Simulating Collaboration From Multiple, Potentially Non-Collaborative Healthcare Systems To Create A Single View of a Patient

Tyrone W A Grandison, Varun Bhagwan, Daniel Gruhl

***Abstract* — There is still significant investment in legacy healthcare information technology (HIT). Current systems are a diverse mix of technologies, standards, platforms and versions. Many of which were never intended to be used together to achieve a common goal. In order to deliver care effectively and efficiently, point-of-care software must navigate this complex maze, coordinating multiple disparate systems, to produce as holistic a view as possible of a patient's treatment history, in a timely manner. In this paper, we introduce a specific problem of enabling the collaboration of HIT to facilitate data ingest and integration, present a solution approach and describe a software embodiment that was deployed.**

***Index Terms* — Collaborative work, Health care, Medical information systems, Information services, Information systems**

I. Introduction

Currently, the delivery of care depends on the healthcare practitioner having as near complete and up-to-date view of the patient's data, based on recent tests, visits, prescriptions, prognoses, etc., at the time of care. Unfortunately, the current healthcare system is faced with the following realities [1]:

1. Patient data is fragmented - A typical patient visit may generate five or more lab documents (of the same or differing modalities); each of which is likely to be stored on a separate server and utilizes different representation formats.
2. Patient data is distributed and mobile - Patient records may exist at several providers, payers, etc. As a patient moves between providers, locations, etc., several records of care are created at treating or service provision organizations.
3. Patient data is replicated - Organizational and or legislative policy often dictates that patient information be duplicated for security and disaster recovery reasons. Additionally, a replica of institutional data is often created for stakeholders, i.e. patients, affiliates, etc., and used as their primary records for service processing and or delivery.
4. Patient data is missing - Best practice in the field is to have interpretative reports accompanying laboratory results. Unfortunately, in real world scenarios, there are lab images with no associated reports [1].
5. Patient data contains errors and redundancies – Depending on the healthcare institution and the input method used, both the error and duplication rates can be considerable [2].

Contemporary wisdom recommends the use of the Digital Imaging and Communications in Medicine (DICOM) [3] standard to solve a lot of the issues involved with healthcare information integration. The focus of DICOM [3] has been to enable the integration of scanners, servers, workstations, printers and network hardware from multiple manufacturers into a Picture Archiving and Communication System (PACS) [4].

DICOM is promoted as a standard for handling, storing, printing, and transmitting information in medical imaging. As the healthcare industry consists mainly of image data, this effort is very important to the sector. Unfortunately, adoption and support of DICOM has been slow; especially in American healthcare [1].

One of the primary reasons for this slow uptake is that healthcare vendors have made significant investment in their current offerings, which tend not to be DICOM-compliant. For a myriad of reasons, it is most often the case that the systems being sold by the healthcare technology giants leverage their own proprietary standards. In this environment, it is understandable that the cost of changes to existing products, to make them DICOM-compliant, is considerable. Thus, there is a strong disincentive to move rapidly towards standardization.

However, it is now recognized that the quality of care will be improved, the cost of providing healthcare services reduced and the efficiency of care delivery increased, if healthcare information from all the relevant data sources, i.e. relevant to a particular patient, is integrated. This integration [5], which was the hope of DICOM standard, is being hastened by the rapid computerization of healthcare assets and the construction of (regional) care networks.

Tyrone W A Grandison is with IBM Services Research, Hawthorne, NY 10532, USA (phone: 408-927-1951; fax: 408-927-3215; e-mail: tyroneg@us.ibm.com).

Varun Bhagwan is with IBM Almaden Research Center, San Jose, CA 95120 USA.

Daniel Gruhl is with IBM Almaden Research Center, San Jose, CA 95120 USA.

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.17

II. The Problem

Formally, the problem can be stated as follows: Given n data sources ($D_1 \ldots D_n$) from which information is to be gathered, where each data source has an associated cost (c_d) and a probability of returning a valid response (p_d), and m data slots ($S_1 \ldots S_m$), one for each segment of a patient record that one is interested in, how does one maximize the probability of obtaining valid results for as many important data slots as possible, while minimizing the cost of acquiring that data?

