# Kreogist Mail
# Development Documentation

**Software Verification and Validation Plans**

**January 25, 2016**

Reference Number: KMKOT05.
This document is capable with IEEE Std 1012-2004.

All information provided here is subject to change without notice. Contact Kreogist Dev Team to obtain the latest Kreogist product specifications and roadmaps.

Kreogist Mail may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

No computer system can be absolutely secure. Kreogist does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

The products described may contain design defects or errors which may cause the product to deviate from published specifications. Current characterized errata are available on request.

All the other documents mentioned in this document could be found at the official site of Kreogist Dev Team. Contact Kreogist Dev Team if there's any trouble.

Intel, Intel Core and the Intel logo are trademarks of Intel Corporation in the U. S. and/or other countries.

Linux is a trademark of Linus Torvalds in the U. S., other countries, or both.

Microsoft and Microsoft Windows are trademarks of Microsoft Corporation in the U. S. and/or other countries.

Macintosh is a trademark of Apple Inc. in the U. S., other countries, or both.

UNIX is a registered trademark of The Open Group in the U. S. and other countries.

*Other names and brands may be claimed as the property of others.

**First Edition (Jan 2016)**

This edition applies to Version 0.1 of Kreogist Mail.

Kreogist Dev Team

# Kreogist Mail
# Development Documentation
# Software Verification and Validation Plans

January 25, 2016

## Revision History

| Revision | Version | Description | Date |
|---|---|---|---|
| KMKOT05 | -001 | Initial commit | Jan. 23th, 2016 |
| KMKOT05 | -002 | Internally accepted | Jan. 25th, 2016 |

## Preface

This document is an update to the specifications contained in the "Affected Documents" table below. This document is a part of product (project) Kreogist Mail.

This document may also contain information that was not previously published.

# Affected Documents

| Document Title | Document Number |
|---|---|
| Kreogist Mail Software Project Management Plans | KMKOT02 |
| Kreogist Mail Software Quality Assurance Plans | KMKOT04 |

# Related Documents

| Document Title | Document Number |
|---|---|
| Kreogist Mail Software Requirement Specification | KMKOT01 |
| Kreogist Mail Software Design Specification | KMKOT03 |

# Contents

# 1 Introduction

## 1.1 Purpose

This Software Verification and Validation Plans (SVVP) specifies the tasks to be performed for Kreogist Mail software project (The project). This SVVP is written in accordance with IEEE Std 1012-1998.

## 1.2 Scope

The scope of this SVVP is to verify and validate all Kreogist Mail software (Mail) and to validate all packaged and embedded software components for conformance to requirements for safety factors, response time, expandability, and external interfaces. Until a customer is under contract, the acceptance testing activities will not be included.

## 1.3 Objectives

The objectives of this SVVP are to verify and validate all developed software and interfaces for conformance with Mail requirements for response time, expandability, and safety factors.

## 1.4 Applicability

This SVVP applies to the products of the requirements, design, implementation, and test phases of Mail development cycle.

## 1.5 Waivers

This SVVP deviates from IEEE Std 1012-1993 in that it will not be applied to the following phases:

- Concept phase
- Installation and checkout phase
- Operation and maintenance phase

This SVVP has been developed during the concept phase and applies to subsequent phases. The V&V planning for the installation and checkout phase, and the operation and maintenance phase will be deferred until a customer installation has been identified for the project.

## 2 Referenced Documents

This section will identify the binding compliance documents, documents referenced by this Plan, and any supporting documents required to supplement or implement this Plan.


[1]. Kreogist Dev Team. "Kreogist Mail Software Requirements Specification", January, 2016.

[2]. Kreogist Dev Team. "Kreogist Mail Software Project Management Plans", January, 2016.

[3]. Kreogist Dev Team. "Kreogist Mail Software Quality Assurance Plans", January, 2016.

[4]. Free Software Foundation, "GNU Free Documentation License", See `http://www.gnu.org/licenses/fdl.html` (last checked January 13th, 2016), November 3, 2008.

[5]. Wikipedia, "Qt (software) - Wikipedia, the free encyclopedia", See `https://en.wikipedia.org/wiki/Qt_%28software%29)`last checked January 16th, 2016), January 4, 2016.

[6]. IEEE Software Engineering Standards Committee. "IEEE Std 1059-1993, IEEE Guide for Software Verification and Validation Plans", December 2, 1993.

[7]. IEEE Software Engineering Standards Committee. "IEEE Std 1063-2001, IEEE Standard for Software User Documentation", December 20, 2001.

[8]. IEEE Software Engineering Standards Committee. "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.

[9]. IEEE Software Engineering Standards Committee. "IEEE Std 1016-1998, IEEE Standard for Information Technology–Systems Design–Software Design Descriptions", July 20, 2009.

