Weapon Zeroing and Warriors' Range Efficiency Analysis System of Small Arms for Bangladesh Army

Software Evaluation Documentation

Group-02
Maj. Sajjad Nowab (201614004)
Maj. Shamim Rahman (201614005)
Akash Poddar (201614051)
Maj. Reazul Haque (201514006)
Shahriar Iqbal (201514079)
Shahriar Kabir Tarafder (201414050)

September 23, 2019

Contents

Chapter 1

Introduction

There are two types of software evaluation approach: criteria-based assessment and tutorial-based assessment according to the Software Sustainability Institute. We have adopted the criteria-based assessment for our integrated design project. Our project is Weapon Zeroing System and Warriors' Range Efficiency Analysis for Bangladesh Army and the aim of this project is to design and develop an automated system for weapon zeroing and warriors' efficiency analysis. Criteria-based assessment is a quantitative assessment of the software in terms of sustainability, maintainability, and usability. This can inform high-level decisions on specific areas for software improvement. A criteria-based assessment gives a measurement of quality in a number of areas. These areas are derived from ISO/IEC 9126-1 Software engineering — Product quality and include usability, sustainability and maintainability. The rest of this document covers each category in greater depth, with lists of questions that is used at the Software Sustainability Institute when compiling detailed software evaluation reports

1.1 Objectives

The assessment involves checking whether the software, and the project that develops it, conforms to various characteristics or exhibits various qualities that are expected of sustainable software. The more characteristics that are satisfied, the more sustainable the software.

Chapter 2

Checklist of Criteria

These assessment criteria for "Criteria-based Software Evaluation" is established by the Software Sustainability Institute which cultivates better, more sustainable, research software to enable world-class research. The assessment criteria are grouped as follows:

Criteria-based Software Evaluation		
Criterion	Sub-criterion	Notes – to what extent is/does the software...
Usability	Understandability	Easily understood?
	Documentation	Comprehensive, appropriate, well-structured user documentation?
	Buildability	Straightforward to build on a supported system?
	Installability	Straightforward to install on a supported system?
	Learnability	Easy to learn how to use its functions?
Sustainability and maintainability	Identity	Project/software identity is clear and unique?
	Copyright	Easy to see who owns the project/software?
	Licencing	Adoption of appropriate licence?
	Governance	Easy to understand how the project is run and the development of the software managed?
	Community	Evidence of current/future community?
	Accessibility	Evidence of current/future ability to download?
	Testability	Easy to test correctness of source code?
	Portability	Usable on multiple platforms?
	Supportability	Evidence of current/future developer support?
	Analysability	Easy to understand at the source level?
	Changeability	Easy to modify and contribute changes to developers?
	Evolvability	Evidence of current/future development?
	Interoperability	Interoperable with other required/related software?

Chapter 3

DETAILED SOFTWARE EVALUATION REPORT

3.1 Usability Evaluation

Usability is the ease of use and learnability of a human-made object such as a tool or device. In software engineering, usability is the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use. The sub-criteria are evaluated in details with respect to our project below.

Usability	
Understandability. How straightforward is it to understand? • What the software does and its purpose? • The intended market and users of the software? • The software's basic functions? • The software's advanced functions?	Yes/No, supporting comments if warranted
High-level description of what/who the software is for is available.	Yes
High-level description of what the software does is available.	Yes
High-level description of how the software works is available.	Yes
Design rationale is available – why it does it the way it does.	Yes
Architectural overview, with diagrams, is available.	Yes
Descriptions of intended use cases are available.	Yes
Case studies of use are available.	Yes

Usability	
Documentation Looking at the user documentation, what is its • Quality? • Completeness? • Accuracy? • Appropriateness? • Clarity?	Yes/No, supporting comments if warranted
Provides a high-level overview of the software.	Yes
Partitioned into sections for users, user-developers and developers (depending on the software).	Yes
Lists resources for further information.	Yes
Is task-oriented.	Yes
Consists of clear, step-by-step instructions.	Yes
Gives examples of what the user can see at each step e.g. screen shots or command-line excerpts.	Yes
For problems and error messages, the symptoms and step-by-step solutions are provided.	Yes
Limitations/constraints are provided clearly in documentation.	Yes
Is on the project web site.	Yes
Documentation on the project web site makes it clear what version of the software the documentation applies to.	Yes