We refer to the *completeness* constraint as the condition of getting as many important data slots filled as possible and the *latency* constraint as the condition of achieving the completeness constraint within time and or budget limits.

Expressed in its simplest form, our (optimization) problem is to minimize (retrieval) costs and maximize (important) slots retrieved.

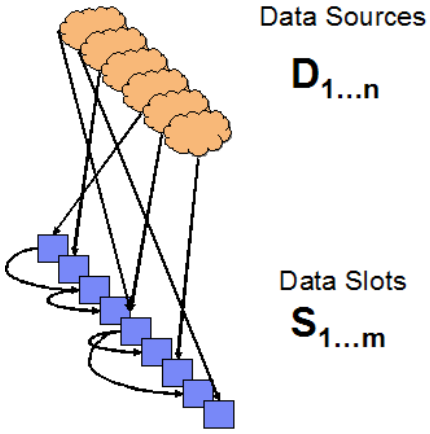

Fig. 1. General Problem Pictorial

Fig. 1 shows a pictorial representation of the problem space. From the figure, it can be seen that 1) data slots may be singular or composite, 2) the number of slots is equal to or greater than the number of data sources and 3) a slot has one or more associated data sources where its data may be retrieved from.

III. Solution Approach

We assume that each slot can be filled by at least one data source, another filled slot or a combination thereof.

We also assume that each slot has an associated importance value. Thus, each S_i has a v_i. where $1 \le i \le m$.

Algorithm 1 outlines the steps executed in solving this problem. As previously stated, the algorithm seeks to maximize the probability of obtaining a valid result, while meeting the completeness and latency constraints.

There is a pre-processing step that involves determining the slots needed in the slot map, based on the artifact of interest. In the case of our scenario, the slot map is composed of the different attributes needed to construct the patient record. After the slot_map, data_source, importance_value, fetch_probability and source_cost arrays are initialized (step 1), the importance values are assigned, based on 1) Subject Matter Expert (SME) knowledge, 2) an expectation of a successful fetch from the associated data source; this is derived from prior fetches of similar datum, and 3) and an expected resource expense, i.e. the cost of performing the fetch.

Then the return on investment (ROI) for each slot is calculated using

$$ROI_i = \frac{p_i * v_i}{cs_i}, \text{ where } 1 \le i \le m$$

cs_i is the cost associated with data slot i and is a measure of the costs associated with retrieving data from the associated data sources.

1. Initialize_Parameters(S, D, v, p, c)
2. Assign_Importance(v, D, c)
3. ROI = Calculate_ROI(p, v, c)
4. Assign_Limits(budget, hard_stop_end_time)
5. Scan S for the highest ROI slot to begin filling.
6. Run Fetch_Process(S, D, p, v, c)
7. Repeat step 5 until ((budget=0) or (current_time > hard_stop_end_time)
8. Return the optimally partially-filled slot-map S to the application.

Algorithm 1: Main Code Segment

At step 4, a budget[1] of resources is assigned and a hard stop end time is set for the entire slot map. Please note that it is often the case that the hard stop time is normally set by the calling application.

At step 5, we use the ROI values to create a virtual priority queue, where the slot with the highest ROI is the one for which the fetch process is to be executed.

The information for the selected slot is sent to an ingestor module, which automatically negotiates the process of acquiring, cleansing and integrating the data. We will discuss the details of this module further in the next section.

Even before attempting data acquisition, the fetch cost/success model (that is maintained by the ingestor) is updated to reflect the fact that some of the budget has been used.

If all associated data sources that could be used to fill the slot have been unsuccessful, then the system goes back to step 5, where another slot is selected.

If the ingest process is successful, then the slot result is analyzed; as it may trigger more slots to be added to the map. If new slots are to be added, then they are assigned cost, time and expectation values from the budget in the fetch cost/success model in the ingestor.

The entire process is repeated until the budget is completely spent or until the hard stop end time has been reached.

At this point, the slot map is returned to the calling application and it embodies the best attempt at getting the most important pieces of data, while meeting the pre-defined constraints.

