SparkMed: A Framework for Dynamic Integration of Multimedia Medical Data Into Distributed m-Health Systems

Liviu Constantinescu, *Student Member, IEEE*, Jinman Kim, *Member, IEEE*, and (David) Dagan Feng, *Fellow, IEEE*

***Abstract*—With the advent of 4G and other long-term evolution (LTE) wireless networks, the traditional boundaries of patient record propagation are diminishing as networking technologies extend the reach of hospital infrastructure and provide on-demand mobile access to medical multimedia data. However, due to legacy and proprietary software, storage and decommissioning costs, and the price of centralization and redevelopment, it remains complex, expensive, and often unfeasible for hospitals to deploy their infrastructure for online and mobile use. This paper proposes the SparkMed data integration framework for mobile healthcare (m-Health), which significantly benefits from the enhanced network capabilities of LTE wireless technologies, by enabling a wide range of heterogeneous medical software and database systems (such as the picture archiving and communication systems, hospital information system, and reporting systems) to be dynamically integrated into a cloud-like peer-to-peer multimedia data store. Our framework allows medical data applications to share data with mobile hosts over a wireless network (such as WiFi and 3G), by binding to existing software systems and deploying them as m-Health applications. SparkMed integrates techniques from multimedia streaming, rich Internet applications (RIA), and remote procedure call (RPC) frameworks to construct a Self-managing, Pervasive Automated netwoRK for Medical Enterprise Data (SparkMed). Further, it is resilient to failure, and able to use mobile and handheld devices to maintain its network, even in the absence of dedicated server devices. We have developed a prototype of the SparkMed framework for evaluation on a radiological workflow simulation, which uses SparkMed to deploy a radiological image viewer as an m-Health application for telemedical use by radiologists and stakeholders. We have evaluated our prototype using ten devices over WiFi and 3G, verifying that our framework meets its two main objectives: 1) interactive delivery of medical multimedia data to mobile devices; and 2) attaching to non-networked medical software processes without significantly impacting their performance. Consistent response times of under 500 ms and graphical frame rates of over 5 frames per second were observed under intended usage conditions. Further, overhead measurements displayed linear scalability and low resource requirements.**

***Index Terms*—Automated systems, biomedical engineering, handheld computing, m-Health, middleware, mobile communication, notebook computers, telemedicine.**

I. Introduction

Mobile healthcare, or m-Health, is one of the fastest growing areas of healthcare computing [1]. As electronic health records become commonplace, and the rapid uptake of mobile and handheld devices puts powerful portable computing devices into the hands of an ever-increasing proportion of the populace (inclusive of those in low-income and disadvantaged areas [2]), the stage is set for future wireless communication technologies to revolutionize patient care by making health services both portable and interoperable.

The next generation of networking is here, in the shape of new 4G and long-term evolution (LTE) wireless technologies such as WiMAX, which are all IP-based heterogeneous networks aimed at vastly expanding the accessibility and usability of any internet-connected system. LTE technologies are portable, lightweight and nonproprietary, and provide mobile devices with access to integrated communications standards that have low transmission costs and rich multimedia support. A major goal of LTE wireless technologies is the provision of personalized, reliable wireless data services that can allow even simple handheld devices to easily make use of multiple multimedia data streams at the same time [3].

Unfortunately, taking advantage of the benefits of LTE to deploy mobile healthcare technology entails major costs (in money, time, and computing resources) for any hospital infrastructure, and consequent problems such as resistance to change and the difficulty in using new systems. The most significant problems, however, result from: 1) the limited scope of access to data in proprietary hospital infrastructure systems; 2) the need to replace or decommission medical applications and data services if they do not support a networked healthcare model; 3) the storage and postprocessing requirements that keep medical data from becoming portable; and 4) the lack of a centralized repository or common standard for most healthcare data.

The motivation of this paper has been to address the aforementioned problems, to allow medical data applications to share data with mobile hosts over a wireless network by binding them to existing software packages and allowing them to operate (or be remotely operated) as m-Health applications. SparkMed integrates techniques from mobile technologies such as multimedia streaming, rich Internet applications (RIA), and remote

Manuscript received November 2, 2010; revised June 11, 2011 and September 30, 2011; accepted October 19, 2011. Date of publication October 28, 2011; date of current version February 3, 2012. This work was supported by ARC grants.

L. Constantinescu and J. Kim are with the School of Information Technologies, University of Sydney, Sydney, N.S.W. 2006, Australia (e-mail: liviu@it.usyd.edu.au; jinman@it.usyd.edu.au).

D. Feng is with the School of Information Technologies, University of Sydney, Sydney, N.S.W. 2006, Australia, and also with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (e-mail: feng@it.usyd.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITB.2011.2174064

procedure call (RPC) frameworks to construct a Self-managing, Pervasive Automated netwoRK for Medical Enterprise Data (SparkMed).

Our SparkMed design ensures: 1) minor interference with the regular operation of the host medical software it is bound to (given the importance of its continued functioning); 2) minimal overhead to make sure that the host system's performance remains unaffected while still allowing it to be utilized interactively across any IP-based connection; as well as 3) providing core functionality to limit the cost and scope of reprogramming.

SparkMed provides a number of automated, self-configuring services including discovery, data monitoring and synchronization, thread pooling for remote functions, collaborative remote control capability, and transcoding to web-based standards. We demonstrate this functionality with a prototype SparkMed system that implements these services for a simulated nuclear medicine workflow environment modeled upon the clinical information system in the Department of PET and Nuclear Medicine at our partner hospital, the Royal Prince Alfred Hospital (Camperdown, N.S.W., Australia). Performance was measured when propagating multimedia medical data to mobile hosts over three distinct network configurations: remote access over WiFi, roaming access via 3G, and "headless" use with an all-mobile network. We observe the performance of the mobile-based medical software, and measure the amount of overhead imposed upon the medical multimedia software we use as our data source.

II. Background

Many successful healthcare applications based on the picture archiving and communication systems (PACS), and using ubiquitous computing technologies, have been presented in the literature. For instance, projects such as Web-PACS [4] and PACSflow [5], and software packages such as the diagnostic image viewer OsiriX [6] allow for interinstitutional and web-based access to multimedia medical data. However, numerous heterogeneous database and diagnostic systems supplement PACS, and they typically require greater computing resources than those that are available on mobile devices [7]. There have been many attempts to develop telemedicine solutions that take advantage of mobile devices; however, the majority rely greatly on dedicated infrastructure, which limits their functionality and the scope of their deployment [8]. There is already speculation that the next generation of PACS is moving toward a cloud-computing-based system, with its data distributed within a network of secure online repositories [9]. Known in the literature as hosted PACS, offsite PACS, or PACS Software as a Service (SaaS), a number of commercial and research systems are already making major steps in this direction (e.g., [10], [11]). With increasing retention requirements, blurring distinctions between data storage and archiving, and an ever-increasing volume of medical data, the cost-effectiveness of cloud-based PACS and its inherent benefits for accessibility and disaster recovery make such systems very attractive to hospital administrators [12], [13].

A. Service-Oriented Architectures

Service-oriented architectures (SOA), e.g., [14], are a common answer to this problem: an approach which consists essentially of breaking down existing software systems into interoperable web services, and leveraging these to create complex medical applications and deploy them over a cloud network. Such an approach combines the benefits of both an in-house and a cloud solution. As a design principle, SOA provides an ideal solution for a healthcare environment, which contains numerous heterogeneous data services and isolated, specialized workstations that are difficult to interface with one another. Further, there is evidence that SOA can provide the necessary scalability and sustainability to support a healthcare information system [15].

In practical terms, however, such SOA implementations rely on the availability of componentized, interoperable web services that support a common syntax and semantics. XML, SOAP, MPEG-7, and MPEG-21 are all powerful standards for semantically rich delivery of data and metadata between web services in a service-oriented architecture. However, these services must be manually developed in a healthcare context; there is no common, accepted middleware standard for doing so; an in-house SOA entails significant overhead; and there is substantial expense incurred in integrating a pre-existing infrastructure to function as an SOA. As such, it is a substantial organizational challenge for any healthcare provider to adopt such an approach. Further, since it is primarily a development methodology to be followed when initially creating a system, a service-oriented architecture can be unfeasible to create from an existing system, largely due to custom-built, proprietary, and legacy components.

B. Prevalence of Mobile Devices in Healthcare

Most specialist healthcare providers own at least one mobile device, such as a SmartPhone, and that number is rising [16]. These mobile devices bring together all of the benefits of pagers, PDAs, and mobile phones, but the social networking and media functionality that is becoming ubiquitous on these devices allows health professionals to expand their practice and their relationships with patients [17], [18]. Further, these devices are showing promise as emergency teleconsultation devices, making true telepresence for medical practitioners possible without even the need for specialized hardware [19]. The current popularity surge of tablet devices such as the Apple iPad, and their increasing popularity in hospitals, has made form-factor considerations for handheld and mobile devices less important. In light of these facts, there is no longer as much of a bias against mobile devices in the healthcare sector as there was in the past.

