ChRIS– A Web-based NeuroImaging and Informatics System for Collecting, Organizing, Processing, Visualizing and Sharing of Medical Data

Rudolph Pienaar, Nicolas Rannou and Jorge Bernal
Department of Radiology
Boston Children's Hospital
Boston, MA 02115
Email: rudolph.pienaar@childrens.harvard.edu

Daniel Hähn
Department of Computer Science
Harvard University
Email: haehn@seas.harvard.edu

P. Ellen Grant
Department of Radiology
Boston Children's Hospital
Boston, MA

***Abstract*—The utility of web browsers for general purpose computing, long anticipated, is only now coming into fruition. In this paper we present a web-based medical image data and information management software platform called *ChRIS* ([Boston] *Ch*ildren's *R*esearch *I*ntegration *S*ystem). *ChRIS*' deep functionality allows for easy retrieval of medical image data from resources typically found in hospitals, organizes and presents information in a modern feed-like interface, provides access to a growing library of plugins that process these data – typically on a connected High Performance Compute Cluster, allows for easy data sharing between users and instances of *ChRIS* and provides powerful 3D visualization and real time collaboration.**

Keywords*—Web, WebGL, Medical Imaging, Real time collaboration

I. Introduction

Modern web browsers are becoming powerful platforms for advanced application development [1], [2]. This is due to many factors, not least of which the increasing maturity of JavaScript (and other related) programming languages, the speed and sophisitication of the browser Just-In-Time (JIT) compilers for these languages, and the ever closing "gap" between actual machine hardware and its exposure to languages such as JavaScript.

In this paper, we introduce *ChRIS* (or the [Boston] *Ch*ildren's *R*esearch *I*ntegration *S*ystem). *ChRIS* is a novel web-based data storage and data processing workflow manager that provides strict data security while also facilitating secure, realtime interactive collaboration over the Internet and internal Intranets. Although *ChRIS* can manage any datatype, it is uniquely suited to medical image data, providing the ability to seamlessly collect data from typical sources found in hospitals (such as Picture Archive and Communications Systems, PACS), import from CDs/DVDs, users desktops, etc. *ChRIS* not only manages data collection and organization, but it also provides a large (and expanding) library of pipelines to analyze imported data. This library is easily extensible via a simple plugin mechanism, and *ChRIS* also provides the ability to directly interact with compute clusters for data analysis. Moreover, a wide variety of 2D, 3D, and 4D medical image data formats are natively supported and can be directly visualized and manipulated within the browser.

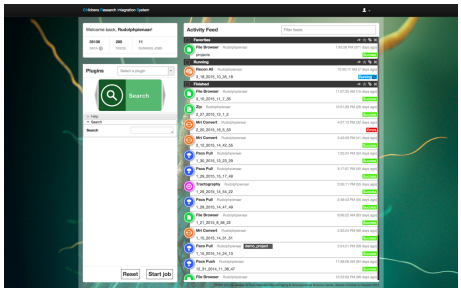


Fig. 1: *ChRIS* overview showing *feeds* and a *plugin*.

The benefits of web-based access to medical data, as well as access to pipelines that can further process this data, are considerable. Due to the ubiquitous spread of the Internet, connectivity is possible in many locations through many mechanisms (esp. cellular network coverage). Due to the web-based nature of a system such as *ChRIS* we not only leverage the traditional web-based benefits (no need for local installation of software – everything runs in the browser, no need for local updates, etc.) but in the context of medical image processing provide the ability for health care practitioners in remote areas to have access to medical data (both raw and processed), as well as for patients.

II. Features and Motivation

ChRIS is motivated by both strong clinical and research needs. On the clinical side, *ChRIS* provides clinicians with easy access to data, allows for powerful collaboration, and provides value added processing not easily available otherwise. At the BCH, *ChRIS* is used to supplement clinical

978-1-4244-9270-1/15/$31.00 ©2015 IEEE

conferences. On the research side, *ChRIS* allows researchers to focus on their research protocols and data processing, without needing to spend time on the minutiae of performing data analysis.

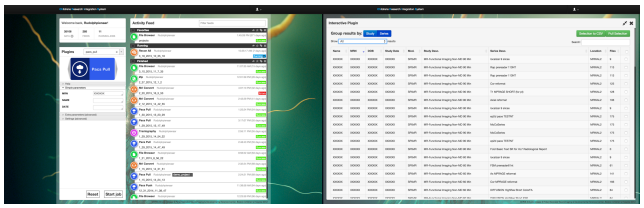
A. UI Philosophy and Data Managing

From user interface experience (UIX) perspective (see Figure 1), *ChRIS* is designed to be easy to use and draws heavily on familiar social Web 2.0 metaphors, such as exemplified by sites such as Facebook and Twitter. The *ChRIS* UI is designed around the central concept of "feeds" organized in a "timeline" with most recent activity at the top and each subsequent feed representing a descending time-ordered event. A *feed* is a component on the web page that contains various perspectives on data. Usually, this perspective is a file browser, centered on a source (or root) directory and providing access to all directories and files underneath this directory. The purpose of the *feed* is to provide the user with the ability to select a file (or directory) that will be processed by some operation.