IV. Technology Embodiment

The ingest technology used here is the MONGOOSE [6] suite, which has been developed over the last few years to

[1] The "budget" is the upper bound on the costs that can be expended.

address the ingest requirements for advanced analytics systems across industries, e.g. media and entertainment [7] and automotive [8].

In this effort, we applied our approach, built on MONGOOSE technology, to the task of gathering data for a cardiology decision support project [9] with a very large healthcare provider.

A. *MONGOOSE*

MONGOOSE [6] is a software library with supporting code for control flow monitoring, analysis and correction that enables *Worst-Case Scenario* Workflow Management. We define *Worst-Case Scenario* systems as those that expect failures to occur and aim to mitigate these incidents.

MONGOOSE allows community-based information extraction around specific phenomena that can be fed into statistical analysis tools. The core components of the MONGOOSE system (Fig. 2) are the Intake Platform (Fig. 3) and the Control Platform (Fig. 4).

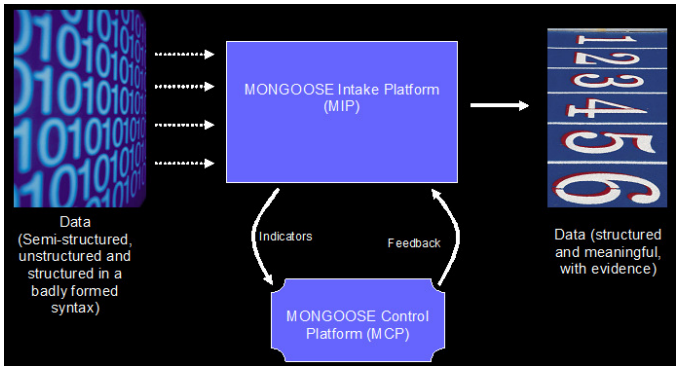

Fig. 2. The MONGOOSE Architecture

The user of the MONGOOSE system can leverage a series of MONGOOSE modules to determine the particular information flow for the application they are developing. The user also tells the MONGOOSE system the form and content of the data they wish to have output.

In leveraging the MONGOOSE modules, the user gets a resilient, fault-aware platform that ingests data irrespective of what happens either with the data sources or with the processing chain. This also allows MONGOOSE instances to be created for a multitude of domains; as instantiation involves plugging in domain knowledge cartridges into the system and then setting the outputs to a form suitable for the domain, e.g. Online Analytical Processing or Business Intelligence consumption.

The intake platform performs a series of tasks that cleans the data, transforms it into a sensible form and combines all the dimensions of the phenomenon under investigation. The control platform monitors the activities of the intake process and performs corrective action.

1) *The Mongoose Intake Platform (MIP)*

The MONGOOSE Intake *Acquisition* Modules (MIAMs) are base constructs used to build specialized ingestors for the application domain of interest. The MONGOOSE Intake *Pre-Processing* Modules (MIPMs) extract individual comments, posts, discussion points, profiles and counts from the ingested data and processes the unstructured content to determine spam and identify on-topic and off-topic information.

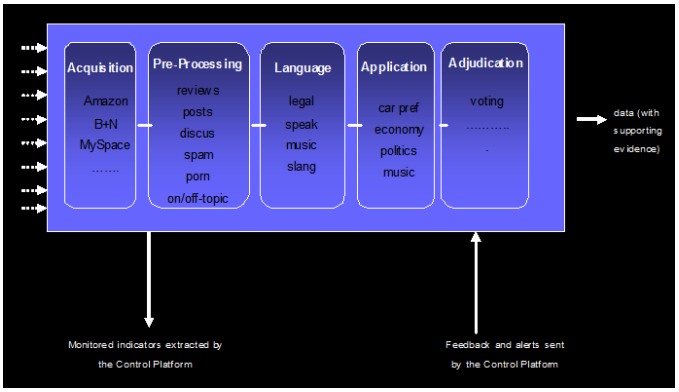

Fig. 3. The MONGOOSE Intake Platform Architecture

The MONGOOSE Intake *Language* Modules (MILMs) allows the jargon of the domain of interest to be included. Dictionaries for terminology in various domains such as healthcare, media and entertainment, law, automobiles, politics, etc. are included in these modules. The MONGOOSE Intake *Application* Descriptors Modules (MIADMs) allows the definition of the key descriptors for the domain and problem of interest. For example, in politics underlying concepts would be Integrity, Record, and possibly Funds. The MONGOOSE Intake *Adjudication* Modules (MAMs) combine multi-modal information in a way that is meaningful for the business task at hand.