C. Integration Methods

Numerous tools and technologies, however, exist for integrating such components to function as interoperable, query-able services compatible with mobile and web-based clients. These have in many cases already been applied to electronic patient records (EPRs) and medical enterprise data. Such techniques include the use of RPC frameworks such as CORBA [20], [21],

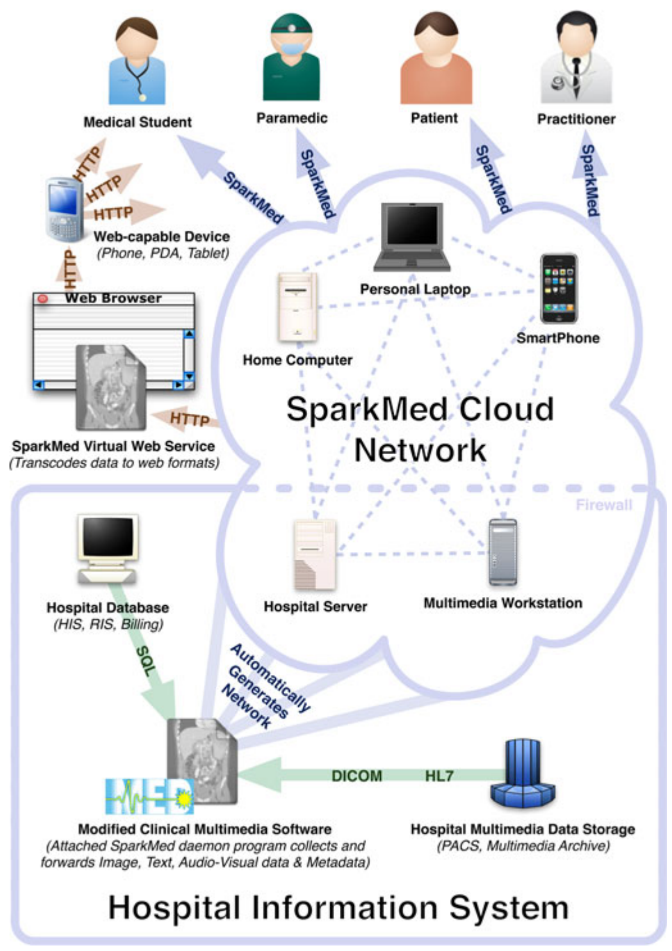
Medical Student
Paramedic
Patient
Practitioner
HTTP
HTTP
HTTP
HTTP
Web-capable Device
(Phone, PDA, Tablet)
SparkMed
SparkMed
SparkMed
SparkMed
Web Browser
Personal Laptop
SmartPhone
Home Computer
SparkMed Cloud Network
HTTP
SparkMed Virtual Web Service
(Transcodes data to web formats)
Firewall
Hospital Database
(HIS, RIS, Billing)
Hospital Server
Multimedia Workstation
SQL
Automatically Generates Network
DICOM
HL7
Modified Clinical Multimedia Software
(Attached SparkMed daemon program collects and forwards Image, Text, Audio-Visual data & Metadata)
Hospital Multimedia Data Storage
(PACS, Multimedia Archive)
Hospital Information System

Fig. 1. SparkMed cloud network, as generated automatically by our framework. Medical data are forwarded to mobile clients by means of a self-configuring, multicomputer network.

federated database systems [22], software agents [23], semantic data sources such as ontologies [24], and various forms of wrapper generation and encoding designed to translate dynamically between medical formats.

D. Our Related Work

Our previous research into mobile and web-based medical imaging technology [25], [26] was developed to address the need for mobile healthcare solutions by enhancing mobile telemedicine. Our new SparkMed framework integrates and extends our group's previous work developing mobile healthcare solutions, such as the mobile, active medical protocol (MAMP) [27] and our mobile "INVOLVE" dual-modality diagnostic workstation [28].

III. Methodology

In this section, we will go over the individual parts of our SparkMed framework for dynamic integration of multimedia medical data. Fig. 1 illustrates the conceptual outline of our network. This network is composed of devices inside and outside of hospitals and medical institutions, both desktop and mobile, and a series of web servers that can be either Intranet based or Internet based. Fig. 2 illustrates the hierarchy of networking layers used by our prototype. This hierarchical architecture is similar to some existing proposed middleware systems for remote application control [29] and medical data propagation [30], but we introduce several improvements. First, our network architecture is automatically self-generated without the need for user input or even network support in the host application. Second, we make use of a novel daemon technique (a daemon is an automated process invisible to the end user) to run a transparent, attached process and adapt network input and output to mimic normal usage, thus allowing compatibility with even legacy medical software. A similar technique is employed by systems that implement autonomic computing techniques to retrofit legacy software with new functionality [31]. However, SparkMed only implements a subset of the official definition of autonomic computing [32] as necessary to ensure that it can make effective data-synchronizing decisions and react intelligently to changes under network conditions.

The following section describes the software architecture and design of SparkMed, as well as the networking techniques it employs, in more detail.

A. Overall Architecture Design

Individual mobile- and desktop-based client nodes are the main components of our SparkMed architecture. A SparkMed client node is a stand-alone daemon program running inside a piece of medical software on any Internet-capable computing device. The attached medical software may in turn be itself running inside a web browser (or as part of the front end of a database system).

The individual SparkMed daemons use a connectionless Zeroconf [33] service discovery approach to connect to other daemons. All SparkMed daemons are identical, but their host applications determine which data items they share or provide, and this information is forwarded whenever the daemons connect to one another (as shown in Fig. 3). Individual nodes retain memory of other available nodes, keep an index of what data each provides or accepts, and are able to reconfigure their network to recover from service interruptions with a minimal disruption in functionality (as shown in Section IV-A).

B. Cloud Networking and Centrality Measurement

Having detected one another, SparkMed nodes communicate to generate a medical data "cloud": effectively a query-able centralized repository integrating every data item in the SparkMed network. The centrality of each node is calculated numerically in order to choose a central "server" node, with the key criteria for determining the centrality of a node being based on reachability, network access time, and security/storage capability. This calculation is performed by first filtering out all nodes which do not meet the data storage or security requirements of the active data sources, and those on peripheral or unreliable physical network connections (as determined over time using observed reachability from each node, and frequency of disconnection). If **G** is the graph representing SparkMed's network of

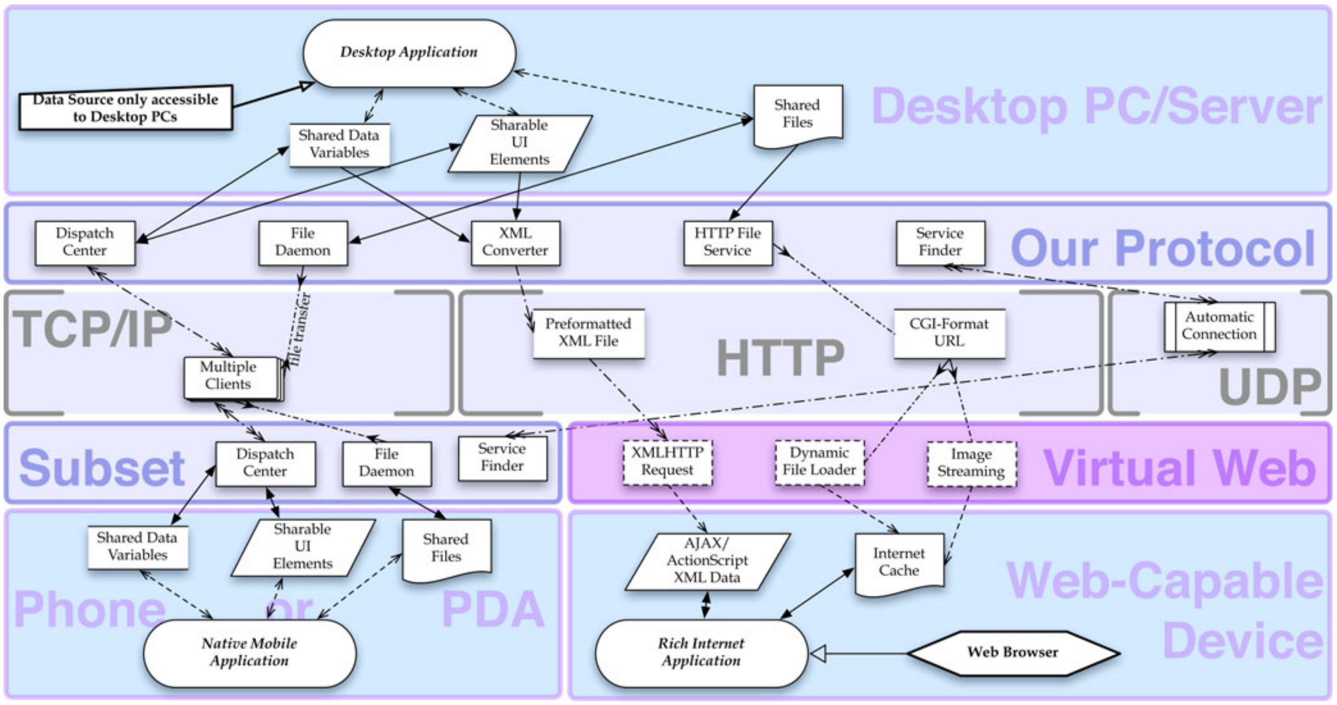

Fig. 2. Networking architecture of our prototype, showing the relationship between data at the hospital (top), and the means by which it is conveyed to mobile and browser-based hosts (bottom).

