Development of a Web Wrapper to Facilitate Radiotherapy Research

Introduction: Medical physics research groups write many computer programs that have unique implementations, usually requiring a great deal of effort on the user’s part to learn how to build, configure, and use. The recent implementations of such codes on GPU make them less usable to researchers without proper hardware. At the same time, the developers of such research software often lack the time and/or money to develop graphical user interfaces for their programs, and face hardware platform and operating system (OS) dependency issues when trying to do so. This project develops a web-based platform to “wrap” radiotherapy software tools into easy to use web apps.

![Diagram](image)

Fig. 1: The web and tool interface are configured by three XML files (green cards). The users upload input data and command the web interface which in turn triggers a process (set of tool operations) to run on the server. Results are accessed and displayed in the web browser.

Methods and Materials: The web wrapper has two main components that run on a server and a user’s web browser respectively. A web app is uniquely configured through XML files and a modular design is used to accommodate virtually any command line program.

The first component is a set of server-side scripts, written in PHP, which handle user requests, generate HTML, manage tool configuration and inputs, and launch command line tools. The server-side scripts act as a modular interface to any command line program installed on the OS. Once data has been uploaded, a user can initiate a process on the server defined by an XML file which outlines the steps required to configure and launch a command line tool. This process can include writing several configuration files and managing command line arguments based on any combination of default, data-dependent, and user-defined parameters. The server-side scripts also manage user registration and uploading data.

The second component to the web wrapper is a collection of HTML and JavaScript code that runs in the user’s web browser to manage work-flow, user defined input for tool parameters, tool execution (launching), and visualization of data (input and output). The client-side interface is also configured by an XML file which describes all the possible user-defined input parameters for a given tool allowing the user to select the proper configuration from a set of drop down menus.

A third XML file describes the data visualization layout and can be adapted to a specific tool’s output. A set of tools written in C++ and Python are included to enable the visualization of raw data and generation of contour plots. The visualization mechanism consists of a slider to
scroll through multiple CT slices and many transparent layers of data may be superimposed in the display panels.

The modular design of the web wrapper allows developers to easily define a web app for their specific purpose by editing three XML files, one defining tool processes, one defining the tool parameters and user options, and the other for data visualization (Fig. 1). The end result is a simple web-based interface able to command a complex set of command line tools.

The web wrapper back-end was developed on Ubuntu Linux and requires the Apache2 web server and PHP5 to run. The hardware requirements for the back-end are dependent on the tools the developer wishes to wrap. The front-end is written in JavaScript and JQuery, and all modern browsers are supported: Firefox, Safari, Chrome, Internet Explorer, and Opera, running on any major platform: Linux, Mac OS X, Windows, as well as all mobile devices running Android and iOS.

Fig. 2: The tool execution display at end of computation (left panel). The viewer console displaying iso-dose contours for plan dose on the left and step-wash for re-computed MC dose on the right.

**Results:** To test this web platform, we successfully wrapped a set of command line tools, developed by our group, into a single web app which provides fluence map generation, CT image processing, GPU-based Monte Carlo (MC) dose calculation. The combined result is a web-based quality assurance (QA) tool. With this tool, users can upload compressed (e.g., .zip) plan parameters in DICOM-RT format, recompute dose using the MC method, and evaluate the results by viewing dose distributions, 3D gamma index distributions and DVH curves. The entire work-flow is completed in under 2 minutes for each plan. Figure 2 shows two representative screen shots from the web app.

**Conclusion:** We successfully designed and developed a web-based platform for wrapping command line research tools into convenient web apps. The ease of use and accessibility of these web apps lowers the learning curve for users who wish to run advanced research software. The developed platform is also friendly to developers, offering a simple modular layout with minimal configuration required.