2) *The Mongoose Control Platform (MCP)*

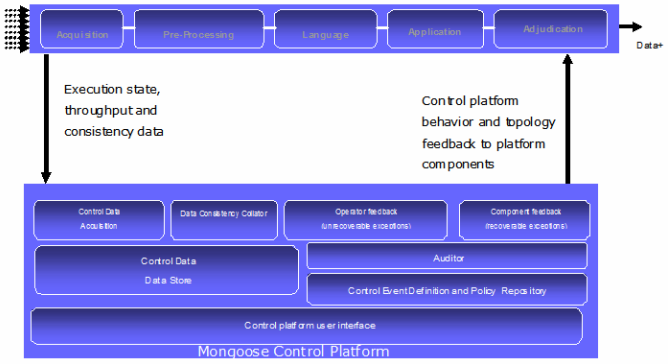

Fig. 4. The MONGOOSE Control Platform Architecture

The MONGOOSE Control Platform (MCP) allows system continuity without visible failure (Fig. 4). The purpose of the MCP is to reduce the Total Cost of Ownership (TCO) of systems both as they are developed and after they transfer either from prototype to development or from research to production. The MCP accomplishes this by providing the following general functionality: 1) standard error detection, handling and reporting, 2) standard data analytics on corruption and consistency reporting, data acquisition layer execution state, 3) data flow integrity and source accessibility assertion, 4) system wide data volume monitoring to ensure an increasingly monotonically increasing input data stream, and

5) system wide data flow and cube output volume monitoring.

B. *The AALIM System*

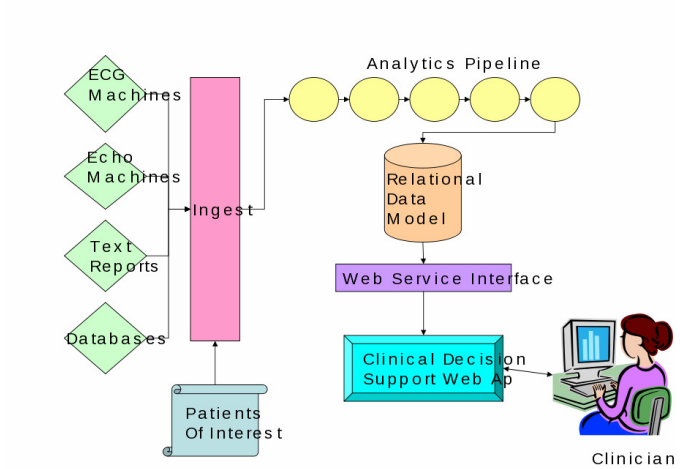

Fig. 5. The AALIM system, which gathers data from a large number of sources, performs analytics on them to extract features, and then uses those features to provide a cardiac clinical decision support system.

The AALIM[2] system (Fig. 5) provides an overview and decision support for patients in the cardiac care space. As such it needs to combine a number of modalities (e.g. textual reports from patient visits, structured data from the pharmacy and labs, signal data from ECGs[3] and video from Echocardiograms) and to perform analytics on all of these to extract features, which then feed the adjudication and clinical support functions. In order to perform these analytics, relevant data needs to be gathered in a timely manner. It is at this step that the challenges arose. Our solution was used to gather the facets of a patient's cardiac.

V. Results

We found that a combination of factors made acquiring cardiac data challenging – and that these challenges are endemic to the health informatics problem space.

A. *MONGOOSE USE OF AALIM*

The initial versions of the AALIM system did not use our solution, and required constant human monitoring. We will now outline how each part of our solution simplified this problem.

Queuing Service: By providing a queuing service of tasks to be run, with failed tasks returned to the queue, the task of batching work became much simpler. The "fetcher" was handed a target machine to log into and a patient's medical record number and it merely had to make an attempt to fetch the data. It could then report its success or failure and exit. The MONGOOSE control framework handled rescheduling failed tasks as well as handing successful work onto the next stage in the pipeline.