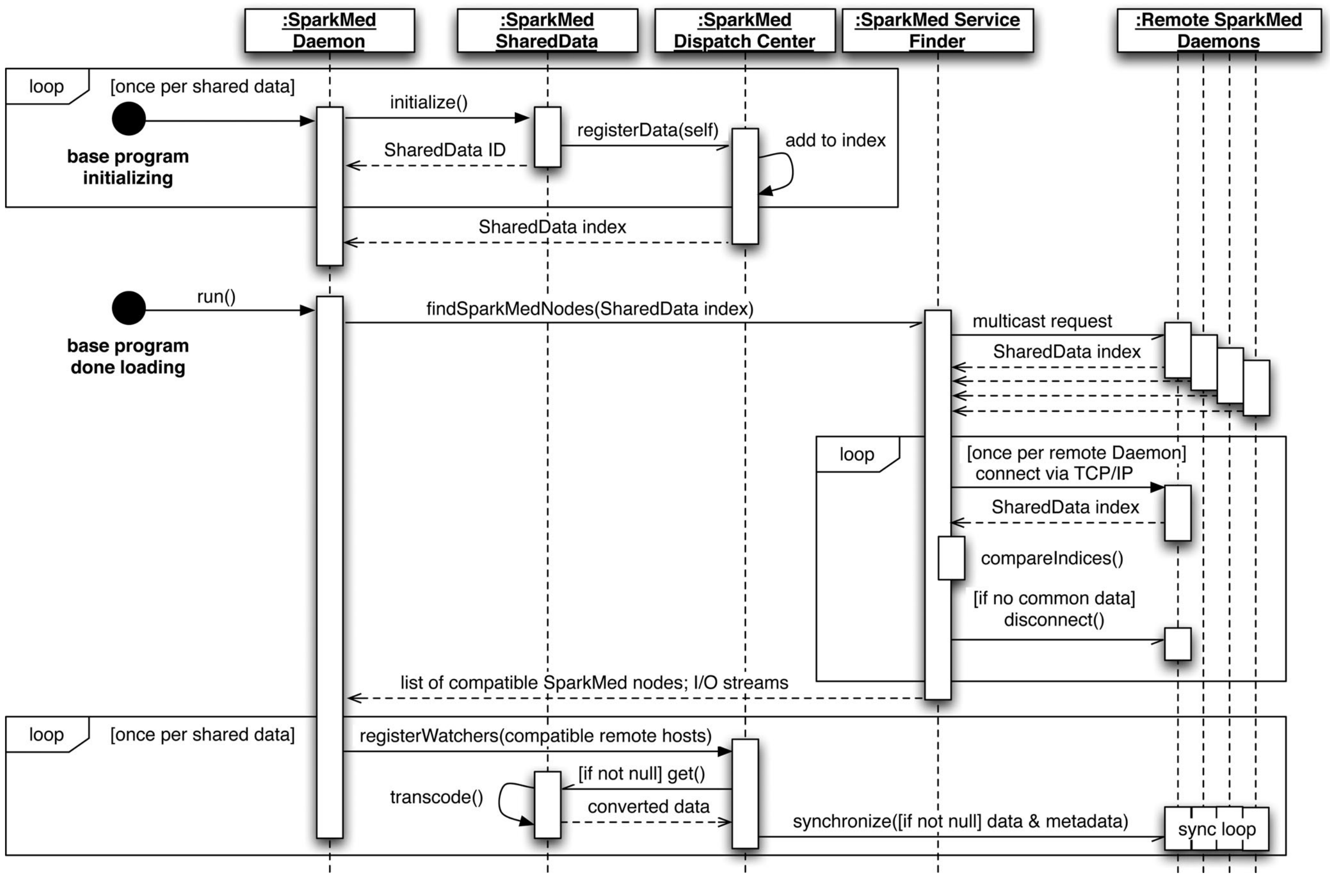

Fig. 3. Abstracted sequence diagram of the process followed by each daemon to create SparkMed's network of nodes, and begin the data synchronization process. Each daemon maintains an index of specific data items it creates and consumes, and an access list of other sources of these data on the network. These are used to notify interested nodes whenever the data change.

interconnected nodes, then for any given node the *delta centrality*, C^{Δ}, is calculated using

$$C^{\Delta} = \frac{(\Delta P)_i}{P} = \frac{P[\mathbf{G}] - P[\mathbf{G}']}{P[\mathbf{G}]} \tag{1}$$

where P is an observed ping value (propagation time of a simple data packet with and without the given node being used to forward the request) and $(\Delta P)_i$ is the ith variation of P where this node is no longer used to forward data as part of the SparkMed network. By $\mathbf{G}'$ we indicate a variation upon the SparkMed network graph $\mathbf{G}$, obtained by removing this node (and hence the graph edges it represents). This represents a similar approach to [34], using observed ping values as a measure of network cohesiveness, to specify the importance of each node to the overall network. This implies that the central node of a SparkMed network will change over time, although in practice, recalculation only occurs when warranted by failure or a loss of connectivity.

This measure determines which node is considered central. Data synchronization is achieved by means of a priority-based synchronization system that propagates any data changes to this central node, which in turn updates the entire network of nodes. Each noncentral node registers data of interest (informed by the host software), representing medical data types the node consumes. The network then ensures that each node is kept synchronized with its respective data source(s) at all times. Conflict resolution is applied in the case of incompatible changes, weighted by the access credentials of the responsible user (where applicable) and the time the change was made. Timing is calculated via POSIX timestamps to an accuracy of approximately 5 ms, while resynchronizing device time at regular intervals to account for clock drift.

Whereas SparkMed is a single integrated service (i.e., there is only one SparkMed network), each application that contains the SparkMed daemon may consume and synchronize different data, and as such numerous separate services can be operating within the SparkMed cloud at any given time. Although each discrete set of application data can be considered a separate web service, there are overlaps between the data used by different applications (such as between mobile and desktop versions of the same application) and a single application may use data from multiple sources (e.g., PACS imaging and hospital information system (HIS) patient data). As such, the number of central nodes is dependent on the number of discrete data sources, which may form a part of any number of applications or services. Specifically, if no single central node is capable of supporting, accessing, or storing all of the data types required, multiple central nodes will ensue, each of which is responsible for a different category of data.

C. Implementation Languages

The majority of autonomous threads operating within the network architecture, including daemons and the RIA interface, were developed using the Java programming language. This choice was made due to the write-once-run-anywhere promise of the Java virtual machine, and for compatibility with CORBA and potential integrability with other leading pervasive medicine frameworks such as the Java Context Awareness Framework [35]. For maximum compatibility with mobile and web-based devices, subsets of the SparkMed daemon were also implemented on the Apple iOS and Adobe Flash platforms. iOS was chosen because it is necessary to run natively on some of the most advanced and prevalent handhelds in the healthcare market. As to non-iOS devices, although Java may be considered ubiquitous on most other handhelds, our investigation of the literature [36] led us to choose the Adobe Flash IDE as the implementation platform for our RIA interface. SparkMed data components have been implemented for the Java, Objective-C, and several RIA environments.

D. Shared Data Propagation Functionality

The lightweight SparkMed daemon interacts with SparkMed data components placed inside the host medical software (referred to herein as “SharedData items”). These are simple data storage and monitoring objects which can replace (or extend) the standard primitive types and user interface (UI) components of the underlying development framework. These components’ functionality is equivalent to the corresponding primitive types or UI components, except that they are thread-safe, will automatically synchronize with networked equivalents, and provide a notification feature so that the base software can watch for changes in value. In the SparkMed UI components, data changes due to synchronization are treated as user input (allowing the base application to respond as normal). This ensures that the functionality of SparkMed does not interfere with the operation of the base medical software. The SparkMed daemon itself can be compiled directly into the application as a library, and need only be invoked in the code. Once started, its operations are automatic.

Where a suitable SparkMed data component is not available, the daemon can instead interface with the software indirectly. It is currently capable of SQL-requests and provides bridging functionality for a variety of medical database and PACS software, including Filemaker Pro. In cases where this is necessary, the database is treated as a subnode with no daemon of its own, and the SparkMed daemon active on the same machine becomes responsible for the synchronization of its data as normal.

E. Overlay Network and Communication Standards

Our protocol’s network architecture is analogous to the concept of a “personal overlay network,” as discussed in [37], in that it straddles the existing network topology separating hospital systems and handheld devices, using these devices as nodes in a peer-to-peer network. The loss of any one SparkMed node should not jeopardize the user’s data or the functionality of the system, as the network as a whole should be able to adapt to its removal (by sourcing data elsewhere, or attempting reconnection).

