Verification and Validation

for

Rapid LDraw Parts Definition

Version 1.0

Prepared by Sohan Tamang, Theron Anderson

PSU, Fall Capstone 2018

11-27-2018

Table of Contents

Revision History

Date	Reason For Changes	Version
11-27-2018	Creation	0.1
3-14-2019	Revision	0.2
3-17-2019	Version 1	1.0

1. Introduction

1.1. Purpose

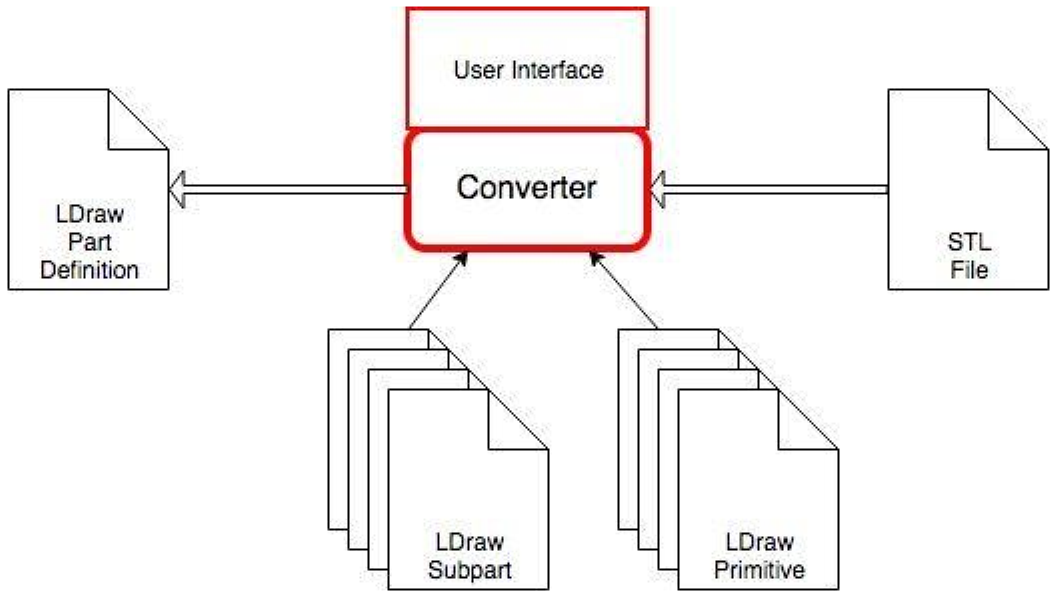
The purpose of this document is to explain the validation and verification processes that will be followed to test and ensure the quality of the Rapid LDraw Parts Definition software. This document is intended primarily for quality assurance engineers as well as the software engineers working on this software application.

1.2. Scope

The scope of this V&V document will include STL file filters, primitives and subparts detector, and the GUI subsystem described in the Requirements and Design documents.

1.3. System Overview

The Rapid LDraw Parts Definition project is a standalone piece of software that will be used to convert 3D STL files of Lego parts to a known, valid LDraw parts definition file. The conversion will take full advantage of the LDraw primitives and subpart definitions and use those where applicable to replace a verbose line, triangle, rectangles. For example, a primitive exists to model the standard Lego™ stud, this will be used instead of a series of rectangles and triangles.

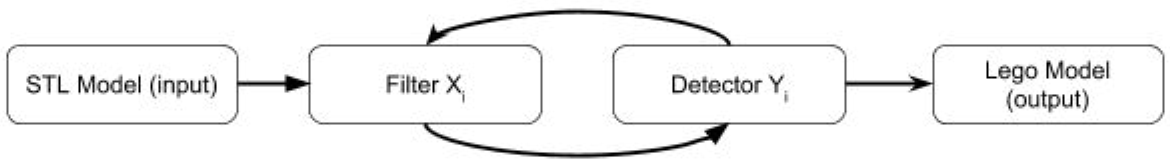


1.4. Terms and Definitions

Term	Definition
CAD (Computer-aided design)	the use of computer systems (or workstations) to aid in the creation, modification, analysis, or optimization of a design
Part Number	a special number LEGO assigns to a specific mold.
LEGO	a line of plastic construction toys that are manufactured by The Lego Group
LEGO Primitive	a highly reusable component of a LEGO part modeled for LDraw.
LDraw	an open standard for LEGO CAD programs that allow the user to create virtual LEGO models and scenes.
GUI	Graphical User Interface
QA	Quality Assurance
Alpha Test	a trial of machinery, software, or other products carried out by a developer before a product is made available for beta testing.
Beta Test	a trial of machinery, software, or other products, in the final stages of its development, carried out by a party unconnected with its development