Usability	
Buildability How straightforward is it to? • Meet the pre-requisites for building the software on a build platform? • Build the software on a build platform?	Yes/No, supporting comments if warranted
Software has instructions for building the software.	Yes
Source distributions list all third-party dependencies that are not bundled, along with web addresses, suitable versions, licences and whether these are mandatory or optional.	Yes
Dependency management is used to automatically download dependencies (e.g. ANT, Ivy, Maven or custom solution).	Yes
All mandatory third-party dependencies are currently available.	Yes
All optional third-party dependencies are currently available.	Yes
Tests are provided to verify the build has succeeded.	Yes

Usability	
Installability How straightforward is it to? • Meet the pre-requisites for the software on a target platform? • Install the software onto a target platform? • Configure the software following installation for use? • Verify the installation for use? Note that in some cases build and install may be one and the same.	Yes/No, supporting comments if warranted
Software has instructions for installing the software.	Yes
Source distributions list all third-party dependencies that are not bundled, along with web addresses, suitable versions, licences and whether these are mandatory or optional.	Yes
Dependency management is used to automatically download dependencies (e.g. ANT, Ivy, Maven or custom solution).	Yes
All mandatory third-party dependencies are currently available.	Yes
All optional third-party dependencies are currently available.	Yes
Tests are provided to verify the install has succeeded.	Yes
All GUIs contain a Help menu with commands to see the project name, web site, how/where to get help, version, date, licence and copyright (or where to find this information), location of entry point into user doc.	Yes
Installers allow user to select where to install software.	No
Uninstallers uninstall every file or warns user of any files that were not removed and where these are.	No

Usability	
Learnability How straightforward is it to learn how to achieve ? • Basic functional tasks? • Advanced functional tasks?	Yes/No, supporting comments if warranted
A getting started guide is provided outlining a basic example of using the software.	Yes
Instructions are provided for many basic use cases.	Yes
Instructions are provided supporting all use cases.	Yes
Reference guides are provided for all command-line, GUI and configuration options.	Yes
API documentation is provided for user-developers and developers.	No

3.2 Sustainability and Maintainability Evaluation

Sustainable development aims to meet present needs while ensuring sustainability of natural systems and the environment so as to not compromise the ability of future generations to meet their own needs. Software maintainability is defined as the ease with which a software system or a component can be modified, to correct faults, improve performance or other attributes, or adapt to a changed environment. The sub-criteria are evaluated in details with respect to our project below.

Identity To what extent is the identity of the project/software clear and unique both within its application domain and generally?	Yes/No, supporting comments if warranted
Project/software has its own domain name.	No
Project/software has a logo.	Yes
Project/software has a distinct name within its application area. A search by Google on the name plus keywords from the application area throws up the project web site in the first page of matches.	Yes
Project/software name does not violate an existing trade-mark.	Yes
Project/software name is trade-marked.	No

Copyright To what extent is it clear who wrote the software and owns its copyright?	Yes/No, supporting comments if warranted
Project/software states copyright.	No
Project/software states who developed/develops the software, funders etc.	Yes
If there are multiple Project/software then these all state exactly the same copyright, licencing and authorship.	Yes
Each source code file has a copyright statement.	No
Each source code file has a licence header.	No

Licencing Has an appropriate licence been adopted?	Yes/No, supporting comments if warranted
Project/software states licence.	No
Project/software (source and binaries) has a licence.	No
Project/software has an open source licence.	No
Project/software has an Open Software Initiative (OSI)-recognised licence.	No

Governance To what extent does the project make its management, or how its software development is managed, transparent?	Yes/No, supporting comments if warranted
Project has defined a governance policy.	No
Governance policy is publicly available.	No

Community To what extent does/will an active user community exist for this product?	Yes/No, supporting comments if warranted
Project/software has statement of number of users/developers/members.	No
Project/software has success stories.	Yes
Project/software has quotes from satisfied users.	No
Project/software has list of important partners or collaborators.	No
Project/software has list of the project's publications.	Yes