SparkMed nodes use a variety of medical and internet standards in their operations, as appropriate. The daemon itself reads medical information directly from the host application, and hence supports DICOM and HL7 formats and protocols as long as the base application does so, as well as containing innate support for consumer image and video formats. Mobile and desktop computer nodes communicate with one another via our own custom communication protocol over TCP/IP. Images are transmitted in JPEG format.

F. RIA Daemon

RIA or browser-based nodes run a stripped-down version of the SparkMed daemon written in a combination of JavaScript and ActionScript 3.0 which communicates via HTTP or HTTPS. This daemon is limited to the communication methods available to RIAs, and hence uses asynchronous XML-based requests, with the remote daemon emulating a web server and providing data in simple web formats in response to specially formed uniform resource indicators (URIs) which encode the requested data and synchronization information. Due to the limitations of Flash, the web browser platform, and the need to support the widest possible array of devices, image data are transcoded to lossless JPEG format, metadata are provided as XML, and entire image stacks (if required) are transmitted as lossy video in FLV format. For further discussion of the RIA subsystem, see Section IV-C.

G. Operational Details

Fig. 2 depicts the layers of the SparkMed protocol. The top represents a standard medical software application, running on a consumer PC or server. Irrespective of its own data processes, the data it loads are stored in memory as program variables and files, and the user manipulates the application via the standard UI components of the operating system. Our framework substitutes these variables, UI elements, and files with equivalent SharedData accessible to the SparkMed daemon. The daemon implements its own suite of services and uses its own protocol on top of the base application as shown in the figure. Each daemon connects to other instances of itself, by using the connectionless UDP protocol to probe the network for other SparkMed nodes. When such nodes are found, they may be one of three types: another piece of medical software implementing SparkMed, or a subset of SparkMed running as a mobile application or RIA. In the case of other desktop applications or mobile native applications, SparkMed establishes a TCP/IP connection and synchronizes the UI elements, variables, and file data between nodes. In the case of web applications, SparkMed implements an AJAX-like interface for XML-based communication of data, and a simple HTTP server for serving files, so as to allow for easy web-based access to the same data. Addressing of data is more complex for web-based applications, since the data are not directly synchronized. SparkMed implements a simple URI model to allow these applications to simply treat remote data as if it were a static file (or CGI script) on a web server, and reference it via dynamically assigned URIs.

Fig. 3 shows the process used by SparkMed to generate a network. Each daemon waits for its host application to start normally, before creating an index of shared data used or created by this application. Once all data have been registered, the daemon seeks out other SparkMed nodes on the network, compares this application's data index with theirs, and hence creates an access list of compatible nodes. These nodes are registered inside the daemon's internal Dispatch Center as watchers for the relevant shared data, and state information is exchanged so as to synchronize these data with the remote daemons. Thereafter, these watchers are notified whenever the data change, and watched for relevant status updates.

IV. Results

The functionality and capabilities of SparkMed were evaluated by means of a simulation study, detailed in Section IV-A, a resilience trial, detailed in Section IV-B, and a case study, detailed in Section IV-D. The results of these studies are discussed under their respective headings, and in Section IV-C.

A. Simulation Experiment

A simulation experiment was conducted to evaluate the interactive usability and overhead costs of our SparkMed framework, under the expected network conditions for normal use in a medical environment. Our simulation was modeled after a radiological workflow, and as such was centered on a server machine running a number of DICOM and HL7 data sources (medical imaging software expanded with our SparkMed daemon component to become networkable nodes). These data sources were combined to create a single medical workstation application, which used SparkMed to retrieve the necessary multimedia data, and transcode image data to lossless JPEG format. Multimodality data sources (e.g., DICOM JPEG) were first split into their component parts (e.g., DICOM metadata and JPEG image data) in order to deploy them as SharedData. Finally, this application was synchronized to a series of mobile devices using the SparkMed framework, thus turning this SharedData into a self-synchronizing network.

In order to model a wide array of potential telemedicine use cases for our system, and prove that it remains responsive and lightweight under various kinds of load, we ran performance and resource usage tests using an increasing number of consumer devices (up to 10), and across multiple network configurations. Specifications for the devices used in the trials are listed in Table I. Mobile SparkMed nodes were connected alternately using WiFi and 3G Internet connections, under the following network configurations.

1) Remote WiFi: The central node is inside a simulated hospital network. Mobile nodes are connected to the Internet over a WiFi connection, to represent standard home or Internet use.
2) Roaming 3G: The central node is within the simulated hospital network. Mobile nodes are outdoors, connected to the Internet over the mobile 3G network to simulate environments, e.g., rural, where a mobile phone's connection might be the only Internet available. Bandwidth fluctuates with reception. The test was performed in an area of limited 3G connectivity, so as to accurately represent a roaming (either rural, or moving between cell towers) environment.
3) Mobile Headless: In order to demonstrate the capability of SparkMed nodes to adapt to connection loss, the central

TABLE I
DEVICES USED IN SPARKMED RADIOLOGICAL SIMULATION

Device	Description	Processor	Memory	Connectivity	OS
Portable Server Node	Apple MacBook Pro 17”	2.33 GHz (dual-core)	3 GB	Wi-Fi/Ethernet	Mac OS X 10.6.7 (build 10J869)
Device 1	Apple iPhone 3G	412 MHz	128 MB	Wi-Fi/3G	Apple iOS 4.2 (build 8C148)
Device 2	Apple iPhone 3G	412 MHz	128 MB	Wi-Fi/3G	Apple iOS 4.2.1 (build 8C148)
Device 3	Apple iPhone 4 (GSM)	1 GHz	512 MB	Wi-Fi/3G	Apple iOS 4.1 (build 8B117)
Device 4	Apple iPhone 4 (GSM)	1 GHz	512 MB	Wi-Fi/3G	Apple iOS 4.2.1 (build 8C148)
Device 5	Apple iPad	Apple A4 1 GHz	256 MB	Wi-Fi	Apple iOS 4.3.3 (build 8J3)
Device 6	Apple iPad	Apple A4 1 GHz	256 MB	Wi-Fi	Apple iOS 4.3.3 (build 8J3)
Device 7	Apple iPad 2	Apple A5 1 GHz (dual-core)	512 MB	Wi-Fi/3G	Apple iOS 4.3.3 (build 8J2)
Device 8	Apple iPad 2	Apple A5 1 GHz (dual-core)	512 MB	Wi-Fi/3G	Apple iOS 4.3.3 (build 8J2)
Device 9	Apple iPad 2	Apple A5 1 GHz (dual-core)	512 MB	Wi-Fi	Apple iOS 4.3.3 (build 8J2)

node is removed from the network, forcing one or more of the mobile nodes (connected to the Internet over WiFi) to become the new central node(s).

Under each of these configurations, the following variables were measured.

1) Propagation time—the time it takes for data to fully propagate through the network, i.e., the time between registering a change in its value, and the entire network having been brought up-to-date. This includes both the transmission time and the time taken to confirm that all nodes are in sync.
2) Frames per second—measuring the FPS of medical image rendering performance observed.
3) Memory usage—the increase in memory use the host application observes, recorded to allow measuring of the memory overhead of SparkMed.
4) Bandwidth usage—the rate, in KB/s, at which SparkMed sends and receives data.
5) Processor load—the increase in strain the central processing unit is under. This represents the overhead added by SparkMed to the base application. Because processor load tended toward inconsistency due to “peaks” in the data, samples were averaged over ten iterations.

In our study, our criterion for acceptably interactive performance was a “near-real-time” updating of the device’s data and visualization content as it was manipulated at the remote side. In this context, we have found this to require at least 5 FPS for image streaming, with a propagation time of under 0.5 s. This value was derived empirically, representing the approximate cutoff at which students and physicians would no longer consider the system “responsive.” These responsive rates ensure that the data and UI of all collaborating mobile hosts are always in synchronicity, with an error margin of under 0.5 s. We suggest that this represents a suitable level of interactivity to provide radiological staff with sufficient responsiveness for various clinical applications.

We ran controlled trials using up to ten active mobile SparkMed nodes, nine of them are handheld devices. In the experiment requiring a handheld to become the server node, there was consequently one less node available. Likewise, not every device supported 3G, limiting the amount that could be used in our 3G trial. Although typically multiple central nodes could be created in the mobile headless trial, the storage and processing capabilities of the iPad 2 nodes ensured that one of them would always be chosen as a central node.

To remove human factors from the simulation, all of the trials were run using the same prescripted sequence of medical image slices (fused PET/CT data), where the viewer application was instructed to render and repeatedly propagate the same series of ten images to every node on the network—navigating to another slice each time all nodes confirmed successful download. Images were 8-bit RGB, with a resolution of 171×111, with an average frame size of 24 210 bytes.

Each of these controlled trials was repeated ten times, and the variables listed earlier were recorded. A series of control results were also recorded, using simulated UI input to perform the same manipulations on the non-networked, nonsynchronizing original version of the software, for the purpose of determining overhead added by SparkMed operations by comparing observed SparkMed results to the control.

