit greater processing and storage capabilities to assist the coordinator in its computation, particularly with respect to machine learning algorithms that operate over a long term log of patient data, as demonstrated by the risk analysis algorithm in section 3.5. Moreover, where the coordinator can only utilize one patient's information, server algorithms may benefit from the contributed data from multiple patients.

Furthermore, in light of the potential benefits of computation on the server coupled with the fact that the coordinator is often disconnected from it, mechanisms for ensuring a cohesive collaboration between the two entities would be needed such that the coordinator can always operate in a standalone fashion yet exploit the resources of the server where possible.

An extension of this is the possibility of reconfiguring the set of virtual sensors and state descriptors that execute on the coordinator in a patient specific manner. It may be possible to match the patient's medical history, condition, and doctor's recommendations to a custom set of objects and their settings, package them, and download them to the coordinator. In this way, the coordinator's operation can evolve programmatically to the patient's changing condition over the long term while the set of installed virtual sensors and state descriptors evolve system operation to the patient's state over a shorter timeframe. This would require mechanisms and APIs for describing virtual sensors and state descriptors for a matching algorithm and securely uploading them to a server where they can make their way to a given patient's coordinator.

Finally, the vision and possibility of tetherless care coupled with the presented framework invites a range of novel algorithms for assessing the patient state and behavior, which are needed to fully determine the scope of the diseases and conditions that are amenable to tetherless care.

6.3 CONCLUSION

The concept of tetherless care describes a paradigm that keeps patients under observation while freeing them to return to their everyday lives. It frees hospital resources for other purposes, which could significantly reduce the cost of care for a large portion of diseases and chronic conditions, and it supports the shift from *disease-centric* to *patient-centric* care.

This work has designed, modeled, implemented, and studied a proof-of-concept prototype for a tetherless care system. It enumerated the requirements of a tetherless care system. It discussed the conflicts among them, particularly between the requirement for pervasive and ubiquitous operation and the context-dependent delay requirement. It then presented a novel solution that mitigates the conflict by saving energy while meeting the delivery deadlines of application data.

This breakthrough has enabled a possible new way of delivering healthcare and has outlined a space into which the ecosystem of tetherless care applications can now grow.

BIBLIOGRAPHY

[1] Alarmnet. http://www.cs.virginia.edu/wsn/medical/, March 2011.

[2] Aware home research initiative. http://awarehome.imtc.gatech.edu/, April 2011.

[3] Smart medical home. http://www.urmc.rochester.edu/future-health/validation/smart-home.cfm, March 2011.

[4] U.s. center for medicaid and medicare health spending projections. http://www.cms.gov/NationalHealthExpendData/downloads/proj2009.pdf, January 2011.

[5] Corventis medical. http://corventismedical.com/, March 2012.

[6] Neurosky mindwave. http://www.neurosky.com, April 2012.

[7] Nike fuelband. http://www.nike.com/FuelBand, April 2012.

[8] Delay tolerant networking research group. http://www.dtnrg.org, 2013.

[9] Oecd health expenditures per country. http://www.compareyourcountry.org/01/health/health-spending-change, March 2014.

[10] Skyhook wireless. http://www.skyhookwireless.com, April 2014.

[11] P. S. Addison. Wavelet transforms and the ecg: a review. *Physiological Measurement*, 26(5):R155–99, Oct 2005.

[12] A. Alesanco, S. Olmos, R. Istepanian, and J. Garcia. Enhanced real-time ecg coder for packetized telecardiology applications. *Information Technology in Biomedicine, IEEE Transactions on*, 10(2):229 –236, april 2006.

[13] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 209–222, New York, NY, USA, 2010. ACM.

[14] S. Brill. Bitter pill: Why medical bills are killing us. *TIME Magazine*, February 2013.

[15] T. Broens, A. V. Halteren, M. V. Sinderen, and K. Wac. Towards an application framework for context-aware m-health applications. *International Journal of Internet Protocol Technology*, 2(2):109–116, 2007.