This processing is provided by "plugins" in *ChRIS* parlance. A *plugin* can ultimately be any (typically) batch processing operation that requires some initialization, then processes its data, and provides outputs – these outputs are in turn additional files and directories. In *ChRIS* the output from a *plugin* in turn forms a new *feed* that can be processed in turn by additional *plugins*.

The organization of *feeds* is time-based sequential. A complete history of a user's activity on the system is captured in the ordered representation of *feeds* and any *feed* can be re-visited to share information, add notes, or even re-process data with the same or different parameters.

B. Data Collecting



((a)) Searching on MRN ((b)) Filtering Results

Within a hospital imaging context, images collected from various clinical devices are stored in a database called the Picture Archive Communications System (PACS) and conform to a file format called DICOM. As shown in the Figure above, *ChRIS* provides the ability to natively query a PACS and retrieve data from this resource. Using the PACS pull *plugin*, a user can enter a list of search terms (typically PatientIDs, or MRNs). Results appear in a table, which the user can filter and choose series of interest. Once pulled, data appears in a *feed* in the users' timeline.

Other mechanisms for collecting data in *ChRIS* are also provided: (1) a special *plugin* (the FileBrowser plugin) allows the user to type an actual file path, and this file path is then presented in a new *feed*– this is most effective when the *ChRIS* server and the end user share a common network filesystem (such as NFS) map; (2) another mechanism allows for drag-and-drop directly from the user's filesystem and into *ChRIS*– the dragged data is uploaded to the server and then presented to the user in a new *feed*; and (3) through a *ChRIS* push which is a special mechanism allowing different *ChRIS* instances to push data to each other via secure copy (scp).

C. Data Processing

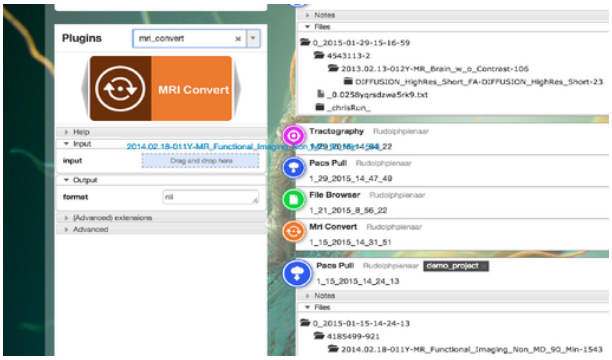


Fig. 2: Dragging data from a *feed* into a *plugin* for additional processing.

Once collected in a *feed*, data can then be processed. This is enabled by choosing an desired *plugin* from the plugin carousel in the UI, and then simply dragging relevant data from a *feed* and dropping it into the *plugin* input field. Once dropped in the *plugin*, the user simply clicks on "Start Job" at the bottom of the *plugin* and the data is processed.

In the simplest processing deployment case, *ChRIS* is setup to run all pipelines on the web server itself. While most portable, this deployment model is limiting in scale. In most cases, *ChRIS* is configured to connect to a High Performance Compute (HPC) resource and process pipelines on that resource. *ChRIS* handles all the interaction with that resource, copying data to the HPC, calling the scheduler, and on completion, copying data back to the server and presenting results in a new *feed*.

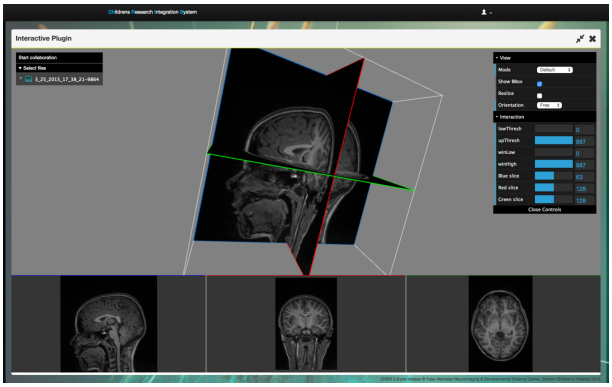


Fig. 3: A set of powerful, full screen visualizers that operate on medical data is built into *ChRIS*.

D. Data Visualizing

ChRIS uses the X toolkit library (XTK [3]) for visualizing medical image data. Any medical image type recognized by the library can be clicked on in the *feed* file view, and the image is rendered directly in the browser. Recognized formats include: DICOM, NIfTI, FreeSurfer Volumetric *mgz,

FreeSufer Surface model types, White Matter Tractographic *trk files. By selecting the "eye" icon next to a file, a fully interactive visualizing window is created allowing full 3D image view as well as orthogonal 2D axes. Multiple different visualizers are supported by *ChRIS*, each specific to particular data set or *plugin* result.

