

Vista Challenge

Proposal B Enhancements

Software Requirements Specification (SRS)

Version 1.2

Draft



December 2004

Department of Veterans Affairs
VistA Health Systems Design & Development (HSD&D)
Infrastructure and Security Services (ISS)

Revision History

Note: The revision history cycle begins once changes or enhancements are requested to an approved SRS.

Date	Revision	Description	Author
04/08/04	1.0	Initial Version – created for VistA Challenge.	
04/16/04	1.1	Edits to grammar, punctuation, and VistA documentation standards compliance.	
11/09/04	1.2	Revised Version – added information for Proposal B enhancements.	

Table of Contents

1. INTRODUCTION.....	5
1.1. PURPOSE.....	5
1.2. SCOPE.....	5
1.3. ACRONYMS AND DEFINITIONS.....	5
1.5. REFERENCES.....	6
2. OVERALL DESCRIPTION	7
2.1. PRODUCT PERSPECTIVE	7
2.1.1. User Interfaces.....	7
2.1.2. Hardware Interfaces	7
2.1.3. Software Interfaces.....	7
2.1.4. Communications Interfaces.....	7
2.1.5. Memory Constraints	7
2.1.6. Special Operations.....	7
2.1.7. Implementation Requirements	7
2.2. PRODUCT FEATURES OR FUNCTION	8
2.3. USER CHARACTERISTICS	9
2.4. DEPENDENCIES AND CONSTRAINTS	9
2.5. APPORTIONING OF REQUIREMENTS	10
3. SPECIFIC REQUIREMENTS	11
3.1. DATABASE REPOSITORY.....	11
3.2. SYSTEM REQUIREMENTS	11
3.3. PERFORMANCE REQUIREMENTS	11
3.4. DESIGN CONSTRAINTS.....	11
3.5. SECURITY	12
3.6. PORTABILITY	12
3.7. OTHER REQUIREMENTS	12
4. FUNCTION POINT ESTIMATION.....	13

1. Introduction

The VistA Challenge is a contest issued by Health Systems Implementation Training and Enterprise Support (HSITES) via FORUM to all field facilities. Its goal was to solicit functionality for streamlining and increasing efficiency in maintaining Veterans Health Information Systems and Technology Architecture (VistA) systems at Veterans Affairs Medical Centers (VAMC). Two proposals were initially selected as VistA Challenge winners for implementation, A and B. Proposal A, also known as the VistA Auto Patch Utility (VAPU) is a Class III software application that was created to automate the Kernel Installation and Distribution System (KIDS) installation steps, giving the VAMCs the ability to automate patch installations. Proposal B is the master servant patching system, which allows site personnel to patch several sites remotely. As work progressed on proposal A, the proposal submitters continued to discuss the proposals and it became clear that they wanted to advance not only proposal B, but an enhanced version of that proposal. This Software Requirements Specification (SRS) addresses enhanced proposal B, which is a combination of all three proposals.

This is a modified SRS for the VistA Challenge project. For purposes of accelerating this project, certain sections of this document do not apply to this release and will be marked Not Applicable within this document.

1.1. Purpose

The purpose of this Software Requirements Specification (SRS) is to outline the requirements for the VistA Challenge project. The Class III software application has been delivered to Health Systems Design and Development (HSD&D) for incorporation into the Class I VistA Infrastructure & Security Services (ISS) suite of applications.

The targeted audience of this document includes HSITES, Development & Infrastructure Support (DaIS), Infrastructure & Security Service (ISS), Health Systems Design and Development (HSD&D), all VistA sites, and VAMCs.

This SRS will provide user functionality as well as technical information.

1.2. Scope

The goal of this project is the reclassification of software from Class III to Class I, and delivery to HSITES for field testing, training, and implementation.

1.3. Acronyms and Definitions

Acronym	Definition
HSD&D	Health Systems Design & Development

Acronym	Definition
HSITES	Health Systems Implementation Training and Enterprise Support
KIDS	Kernel Installation and Distribution System
VAMC	Veterans Affairs Medical Center
VAPU	VistA Auto Patch Utility
VISTA	Veterans Health Information Systems and Technology Architecture

1.5. References

The following references are addressed in the SRS:

- VistA Challenge - Initial Assessment and Briefing
- VistA Challenge – Project Management Plan

2. Overall Description

2.1. Product Perspective

Per consideration from HSITES, enhanced proposal B is to become VistA ISS Class I software as soon as possible, estimated time for completion is June 2005.

2.1.1. User Interfaces

User enters data into KIDS via the use of the VistA Character-based User Interface (a.k.a. roll-and-scroll). Patch XX*XX*XX will provide this functionality.

2.1.2. Hardware Interfaces

There are no hardware/system interface issues with this project.

2.1.3. Software Interfaces

VAPU interfaces with the Infrastructure & Security Services side of VistA. The account(s) must contain the fully patched versions of the following software:

- KIDS V.8.0 or greater,
- MailMan V.8.0 or greater,
- Kernel V.8.0 or greater, and
- VA FileMan V.22 or greater

2.1.4. Communications Interfaces

Transmission Control Protocol/Internet Protocol (TCP/IP) and MailMan will be the communication protocol.

2.1.5. Memory Constraints

None.

2.1.6. Special Operations

None.

2.1.7. Implementation Requirements

None.

