**Reliance General Insurance**

**Software Quality Assurance Plan**

October 11, 2018

## 1.1 Purpose

The purpose of this Software Quality Assurance Plan (SQAP) is to define the techniques, procedures, and methodologies that will be used for “Reliance General Insurance” (hereafter referred to as RGI) to assure timely delivery of the software that meets specified requirements within project resources. The format of this plan follows the requirements in the IEEE Std. 730-2002.

## 1.2 Scope

The primary audience for this document is the RGI project team. The team members are responsible for following the quality standards laid out while developing the application, documenting the results, monitoring the project progress, and testing the project quality. This SQAP (Software Quality Assurance Plan) describes the procedures and control methods to obtain the desired quality of the end products and the process by which these end products are created. It will therefore specify product and process quality standards and will serve as a guide for the client and development team.

## 1.2 Reference Documents

**IEEE Std. 730-2002**

IEEE Standard for Software Quality Assurance Plans. This document defines the standards for making the SQAP document.

# 2. SQA Description

## Quality Assurance

To obtain the desired quality for the end products and the process by which these end products are created a project plan will utilized. The quality plan should set

out the desired software qualities and describe how these qualities are to be assessed. It

defines what “high-quality” software actually means for a particular system.

Quality Assurance for this project will include at least one analysis of all current draft deliverables and selected work products in each stage of development. The reviews will assure that the established system development and project management processes and procedures are being followed effectively, and exposures and risks to the current Project Plan are identified and addressed

**SQA Roles and Responsibilities**

The following chart defines the SQA roles and responsibilities of the members of the project

team and their function at stage exit.

|  |  |
| --- | --- |
| **Roles** | **SQA Responsibility** |
| QA  Manager | Manages the Quality Assurance  function. |
| System  Owner | Helps define product quality  expectations. Represents  procurement users. Determines  final acceptance of RGI |
| QA  Consultant | Audits and approves project  deliverables from QA  perspective. Reviews plans and  deliverables for compliance with  applicable standards. Provides  guidance and assistance on  process matters. |
| Project  Manager | Ensures implementation of  quality activities. Coordinates  resolution of issues. Provides  regular and timely  communications. |

**Required Skills**

The Quality Assurance consultant must be able to review iterations of the Project Plan and

lifecycle work products to determine industry standards, as modified and

documented in the Project Plan, and provide expert assistance on project management practices

and software development process related matters.

This position will work independently from the development team to ensure objective audits of

the work products as they are being developed and objective reviews of project management

processes and stage exit.

**Verification and Validation of Requirements**

Verifying RGI requirements at the end of the Preparation stage will establish the proper basis

for initiating the Software Design stage activities. The Functional Requirements Document

(FRD) must contain, at a minimum, documentation on the essential requirements (functions,

performance, design constraints, and attributes) of the software and its external interfaces.

The following IEEE definitions apply in this RGI:

**verification**: The process of determining whether or not the products of a given stage of the

software development cycle fulfills the requirements established during the previous stage.

**validation**: The process of evaluating software at the end of the software development

process (acceptance testing activity in the Testing stage) to ensure compliance with software

requirements.

The term **requirements** encompass the areas of hardware, user interface, operator, software

interface, functionality, performance, communications, security, access, and backup and

recovery.

**Verification**

The following activities will be performed as part of requirements verification:

* Produce a traceability matrix tracing all FRD requirements back to system objectives in the RGI Objectives Statement and forward to Software Design elements.
* Evaluate FRD requirements and relationships for correctness, consistency, completeness,

accuracy, readability and testability.

* Assess how well the FRD satisfies the RGI system objectives.
* Assess the criticality of requirements to identify key performance or critical areas of software.

**Validation**

The following activities will be performed as part of requirements validation:

* Plan acceptance testing, including criteria for:
* compliance with all requirements
* adequacy of user documentation
* performance at boundaries and under stress conditions.
* Plan documentation of test tasks and results.
* Execute the Acceptance Test Plan.
* Document acceptance test results.

# Documentation

This section shall perform the following functions:

1. Identify the documentation governing the development, verification and validation, use, and maintenance of the software.
2. List which documents are to be reviewed or audited for adequacy. For each document listed, identify the reviews or audits to be conducted and the criteria by which adequacy is to be confirmed.

To ensure that the implementation of the software satisfies the technical requirements, the following documentation is required as a minimum.

## Software Requirements Document (SRD)

Software specification review is to be used to check for adequacy and completeness of this documentation. The Software Requirements Document, which defines all the functional requirements, quality attributes requirements and constraints on the RGI project.

## Software Test Reports

Software Test Reports are used to communicate the results of the executed test plans. This being the case, a particular report should contain all test information that pertains to the current system aspect being tested. The completeness of reports will be verified in walkthrough sessions.

## Software Architecture and Design

Software Architecture and Design reviews are to be used for adequacy and completeness of the design documentation. This documentation should depict how the software will be structured to satisfy the requirements in the SRD. The SDD should describe the components and subcomponents of the software design, including databases and internal interfaces

## User Documentation

User documentation guides the users in installing, operating, managing, and maintaining software products. The user documentation should describe the data control inputs, input sequences, options, program limitations, and all other essential information for the software product. All error messages should be identified and described. All corrective actions to correct the errors causing the error messages shall be described.

# Testing strategy

Testing for the RGI project seeks to accomplish two main goals:

1. Detect failures and defects in the system.
2. Detect inconsistency between requirements and implementation.

To achieve these goals, the testing strategy for the RGI system will consist of four testing levels. These are unit testing, integration testing, acceptance testing, and regression testing. The following subsections outline these testing levels, which development team roles are responsible for developing and executing them, and criteria for determining their completeness.

## Unit Testing

The target of unit tests is a small piece of source code. Unit tests are useful in detecting bugs early and also in validating the system architecture and design. These tests are done one function at a time and written by the developer. Ideally each logic path in the component and every line of code are tested. However, covering every line of code with unit tests is not time or cost effective in most cases. Code coverage goals will be defined to ensure that the most important code is well covered by tests while still making efficient use of developer time.

All unit tests must be executed and passing before each check-in to the source control system. Unit tests will also be run automatically as part of the continuous integration process. The results of these test runs will be stored by the continuous integration system and emailed to the development team.

## Integration Testing

Integration testing will execute several modules together to evaluate how the system as a whole will function. Integration tests will be written and executed by the testing team. Attempting to integrate and test the entire system all at once will be avoided as it makes finding the root cause of issues more difficult and time consuming. Instead, integration tests will be done at specific points, ideally where one component interacts with another through an interface. Integration tests will focus on these specific points of interaction between two components. This testing of interaction between two modules ultimately leads to an end-to-end system test. Each test is written to verify one or more requirements using the scenarios or use cases specified in the requirements document. Integration tests also include stress or volume testing for large numbers of users.

## Acceptance Testing

Acceptance testing is functional testing that the customer uses to evaluate the quality of the system and verify that it meets their requirements. The test scripts are typically smaller than integration or unit testing due to the limited time resources of the customer. Acceptance tests cover the system as a whole and are conducted with realistic data using the scenarios or use cases specified in the requirements as a guide.

## Regression Testing

The purpose of regression testing is to catch any new bugs introduced into previously working code due to modifications. As such, the regression test suite will be run every time the system changes. Regression tests will be created and run by the testing team. Regression testing will consist of running previously written automated tests or reviewing previously prepared manual procedures. It is common for bug fixes to introduce new issues and therefore several “test/fix” cycles will be planned and conducted during regression testing.