2. Test Scope and Overview

Coverage and the scope of the tests



2.1. Acceptance Criteria

1. The product will execute on a standard commercial computer (it will not require an custom computing platform).
2. Conversion of an STL scan into a .dat file that minimizes the complexity of the triangulation of the STL file. This is assuming that the STL is not already in its simplest form.
3. The converted .dat file can be seamlessly used by most (if not all) existing CAD programs that use the LDraw parts library. The initial selection of CAD programs to be used for testing will include Bricksmith and LeoCAD.
4. A minimal user interface with the following features
    a. GUI (not command line) that uses standard GUI paradigms (mouse control, menus, buttons, etc.).
    b. Allows file selection of an STL file to be converted. The file selection process should highlight allowed file extensions.
    c. Allows for entry of new LDraw part metadata, as well as the ability to edit existing LDraw part metadata. Where possible, fields should be pre-populated. Allows a user to edit pre-populated fields. At a minimum, the metadata required by the LDraw parts organization will be added to the resulting output .dat file. A link to this part of the specification is posted in the references section of this document.
    d. Where possible, automatically creates the converted .dat file name consistent with the part under conversion. Allows a user to edit the file name.
    e. “Go”, “Pause” and “Cancel” buttons
    f. Aesthetics (like animated transitions or fancy logos), while they enhance the professional appearance, are not necessary. However, the UI should be perceived as “usable” (among other things, this implies a responsive and performant interface).
    g. Localization is not required. The user interface will be in en-US.
5. A mechanism for comparing official LDraw part definition with the converted .dat file. *The exact measure of the “comparison” is to be determined*.

2.2. Functional Requirements

2.2.1 UI

The design and expectation of the UI are available in the UI Specification documentation. The UI should be easily understood by all users with the documentation provided. All functions should be accessible during operation as explained in UI documentation.

2.2.2 Converter

Testing will be done throughout the conversion process to verify that the final output will be correct and accurate. It will also be verified by opening the final product in both of the specified CAD programs.

2.2.3 Triangulation

If the selected STL file is not in its simplest form, this process should simplify the data and lessen the number of triangles that will be converted and passed to the .dat file.

2.3. Non-functional Requirements

2.3.1 Maintainability

This software should be easily maintained by a competent developer. Any improvements or additions should be completed with little difficulty with the help of documentation and the modular nature of the software.

2.3.2 Performance

Conversion completes rapidly. Definition of rapidly will depend on available processing power, the complexity of algorithms and the part being converted. However, the expectation is that the elapsed time will be no more than 2-3 hours (excluding the parts scan). *The time will be negotiable once enough of the product is implemented to gain a full understanding of the complexities involved*

3. Test Strategy

The software will be tested in completeness to assure all requirements are met and it gives a good user experience.

3.1. Feature development and Unit Test

A feature is a functionality added to the current code. We describe a unit of code as any component that relies on an input, performs work on some data related to the input, and returns a result based on actions performed related to the input. Unit testing will be performed alongside development. Developers working on a feature will write unit tests before going out for QA. The unit tests will cover methods and class that a developer touches. The unit test will be conducted on the method level where each method will be passed the set of parameters it takes and be asserted on the expected output. Unit tests are very important because it makes sure that our software works as expected during a change.

3.2. Integration Test

Integration tests are tests that make sure that new features do not break the current system. Continuous Integration makes sure that new commit and pull requests do not break the current system. We will use Travis CI as our continuous integration tool. It creates an environment executes all the tests to make sure changes are not breaking the system.

3.3. System Test

Manual system test will be performed to perform black box testing. Various tests cases are created to meet the acceptance criteria mentioned in the requirements document. A report is generated to track the results of a system test. Development issues and bugs are created to report the failures. Severity level of each issue indicates how critical is the fix to the system.

3.4. Acceptance Test

These tests will be completed to verify that all requirements are met and that the program is usable by all users assuming they have read the instructions and have met user expectations.

3.4.1. Alpha test

The Alpha test will be performed with the sponsor to assure it is built as expected and if not, that any unfulfilled requirements are addressed.

3.4.2. Beta test

The Beta test will be performed by both beta testers and the sponsor, to assure all requirements are met and the software is usable by a user who is unfamiliar with the system.

4. Test Environment

The purpose of this section is to instruct software testers to set up environments to perform all the tests. This section lists all the necessary hardware and software setup to prepare for testing.

4.1. Hardware Setup

- A standard computer
- A standard keyboard
- A standard mouse

4.2. Software Setup

4.2.1 Hardware

System tests will be performed on three major operating systems. Testers should perform all the system tests in following operating systems:

- Windows OS
- Mac OS

4.2.2 Software

The moonshot of this project is to create an installer with users can download and install easily. However, before the software release, all the tests will be performed with manual setup. Software required to perform system testing are:

- Python 3.7 with standard packages
- Other packages mentioned in requirements.txt
- PyCharm
- Bricksmith and LeoCAD

5. References

- Requirement Specification Document
- System Test Plan
- LDraw.org

6. Disclaimers