Software Technical Specification

**Messenger**

**Client – Server Application**

**Simple Message Protocol (SMP)**

**Version 1**



# Table of Contents

[Table of Contents 2](#_Toc132795706)

[Revision History 5](#_Toc132795707)

[1 Introduction 6](#_Toc132795708)

[2 Technical Specifications – SMP Server 6](#_Toc132795709)

[2.1 Technical Specification 1: *Application Start Feature* 6](#_Toc132795710)

[2.1.1 Associated Requirement 6](#_Toc132795711)

[2.1.2 Specification Details 6](#_Toc132795712)

[2.1.3 Security Details 6](#_Toc132795713)

[2.1 Technical Specification 2: *Show Messages Feature* 6](#_Toc132795714)

[2.1.1 Associated Requirement 6](#_Toc132795715)

[2.1.2 Specification Details 7](#_Toc132795716)

[2.1.3 Security Details 7](#_Toc132795717)

[3 Technical Specifications – SMP Client Producer 7](#_Toc132795718)

[3.1 Technical Specification 1: *Send Message Feature* 7](#_Toc132795719)

[3.1.1 Associated Requirement 7](#_Toc132795720)

[3.1.2 Specification Details 7](#_Toc132795721)

[3.1.3 Security Details 7](#_Toc132795722)

[3.1 Technical Specification 2: *Name of Technical Specification* 7](#_Toc132795723)

[3.1.1 Associated Requirement 7](#_Toc132795724)

[3.1.2 Specification Details 7](#_Toc132795725)

[3.1.3 Security Details 7](#_Toc132795726)

[4 Technical Specifications – SMP Client Consumer 7](#_Toc132795727)

[4.1 Technical Specification 1: *Get Message Feature* 8](#_Toc132795728)

[4.1.1 Associated Requirement 8](#_Toc132795729)

[4.1.2 Specification Details 8](#_Toc132795730)

[4.1.3 Security Details 8](#_Toc132795731)

[4.2 Technical Specification 2: *Name of Technical Specification* 8](#_Toc132795732)

[4.2.1 Associated Requirement 8](#_Toc132795733)

[4.2.2 Specification Details 8](#_Toc132795734)

[4.2.3 Security Details 8](#_Toc132795735)

[5 Technical Specifications – SMP Library 8](#_Toc132795736)

[5.1 Technical Specification 1: *SMP Protocol PUT Packet Format* 8](#_Toc132795737)

[5.1.1 Associated Requirement 8](#_Toc132795738)

[5.1.2 Specification Details 8](#_Toc132795739)

[5.1.3 Security Details 9](#_Toc132795740)

[5.2 Technical Specification 1: *SMP Protocol GET Packet Format* 9](#_Toc132795741)

[5.2.1 Associated Requirement 9](#_Toc132795742)

[5.2.2 Specification Details 9](#_Toc132795743)

[5.2.3 Security Details 9](#_Toc132795744)

[5.3 Technical Specification 2: *Name of Technical Specification* 9](#_Toc132795745)

[5.3.1 Associated Requirement 9](#_Toc132795746)

[5.3.2 Specification Details 9](#_Toc132795747)

[5.3.3 Security Details 9](#_Toc132795748)

[5.1 Technical Specification 2: *Name of Technical Specification* 9](#_Toc132795749)

[5.1.1 Associated Requirement 10](#_Toc132795750)

[5.1.2 Specification Details 10](#_Toc132795751)

[5.1.3 Security Details 10](#_Toc132795752)

[6 Technical Specifications – SMP Server Data Store 10](#_Toc132795753)

[6.1 Technical Specification 1: *Data Store Type* 10](#_Toc132795754)

[6.1.1 Associated Requirement 10](#_Toc132795755)

[6.1.2 Specification Details 10](#_Toc132795756)

[6.1.3 Security Details 10](#_Toc132795757)

[6.2 Technical Specification 2: *Name of Technical Specification* 10](#_Toc132795758)

[6.2.1 Associated Requirement 10](#_Toc132795759)

[6.2.2 Specification Details 10](#_Toc132795760)

[6.2.3 Security Details 10](#_Toc132795761)

[7 User Interfaces – GUI Version 10](#_Toc132795762)

[7.1 SMP Server Interface 11](#_Toc132795763)

[7.1.1 Start Server Button 11](#_Toc132795764)

[7.1.2 Show Messages Button 11](#_Toc132795765)

[7.2 Message Producer Client Interface 12](#_Toc132795766)

[7.2.1 Send Message Button 12](#_Toc132795767)

[7.3 Message Consumer Client Interface 12](#_Toc132795768)

[7.3.1 Get Message Button 12](#_Toc132795769)

[8 User Interfaces – CLUI Version 13](#_Toc132795770)

[8.1 SMP Message Producer Client Interface 13](#_Toc132795771)

[8.2 SMP Message Server Interface 13](#_Toc132795772)

[8.3 SMP Message Consumer Client Interface 14](#_Toc132795773)

[8.4 SMP Server Admin Interface 14](#_Toc132795774)

[9 Technical Considerations 14](#_Toc132795775)

[9.1 Dependencies 14](#_Toc132795776)

[9.2 Assumptions 14](#_Toc132795777)

