RAT-STATS 17

Statistical Software

Developer Guide

A Challenge Solution Submitted by

J&J Cyber Security LLC

in response to

Dept. of Health and Human Service's

"Statistical Software for Healthcare Oversight" Challenge

Prepared By:

J&J Cyber Security LLC
POBOX 9585
Moscow ID 83843
(208) 883-8181

Executive Summary

This document provides an overview of the software solution submitted in response to the Dept. of Health and Human Services' "Statistical Software for Healthcare Oversight" challenge. The intent of this document is to provide sufficient discussion of the software structure to allow for future modification.

This submission provides the required 4 functions specified in the challenge.

This submission also provides a Graphical User Interface (GUI) like that of the current RAT-STATS program.

This submission also supports input and output from Excel .xlsx files. This support is provided through use of library routines and does not actually run Excel. This submission does not support input or output to MS Access database. A user should be able to import the Excel files into Access.

This submission also does not provide an updated user manual or online help.

This submission contains the following files, the reader of this document is expected to be familiar with the contents of *JJCyber-Overview*.

(1)	JJCyber-RATSTATS.exe.zip	A zip file containing the executable solution and related DLL files. This solution replicates the 4 target RAT-STATS functions: Single Stage of Random Numbers, Unrestricted Attribute Appraisal, Unrestricted Variable Appraisal, and the Stratified Variable Appraisal.
(2)	JJCyber-RATSTATS.src.zip	A zip file containing the source code for the executable, header files, resource files, makefile and relevant DLLs.
(3)	JJCyber-Overview.docx	A document providing a detailed discussion of the submission, including a discussion of differences between the original RAT-STATS and this submission.
(4)	JJCyber-Devel.docx ***This Document***	A document describing the overall code structure and design with guidance how to modify the solution to add additional functionality.
(5)	JJCyber-Licenses.docx	A document listing all software licenses associated with the source code and libraries used as part of the project.
(6)	JJCyber-Summary.docx	A document providing summary submission information as requested on the challenge website.
	README.txt	A simple text version of the preceding information.

Contents

Executive Summary 2

1 Project Overview 4

1.1 Background 4

1.2 Features 4

1.3 Risks / Future Work 4

1.4 Building/Using/Installing 5

1.5 Directory Structure 5

1.6 Modules 5

1.6.1 Main module (winmain.cpp) 6

1.6.2 I/O module 6

1.6.3 General Utilities module 6

1.6.4 Random module 6

1.6.5 Atrtibute Appraisal module 7

1.6.6 Variable Appraisal module 7

2 Licenses 7

3 References 7

3.1 Development Tools 7

3.2 Obtaining Necessary Tools and Libraries: 8

3.2.1 Visual Studio 8

3.2.2 GNU windres 8

3.2.3 OpenXML SDK libraries 8

3.3 resource.rc File: 8

1 Project Overview

1.1 Background

Work began on this submission at the end of November 2016. The first part was to experiment with translating the Visual Basic and R software into different languages to determine ease of translation, functional equivalence and performance. Tests with Python and Java led to the conclusion that these interpreted languages had too many performance issues for confidence interval calculations. Tests with embedding native 'C' code with Python produce acceptable performance results, but greatly increased the maintainability of the code.

A final decision was made to implement the code in C++ (with limited use of objects). This also enabled the use of standard Microsoft tools and controls. The solution is implemented using freely available tools to simplify future maintenance.

The submission is not commercial release/production quality. However, it has been designed to allow additional testing and modification to allow it to grow to production quality.

1.2 Features

This solution to the challenge was designed to utilize the standard Windows Desktop interface and user customizable attributes. As can be seen in the screen shots, it utilizes the standard windows desktop display, with the standard minimize, maximize and exit icons. It utilizes the standard menu bars, both mouse and keyboard selectable.