Accessibility To what extent is the software accessible?	Yes/No, supporting comments if warranted
Binary distributions are available (whether for free, payment, registration).	No
Source distributions are available (whether for free, payment, registration).	No
Access to source code repository is available (whether for free, payment, registration).	No
Ability to browse source code repository online.	No
Repository is hosted externally to a single organisation/institution in a sustainable third-party repository (e.g. SourceForge, GoogleCode, LaunchPad, GitHub) which will live beyond the lifetime of any current funding line.	Yes
Downloads page shows evidence of regular releases (e.g. six monthly, bi-weekly, etc.).	No

Testability How straightforward is it to test the software to verify modifications?	Yes/No, supporting comments if warranted
Project has unit tests.	Yes
Project has component tests.	Yes
Project has integration tests.	Yes
GUI tests are available for project.	Yes
Project has scripts for testing scenarios.	Yes
Project uses automated testing tools.	No
Project has automated tests to check conformance to coding standards.	No
Continuous integration is supported – tests are automatically run whenever the source code changes.	Yes
Test results are visible to all developers/members.	Yes
Test results are visible publicly.	Yes
Tests create their own files, database tables etc.	Yes

Portability To what extent can the software be used on other platforms?	Yes/No, supporting comments if warranted
Application can be built on and run under Windows.	Yes
Application can be built on and run under UNIX/Linux.	Yes
Application can be built on and run under MacOSX.	Yes
Browser applications run under Internet Explorer.	Yes
Browser applications run under Mozilla Firefox.	Yes
Browser applications run under Google Chrome.	Yes

Supportability To what extent will the product be supported currently and in the future?	Yes/No, supporting comments if warranted
Project/software has page describing how to get support.	Yes
User doc has page describing how to get support.	Yes
Software describes how to get support (in a README for command-line tools or a Help=>About window in a GUI).	Yes
Project has an e-mail address.	Yes
Project e-mail address has project domain name.	No
Project/software has site map or index.	Yes
Project/software has search facility.	Yes
Project resources are hosted externally to a single organisation/institution in a sustainable third-party repository (e.g. SourceForge, GoogleCode, LaunchPad, GitHub) which will live beyond the lifetime of the current project.	Yes
If there is a blog, is it is regularly used.	No
E-mail lists or forums, if present, have regular posts.	No

Analysability How straightforward is it to analyse the software's source release to? • To understand its implementation architecture? • To understand individual source code files and how they fit into the implementation architecture?	Yes/No, supporting comments if warranted
Source code is structured into modules or packages.	Yes
Source code structure relates clearly to the architecture or design.	Yes
Project files for IDEs are provided.	No
Source code is commented.	Yes
Source code comments are written in an API document generation mark-up language e.g. JavaDoc or Doxygen.	No
Source code is laid out and indented well.	Yes
Source code uses sensible class, package and variable names.	Yes
Project-specific coding standards are consistent with community or generic coding standards (e.g. for C, Java, FORTRAN etc.).	Yes

Changeability How straightforward is it to modify the software to? • Address issues? • Modify functionality? • Add new functionality?	Yes/No, supporting comments if warranted
Project has defined a contributions policy.	Yes
Contributors retain copyright/IP of their contributions.	No
Users, user-developers and developers who are not project members can contribute.	No
Releases document removed/changed components/APIs in that release.	Yes
Changes in the source code repository are e-mailed to a mailing list.	No

Evolvability To what extent will the product be developed in the future: • For a future release? • within a roadmap for the product?	Yes/No, supporting comments if warranted
Project/software describes project roadmap or plans or milestones (either on a web page or within a ticketing system).	Yes
Project/software describes how project is funded/sustained.	Yes
Project/software describes end dates of current funding lines.	Yes

Interoperability To what extent does the software's interoperability: • Meet appropriate open standards? • Function with required third-party components? • Function with optional third-party components?	Yes/No, supporting comments if warranted
Uses open standards.	Yes
Uses mature, ratified, non-draft open standards.	Yes
Provides tests demonstrating compliance to open standards.	Yes

Appendices

The following links are used to develop the document:

1. https://software.ac.uk/sites/default/files/SSI-SoftwareEvaluationTutorial.pdf
2. https://en.wikipedia.org/wiki/Usability
3. https://isr.uci.edu/content/software-engineering-sustainability-se4s
4. https://ieeexplore.ieee.org/document/7965364