B. Resilience Trial

A secondary experiment was conducted in order to prove the resilience of the SparkMed framework to common wireless network issues. Several configurations and usage scenarios were tested, as listed later. In each case, the variable recorded was the time taken for the SparkMed network to re-establish the links between the active nodes, and resume sending data.

1) Congestion: A SparkMed network was set up containing one data source node and one mobile client. A simulated congested-network situation was created by artificially introducing data loss on the band, causing the client node to attempt another means of connection. This was tested over 3G and WiFi, with the node switching over to the alternate method in each case.
2) Crash: A SparkMed network was set up containing one data source node and two mobile clients. A synchronization loop was entered, with each node changing the data and propagating that change in turn. A simulated crash was induced in one of the client nodes, by using a remote debugger to induce an inconsistent state. The time taken between network failure and successful re-establishment of a consistent synchronization loop was measured over both WiFi and 3G.
3) Headless: A SparkMed network was set up containing two data source nodes and one mobile client. The particular data source chosen by the client node to request data from was then physically removed from the network. The time

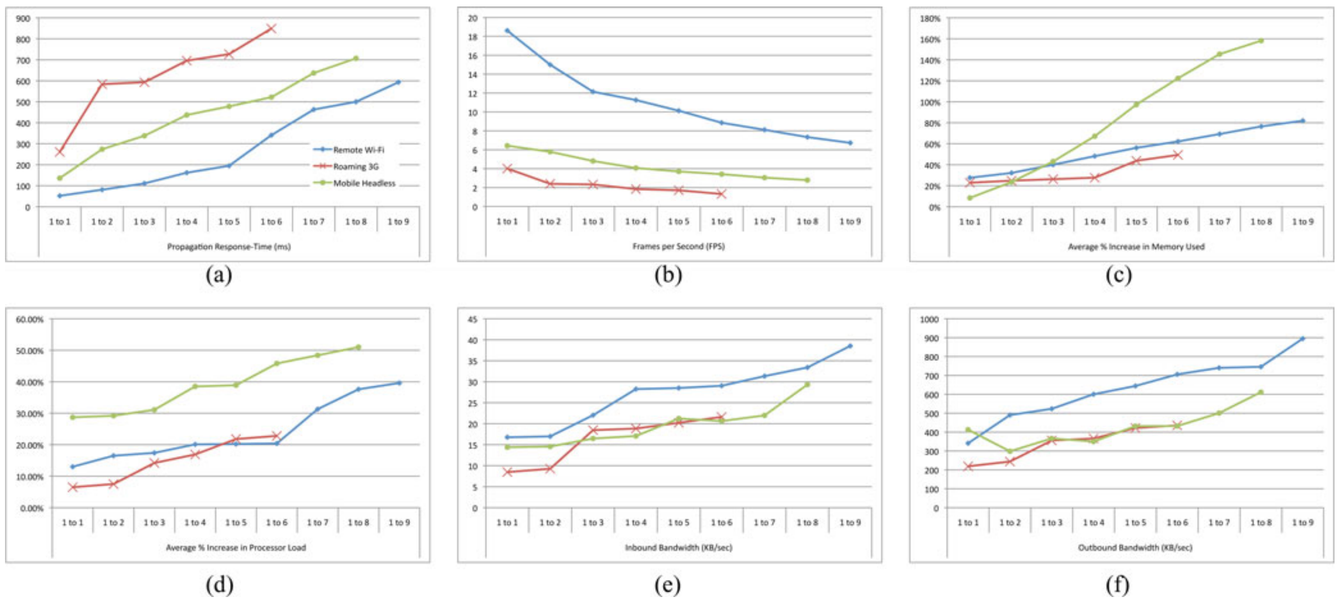

Fig. 4. Simulation results for SparkMed radiology workflow under remote WiFi, roaming 3G, and mobile headless conditions. (a) Time taken for multimedia data to fully propagate throughout the SparkMed network. (b) Frames per second of SparkMed graphical performance over WiFi, 3G, and using a mobile phone as a server. (c) Average memory usage in MB of PET/CT fusion viewer workstation with SparkMed active. (d) Average processor load of PET/CT fusion viewer workstation with SparkMed active. (e) Inbound bandwidth usage of SparkMed daemon during our trials. (f) Outbound bandwidth usage of SparkMed daemon during our trials.

taken for the client node to shift to the node with lesser centrality and resume normal operation was measured.

The results are shown in Fig. 5, displaying both the time taken with SparkMed's automatically created access list, and without. By keeping a prioritized list of other node, each SparkMed daemon can significantly improve its recovery time in case of service interruption. As shown in the diagram, this is vital over 3G, as our daemon is unable to discover a pre-existing SparkMed network over 3G without at least one server in its access list.

C. *Results Analysis*

Fig. 4(a) graphs the time taken for data to propagate through the entire SparkMed network, where the responsiveness of SparkMed is shown to be high, with propagation times remaining under 200 ms irrespective of what kind of device (server or mobile device) is acting as the central server node. The same is true of connecting to SparkMed solutions over 3G Internet. If more than one remote client per workstation chooses to access it over 3G Internet, this response time increases sharply. These data suggest that, under the intended usage criteria for SparkMed, propagation of data through the SparkMed network happens well under the desired 0.5 s. In the case of more than five simultaneous users, however, worst-case performance is shown to fall short of our requirement with six clients using a mobile server, or nine for a laptop server. Further, the unreliable nature of 3G Internet makes desynchronization common if there are multiple roaming 3G nodes providing constant activity. As seen in the graph, a single 3G node is relatively reliable on the network, but additional 3G clients beyond that point increase the error rate of the channel sufficiently to significantly reduce the data rate that can be guaranteed.

Fig. 4(b) shows the graphical performance of the SparkMed-based PET/CT viewer in our simulation, under a number of network conditions. In every case, the frame rate of the application drops as expected in response to the addition of further networked nodes. Our results demonstrate that even with a single data source, a single laptop can service up to nine mobile devices without losing interactive frame rates, and one mobile device can reliably furnish data to at least two others before the loss of a nonhandheld data source even becomes noticeable. When connecting over 3G, however, the lower data rate lowers the observed worst-case FPS significantly, particularly when numerous clients connect over 3G. Nevertheless, a one-to-one telemedicine collection exhibits frame rates very close to the 5 FPS target.

Fig. 4(c) shows the average memory usage of our data source during the trial (although the observed value is skewed upward by the observer effect, due to SparkMed simultaneously recording the charted data). This value is expressed as a percentage increase over the memory requirements of the base system, and can be shown to increase steadily as more clients are added. As expected, much more of the device's memory must be set aside in the case of a mobile device becoming the central node, as the synchronization status of every shared-data item on the network must be stored in its memory.

Fig. 4(d) shows processor load incurred through the use of SparkMed. This is expressed as a percentage increase over the CPU usage of the base application, inclusive of the additional processing that must be done to render and transcode images for mobile deployment. These data are highest for handheld device central nodes as expected. The data also illustrate that a laptop device is able to service up to six handheld nodes without incurring more than approximately 20% overhead. CPU usage

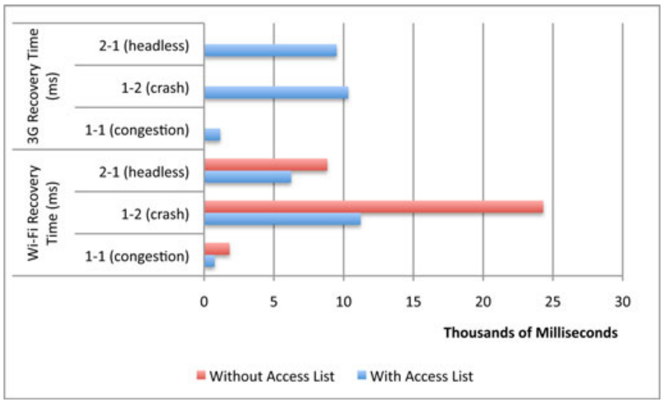

Fig. 5. Time taken for the SparkMed network to recover from adverse conditions.

is lowest in the 3G trial, as the system is forced to do more waiting.

Figs. 4(e) and (f) show the bandwidth usage of SparkMed, during each trial. As expected, much higher data rates can be sustained using a nonhandheld WiFi node as the data source. Although handheld servers perform reasonably for small numbers of clients, the mobile and 3G trials display similar bandwidth usage with larger numbers of concurrent users, as in each case there is a limiting factor (the network, or the capabilities of the device) that reduces the potential of the method. Note that due to the design of this trial, the performance results recorded do not in fact represent the highest possible frame rate and response time possible for the SparkMed network: rather, this is the highest rate of performance which can be guaranteed for *every* device in the network, ensuring that they remain in sync at all times.

Fig. 5 shows that recovery times for the SparkMed protocol are very similar across both 3G and WiFi, owing largely to the access list of compatible nodes it maintains. Sourcing data via an alternate connection to overcome congestion is a rapid switch, causing very little downtime, and even breakdowns in the network are automatically overcome within a short timeframe. When the access list contains no known-good nodes, rerouting and crash recovery are at their weakest, but still complete successfully. The service discovery process, however, typically discovers other nodes on the network very quickly.