[16] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *Mobile Computing, IEEE Transactions on*, 6(6):606 –620, june 2007.

[17] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2):4:1–4:26, June 2008.

[18] Y. Chon, E. Talipov, H. Shin, and H. Cha. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 82–95, New York, NY, USA, 2011. ACM.

[19] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*, page 48, Washington, DC, USA, 2006. IEEE Computer Society.

[20] R. Cristescu, B. Beferull-Lozano, and M. Vetterli. On network correlated data gathering. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2571–2582 vol.4, March 2004.

[21] B. de Silva, A. Natarajan, and M. Motani. Inter-user interference in body sensor networks: Preliminary investigation and an infrastructure-based solution. In *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, pages 35–40, June 2009.

[22] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 588–599. VLDB Endowment, 2004.

[23] E. E. Egbogah and A. O. Fapojuwo. A survey of system architecture requirements for health care-based wireless sensor networks. *Sensors*, 11(5):4875–4898, 2011.

[24] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. Diversity in smartphone usage. In *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 179–194, New York, NY, USA, 2010. ACM.

[25] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM Press.

[26] K. Fall and S. Farrell. Dtn: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, June 2008.

[27] W. Gao, Q. Li, B. Zhao, and G. Cao. Multicasting in delay tolerant networks: a social network perspective. In *MobiHoc '09: Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*, pages 299–308, New York, NY, USA, 2009. ACM.

[28] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Transactions on Networking*, 10(4):477–486, 2002.

[29] H. Gupta, V. Navda, S. Das, and V. Chowdhary. Efficient gathering of correlated data in sensor networks. *ACM Trans. Sen. Netw.*, 4(1):1–31, 2008.

[30] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer. Delay tolerant mobile networks (dtmns): Controlled flooding in sparse mobile networks. *IFIP Networking*, May 2005.

[31] W. Heinzelman, A. Murphy, H. Carvalho, and M. Perillo. Middleware to support sensor network applications. *Network, IEEE*, 18(1):6 – 14, 2004.

[32] F. Hu, M. Jiang, L. Celentano, and Y. Xiao. Robust medical ad hoc sensor networks (masn) with wavelet-based ecg data mining. *Ad Hoc Netw.*, 6(7):986–1012, Sept. 2008.

[33] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4g lte networks. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 225–238, New York, NY, USA, 2012. ACM.

[34] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, pages 244–251, New York, NY, USA, 2005. ACM.

[35] P. Hui and J. Crowcroft. How small labels create big improvements. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference on*, pages 65 –70, mar. 2007.

[36] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 241–250, New York, NY, USA, 2008. ACM.

[37] P. Hui and A. Lindgren. Phase transitions of opportunistic communication. In *Proceedings of the third ACM workshop on Challenged networks*, CHANTS '08, pages 73–80, New York, NY, USA, 2008. ACM.

[38] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft. Distributed community detection in delay tolerant networks. In *MobiArch '07: Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, pages 1–8, New York, NY, USA, 2007. ACM.

[39] S. Intille, K. Larson, E. Tapia, J. Beaudin, P. Kaushik, J. Nawyn, and R. Rockinson. Using a live-in laboratory for ubiquitous computing research. In K. Fishkin, B. Schiele, P. Nixon, and A. Quigley, editors, *Pervasive Computing*, volume 3968 of *Lecture Notes in Computer Science*, pages 349–365. Springer Berlin / Heidelberg, 2006.

[40] R. Istepanian and A. Petrosian. Optimal zonal wavelet-based ecg data compression for a mobile telecardiology system. *Information Technology in Biomedicine, IEEE Transactions on*, 4(3):200–211, Sept. 2000.

[41] G. Jackson and C. Crocker. The use of altimeters in height measurement. http://www.hills-database.co.uk/altim.html, March 2014.

