Open-Source, Web-Based Dashboard Components for DICOM Connectivity

Catalina Bustamante¹ · Julian Pineda¹ · Simon Rascovsky¹ · Andres Arango¹

Published online: 8 February 2016
© Society for Imaging Informatics in Medicine 2016

Abstract The administration of a DICOM network within an imaging healthcare institution requires tools that allow for monitoring of connectivity and availability for adequate uptime measurements and help guide technology management strategies. We present the implementation of an open-source widget for the Dashing framework that provides basic dashboard functionality allowing for monitoring of a DICOM network using network “ping” and DICOM “C-ECHO” operations.

Keywords Digital Imaging and Communications in Medicine (DICOM) · PACS · Administration · Open-source information visualization

Background

There is a worldwide trend towards more cost-effective healthcare delivery across all disciplines, in which radiology plays a major part as a component in the healthcare value chain with some of the most complex and expensive technology. Medical imaging information technology plays a critical part in the appropriate delivery of quality medical imaging, but as the size of the deployment gets smaller, the cost of advanced tools to manage a radiology operation becomes prohibitive, especially in developing countries.

Open-source software already holds a privileged position in developing countries when it comes to medical imaging applications. Open-source picture archiving and communications system (PACS) [1], diagnostic [2], and clinical viewers [3] have a very large user base and have proven their worth where financial resources are limited.

In the past few years, well-established business and technology practices have inched their way towards medical imaging. Both business and infrastructure dashboards have become more common for management of a radiology business, and several commercial offerings [4] are in use today.

Open-source, as is the case with most commercial software categories, offers a mirror solution both for business or technology dashboards. Some solutions are general purpose dashboards that can aggregate incoming information using modern technologies and can be used for real-time visualization of heterogeneous data. Apart from certain trailblazing projects for PACS performance monitoring however [5, 6], the application of open-source infrastructure dashboards and analytics to medical imaging deployments is a relatively recent development.

A critical component of timely delivery of medical imaging is reliable communication with high availability to ensure the interoperability between the systems used that communicate via the Digital Imaging and Communications in Medicine (DICOM) protocol [7]. DICOM downtimes are the periods when any DICOM server or modality is unavailable; they are often the cause of significant delays in the radiology process, and ensuring a fast and appropriate response to these issues is of the highest importance. The health of the DICOM network, which refers specifically to the data network connecting DICOM nodes [8], is often the main priority of medical imaging IT support teams, and real-time visibility of all modalities and systems involved can be an invaluable aid.

Standard network management and monitoring tools can evaluate and measure connectivity using simple checks such

✉ Catalina Bustamante
catalina.bustamante@iatm.com.co

¹ Instituto de Alta Tecnología Médica - IATM, Carrera 81 No 30A-99, Medellín, Antioquia, Colombia

as “ping” which helps determine if a connectivity issue is related to a lower level problem such as hardware failures and network problems, as well as bandwidth and packet routing issues. After a basic network-level fault is ruled out, the next step is to evaluate the DICOM service layer connectivity. This is usually performed using a standard check specified by the DICOM standard called “C-ECHO,” which builds upon a basic connectivity check but engages in a full DICOM negotiation with the application entity (AE). This is a simple but useful way of determining basic DICOM connectivity since all application entities are required to respond to an echo request.

The availability of a tool that allows for real-time visualization, monitoring, and alerting of the status of DICOM nodes within a network is quite useful for imaging system administrators, who can then respond promptly and appropriately to minimize downtime, before the final user is affected increasing service level agreement compliance and end user satisfaction.

We present our institutional experience developing a simple yet functional DICOM connectivity monitoring component for an open-source dashboard framework and show how additional basic alerting functionality of connectivity issues can be implemented with a simple integration.

Methods

We implemented a DICOM connectivity check component for an open-source dashboard written in Ruby, which gives a visual indicator of the DICOM status of preconfigured servers. We further augmented the functionality of this dashboard by implementing an alert system to our communication hub (Fig. 1).

We set about to create a widget for the Dashing framework, an easy-to-use Ruby-based dashboard tool, that allows for the monitoring of a DICOM network using network ping and DICOM C-ECHO operations. Our objectives were to make this widget easily configurable and very modular and to offer it as an open-source third-party widget that can be easily extended to any institution that manages DICOM nodes and can be integrated with other dashboard platforms.