Pipeline Management: Every modality underwent private patient health information removal immediately after fetching. This allowed the analytics to run on "clean" data, and allowed developers to debug their code without being exposed to personally identifiable information (PII). This is an example of a step that needed to be developed and run after every successful fetch. Our solution allowed steps such as this to be registered in the pipeline, and automatically flagged areas where PII removal failed.

Fail Over Mechanism: If data acquisition failed and there were alternate paths to the data, the system queued failed fetches via alternate paths. Additionally, it took remedial action, such as flushing DNS caches, resetting connections, etc. and attempting to retry main code before failing over. This provides a natural place to decompose complex code into stand alone tasks that are easier to develop and debug.

Central Reporting: By gathering statistics such as how long data acquisition took, how many results were fetched, which source servers were up and down, etc. and providing all this information at a central summary page it became very easy for an operator to get an "at a glance" idea of how the system was working, as well as identify when problems arose where they started and what might need to be done to address them.

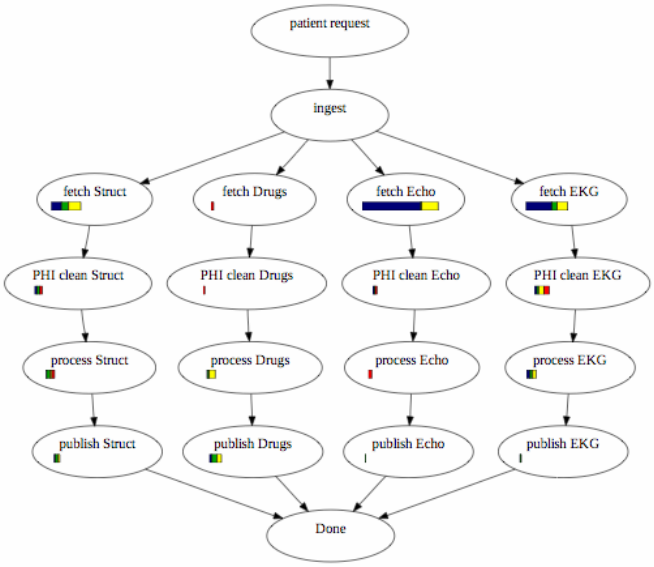

Fig. 6. A visualization that shows the flow of data through multiple processing pipelines. The colors indicate the status of the *work-items*, and the width of the bar indicates the number of *work-items* being processed in any stage.

In Fig. 6, we show how obtaining and processing of a patient record is done in the deployment. Moving from top to bottom, *patient request* denotes the batch of MRNs (Medical Record Number), also referred to as *work-items*, whose data needs to be acquired and processed through the system. This triggers *Ingest* of the relevant data, which itself is forked off into four parallel pipelines (one per data-modality). As each *work-item* is processed successfully, it is passed through the

[2] AALIM (Advanced Analytics for Information Management) work is performed at IBM Almaden Research Center. More information can be retrieved from http://www.almaden.ibm.com/cs/projects/aalim/

[3] ECG refers both to an electrocardiograph and an electrocardiogram. An electrocardiograph is a transthoracic interpretation of the electrical activity of the heart over time captured and externally recorded by skin electrodes. An electrocardiogram is a graphical recording of the cardiac cycle produced by an electrocardiograph.

next stage in the pipeline, e.g., upon successful completion of *'fetch Struct'* for a *work-item*, it is automatically sent to the *'PHI clean Struct'* stage and so forth.

In addition to displaying the movement of work-items through the entire AALIM pipeline, the visualization also indicates the *status* of *work-items* in each stage. This is done by means of color-coding, where the color Blue indicates a queued and *Pending* item, Green implies an item *In Process*, Yellow implies *Transient Failure* (where the number of retries haven't been exhausted); and Red implies *Permanent Failure* (where the threshold of failures has been exceeded and human intervention is required). Additionally, the width of the bar indicates the relative number of *work-items* in a given status in a given stage within the system. Our system has the capability to provide additional details, such as individual counts, processing throughput, etc., as required.

Clearly, Fig. 6 is a fairly simplistic visualization. However, it forms the base for a more sophisticated version, which is in the works.