Overall, our results show that, for the identified medical use cases (typically 1-1 or 1-2) especially, SparkMed performance lies within our targeted "near-real-time" performance (5 FPS, 0.5 s response time) over WiFi and retains usable performance under 3G. Further, the protocol recovers gracefully from errors, and can retain this performance for each application with a much larger number of users if a nonhandheld device is the central node. The overhead from implementing SparkMed is relatively low, although memory usage rises steeply if SparkMed's intended usage parameters (i.e., the radiology use cases of 1-1 or 1-2) are exceeded.

D. Case Study: Web-Based Radiology Workflow for Nuclear Medicine

The workflow of clinical radiologists primarily involves generating a variety of multimedia metadata, including annotations to the volumetric image data, textual descriptions, and amendments to the patient record, as well as audio-visual reports and diagnoses. In order to generate these data, they require access to not only the source radiological imagery but also the full patient record and the interactive capability to adjust color, contrast, brightness, viewing angle, multimodality fusion, and navigation.

At our partner institution, image and report data are stored in two places: on a PACS in DICOM and HL7 formats, and on a purpose-built radiology information system (RIS) implemented in FileMaker Pro. The data generated by radiologists are typically communicated to the referring physician/staff by means of analogue delivery methods (CD, printed report) or by email. Such delivery limits the varieties of media available, and transfers only a subset of the full diagnostic data from the RIS, e.g., a textual description, sometimes supplemented with keyframe images from the PACS. Thus, the rich, detailed information in the volumetric image data and the patient's medical history—as included in the EPR or HIS—are often effectively discarded.

To address the problem of accessing these data as a whole and retaining the rich contextual information, a web-based RIA subsystem (based on our previous work in browser-based telemedicine, for the technical details of which see [38]) has been integrated into our SparkMed framework to support volumetric image navigation in a browser, and provide concurrent access to the hospital database system (HIS or RIS). By making use of this RIA subsystem, we have created a portable radiology workstation system that requires no installation and runs in any standard web browser on any Internet-capable device. It is fully DICOM compliant, and supports multimodality PET/CT navigation and fusion, allowing slices to be visualized in any orthogonal plane, or as a 3-D maximum intensity projection (MIP). It supports image and lookup-table manipulation, and has the potential to improve remote collaboration, medical education, and emergency record access.

The overall goal of this system is to actively collect and forward data from a series of heterogeneous systems within the hospital to a client machine outside of the same network, with an unknown configuration and capabilities. The technical challenges in developing the system were the effective and interactive streaming of volumetric 3D data to an unknown device, transcoding said data on-the-fly into web-compatible form, and the human interface concerns of providing the same interaction capability as three separate desktop-based medical systems within a potentially very small handheld UI. We were able to solve the problem using a simple mobile UI, connected to an advanced visualization system using our SparkMed framework. Figs. 5 and 6 show the resulting web-based radiology software running on an iPod Touch and in the browser of a low-end SmartPhone (Nokia N95) device, respectively.

To illustrate the operation of our framework, the following example scenario is outlined.

Due to the limited number of fully trained radiology personnel, as is becoming increasingly common, a radiology assistant is taking the evening shift. Given the size of the waiting list, and the number of scans that must be performed each day, a significant backlog of cases is building, which these radiology trainees clear during the night.

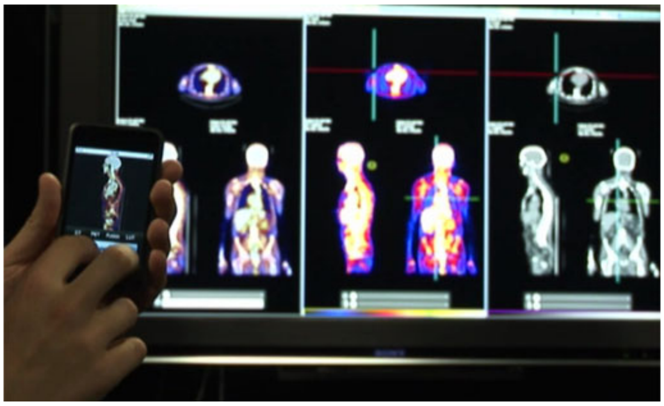
Fig. 6. SparkMed in use synchronizing a handheld visualization system (shown being manipulated by hand on the left of the image) with the hospital's corresponding data source (shown on the screen in the background).

Fig. 7. Our framework's web-based transcoding allows the seamless operation of advanced hospital imaging software from any Internet-capable device, such as this Nokia N95.

The assistant's work, however, is interrupted by an uncertainty. The abnormality on a scan indicates that the patient's lymphoma may be spreading to his/her lungs, or it may represent a new condition. To confirm or deny this uncertainty, a request for an outside opinion is made using the desk phone to contact the on-call expert radiologist's mobile. Having received the call and spoken briefly, the expert starts the relevant m-Health application—or, if there is no such application available, the on-call radiologist can instead visit a URL via the Smartphone's mobile browser, thus launching a RIA to the same effect.

The SparkMed application networks itself automatically (as shown in Fig. 2), and the required data items are then automatically synchronized by the SparkMed framework: the remote user is immediately delivered the same database data, metadata, imagery, and variables in use on the initiator's machine (or machines), at which point he/she is able to manipulate each connected system directly, with results visible immediately to both parties. With collaboration thus established, meaningful diagnostic discussion can occur.

1) Lessons From the Case Study: A great deal was learned in creating this web-based radiological system, particularly with respect to user acceptance. It was found that users familiar with high-end radiological software considered the mobile equivalent far more usable if, despite the mobile form factor, it presented the same options as the equivalent hospital workstation at a glance. Although some display space must be sacrificed, clearly labeled buttons for every function and view resulted in more positive feedback, particularly if the resulting interface used "push" or "slide" animations in such a way as to suggest the user was navigating a much larger interface, with related tools grouped into consistently placed subinterfaces which could be animated into view when needed. Our web-based radiology workstation implements these techniques directly.

Also of interest was the observation that a system that exhibited hysteresis was considered much less acceptable. Users expected immediate feedback when manipulating familiar UI elements, and delays between action and reaction led to user dissatisfaction. Because responsivity is often low in a browser-based or web environment, mostly due to additional overhead and the limitations of individual devices, it was found preferable to perform preliminary image processing on the mobile device and use video streaming techniques, so as to limit insofar as possible the communication delay observed by the user. For this reason, SparkMed's RIA subsystem implements FLV-based streaming of entire image stacks, to ensure smooth navigation, and non-networked simple image transforms (to avoid or supplement server requests).

Finally, it was found insufficient to simply translate interface elements directly to the mobile client. Users engaging with the same interface on a mobile device immediately attempt to engage by means of mobile and touchscreen gestures such as the "swipe" and "pinch." For this reason, SparkMed implements specific networked UI elements, the implementation of which differs for mobile and web clients, so that each element can adapt its functionality to the context.

2) Limitations and Future Potential: To use our framework in a browser-based environment, sacrifices must be made in terms of acceptable formats, Quality of Service (QoS), and data synchronization. Due to browsers' security restrictions and because all data are carried via HTTP across asynchronous XML-based channels, or as discrete files, communication is largely one-way and it is difficult to keep RIA-based clients in sync with the remainder of the SparkMed network. At present, the solution adopted by our RIA subsystem is to use a regular SparkMed node (typically on a separate machine) as a "gateway" to the SparkMed network, using HTTP to synchronize data and requests with that machine, and discovering data changes by polling periodically. As such, the burden of maintaining synchronization is shifted to this second node.

While this approach results in a functional system with very wide compatibility, which appears identical to other SparkMed nodes, the need for polling uses additional bandwidth. Further, functionality is limited by the one-way channel, and complexity increased by the need to avoid hysteresis. Finally, the resolution and lack of input capabilities of some mobile devices (such as many clamshell phones) require a reasonably complex interface to be implemented: there are a number of potential display configurations to adapt to, and input options such as the thumbstick, keypad, touchscreen, hot-pluggable keyboard, mouse, etc., all must be catered for. This requires significant additional programming, undermining the ability of SparkMed to automate the mobile deployment process.

With the advent of HTML5 and CSS3, it may be possible to perform the necessary processing directly in the browser to improve efficiency by not using extensions such as Adobe Flash. Further, if the RIA subsystem were to generate HTML5 pages directly, CSS could be used to automate much of the process of adapting applications for specific devices and input methods. Also, this new web standard supports two-way communication not available in current AJAX approaches. As such, SparkMed has the potential to deploy fully functional nodes directly into web browsers in future, and integrate them seamlessly into the SparkMed cloud.

V. Discussion

Mobile devices have been an important tool in healthcare for a long time. Fischer *et al*. (in [39]) listed the uses of handheld mobile devices in medicine in 2003, and this drive toward adoption of mobile devices at every level of the healthcare enterprise has gained ground in recent years [40]–[42]. There has been a huge increase over time in how many mobile devices, such as PDAs, are used by medical doctors. The next generation of medical practitioners is likely to be avid users of mobile technology, and to expect the devices to expand their usability and capabilities quickly, with m-Health solutions adapting and expanding accordingly.