1.3 Risks / Future Work

This submitted solution does not provide the full functionality of the current RAT-STATS application, and requires a substantial of work to include all of these functions and to become a fully functional product. The following summarize main issues with respect to this submission and future expansion.

English Only: This application has not been internationalized, and was not designed to support multiple languages. Therefore, there is no support for localization of character sets.

Development: The full solution utilizes several technologies, which may complicate future development and support. The implementation is written in C++, with a strong leaning on standard C coding styles. The GUI interface utilizes standard MSDN Windows controls and dialogs. The expansion of these is documented, however the implementation does not tie into a specific GUI development environment. Therefore the placement of the controls is a manual process. This is documented with several examples, such that an experienced C/C++ programmer should be able to expand the software. Place-holders for unimplemented menu items have been placed in the code and GUI. *JJCYBER-Devel* guide provides detailed instructions for expansion.

User Testing: A well-designed program should undergo extensive user testing. This solution has not gone through that process. The design of the GUI interface should allow for relatively simple modifications to improve the user experience. The solution provides a couple of different GUI options, which will be discussed further, that should be evaluated during user testing.

1.4 Building/Using/Installing

- Install VisualStudio Community addition (see last section for links).
- Extract the JJCyber-Src.zip file into a working directory.
- Open a "Developer Command Prompt for Visual Studio" – this link should have been installed when you install visual studio. It will set the appropriate build environment variables for you.
- Change to the working directory where you installed the RAT-STATS 2017 source code.
- Type "nmake" -- this should compile and build the application. The executable is placed in the "build"subdirectory.
- You can type "nmake clean" to remove any temporary files to start a clean build. This is required if you change any of the shared data structures or the numbering in "resource.h".

1.5 Directory Structure

The submission directory is structured as follows:

```
\RATSTAT17: Parent directory
|    Makefile: for building executable, using Visual Studio's nmake
|    README: basic information about the submission
|    devNotes: detailed notes about the submission
|
|---\bin: Binary sub directory, contains extra build tools
|      |    . . .
|
|---\build: Directory where final executable is placed
|
|---\lib: Library sub directory, contains necessary redistributable library components.
|      |    . . .
|
|---\obj: Object code sub directory, contains object code and .res file compiled from the sources
|      |    . . .
|
|---\res: Resource sub directory, contains resource definition script, images and manifest
|      |    . . .
|
|---\src: Source code sub directory, contains source code for the project
       |    . . .
```

1.6 Modules

The software is organized into modules. A main module, an I/O, a general utilities module, and one module for each of the 3 specified functional areas (4 specified functions). Each source file contains internal documentation. The remainder of this section summarizes the modules. The rest of the document provides an overview of some key points in the implementation.

1.6.1 Main module (winmain.cpp)

Files: `src/main.cpp, res/resource.rc, res/Application,manifest, res/*.ico, res/*.bmp`
Headers: `include/resource.h`

The main program is specified and executed from `winmain.cpp`. This creates the application main loop, displays the main screen and menu.

The layout and organization of menus and dialog boxes is described in `resource.rc`, which relies heavily on the definitions in `resource.h`. This is compiled by `wtools/windres` into a windows resource file (`resource.res`).

The application manifest (no need to edit this) and images are stored in the `res` directory as well.

1.6.2 I/O module

Files: `src/printText.cpp, src/excelIO.cs, src/excelWrapper.cpp`
Headers: `include/printText.h, include/excelWrapper.h`

The solution allows for native reading and writing of Excel "*.xlsx" files, utilizing the Microsoft OpenXml SDK. The SDK provides a C# interface for access to openxml files, specifically through the DocumentFormat.OpenXml.dll library. This solution provides a set of C# functions in `excelIO.cs` (compiled as `excelIO.dll`) to manage this access for Excel files for RAT-STATS 2017. To simplify the interface, `excelWrapper.cpp` (compiled as `excelWrapper.dll`) provides a C++ class, ExcelAPIWrapper, which is C/C++ callable, (written in C++/CLI) that interfaces with the C# code. Developers should be able to leave these files untouched and just use the ExcelAPIWrapper methods for Excel file access.