Alerting: By providing an email and page-out system, our solution can notify developers when they need to pay attention to (batch) processes that they wish to monitor, as well as to inform the operator when important steps failed.

VI. Discussion

The cardiological instance of our solution provided a glimpse into the difficulties that IT departments in healthcare providers across the US face; due to the lack of widespread uptake and implementation of standards, e.g. DICOM [3], HL7 [10], etc. In this section, we present some of the lessons learned from the AALIM effort and then generalize to articulate the challenges in healthcare IT departments globally.

A. AALIM Lessons Learned

Large health care environments are a collection of machines and systems dating back, in some cases, over two decades. With the constant financial pressures in this space it is not surprising that systems that work well tend not to be replaced simply because another file format has emerged. Additionally incremental support of systems by vendors does not continue indefinitely. Taken together, this results in a large number of systems that contain data critical to understanding patient health that may not have been updated in a long time. One specific example we encountered was a set of machines that stored ECG data. These machines were over 15 years old, yet contained critical information on how a patient's heart was performing a decade ago. These machines were not developed with an API[4] for integration in mind, and in fact, in the life cycle of upgrades they did receive, they ended up in an unstable state, where remote access to the underlying data storage of the machine was impossible (due to conflicting co-installed versions of the ODBC[5] stack). Additionally, the waveform data was stored encoded in a BLOB[6] in a relational database. The format of this encoding was not available. This left the only way to retrieve this coding through navigation of the (at the time) cutting edge Web interface, followed by "pressing" the *print* button and downloading the resultant HP Printer Control Language (PCL). This PCL was decoded into a raster which was then converted into a vector of measurements for the waveform.

An additional complication was that these machines were not completely stable, and at any given time there was approximately 5% of them that were not available. Fortunately, some of the data from the ECG (mostly raw measurements such as *P* and *Qt* intervals as well as diagnosis) was propagated to a mainframe system much of the time. Thus, to draw this data for analysis, a complex set of tasks needed to be authored. These tasks needed to be monitored. If machines were unavailable, then alternate data acquisitions paths needed to be pursued, with the "full record" perhaps being monitored and later included if it became available.

Despite the best of intentions, modern health care systems are a heterogeneous collection of systems that span many years and many versions. It is clear that as newer systems move to more standards based approaches, like DICOM, this kind of data integration will become easier, it is also clear that systems that wish to provide health care informatics today cannot wait for these standards based systems, but must instead work with complex, messy environments to extract the data they need to empower the transformation of the health care system.

B. General Challenges in Ingestion and Integration

The first challenge is that a unified, coherent and complete picture of the patient is required to provide the best possible care; in environments that do not support this goal [11-13]. A patient's data is fragmented due to current mode of care delivery. The record for a single instance of care for a patient who goes to a general practitioner and gets referred to a specialist, who then refers them to a laboratory for tests is distributed across at least three systems and is made up of segments of differing modalities (e.g. text documents, scanned files, X-Rays, CT Scans, etc.). Each modality has its own class of system and all of them are required to get a complete picture of the patient's condition.

The second challenge is the struggle between legacy systems and new technology. While legacy systems are expected to eventually be shut down, they are often the authoritative sources in some departments for particular pieces of information. Thus, the *older* the systems that one ingest data from, the better the picture of the patient history that can be

[4] API (Application Programming Interface) is a set of routines, data structures, object classes and/or protocols provided by libraries and/or operating system services in order to support the building of applications,

[5] ODBC (Open DataBase Connectivity) is a standard database access method developed to make it possible to access any data from any application, regardless of which database management system is handling the data.

[6] BLOB (Binary Large OBject) is a collection of binary data stored as a single entity in a database management systems (DBMS). BLOBs are used primarily to hold multimedia objects such as images, videos, and sound, Unfortunately not all DBMSs support BLOBs.

created. Simply because, this piece of data may only be available from that *older* system. The MONGOOSE-AALIM deployment provides evidence of this.

The third challenge is that an *episode of care* may span years. Within those years, multiple technology cycles would have been experienced. With these cycles come a myriad of device interface and technology obsolescence issues and process and policy changes. These factors exacerbate the problem of the number of device formats and (proprietary) standards that must be supported in order to deliver care.