Our proposed SparkMed framework was evaluated within our own simulation environment, which we modeled on an actual radiology department. Due to the data management needs of this hospital and the nature of its clinical information system with relation to those of other Australian institutions [43], we believe that our simulation environment represents a roughly equivalent information and network architecture to that of most medium-to-large-sized Australian healthcare providers. Further, accessing this same simulation environment via 3G Internet was used to exemplify the situation in rural Australia, where medical data transfer is a perennial issue, and patients can expect little more connectivity than a mobile phone connection with which to access multimedia data [44].

SparkMed's ability to bind to medical applications and transcode-and-forward their data for mobile use bypasses many of the most intransigent issues associated with medical data. The cloud network created by SparkMed has the potential to extend the reach of hospital infrastructure to allow data access beyond the typical limitations of proprietary systems, and the fact that it is generated and managed automatically may reduce the need to expensively decommission legacy software without in-built networking support, by potentially allowing it to be retrofitted with SparkMed instead. Likewise, SparkMed's automatic transcoding feature could avoid issues relating to storage and processing of medical data, by transferring data progressively in web-appropriate formats, and allowing the necessary processing to be done in the cloud. Finally, the use of SparkMed can avoid the need for a centralized repository or common standard: this cloud approach could potentially serve as a powerful alternative, providing the benefits of these methods through a distributed data store and automatic transcoding.

A. *Storage and Security*

As a cloud solution that requires little to no infrastructure support, SparkMed aims to provide many of the benefits of an off-site PACS or SOA solution, without the high associated costs. The majority of these costs, however, are associated with storage and security: moving personal data into the cloud and making it widely accessible, but at the same time maintaining the privacy and security inherent in the in-house hospital system. This is a difficult challenge, and a major research field in and of itself, but despite being a prototype system SparkMed has made some major steps in that direction.

In order to avoid the prohibitive cost and infrastructure requirements of off-site storage, and avoid the security risks of lost or stolen mobile devices, the SparkMed framework leaves medical source data within the hospital network, forwarding only state data and the necessary in-use imagery to its network nodes. SparkMed information is held in memory without being saved to disk, to the limits of the mobile device, and hence does not represent a security risk from being available on a mobile device unless the medical application is logged in, running, and in the middle of the diagnostic process when stolen or lost. Even then, the remote user can remove that node, or control what that user sees (if anything). Further, from a transmissions standpoint, SparkMed already supports industry-standard encryption in the form of HTTPS. While the prototype does not do so for performance reasons, the framework also has the potential to encrypt all communications by using encrypted data streams. 3G transmissions are, of course, already encrypted by the provider, providing an additional layer of protection when the user is not within a trusted network.

The real barriers to the deployment of SparkMed are, in fact, legal rather than technical. While the legal environment differs between countries (for the case in Australia, see [45]), transmitting patient data over the network for the purposes of collaboration typically necessitates certification by a variety of bodies, and may fall afoul of local or federal guidelines. Further, interorganizational collaboration and teleradiology are not currently covered adequately by legislation and insurance contracts, making it difficult for radiologists to be compensated for this kind of work. Our framework presents a technical solution to the problem, such that when need for such electronic medical record transmission has been legally ratified, a practical approach exists to achieve multimedia medical data integration and distribution.

B. *Future Work*

Our upcoming work will expand our framework to incorporate scalability and QoS issues which are expected to largely optimize our performance, in addition to investigating more advanced modes of streaming multidimensional image data, such as the method proposed by Lamberti and Sanna in [46], and adapting the peer-to-peer networking code to reduce power consumption of individual nodes, and maximize throughput, as in [47]. Future work will also include more fine-grained QoS through the use of redundant preprocessed streams at varying bit rates; improvements to its capability for real-time visualization by harnessing medical region-of-interest knowledge and the processing power of the client device to deliver the important data first, and take on some of the necessary image processing tasks with mobile hardware.

VI. Conclusion

This paper presented SparkMed, a framework to enable mobile access to multimedia medical data to a wide range of Internet-capable and mobile devices. We have outlined the functionality of the system, demonstrated its capability to interactively deploy medical multimedia systems to mobile device clients, and intelligently synchronize and propagate medical data from a variety of heterogeneous sources in a convenient, reliable manner without appending significant overhead to the underlying process.

Our prototype and case scenario evaluated the effectiveness of the SparkMed architecture in an environment designed to simulate a real hospital and telemedicine setting. Within the context of our simulated radiological workstation, our prototype demonstrated highly interactive usability and low overhead cost requirements, proving its suitability and effectiveness in similar hospital contexts.

Acknowledgment

The authors would like to thank their collaborators at the Royal Prince Alfred Hospital for their invaluable help and clinical advice, and access to their PACS and RIS infrastructure.

References

[1] Foundation for the National Institutes of Health. (2009). *The Inaugural mHealth (mobile health) Summit*, Washington D.C. [Online]. Available: http://www.fic.nih.gov/news/events/mhealthsummit.htm

[2] D. Vatsalan, S. Arunatileka, K. Chapman, G. Senaviratne, S. Sudahar, D. Wijetileka, and Y. Wickramasinghe, "Mobile technologies for enhancing eHealth solutions in developing countries," in *Proc. 2nd Int. Conf. eHealth, Telemed., Soc. Med., eTELEMED 2010, Includes MLMB 2010; BUSMMed 2010*, pp. 84–89.

[3] R. Istepanian, C. S. Pattichis, and S. Laxminarayan, "Ubiquitous m-health systems and the convergence towards 4G mobile technologies," in *M-Health: Emerging Mobile Health Systems*. New York: Springer-Verlag, 2005, pp. 3–14.

[4] J. A. Hernandez, C. J. Acuna, M. V. de Castro, E. Marcos, M. Lopez, and N. Malpica, "Web-PACS for multicenter clinical trials," *IEEE Trans. Inf. Technol. Biomed.*, vol. 11, no. 1, pp. 87–93, Jan. 2007.

[5] I. Balasingham, H. Ihlen, W. Leister, P. Roe, and E. Samset, "Communication of medical images, text, and messages in inter-enterprise systems: A case study in Norway," *IEEE Trans. Inf. Technol. Biomed.*, vol. 11, no. 1, pp. 7–13, Jan. 2007.

[6] A. Rosset, L. Spadola, and O. Ratib, "OsiriX: an open-source software for navigating in multidimensional DICOM images," *J. Digital Imag.*, vol. 17, no. 3, pp. 205–216, Sep. 2004.

[7] A. Kailas, C. Chong, and F. Watanabe, "From mobile phones to personal wellness dashboards," *IEEE Pulse*, vol. 1, no. 1, pp. 57–63, Jul.–Aug. 2010.

[8] 2007 IBM Report on Health Care. (2007), "Healthcare 2015: Win-win or lose-lose? A portrait and a path to successful transformation," IBM Institute for Business Value, pp. 1–8. [Online]. Available: http://www-935.ibm.com/services/us/gbs/bus/pdf/healthcare2015-win-win_or_lose-lose.pdf

[9] J. Philbin, F. Prior, and P. Nagy, "Will the next generation of PACS be sitting on a cloud?" *J. Digital Imag.*, vol. 24, no. 2, pp. 179–183, Apr. 2011.

[10] C. Costa, C. Ferreira, L. Bastiao, L. Ribeiro, A. Silva, and J. L. Oliveira, "Dicoogle—An open source peer-to-peer PACS," *J. Digital Imag.*, vol. 24, no. 5, pp. 848–856, Oct. 2010.

[11] Merge Healthcare, Inc. (2011, Jun. 1). "AMICAS PACS—The first 100% web-based PACS system," [Online]. Available: www.merge.com/products/pacs/amicas-pacs/index.aspx

[12] S. G. Langer, "Challenges for data storage in medical imaging research," *J. Digital Imag.*, vol. 24, no. 2, pp. 203–207, Apr. 2011.

[13] T. J. Farnsworth, "PACS for imaging centers," *Radiol. Manage.*, vol. 25, no. 3, pp. 36–41, 2003.

[14] S. B. El-Ghatta, T. Clade, and J. C. Snyder, "Integrating clinical trial imaging data resources using service-oriented architecture and grid computing," *Neuroinformatics*, vol. 8, no. 4, pp. 251–259, Dec. 2010.

[15] T.-H. Yang, Y. S. Sun, and F. Lai, "A scalable healthcare information system based on a service-oriented architecture," *J. Med. Syst.*, vol. 35, no. 3, pp. 391–407, Jun. 2011.

[16] B. Silverman, O. Sokolsky, V. Tannen, A. Wong, and L. Lang, "HOLON/CADSE: Integrating open software standards and formal methods to generate guideline-based decision support agents," in *Proc. AMIA Annual Symp.*, 1999, pp. 955–959.

[17] R. Kapitza, H. Schmidt, G. Soeldner, and F. Hauck, "A framework for adaptive mobile objects in heterogeneous environments," in *OTM Conference* (Lecture Notes in Computer Science). New York: Springer, 2006, pp. 1739–1756.