[42] S. F. Jencks, M. V. Williams, and E. A. Coleman. Rehospitalizations among patients in the medicare fee-for-service program. *New England Journal of Medicine*, 360(14):1418–1428, 2009.

[43] B. D. Jones, C. R. Winegarden, and W. A. Rogers. Lifelong interactions: Supporting healthy aging with new technologies. *interactions*, 16:48–51, July 2009.

[44] V. Jones, A. van Halteren, I. Widya, N. Dokovski, G. Koprinkov, R. Bults, D. Konstantas, and R. Herzog. Mobihealth: mobile health services based on body area networks. Technical Report TR-CTIT-06-37, Centre for Telematics and Information Technology, University of Twente, Enschede, January 2006. Imported from CTIT.

[45] J. Ko, J. H. Lim, Y. Chen, R. Musvaloiu-E, A. Terzis, G. M. Masson, T. Gao, W. Destler, L. Selavo, and R. P. Dutton. Medisn: Medical emergency detection in sensor networks. *ACM Trans. Embed. Comput. Syst.*, 10:11:1–11:29, August 2010.

[46] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh. Wireless sensor networks for healthcare. *Proceedings of the IEEE*, 98(11):1947 –1960, nov. 2010.

[47] D. Konstantas, V. Jones, R. Bults, and R. Herzog. Mobihealth-innovative 2.5/3g mobile services and applications for health care. In *IST Mobile & Wireless Telecommunications Summit 2002*, Thessaloniki, Greece, June 2002. EC IST. Imported from research group ASNA (ID number 128).

[48] K. V. Laerhoven, B. P. L. Lo, J. W. P. Ng, S. Thiemjarus, R. King, S. Kwan, H. werner Gellersen, M. Sloman, O. Wells, P. Needham, N. Peters, A. Darzi, C. Toumazou, and G. zhong Yang. Medical healthcare monitoring with wearable and implantable sensors. In *Proc. of the 3rd International Workshop on Ubiquitous Computing for Healthcare Applications*, 2004.

[49] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile data offloading: how much can wifi deliver? In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 26:1–26:12, New York, NY, USA, 2010. ACM.

[50] A. Lindgren, C. Diot, and J. Scott. Impact of communication infrastructure on forwarding in pocket switched networks. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS '06, pages 261–268, New York, NY, USA, 2006. ACM.

[51] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[52] C. Liolios, C. Doukas, G. Fourlas, and I. Maglogiannis. An overview of body sensor networks in enabling pervasive healthcare and assistive environments. In *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '10, pages 43:1–43:10, New York, NY, USA, 2010. ACM.

[53] D. Lowe. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image, Mar. 23 2004. US Patent 6,711,293.

[54] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *International Workshop on Wearable and Implantable Body Sensor Networks*, April 2004.

[55] A. Milenković, C. Otto, and E. Jovanov. Wireless sensor networks for personal health monitoring: Issues and an implementation. *Comput. Commun.*, 29(13-14):2521–2533, 2006.

[56] A. Nait-Ali, R. Borsali, W. Khaled, and J. Lemoine. Time division multiplexing based method for compressing ecg signals: application for normal and abnormal cases. *Journal of Medical Engineering and Technology*, 31(5):324 – 331, 2007.

[57] A. Natarajan, B. de Silva, K.-K. Yap, and M. Motani. Link layer behavior of body area networks at 2.4 ghz. In *MobiCom '09: Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 241–252, New York, NY, USA, 2009. ACM.

[58] A. Natarajan, B. de Silva, K.-K. Yap, and M. Motani. To hop or not to hop: Network architecture for body sensor networks. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON '09. 6th Annual IEEE Communications Society Conference on*, pages 1–9, June 2009.

[59] A. Natarajan, M. Motani, B. de Silva, K.-K. Yap, and K. C. Chua. Investigating network architectures for body sensor networks. In *HealthNet '07: Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, pages 19–24, New York, NY, USA, 2007. ACM.