E. Data Sharing

An important design goal of *ChRIS* is the ability to allow users to *share* data and results, without needing to resort to complex mechanisms like sending emails with large attachments. Since the core element of *ChRIS* is a *feed*, which in turn is a construct built around data and shown as a file browser, we provided the ability to share *feeds* between users. All such shared *feeds* are in fact linked to a single copy and all links are thus updated in real time as different users add information, notes, comments, etc. to a *feed*. All users who share a *feed* share also all the data within that *feed*.

F. Data RealTime Collaboration

As a powerful conceptual extension to data sharing, *ChRIS* also offers RealTime visual collaboration between users who share a *feed*.

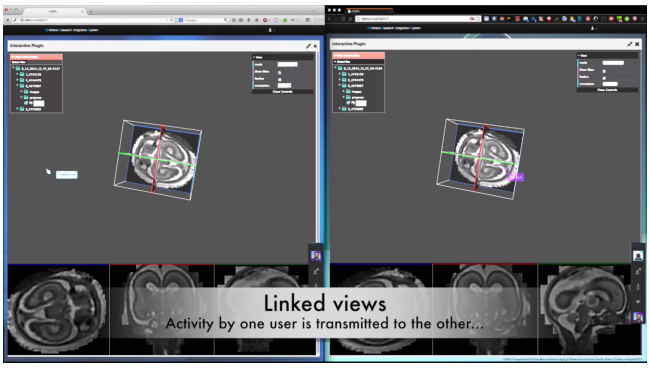


Fig. 4: Realtime collaboration on visual data in *ChRIS*. Here, two viewers are shown. As one viewer updates, so too does the other.

In this mechanism (see Figure 4) any number of browsers can collaborate. The concept of a "room" (much like an IRC chat room) is used and all viewers "enter" the room. The first user is the room creator and all subsequent visitors have their views initially sourced to the creator's current view state.

Unlike screensharing, real time collaboration in *ChRIS* is more akin to networked multi-player gaming architectures. Each viewer/browser has its own complete copy of the data being viewed, and renders this data locally. Shared between users in a "room" are the update events, i.e. mouse events, touch events, etc. In this manner, each connected viewer shows updates to its own internal data in such a way that all viewers show identical information.

An important component of this collaboration is the sharing of connected viewer's mouse pointers between all parties. Consider a situation where one user is discussing a case that is being visualized and with real time collaboration between multiple participants. As the user discusses with others (via microphone, for example) and moves the mouse over parts of the image, the other viewers see exactly what is being highlighted.

III. Design

ChRIS uses the Model-view-controller software architecture pattern. No third party framework is currently in use – the system has been developed from the ground up to be as clean, fast, and efficient as possible. The client is "thin" and the server effects most of the UI processing. In *ChRIS*, the three interconnected parts are:

- **Model**. The model provides a description of the system internal state – this data state is stored in a MySQL DB and describes the users, the feeds, etc.
- **View**. User interface (Javascript) requests information from the **Model** and changes the UI accordingly by adding new feeds, updating the contents of feeds, etc.
- **Controller**. The controller sends commands to update the bf Model's state, i.e. the database maintaining the feeds, and importantly the control interface between *ChRIS* and an external compute resources that processes and analyzes data.

A. Core Runtime Environment

Although *ChRIS* is a web-based service, and thus accessible from any browsing device, the system itself has specific requirements on the server. *ChRIS* requires a Linux environment, providing apache2, php, and MySQL database. If configured to connect to a HPC, *ChRIS* assumes that the HPC is essentially a Linux-based resource with a conventional cluster scheduling paradigm.

B. ChRIS *Filesystem Representations*

A MySQL database is used to track the users in the system, their feeds, and even some of the jobs as they are executed. Importantly, image data is not stored in the database, but directly in the file system. The primary reason for this is to allow access to data without needing *ChRIS* per se; users who can navigate the file system can always get to their data. In fact, feeds are nothing more than directories on a filesystem.

DICOM image data that is transmitted to a *ChRIS* system is packed out directly onto the filesystem, typically to the directory `/home/chris/data` and then organized in directories according to the DICOM `PatientID` field. Within this housing directory, a dated study directory is created, containing directories names according to the `SeriesDescription` tag, and finally the actual DICOM are files are stored in the appropriate directory.

Leverating the filesystem in this manner means that the data is available to be used by any process and not necessarily only within the *ChRIS* system. Similarly, the *feeds* of each user are stored directly in the filesystem, in a predictable location under each user's name in the `/home/chris/users/` directory.