[18] W. Grimson, D. Berry, J. Grimson, G. Stephens, and E. Felton, "Federated healthcare record server—the Synapses paradigm," *Int. J. Med. Informat.*, vol. 52, no. 1, pp. 3–27, 1998.

[19] D. Sullivan, K. Farion, S. Matwin, and W. Michalowski, "A concept-based framework for retrieving evidence to support emergency physician decision making at the point of care," in *Knowledge Management for Health Care Procedures*, (Lecture Notes in Computer Science vol 4924), 2008, pp. 117–126

[20] M. Tsiknakis, M. Brochhausen, J. Nabrzyski, and J. Pucacki, "A semantic grid infrastructure enabling integrated access and analysis of multilevel biomedical data in support of postgenomic clinical trials on cancer," *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 2, pp. 205–217, Mar. 2008.

[21] W. Cai, D. Feng, and R. Fulton, "Web-based digital medical images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 1, pp. 44–47, Jan./Feb. 2001.

[22] E. Lim, R. Khoury, L. Tarlinton, A. Mohamed, D. D. Feng, and M. Fulham, "A medical imaging web-based electronic patient history (EPH) via a personal assistant (PDA) in a wireless environment," *J. Nuclear Med.*, vol. 47, no. 1, p. 367, 2006.

[23] L. Constantinescu, J. Kim, C. Chan, and D. D. Feng, "Automatic mobile device synchronization and remote control system for high-performance medical applications," in *Proc. 29th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Jul. 2007, pp. 2799–2802.

[24] L. Constantinescu, J. Kim, M. Fulham, and D. Feng, "Rapid interactive smartphone access to PET-CT data for improved patient care," *J. Nucl. Med.*, vol. 50 (Suppl. 2), p. 427, 2009.

[25] K. Leiviska, "Physicians' views toward mobile HIS," in *Proc. 30th Inf. Syst. Res. Seminar in Scandinavia*, 2007, p. 14.

[26] C. Lefebvre, "Integrating cell phones and mobile technologies into public health practice: A social marketing perspective," *Health Promot. Pract.*, vol. 10, no. 4, pp. 490–494, Oct. 2009.

[27] M. N. K. Boulos, S. Wheeler, C. Tavares, and R. Jones, "How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from eCAALYX," *Biomed. Eng. Online*, vol. 10, p. 24, 2011.

[28] R. J. Toomey, J. T. Ryan, M. F. McEntee, M. G. Evanoff, D. P. Chakraborty, J. P. McNulty, D. J. Manning, E. M. Thomas, and P. C. Brennan, "Diagnostic efficacy of handheld devices for emergency radiologic consultation," *Amer. J. Roentgenol.*, vol. 194, no. 2, pp. 469–474, Feb. 2010.

[29] F. Lamberti and A. Sanna, "Extensible GUIs for remote application control on mobile devices," *IEEE Comput. Graph. Appl.*, vol. 28, no. 4, pp. 50–57, Jun. 2008.

[30] C. Rolim, F. Koch, C. Westphall, J. Werner, and Fracalossi, "A cloud computing solution for patient's data collection in health care institutions," in *Proc. 2010 2nd Int. Conf. eHealth, Telemed., Soc. Med.*, 2010, pp. 95–99.

[31] J. Parekh, G. Kaiser, P. Gross, and G. Valetto, "Retrofitting autonomic capabilities onto legacy systems," *Cluster Comput.*, vol. 9, no. 2, pp. 141–159, 2006.

[32] A. Gonsalves, "IBM releases blueprint for automated computing," *TechWeb News* [Web article], Apr. 4, 2003.

[33] D. Steinberg, S. Cheshire, and M. Loukides, *Zero Configuration Networking: The Definitive Guide*, M. Loukides, Ed. Cambridge, MA: O'Reilly Media, Inc., 2006.

[34] V Latora and M Marchiori, "A measure of centrality based on network efficiency," *New J. Phys.*, vol. 9, no. 6. p. 188, 2007.

[35] J. E. Bardram, "The Java Context Awareness Framework (JCAF)—A service infrastructure and programming framework for context-aware applications," in *Pervasive Computing*, Munchen, 2005, pp. 98–115. Available: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.94.9175.

[36] T. Lammarsch, W. Aigner, A. Bertone, S. Miksch, T. Turic, and J. Gartner, "A comparison of programming platforms for interactive visualization in web browser based applications," in *Proc. 12th Int. Conf. Inf. Vis.*, Jun. 2008, pp. 194–199.

[37] T. Roscoe, "Network architecture test-beds as platforms for ubiquitous computing," *Philos. Trans. Series A, Math., Phys., Eng. Sci.*, vol. 366, no. 1881, pp. 3709–3716, Oct. 2008.

[38] L. Constantinescu, J. Kim, and D. Feng, "Integration of interactive biomedical image data visualisation to Internet-based Personal Health Records for handheld devices," in *Proc. 11th Int. Conf. e-Health Netw., Appl. Serv. (Healthcom2009)*, pp. 79–83.

[39] S. Fischer, T. Stewart, S. Mehta, R. Wax, and S. Lapinsky, "Handheld computing in medicine," *J. Amer. Med. Informat. Assoc.*, vol. 10, no. 2, pp. 139–149, Mar. 2003.

[40] K. Leiviska, "Physicians' views toward mobile HIS," in *Proc. 30th Inf. Syst. Res. Semin. Scand.*, 2007, p. 14.

[41] J. E. Bardram, A. Mihailidis, and D. Wan, *Pervasive Computing in Healthcare*. Boca Raton, FL: CRC Press, 2006.

[42] Y.-C. Lu, Y. Xiao, A. Sears, and J. A. Jacko, "A review and a framework of handheld computer adoption in healthcare," *Int. J. Med. Informat.*, vol. 74, no. 5, pp. 409–422, 2005.

[43] L. Robertson, "Selecting a core clinical computerised information system," *Health Inf. Manage.*, vol. 24, no. 3, pp. 104–107, Sep. 1994.

[44] D. Pierce and G. Fraser, "An investigation of medication information transfer and application in aged care facilities in an Australian rural setting," *Rural Remote Health*, vol. 9, no. 3, p. 1090, 2009.

[45] NEHTA Australia. (2011, Sep. 29). *NEHTA—National E-Health Transition Authority* [Online]. Available: http://www.nehta.gov.au/

[46] F. Lamberti and A. Sanna, "A streaming-based solution for remote visualization of 3D graphics on mobile devices," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 2, pp. 247–260, Mar./Apr. 2007.

[47] G. Jourjon, T. Rakotoarivelo, and M. Ott, "Models for an energy-efficient P2P delivery service," in *Proc. 18th Euromicro Conf. Parallel, Distrib. Netw.-Based Process.*, 2010, pp. 348–355.

Liviu Constantinescu (S'07) received the BIT degree (Hons.) from the University of Sydney, Sydney, N.S.W., Australia, in 2006, majoring in software development and multimedia technology. He is currently working toward the Ph.D. degree in the Biomedical and Multimedia Information Technology (BMIT) Research Group, School of Information Technologies, University of Sydney.

He has worked part time in medical research since 1998, transitioning from IT support to research work. His research interests include cloud computing, pervasive web-based systems, and mobile healthcare.

Jinman Kim (S'02–M'06) received the Ph.D. degree in computer science from the University of Sydney, Sydney, N.S.W., Australia, in 2005.

He has been an Australian Research Council Postdoctoral Research Fellow since 2007. In 2009, he joined MIRALab, University of Geneva, Switzerland, as an Experienced Researcher for a European Marie Curie research project—3-D Anatomical Human. He is currently the Director of the Telemedicine Lab, the Biomedical and Multimedia Information Technology Research Group, School of Information Technologies, University of Sydney. He is also a member of the Research Committee and Coordinator of the Imaging, Visualization and Information Technologies Research Theme of the newly established Institute of Biomedical Engineering and Technologies, Faculty of Engineering and Information Technologies, University of Sydney. His research focuses on the convergence of information technologies and biomedicine, aiming at improving our quality-of-life and increasing our understanding of diseases such as cancer and dementia. His research interests include medical image data analysis, multimodal visualization, and eHealth/telemedicine.

(David) Dagan Feng (S'88–M'88–SM'94–F'03) received the M.E. degree in electrical engineering and computer science from Shanghai Jiao Tong University, Shanghai, China, in 1982, the M.Sc. degree in biocybernetics and the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), in 1985 and 1988, respectively. During his Ph.D. studies, he received the Crump Prize for Excellence in Medical Engineering.

He has been a Professor and Head of the Department of Computer Science/School of Information Technologies, and is currently the Director of the Institute of Biomedical Engineering and Technology, University of Sydney, Sydney, N.S.W., Australia. He has published more than 500 scholarly research papers, pioneered several new research directions, and made a number of landmark contributions in his field.

Dr. Feng is also a Fellow of the Australian Computer Society, The Hong Kong Institution of Engineers, Institution of Engineering and Technology, and the Australian Academy of Technological Sciences and Engineering.