[10]. IEEE Software Engineering Standards Committee. "IEEE Std 1058-1998, IEEE Standard for Software Project Management Plans", December 22, 1998.

[11]. IEEE Software Engineering Standards Committee. "IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Processes", 1997.

[12]. IEEE Software Engineering Standards Committee. "IEEE Std 1012-1998, IEEE Standard for Software Verification and Validation", 1998.

[13]. IEEE Software Engineering Standards Committee. "IEEE Std 828-2005, IEEE Standard for Software Configuration Management Plans", August 12, 2005.

# 3 Definitions

This section will define or provide a reference to the definitions of all terms required to properly interpret the Plan. This section shall describe the acronyms and notations used in the Plan.

Table 4: Definitions, Acronyms & Abbreviations

| Acronyms | Definition |
|---|---|
| AD | Architectural Design |
| ADD | Architectural Design Document |
| ATP | Acceptance Test Plan |
| baseline | a work product that has been formally reviewed and accepted by the involved parties |
| Client | Monitor, Agent or Submitter |
| CM | Configuration Management |
| FFP | Firm-Fixed-Price; a price for fulfilling the contract that will not be under- or over-run |
| IEEE | Institute of Electrical and Electronics Engineers |
| milestone | a scheduled event used to measure progress |
| MG | Merge Phase |
| QAM | Quality Assurance Manager |
| RA | Reanalysis Phase |
| SC | Sprint Check Phase |
| SCMP | Software Configuration Management Plan |
| SDS | Software Design Specification |
| SM | Senior Management |
| SPMP | Software Project Management Plan (this document) |
| SQA | Software Quality Assurance |
| SQAP | Software Quality Assurance Plan |
| SR | Software Requirements |
| SRD | Software Requirements Document |
| SRS | Software Requirements Specification |
| STD | Software Transfer Document |
| STP | Software Testing Plan |
| SUM | Software User Manual |
| SVVP | Software Verification and Validation Plan |
| TR | Transfer Phase |
| UTP | Unit Test Plan |
| VPM | Vice Project Manager |
| VQAM | Vice Quality Assurance Manager |

# 4 V&V overview

This section will describe the organization, schedule, resources, responsibilities, tools, techniques, and methodologies necessary to perform the software verification and validation.

## 4.1 Organization

For the organization within the project and responsibilities of individual members of teams see *Kreogist Mail Software Project Management Plan*[2] (SPMP). Team leader leads the V&V team. Furthermore team leader is responsible for the project and performance of V&V team. When the project is finish, V&V team has to verify and validate the whole part of the software.

## 4.2 Master schedule

Main task of V&V team is to check whether procedures are followed properly and that standards are handled correctly as defined in this document. Additionally V&V team will check whether the software is enabled to be released. Besides described main task, V&V team has to following additional tasks:

1. Organize V&V meeting.
2. Attend V&V reviews.
3. Notify Kreogist Dev Team to release the software when Mail is recognized as accept.
4. Notify team leader and Quality Assurance Manager to check the product when Mail is recognized as failed.

## 4.3 Software integrity level scheme

Table 5 defines the software integrity levels used for Mail.

Table 5: Assignment of software integrity levels

| **Software integrity level** | **Description** |
|---|---|
| 1 | Software element must execute correctly or intended function will not be realized, causing negligible consequences. Mitigation not required. |
| 2 | Software element must execute correctly or an intended function will not be realized, causing minor consequences. Complete mitigation possible. |

Continued

| Software integrity level | Description |
|---|---|
| 3 | Software element must execute correctly or the intended use (mission) of the system/software will not be realized, causing serious consequences (permanent injury, major system degradation, economic or social impact). Partial to complete mitigation is possible. |
| 4 | Software element must execute correctly or grave consequences (loss of life, loss of system, economic or social loss) will occur. No mitigation is possible. |

### 4.4 Resources summary

The project resource summary please check the reference section in SPMP[2]. All the resource for V&V team is list below:

1. Team leader
2. Project manager
3. Quality Assurance Manager

V&V team will be setup after all the project is done and Software Quality Assurance team ensures that the software is ready to be released. Software Quality Assurance team should hand in a report that the latest version is ready to be checked by V&V team.

### 4.5 Responsibilities

Main responsibility for V&V team tasks, as described in section 4.2, i.e. *Master schedule*, lies with team leader. Team leader can delegate the tasks within V&V team. All the problems have to be reported to team leader, and all the problems will be gathered and report to Kreogist Dev Team. In case team leader will be unavailable for a short period of time, one of the team member will assume his tasks. If team leader will be unavailable for a longer period of time, V&V team must be expanded and the tasks reorganized. Project Manager will be responsible for this.