[60] J. W. Ng, B. P. Lo, O. Wells, M. Sloman, C. Toumazou, N. Peters, A. Darzi, and G.-Z. Yang. Ubiquitous monitoring environment for wearable and implantable sensors (ubimon). In *Proceeding of The Sixth International Conference on Ubiquitous Computing*, September 2004.

[61] M. Nielsen, E. N. Kamavuako, M. M. Andersen, M.-F. Lucas, and D. Farina. Optimal wavelets for biomedical signal compression. *Med Biol Eng Comput*, 44(7):561–568, July 2006.

[62] C. Otto, A. Milenkovic, C. Sanders, and E. Jovanov. System architecture of a wireless body area sensor network for ubiquitous health monitoring. *Journal of Mobile Multimedia*, 1(4):307–326, January 2006.

[63] S. Patel, K. Lorincz, R. Hughes, N. Huggins, J. Growdon, D. Standaert, M. Akay, J. Dy, M. Welsh, and P. Bonato. Monitoring motor fluctuations in patients with parkinson's disease using wearable sensors. *Trans. Info. Tech. Biomed.*, 13(6):864–873, 2009.

[64] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay tradeoffs in smartphone applications. In *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 255–270, New York, NY, USA, 2010. ACM.

[65] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong. On the levy-walk nature of human mobility. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 924 –932, april 2008.

[66] J. Scott, P. Hui, J. Crowcroft, and C. Diot. Haggle: a networking architecture designed around mobile users. In *Proc. Third IFIP Wireless on Demand Network Systems Conf.*, 2006.

[67] V. Shnayder, B.-r. Chen, K. Lorincz, T. R. F. F. Jones, and M. Welsh. Sensor networks for medical care. In *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 314–314, New York, NY, USA, 2005. ACM.

[68] V. Shnayder, B. rong Chen, K. Lorincz, T. Fulford-Jones, and M. Welsh. Sensor networks or medical care. Technical Report TR-08-05, Harvard University, 2005.

[69] F. Spadini, F. Vergari, L. Nachman, C. Lamberti, and T. Cinotti. A wireless and context-aware ecg monitor : An imote2 based portable system. In *Computers in Cardiology, 2008*, pages 997 –1000, 14-17 2008.

[70] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM Press.

[71] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. *IEEE International Conference on Pervasive Computing and Communications Workshops*, 0:79–85, 2007.

[72] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[73] A. H. van, R. Bults, K. Wac, D. Konstantas, I. Widya, N. Dokovski, G. Koprinkov, V. Jones, and R. Herzog. Mobile patient monitoring: The mobihealth system. *The Journal on Information Technology in Healthcare*, 2(5):365–373, October 2004.

[74] U. Varshney. Enhancing wireless patient monitoring by integrating stored and live patient information. In *Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on*, pages 501–506, 0-0 2006.

[75] U. Varshney. Transmission of emergency messages in wireless patient monitoring: Routing and performance evaluation. In *System Sciences, 2006. HICSS '06. Proceedings of the 39th Annual Hawaii International Conference on*, volume 5, pages 91a–91a, Jan. 2006.

[76] U. Varshney. A framework for supporting emergency messages in wireless patient monitoring. *Decis. Support Syst.*, 45(4):981–996, 2008.

[77] A. Wood, J. Stankovic, G. Virone, L. Selavo, Z. He, Q. Cao, T. Doan, Y. Wu, L. Fang, and R. Stoleru. Context-aware wireless sensor networks for assisted living and residential monitoring. *Network, IEEE*, 22(4):26 –33, 2008.

[78] J. Yao, R. Schmitz, and S. Warren. A wearable point-of-care system for home use that incorporates plug-and-play and wireless standards. *Information Technology in Biomedicine, IEEE Transactions on*, 9(3):363–371, Sept. 2005.

[79] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft. A socio-aware overlay for publish/subscribe communication in delay tolerant networks. In *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, pages 225–234, New York, NY, USA, 2007. ACM.