The fourth challenge is the siloed nature of healthcare institutions, particularly providers, where each silo acts with a lot of autonomy and very little coordination with the other silos in order to ensure consistency. Thus, each silo may make budget decisions on technology investment that widen the integration gap. This eventually leads to a situation with lots of groups of specialized systems and groups of varying system types, e.g. the radiology department purchasing groups of General Electric and Siemens equipment and then multiple versions of a specific brand. In this scenario, the incompatibilities in the devices (from the same manufacturer or across manufacturer), when devices are used randomly for different patients, exacerbates the integration problem.

These challenges would be greatly reduced (if not eliminated) were the DICOM standard widely adopted. In such a case, MONGOOSE technology would simply be used for monitoring and fault-tolerance.

VII. Conclusion

Currently, integration of healthcare data to provide a complete view of the patient is a messy task, which cannot be easily performed. Use of standards would greatly enhance the task of integration. Unfortunately, current systems require extraction and integration of some systems that are and will never be standards-compliant and were not intended to work in a collaborative manner. In this paper, we provided a possible solution. We explained how the technology was used in a cardiac decision support solution and presented the observations and lessons learned in the process.

Acknowledgment

We would like to thank the AALIM team for their invaluable contributions.

References

[1] Bhagwan, V, Grandison, T, Gruhl, D. "Ingest and Integration of Medical Data in a World with very little DICOM". The proceedings of the 13th World Congress on Medical and Health Informatics (MEDINFO) 2009. September 12-15, 2010. Cape Town, South Africa.

[2] HIMSS. "Patient Identity Integrity". White Paper. December 2009. Retrieved From http://www.himss.org/content/files/PrivacySecurity/PIIWhitePaper.pdf

[3] Pianykh, OS. "Digital Imaging and Communications in Medicine (DICOM): A Practical Introduction and Survival Guide". Springer. Edition 1. July 24, 2008. ISBN-13: 978-35407455709.

[4] Wiley, G. "The Prophet Motive: How PACS Was Developed and Sold". Imaging Economics, May 2005. http://www.imagingeconomics.com/issues/articles/2005-05_01.asp Accessed on October 15, 2009.

[5] Brailer, DJ. "Interoperability: The Key To The Future Health Care System". Health Affairs: The Policy Journal of the Health Space. http://content.healthaffairs.org/cgi/content/full/hlthaff.w5.19/DC1 Accessed October 15, 2009.

[6] Bhagwan, V, Grandison, T, Alba, A, Gruhl, D and Pieper J. "MONGOOSE: MONitoring Global Online Opinions via Semantic Extraction. The proceedings of the 2009 Service Quality and Assurance Management (SQAM) workshop at IEEE 2009 International Conference on Cloud Computing (CLOUD-II 2009), September 21-25, 2009, Bangalore, India.

[7] Bhagwan, V, Grandison, T and Gruhl, D. "Sound Index: Music Charts By The People, For The People". Communications of the ACM. September 2009. Vol 52, No 9.

[8] Zakharian, Z, Mishra, M and Chandramohan, S. "Cars 2.0". Masters Thesis, San Jose State University, December 2008.

[9] Syeda-Mahmood, T, Wang, F, Beymer, D, Amir, A, Richmond, M, and Hashmi SN. Multimodal mining for cardiac decision support. In Computers in Cardiology, pages 209-212, 2007.

[10] Health Level Seven Inc., "HL7 Standard". http://www.hl7.org/ Accessed October 15, 2009.

[11] Rigby, MJ, Robins, SC. "Building healthcare delivery and management systems centred on information about the human aspects". Computer Methods and Programs in Biomedicine, Volume 54, Issue 2, Pages 93-99.

[12] Gilhooly, K. "Rx for better health care: interoperable electronic health records promise to streamline health care delivery, improve quality and help contain costs. But financing, a lack of standards and the scope of implementation stand in the way". ComputerWorld, January 31, 2005.

[13] Sarkar, D. "NHIN save lives reduce costs E-records vital to health strategy," Federal Computer Week, August 2, 2004