### 4.6 Tools, techniques, and methods

All the tools and development environment could be found at section *Methods, Tools, and Techniques* of SPMP[2].

V&V tools is defined at section *Verification and Validation Plan* of SPMP[2]. The process of V&V can be found at section 5, i.e. *V&V processes*.

# 5 V&V processes

SVVP will identify V&V activities and tasks to be performed for each of the V&V processes, and will document those V&V activities and tasks. The following contain an overview of the V&V activities and tasks for all software life cycle processes.

SVVP will address the following topics for each V&V activity.

## 5.1 Dependencies

To start up V&V process, the following items should be handed in to team leader:

1. Software program and library binaries
2. Software source code
3. Software Requirements Specification (SRS), ID: KMKOT01
4. Software Quality Assurance team report.

The following items should be prepared:

1. Target platform machine.
2. System interfaces on target platform.

The following tools should be prepared for the V&V process:

1. cppcheck
2. Valgrind
3. System Process Manager

The process will be divided into the following steps:

- Stability verification.
- Function verification.
- Quality verification.

The following section will show up the whole verification check process.

## 5.2 Stability Verification

Code stability test will be done on the following platform:

**Software Requirement**

- OS: Ubuntu Linux 15.10 64-bit
- Tools: cppcheck (static analysis tool), Valgrind (dynamic analysis tool)

**Minimal Hardware Requirement**

- CPU: 2.0GHz Intel® Core™ 2 Duo CPU
- Memory: 1GB DDR II 800
- Disk Space: 1GB of available storage

Besides, Qt platform and all the other dependencies of Mail and testing tools will be installed.

First, use `cppcheck` to do the static check of the code. There shouldn't be any critical errors in the following list:

1. Local auto-variable usage error.
2. Array out of bound error.
3. Class error.
4. Abandon function usage error.
5. Abnormal memory usage error.
6. Memory leak.
7. Abnormal STL function usage error.
8. Other critical error in the cppcheck critical error list.

Next, compiled the source code and do the dynamic check of the code in Qt Creator with Valgrind. Valgrind shouldn't detect any of the following critical memory errors:

1. Accesses memory it shouldn't (areas not yet allocated, areas that have been freed, areas past the end of heap blocks, inaccessible areas of the stack).
2. Uses uninitialised values in dangerous ways.
3. Leaks memory.
4. Does bad frees of heap blocks (double frees, mismatched frees).
5. Passes overlapping source and destination memory blocks to memcpy() and related functions.

The report of cppcheck and Valgrind will be including into V&V report.

### 5.3 Function Verification

Black-box testing will be performed on the implemented software interfaces to ensure that the outputs of each interface are consistent with what is input, based on the interface design.

All the requirements define in SRS will be used as input in black box test. All the requirement should be pass the black-box testing with all the known exceptions processed. All the test will be recorded in the report.

### 5.4 Quality Verification

Tools included in `Valgrind` will be used as quality verification. The usage of tools could be founded on the official site of Valgrind: `http://valgrind.org/`. All the output of the tools will be recorded as a part of the report.

### 5.5 Summary

In any of the process, if there's one critical error detected, the process will be forced stop, and the V&V meeting will be marked as failed. The problem will be marked as an issue and it should be setup in Github issue tracking system. The report will be abandon.

After all the processes done, the report will be marked as accept. V&V team will notify Kreogist Dev Team to release Mail.

## 6 V&V reporting requirements

Summarize all V&V activities and results, including status and disposition of anomalies in the V&V final report.

The V&V report should contains the following things:

1. Introduction
2. Stability Verification Report
    - cppcheck Output History
    - Valgrind Output History
3. Function Verification Report
4. Quality Verification Report

5. Summary

6. Signature

# 7 V&V administrative requirements

This section will describe, at a minimum, the V&V administrative procedures described in the following sections.

## 7.1 Anomaly resolution and reporting

All the problem in any section will be recorded as an issue. On Github issue tracking system, all the issue will be saved as an issue item. The category of issue should be one of the following:

- bug
- enhancement
- duplicate
- help wanted
- invalid
- wontfix

After the issue set up, one of the team member will be allocated as assignee of the issue. The assignee will be response for the issue. After the issue has been solved, the issue will be closed. After the issue has been verification by Software Quality Assurance team, V&V process will be start from very beginning.

## 7.2 Control procedures

All the issues which are checked in V&V process will be marked as critical issue. The issue should be notify team leader. Team leader will decide the way to solve the issue. Team leader will be response for all the issue.

## 7.3 Standards, practices, and conventions

The whole V&V process will be planned followed by IEEE std 1012-2004.