2.2. Product Features or Function

- The goal of this software is to improve VistA and HealtheVet software patch installation efficiency. This will assist in maintaining VistA systems at VAMCs. This benefits the Veterans Health Administration by significantly reducing the time spent on installing patches.
- Streamline maintenance, specifically for the installation of patches.
- In Enhanced Proposal B, there is an ‘auto-load’ process for patches received via FORUM. Although all patches cannot be auto-loaded, there are site files that can be used to prevent application patches from being auto-loaded. See the following bulleted list for more detail on this functionality:
 - Parameters can be set up to turn off Taskman and lock the system during patch installation if you want to turn off certain packages. Therefore, certain patches could auto-load and others would require human intervention.
 - If certain variables are present, the package makes changes to national patching routines to allow the package to “kick in”. This will allow the package to lie dormant if it is not “turned on” and patches could be installed as normal.
 - If a local field in the package file were set to enabled, then patches would auto-install when the variables show up in the routines. The package saves copies of the verify process and the install process in a directory on the hard drive for each patch that gets installed. A backup of the install is kept in a mail box of the person running the patch.
- Additional functionality of Enhanced Proposal B is the master-servant patching process. This will satisfy the requirement for a package that could be used by sites to patch other sites remotely. See the following bulleted list for more detail on this functionality:
 - This allows site personnel to patch several sites remotely. Sites could allow other sites to patch them; however, there is a way to make sure that only the authorized site would be able to send patches that would auto-install.
 - To make sure only one site could do this, Enhanced Proposal B suggests the idea of a master site that could send patches to self-install on a subordinate or “servant” site, similar to master-client servers.
 - Site parameters would be established allowing the master site to send patches for install to the servant site under pre-specified conditions. The messaging that occurred between the two sites would be recorded by the package in the

site files and reports could be generated that would allow a user to quickly see the status of the auto-install of those patches.

- Once this master-servant package was set up, a package could be developed that would run all components of a standard patch install by calling the same entry points in the KIDS package with pre-defined variables, and the patching process would proceed as if the patch were being run interactively.
- Enhanced Proposal B is much like the package that the White River Junction VAMROC currently uses. This package keeps track of messages as they arrive on station in the message file. See the following bulleted list for more detail on this functionality, which also describes the types of reports can be generated:
 - A server would populate a file with the information about patches when they arrive and would calculate when they are due, and then establish links from this file to the install file when patches are installed.
 - This package will notify IRM personnel when patches are going to be late. In addition, this package will also generate statistics at the end of the month to determine IRM's compliance level with patch install dates.
 - As mail messages are purged, the patch file would still keep the information about the message so a trail of the date the patch arrived on station and when it was installed will still be on record, though the actual message no longer exists and, therefore, the record of arrival died with the message.
 - This does not install patches, only keeps track of local patching.

2.3. User Characteristics

Users of the Vista Auto Patch Utility include all VAMCs.

2.4. Dependencies and Constraints

Dependency—Field testing (alpha & beta), training, implementation, and support will be performed by HSITES.

Constraints—Many patches would still have to be manually installed:

- Kernel patches that require a running of ZTMGRSET.
- Patches that require routines to be moved to another UCI.
- Patches that require files to be manually populated.
- Patches that need mail groups to be set up, domains to be created, site parameters to be entered, and options queued up in Taskman, could not be auto installed or remotely installed.

These and similar issues would need to be resolved regardless of individual or combinations of proposals as all were not considered in the original proposals.

This section shall be completed after the draft SRS has been reviewed by the development team and user group; constraints shall be identified and documented prior to design completion.

2.5. Apportioning of Requirements

No requirements have been deferred.

3. Specific Requirements

3.1. Database Repository

The VistA Auto Patch Utility maintains a record of patch installs through the INSTALL file (#9.7) and the VISTA AUTOPATCH INSTALL MODIFIERS file (#619070).

3.2. System Requirements

Infrastructure & Security Service (ISS) is responsible for modifying the code for the national release of Enhanced proposal B.

Enhanced proposal B consists of routines all of which begin with the letters AND. Files have internal numbers of 619066-619072 assigned by the database administrator.

The following VistA changes are required to bring Enhanced proposal B up to the standards required for a VistA Class I software release:

- Modify four Files into a supported KIDS name and number space:
 - VISTA AUTOPATCH INSTALL MODIFIERS (619070)
 - VISTA AUTOPATCH PACKAGE MODIFIERS (619072)
 - VISTA AUTOPATCH SITE PARAMETERS (619066)
 - VISTA AUTOPATCH FTP SITES (619068)
- Modify a field in the PACKAGE FILE to move it into the supported namespace.
- Modify a field in the INSTALL FILE to move it into the supported namespace.
- Change routines from the AND namespace to the KIDS namespace.
- Modify options that need to be moved into the KIDS namespace.
- Modify the necessary Edit Templates.

3.3. Performance Requirements

There are no special performance requirements.

3.4. Design Constraints

None are foreseen at this time; however, they may arise during the design phase and will be documented then.

3.5. Security

There are no specific security features or requirements introduced by this enhancement.

3.6. Portability

The solution must be compatible with VMS/DSM, NT/Cache, and VMS/Cache configurations currently being utilized in the VistA environment.

3.7. Other Requirements

None.

4. Function Point Estimation

A table will be provided with a FP count.