The widget requires minimal or no knowledge of the Ruby language or the Dashing gem to configure, although more sophisticated solutions can be built with only a few lines of Ruby code if so desired.

Dashboard (Dashing)

Of the available general purpose open-source dashboard frameworks, Dashing (<http://dashing.io>) seems to be quite popular due to its ease of deployment, configuration, and extensibility. It is a Ruby-based software that uses the web framework Sinatra (<http://www.sinatrarb.com/>) and is fully modular, taking advantage of a simple domain-specific language (DSL) that makes it easy for developers to create their own widgets for simple, yet effective visualization.

Dashing provides several widgets out of the box, and it is quite easy to develop widgets for specific functionality using common web technologies (CSS, HTML, and CoffeeScript—a javascript compiler).

Jobs

Dashing provides two ways to input information to the widgets: via HTTP calls for external updating, and scheduled jobs

Fig. 1 Implementation overview

run internally that can implement any functionality through Ruby. Since all of our DICOM systems lack the functionality to push their status via HTTP calls, we wrote Ruby code that is run periodically and performs both ICMP ping and DICOM Echo checks and updates the widgets and alerting system accordingly.

DICOM Connectivity

DICOM connectivity checks were performed using the Ruby DICOM library/gem. Ruby DICOM is “a cross-platform library for handling DICOM files and network communication in Ruby language” [9]. The library offers a full range of DICOM manipulation tools as well as basic network operations (C-STORE, C-MOVE) and, importantly, a very simple method to test a DICOM association.

A DICOM Client is instantiated by passing an IP, port, and AE-Title (AET) as parameters, as well as an AET with which the client will identify itself to the called entity:

```
client = DClient.new(self.ip, self.port, {ae:
  CALLING_AE, host_ae: self.aet})
```

In order to check DICOM connectivity to a server, *DClient:test* method performs a simple negotiation of an association and then releases it, returning true if the association was successful [10]:

```
client.test # => true
```

The Widget

Simplicity was the aim for the development of the widget. As with any dashboard component, a quick glance from across the room should tell the user the main criteria for system health, and additional information can be gained upon closer inspection.

The widget consists of a simple colored tile, for which the color of the tile represents the connectivity status. A green tile represents a server that responds to both ICMP ping and DICOM Echo, an orange tile means a server answered to ping but not DICOM Echo, and a red tile in which both checks

failed. A blue tile means no checks have yet been performed for that server, such as in a dashboard initialization (Fig. 2).

Inside the widget is the name of the server being checked, as well as the IP address and the time of last check are prominently displayed. Additionally, if configured, two icons are displayed, one that links to a web interface for that particular server and another that links to its configuration/management URL. These links are provided for fast and easy access to the tools necessary for incident resolution.

Alerting (Slack)

Alerting functionality can be provided in several ways, and the most common of which is via e-mail. More recently, push notifications have become the standard, and with multiple applications sending notifications, it is often desirable to aggregate these into a single communication “hub.”

Slack (<https://slack.com/>) is a communication/collaboration tool that has gained widespread acceptance [11] and is used in our institution for all internal communications. Part of Slack’s appeal to developers is the ease with which information can be sent to the application via multiple existing integrations. When an integration is not readily available, custom integrations are very easy to perform due to a very robust API [12] and Slack’s support for incoming Webhooks that provide a URL for posting messages into a specific Slack channel.

Although direct HTTP calls are supported, libraries are available for most major programming languages, and in Ruby, the slack-notifier gem provides a Ruby-like interface to send notifications to Slack Webhooks [13].

Configuration of the Slack interface is very simple, requiring only the Webhook URL and the channel or user the message should be posted to:

```
notifier = Slack::Notifier.new "IATM",
  "***WEBHOOK_URL goes here***", channel: '#alerts',
  username: 'alers'
```

Once configured, instant push notifications to Slack can be invoked with just:

```
notifier.ping "A server connectivity issue has been de-
  tected for server: #{server[:name]}"
```

Fig. 2 Difference of color tone representing connectivity status

Fig. 3 DICOM network dashboard. The *darkest* server (or *red* in color version) failed the ping test, the *lighter* (or *orange* in color version) one failed a DICOM Echo, and the *lightest* (or *green* in color version) server passed both tests

Results

Development of the DICOM widget was quite straightforward due to the expressiveness of the Ruby language and Dashing's simple DSL. The complete widget and job algorithm required less than 200 lines of well-commented code across five files. The source code can be found at <https://gist.github.com/simonmd/cc2ca1e5e94831013b93>.