The `printText.cpp` file contains the output functions for all of the functional modules, for both text and excel files. This provides all of the formatting needed for these outputs. Header file is `printText.h`

1.6.3 General Utilities module

Files: `src/util.cpp src/winUtil.cpp`
Headers: `include/util.h, include/winUtil.h`

The `util.cpp` file contains a set of conversion functions usable by the functions. Header file is `util.h.`

The `winUtil.cpp` file contains common Windows/GUI management functions. Header file is `winUtil.h.`

1.6.4 Random module

Files: `src/rnd.cpp src/rndUI.cpp`
Headers: `include/rnd.h`

The function capabilities for generation of Random numbers is implemented in `rnd.cpp`, the user interface is `rndUI.cpp`. Header file is `rnd.h.`

1.6.5 Atrtibute Appraisal module

Files: `src/attr.cpp src/attrUI.cpp`
Headers: `include/attr.h`

The function capabilities for Attribute Appraisal is implemented in `attr.cpp`, the user interface is `attrUI.cpp`. Header file is `attr.h.`

1.6.6 Variable Appraisal module

Files: `src/var.cpp, src/varUI.cpp, src/excelDialog.cpp`
Headers: `include/var.h`

The function capabilities for Variable Appraisal (both Unrestricted and Stratifies) is implemented in var`.cpp`, the user interface is `varUI.cpp`. In addition, selection of data value locations from excel (or text) files is implanted separately in `excelDialog.cpp`. Header file is `var.h.`

2 Licenses

The source code is copyrighted by J&J Cyber Security LLC. License to use this software is provided in compliance with the Challenge rules.

Microsoft DLLs are copyrighted by Microsoft and distributed under their End User License Agreement.

This software was developed using Microsoft Visual Studio 2015, Community Edition. This development software is freely available from Microsoft. The GUI libraries utilize the standard freely redistributable libraries (DLLs) made available with MS Windows and Visual Studio.

Input and output to MS Excel files is implemented through use of Microsoft OpenXml Software Development Software Development Kit libraries. These libraries require access using the C# language. This submission includes custom libraries, developed by J&J Cyber Security LLC, that allow C/C++ access to Excel files. These are explained in *JJCyber-Devel*. The SDK DLLs and the custom DLLs are redistributable.

3 References

3.1 Development Tools

To simplify future development and modification, we used Visual Studio Community Edition (free version) to build the executable.

- Programs are written in C++ for GUIs, calculations and overall program flow.

- Programs are written in C# for reading and writing Excel data files using Windows OpenXML SDK.

- There is a small amount of code written in C++/CLI to manage interoperability between the C++ and C# code.

- GUI information is specified in a .rc resource definition script. We use GNU windres to compile the resource file into a binary .res file linkable to the

3.2 Obtaining Necessary Tools and Libraries:

3.2.1 Visual Studio

To download and install Microsoft Visual Studio, visit the following link:

https://www.visualstudio.com/downloads/

3.2.2 GNU windres

This is precompiled and included wint minggw. To make things simpler, we have included this program with the submission source code, in the “bin” directory.

3.2.3 OpenXML SDK libraries

To make compilation easier, we have included the relevant OpenXML SDK files (a .dll and a .xml file) with the submission source code in the “lib” directory.

3.3 resource.rc File:

MSDN documentation about resource files can be found at the following link, specifically look at the section on Resource-Definition Statements:

https://msdn.microsoft.com/en-us/library/aa380599(v=vs.85).aspx

GNU windres documentation van be found at:

http://www.cs.colorado.edu/~main/cs1300/doc/gnu/binutils_13.html

MSDN Desktop API UI documentation can be found at the following link, pay specific attention to the sections on dialog boxes, menus, and windows.

https://msdn.microsoft.com/en-us/library/ff657751(v=vs.85).aspx