By storing all data directly on the file system we allow knowledgeable users the ability to always access their data and plugin results where necessary (and of course with appropriate permission).

C. ChRIS *Plugin Infrastructure*

The *plugin* infrastructure is built around the idea of processing pipelines that require initial setup, and which then process data to achieve a specific result. These pipelines are for the most part non-interactive, and once started run without interruption until reaching their conclusion. In *ChRIS*, *plugins* are any executable process – compiled C/C++ code, python scripts, bash scripts, etc. Connecting the executable process and *ChRIS* is a thin python-layer/wrapper class that is provided by *ChRIS*. A *plugin* is really just a simple python script that sub-classes the `ChRISplugin` object and creates a UI component, and a `run()` method. Typically in the `run()` method, a command line string is generated that contains the actual executable process that will analyze the *plugin* input. *ChRIS* handles the particulars of associating actual data on the server filesystem with a given process and handles all the translations of the same when executed on remote cluster resource. All *plugins* generate an XML description of themselves, category, version, and all input parameters/types. This XML description is used by *ChRIS* to generate the actual HTML UI of the *plugin* itself. Effectively this means that the python code in a *plugin* is directly used to create the UI for that *plugin*.

IV. Security

ChRIS is designed around Health Insurance Portability and Accountability Act (HIPAA) considerations. Ultimately, however, *ChRIS* assumes that users in the system conform to IRB requirements as pertain to their data. While is it not possible for *ChRIS* to police the activities of users, it does enforce certain behaviors. For example, all data that is transmitted by *ChRIS* to a remote resource (be it another instance of *ChRIS* or a compute cluster) is **always** anonymized by default.

Within *ChRIS* itself, data that is captured by the system (typically DICOM input data) is available to all users of that *ChRIS*. The *feed* results from *plugins* processing that data, however, is only available to the user who created the analysis, and at their discretion, other users with whom the *feed* is explicitly shared.

Data compartmentalization, for example only allowing certain users to see certain input data is currently achieved by instantiating different *ChRIS* systems, each system populated only with the users who can view the data received by that system.

V. Getting *ChRIS* and Deployments

A. *Getting* ChRIS

ChRIS is a completely opensource platform, with its source code available on `github`. Note however, that many *plugins* are in fact third party applications (such as FreeSurfer and The Diffusion Toolkit) and these packages are not distributed with the *ChRIS* source repository.

The easiest mechanism of creating a *ChRIS* instance is to download from the main developers a Virtual Machine image. This image is currently based on Ubuntu 14.10 and contains a complete, stand alone system with many core plugins installed and available. This VM is configured to run all its plugins on the web server, and is thus not a highly scalable option. If an HPC is available, then this *ChRIS* VM can be easily configured to access this HPC.

B. *Deployment*

ChRIS's main "home" is the Boston Children's Hospital, from which it derives part of its name. Development is active, with improvements to its user interface and plugin system being currently designed. Other deployments, as of early 2015, include:

- Massachusetts General Hospital / NMR Center, Boston, MA.
- Massachusetts Green High Performance Compute Center, Boston, MA.
- Cape Universities Brain Imaging Center, Western Cape, South Africa.

with evaluating deployments in

- Scientific Institute IRCCS, E Medea/La Nostra Famiglia, Bosisio, Italy.
- Buzzi Children's Hospital, Milan, Italy.
- Children's Hospital of Philadelphia, Philadelphia, PA.

VI. Conclusion and Future Directions

In its basic philosophy, *ChRIS* seeks to reduce the barrier of accessibility to medical data, to simplify the complex processing of data, and to leverage powerful tools to share and collaborate. *ChRIS* is not only a medical data management system, but strives to improve the quality of healthcare. Consider that *ChRIS* is not only applicable in modern research and clinical institutions but is also use-able *anywhere* there is a viable network signal. For example, in remote, rural communities without direct access to healthcare, or where patient images might have been acquired far away, a network system such as *ChRIS* could offer health care providers access to data in these remote areas. The real time collaboration visualization features would allow for direct consultation between remotely located professionals and help dramatically lower the bar to high quality health care.

All analysis and development conducted by the ChRIS system at the Boston Children's Hospital was conducted under relevant Institutional Review Board approval, which governed access to image data and controlled the scope of sharing of such data.

References

[1] D. Hähn, N. Rannou, P. E. Grant, and R. Pienaar, "Slice:drop," in *IEEE VisWeek 2012*, 2012.

[2] D. Ginsburg, S. Gerhard, J. E. C. Calle, and R. Pienaar, "Realtime visualization of the connectome in the browser using webgl," *Frontiers in Neuroinformatics*, 2011.

[3] D. Hähn, N. Rannou, B. Ahtam, P. E. Grant, and R. Pienaar, "Neuroimaging in the browser using the x toolkit," in *Neuroinformatics 2012*, 2012.