The widget was included into our main dashboard and configured to perform checks on all DICOM-enabled devices on our network. This dashboard has been in operation for 11 months and has provided our support staff with a real-time overview of the health of the DICOM network across five sites. The main dashboard also includes other widgets with information about reports and order generation and was easily integrated with the DICOM connectivity widgets (Fig. 3).

The usefulness of this setup was further enhanced when the widget was paired with the integration to our communication hub.

The use of this tool has allowed us to obtain real-time visibility of our DICOM network and reduced our resolution times for connectivity issues, many times before the end user is affected.

Discussion

We set about to create a widget for the Dashing framework that allows for the monitoring of a DICOM network using

network ping and DICOM C-ECHO operations. Our objectives were to make this widget easily configurable and very modular and to offer it as an open-source third-party widget that can be easily extended to any institution that manages DICOM nodes. The widget requires minimal or no knowledge of the Ruby language or the Dashing (DSL) to configure, although more sophisticated solutions can be built with only a few lines of Ruby code if so desired.

Adequate management of a DICOM network benefits from tools that allow for real-time visualization of the availability status of the different nodes within the network. Monitoring and alerting can be performed with open-source frameworks that provide basic dashboard functionality, coupled with widgets designed for network ping and DICOM C-ECHO operations.

Conclusions

This project showcases the possibility of using simple open-source software to address common needs in medical imaging informatics, without the need for major investments or support staff. This is especially meaningful in developing countries where digital medical imaging is taking hold, but often without the financial resources to support software acquisitions that would enhance operations.

As technology teams become smaller, more efficient, and more transparent, these general-purpose dashboards can provide a medical imaging institution with valuable operational and infrastructural insights.

References

1. Dcmechee.org, Open Source Clinical Image and Object Management [Online]. [Accessed 16th May 2015]. Available at: <http://www.dcm4che.org>.
2. Osirix Imaging Software, [Online]. [Accessed 16th May 2015]. Available at: <http://www.osirix-viewer.com/AboutOsiriX.html>.
3. Welcome to Oviyam [Online]. [Accessed 16th May 2015]. Available at: <http://www.dcm4che.org/confluence/display/OV/Home>
4. PACSHealth [Online]. [Accessed 17th May 2015]. Available at: <http://www.pacshealth.com/features/>
5. Sweeney, Ron. Monitoring Your Enterprise PACS with Nagios, Cacti and Smokeping [Online]. [Accessed 10th May 2015]. Available at: <http://www.clubpacswestmi.net/EnterprisePACSMonitoringwithNagiosSmokepingandCacti.pdf>
6. Nagy, PG, Daly, M, Warnock, M, Ehlers, KC, Rehm, J. PACSPulse: a Web-based DICOM network traffic monitor and analysis tool. *Radiographics* 2003;23:795–801.
7. DICOM. Digital Imaging And Communication in Medicine [Online]. [Accessed 16th May 2015]. Available from: <http://medical.nema.org/standard.html>
8. The DICOM Network [Online]. [Accessed 4th January 2016]. Available at: <https://www.neologica.it/html/Tutorial/DICOMNetwork>
9. GitHub: RubyDICOM [Online]. [Accessed 25th April 2015]. Available at: <https://github.com/dicom/ruby-dicom>
10. Documentation for dicom/ruby-dicom (master) [Online]. [Accessed 26th April 2015]. Available at: www.rubydoc.info/github/dicom/ruby-dicom/DICOM/DClient:test
11. Augustine A. Slack Sets New Standard for Team Communication Online [Online]. [Accessed 27th April 2015]. Available at: <http://collaboration.about.com/od/groupcommunication/fl/Slack-Sets-New-Standard-for-Team-Communication-Online.htm>
12. SlackAPI: Incoming Webhooks [Online]. [Accessed 2nd May 2015]. Available at: <https://api.slack.com/incoming-webhooks>
13. GitHub: stevenosloan/slack-notifier [Online]. [Accessed 28th April 2015]. Available at: <https://github.com/stevenosloan/slack-notifier>