**READ ME.docx for RHOK 2010**

**Glenn Pearson Glenn\_Pearson@nlm.nih.gov**

**National Library of Medicine, Bethesda, Maryland, USA**

**Overview of the Project**

Start with the document “LPF 3pg ATA.pdf”, a 3-page handout (from the recent American Telemedicine Associate meeting) that gives a brief overview of the project.

More details about how the project was used during CMAX 2009 can be found here:

<http://wiki.sahanafoundation.org/doku.php/dep:bethesda_2009>

**About TriagePic as a Project**

This is a Visual Studio 2005 project, distributed here as a zip file. Feel free to update it to VS 2008 or VS 2010 (we plan to do the latter shortly anyway). At the end of the project, please return as zipped file (sorry, no public repository yet, but we’re working on it).

TriagePic to date has been programmed by me, at the National Library of Medicine. NLM by law does not assert copyright to its productions, but credits are appreciated.

Parts of TriagePic are based on code samples. TriagePic also uses separately-built DLLs: Filmstrip (slightly modified for this project), Pop3 and Lumisoft (from same source), DialogBox. URLs and credits for these can be found in the TriagePic source files. I have included the modified Filmstrip source code and VS project, just in case it is needed for RHOK. The others shouldn’t be. Details about the filmstrip control at www.codeproject.com/KB/miscctrl/CustomFilmstripControl.aspx

TriagePic interacts, by email and [if you build it] web services, with a customized/extended LPF version of the open-source Sahana disaster management system. No coding of the latter is directly planned for RHOK. While bi-directional web services through LPF/Sahana are planned, at the moment only unidirectional service is implemented at LPF.

**Installation**

There is no install procedure. Copy the unzipped tree root and subtrees to wherever you’d like. The program doesn’t use the Registry, but does have required subdirectories and files, discussed next. (If used with a Bluetooth camera, there would be Bluetooth driver install and configuration issues. However, Bluetooth use will not be part of the RHOK activity.)

**Required Runtime Directory Structure and Files**

*My Documents\Bluetooth Exchange Folder*

This is where the app expects pictures received from the Bluetooth driver to be placed. It is also where the “simulated camera” image is placed, so the directory must be present. (If building a loosely-coupled webcam app, you can place your images here instead. You must follow a camera-like file naming convention.)

*Where the .exe lives (i.e., in Debug, Release, etc.)*

The file TriagePicSharedSettings.xml is the config file for the app’s settings, shared across user accounts on this machine. It’s usually edited indirectly, using the various tabbed pages that the app exposes.

Below that, there are 3 subdirectories:

* 1. text attachments. There are 2 files here with example-legal boilerplate, 1 of which will be included as an email attachment with each file send. (Ignore for RHOK)
  2. outbox queue. Contains outbox.xml, that has the history of sent patient records (and attached images) viewable with the app’s “Outbox” tab.
  3. images. Has 4 subdirectories:
     + just taken. The app moves images here from the My Documents\Bluetooth Exchange Folder.
     + processed. Associated with a patient ID (i.e., renamed), and ready to sent.
     + sent. Done. Images associated with outbox.xml.
     + special. Contains the “no image” and “simulated shot of patient” pictures.

**Running TriagePic the First Time**

**CheckList Tab**

You can specify whether the disaster event is a test or demo (optionally name-able), a named drill, or an actual event. The names are stored locally.

One of the RHOK ideas is to fetch the events from the LPF web service. (See separate doc for specs on that. The web services include a search interface too, but that’s not needed in TriagePic.)

There’s also a map with “disaster radii”. This is a placeholder for future reporting using the TEP/DMOpen network (way too early and complicated for an RHOK task).

At the bottom, feel free to add your own names as the operators of the triage station.

**Email Setup**

TriagePic wants to send information to one or more email accounts. If the nature of your RHOK work doesn’t require actually sending out results, then you don’t have to setup email. Otherwise, read on.

(Also, there’s a “practice” checkbox at the bottom of the page. If you check that, the patient numbering is different, but also it will ask you everytime if you want to actually send out by email.)

The email service is configurable from within the app, through the Email Distribution and Email Setup tabs.

As distributed here, Google is used as the default email server.

**Using and Testing TriagePic**

On the Main tab, hit the “Simulate camera” button, to get a picture. Notice the Patient-ID number below. Ordinarily, 1 photo is associated with ID. However, you can do more than 1. Keep in mind that this is designed for an environment where the picture taker and app user are different people, and there may be a flow of images being transmitted. By default, when there are no images initially seen and a new one arrives, it becomes the “primary” image for that patient ID. As more arrive, they are considered unassigned. In the “filmstrip” viewer, select one, and set its tag to “secondary”. When you send that patient’s data off (by clicking on a colored button), the tagged images (and there can be any number of secondary ones) are sent and disappear from the filmstrip; the patient number is incremented and the next queued image if any automatically made the primary for it.

The colored buttons represent triage zones, based on the quick assessment at the triage station: Green (OK or minor injury), Green BH (behavioral health – physically OK, but needs counseling/psych.), Yellow (urgent but delayable), Red (immediate), Gray (can’t be saved), Black (deceased).

Entering a patient name is optional (often no time to do this in practice), but gender and adult/ped (pediatric patient, i.e., non-adult) is required; you will be prompted for these if you leave them blank (which is OK, but indicates a special case, so it double-checks).

As mentioned earlier, there’s a “practice” checkbox at the bottom of the page. If you check that, the patient numbering is different (prefixed with “Practice” aka “Prac”), but also it will ask you every time if you want to actually send out by email.

**Adding First Web Service to TriagePic**

We are just starting to provide web services, so there’s not much implemented. For the services done so far (and they’re SOAP only, not RESTful), see:

<http://hepldemo.nlm.nih.gov/index.php?stream=soap&act=reg&wbsmod=pls&mod=ws>

and/or

https://docs.google.com/Doc?docid=0ARbUIKLMD\_CSZHR2azJuY184Nzlna2hkbTljeg&hl=en

The shn\_pls\_get\_incident\_list function is what’s of interest. This returns an array of (incident\_id, parent\_id, name, shortname, date, type, latitude, longitude) of listed disaster incidents.

“Name” and “Type” fields are of immediate interest. For the rest, the incident\_id and parent\_id define are positive integers that define a potential hierarchy of events. But for now, parent\_id will always be null. (In theory, a 3-level hierarchy is supportable by our database structure.) Short name might have some use if the regular name is “way too long”. Date, latitude, and longitude (intended to be when and where a real-world event started) are not currently exposed in TriagePic.

Use this function to populate the event radios and (your added) pulldown on the “Checklist” page. Returned value “type” should (I hope) be one of “TEST”, “DEMO”, “DRILL”, or “REAL” [maybe case will be different], associated with radios (You’ll need to separate out the “TEST or DEMO” category to make 4 radios instead of 3. Revise rest of code to support this changed enumeration.). (If you get back some other category, ask user which of 4 radios it should be?)

The web services have simple search services available, too, which are of interest to the ReUnite clone micro-projects, but TriagePic should not do search (that is handled elsewhere).

Equivalent web services are available from hepl… and hepldemo… For the purposes of RHOK, please use hepldemo.