

DEVELOP National Program Python Package (dnppy)

Software Assurance Classification Report

Prepared by:

Name	Date
Leslie J. Johnson, NASA LaRC Software Assurance Engineer Mission Assurance Branch	11/19/2014

NASA Langley Research Center
Hampton, VA 23681

Revision and History Page

Revision No.	Description	Release Date
-	Initial Release	11/19/2014

Table of Contents

SECTION	PAGE
1. INTRODUCTION	4
1.1 <i>BACKGROUND</i>	4
2. REFERENCE DOCUMENTS.....	4
3. SUMMARY	4
3.1 <i>SOFTWARE CLASSIFICATION</i>	5
3.2 <i>SOFTWARE SAFETY</i>	5
3.3 <i>SOFTWARE ASSURANCE EFFORT</i>	6
APPENDIX A: ACRONYMS	6

1. INTRODUCTION

This report contains the software assurance classification assessment which identifies and evaluates the characteristics of software in determining the software's classification, software safety-criticality, and level of software assurance to be applied to a Project.

1.1 Background

Teams in the DEVELOP program increasingly find themselves using some level of programming to manipulate data. Much of the time, this manipulation is performed in Python. The DEVELOP Python package, referred to as "dnppy" is being created to improve institutional knowledge retention, open the DEVELOP toolkit for public contributions and use, and represent DEVELOP in the public domain. It is a joint social media and programming capacity building endeavor.

The dnppy package will be used to functionalize common programming tasks in the geospatial community, specifically for working with NASA data products. It will include functions for processing satellite data and assist in structuring analysis to reduce the startup time for DEVELOP teams to learn programming and create tools for end users.

This software can streamline common processing tasks of NASA data products.

The package is a python module, written entirely in python.

Assumptions, Limitations, & Errors

- Many functions require the arcpy module, which comes with Arcmap
- There is presently no functionality built into the module that subjects the users to functionally inherent errors.

2. REFERENCE DOCUMENTS

The following documents were used or referenced in the development of this report:

Document No.	Document Title
NPR 7150.2A	NASA Software Engineering Requirements
NASA-STD-8739.8	NASA Software Assurance Standard
NASA-STD-8719.13B	NASA Software Safety Standard
LAPD 5300.1	Program/Product Assurance
LPR 7150.2	LaRC Software Engineering Requirements
LPR 5300.1	Product Assurance Plan
LMS-CP-4754	Software Assurance (SA) for Development and Acquisition

3. SUMMARY

The following paragraphs summarize the results and describe the details used to determine the software classification assessment for this report.

3.1 Software Classification

According to LPR 7150.2, the software component for this Project is classified as Class E – Small Light Weight Design Concept and Research and Technology Software which is defined as

1. *Software developed to explore a design concept or hypothesis, but not used to make decisions for an operational Class A, B, or C system or to-be built Class A, B, or C system, or*
2. *Software used to perform minor desktop analysis of science or experimental data.*

As such, the Project shall follow the instructions and complete the compliance matrix in LMS-CP-7150.6, *Class E Software*, which applies to all Class E software that is not safety-critical.

3.2 Software Safety

The Software Safety Litmus Test below is applied to all projects with software to determine if the software is safety-critical. If the software is determined to be safety-critical, then the project must adhere to the NASA-STD-8719.13, NASA Software Safety Standard.

A software component is considered safety-critical if it meets **any** of the following criteria:

Criteria:	Software components
a. Resides in a safety-critical system (as determined by a hazard analysis) AND at least one of the following apply:	No
(1) Causes or contributes to a hazard	
(2) Provides control or mitigation for hazards	
(3) Controls safety-critical functions	
(4) Processes safety-critical commands or data	
(5) Detects and reports, or takes corrective action, if the system reaches a specific hazardous state	
(6) Mitigates damage if a hazard occurs	
(7) Resides on the same system (processor) as safety-critical software	
b. Processes data or analyzes trends that lead directly to safety decisions	No
c. Provides full or partial verification or validation of safety-critical systems, including hardware or software subsystems.	No

The software components in this Project do not reside in a safety-critical system; process data or analyze trends that lead directly to safety decisions or provide full or partial verification or validation of safety-critical systems.

The LaRC Safety and Mission Assurance Office have determined that the software components in this Project are not safety-critical.

3.3 Software Assurance Effort

The software assurance effort is based on the software class and impacts from potential failure. In accordance with LMS-CP-4754 software assurance is not applicable for non-safety critical Class E software developments.

APPENDIX A: ACRONYMS

CP	Center Process
LaRC	Langley Research Center
LAPD	Langley Policy and Directives
LMS	Langley Management System
LPR	Langley Procedural Requirements
NASA	National Aeronautics and Space Administration
NPR	NASA Procedural Requirement
SA	Software Assurance
STD	Standard