[9.3 General Constraints 14](#_Toc132795778)

[9.4 Guidelines 15](#_Toc132795779)

[9.5 Development Methods 15](#_Toc132795780)

[10 Software Configuration 15](#_Toc132795781)

[11 Software Build Instructions 15](#_Toc132795782)

[11.1 Build Tools 15](#_Toc132795783)

[11.2 Compiler Command Line 15](#_Toc132795784)

[11.3 Linker Command Line 15](#_Toc132795785)

[12 Program Execution 15](#_Toc132795786)

[12.1 Command-Line Arguments 15](#_Toc132795787)

[13 Package Installer 15](#_Toc132795788)

[14 Access and Security 15](#_Toc132795789)

[14.1 Server Access 16](#_Toc132795790)

[15 Appendix 16](#_Toc132795791)

[15.1 Acronyms 16](#_Toc132795792)

[15.2 Engineering Terms 16](#_Toc132795793)

[15.2.1 Embedded Processing 16](#_Toc132795794)

[16 Notes 16](#_Toc132795795)

[17 Schedule 16](#_Toc132795796)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Authors | Comments |
| 1 |  |  | 1. Initial release. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

While the Software Requirements Specification document captures the requirements of the software system, it’s not intended to describe the full technical details of the software system. The role of the Software Technical Specification document is to define in great detail the technical specifications of each of the software requirements. The Software Technical Specification definition process is an iterative process. During each iteration, the team develops a better understanding of the system and the system’s operating environment.

This Software Technical Specification document contains SMP Messenger client and server application technical specifications.

NOTE: This document contains software application technical specifications specific to customers contracting with Never Crash Software Services and vendors developing a software or hardware product or software subsystem for Never Crash Software Services. This document is NOT intended to serve as a non-disclosure agreement, letter of intent, request for proposal or quotation, joint marketing agreement or purchase agreement. These or other documents that may be required to define any business relationship or agreements between any vendor and Never Crash Software Services shall be executed separately as required.

All licenses for this software application are owned by the principal contracting agency, Reality Software.

# Technical Specifications – SMP Server

## Technical Specification 1: *Application Start Feature*

## Associated Requirement

SMP Software Requirements Specification

SMP Server Requirements: 2.1.1 Requirement 1: Application Start Feature

## Specification Details

Implement a GUI Command Button Event Handler that starts the SMP server. The Event Handler should:

* Create a thread of execution to process SMP Client Producer SMP PUT messages and SMP Client Consumer SMP GET messages sent over a TCP/IP network.
* Implement the communications using the Socket API.

## Security Details

TBD

## Technical Specification 2: *Show Messages Feature*

## Associated Requirement

SMP Software Requirements Specification

SMP Server Requirements: 2.1.2 Requirement 2: Show Messages Feature

## Specification Details

Reads SMP message records from the SMP message file and displays the message date, message priority, and message content for each of the messages in the file and outputs the messages. The UI for this feature can be a command-line option in the case of a command-line UI (CLUI), or a clickable button in the case of a graphical user interface (GUI).

## Security Details

Local (inside the network’s firewall) unauthorized accessed to the SMP server that’s hosting the SMP server application allows the unauthorized actor to read SMP messages.

# Technical Specifications – SMP Client Producer

## Technical Specification 1: *Send Message Feature*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Producer Requirements: 2.2.1 Requirement 1: Send Message Feature

## Specification Details

Implement a GUI Command Button Event Handler that sends an SMP PUT message to a SMP server. Implement the communications using the Socket API.

## Security Details

TBD.

## Technical Specification 2: *Put Message Encryption*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Producer Requirements: 2.2.2 Requirement 2: Put Message Encryption

## Specification Details

Using RSA encryption, encrypts the SMP packet’s message field before sending the SMP packet to the SMP Server. RSA encryption asymmetry works with two different keys: a public key and private key. The public key encrypts the data. The private key decrypts the data. Each Client Producer receives the same public key. Each Client Consumer receives the same private key.

## Security Details

Distribution of public and private keys is a security concern. The method used to distribute the public and private keys are to be determined.

## Technical Specification 3: *Name of Technical Specification*

## Associated Requirement

TBD.

## Specification Details

TBD.

## Security Details

TBD.

# Technical Specifications – SMP Client Consumer

## Technical Specification 1: *Get Message Feature*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Consumer Requirements: 2.3.1 Requirement 1: Get Message Feature

## Specification Details

Implement a GUI Command Button Event Handler that sends an SMP GET message to a SMP server. Implement the communications using the Socket API.

## Security Details

TBD.

## Technical Specification 2: *Get Message Decryption*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Consumer Requirements: 2.3.2 Requirement 2: Get Message Decryption

## Specification Details

Using RSA encryption, decrypts the SMP packet’s message field sent from the SMP Server. The RSA encryption algorithm is an asymmetric cryptographic algorithm. RSA encryption asymmetry works with two different keys: a public key and private key. The public key encrypts the data. The private key decrypts the data. Each Client Producer receives the same public key. Each Client Consumer receives the same private key.

## Security Details

Distribution of public and private keys is a security concern. The method used to distribute the public and private keys are to be determined.

## Technical Specification 3: *Name of Technical Specification*

## Associated Requirement

TBD.

## Specification Details

TBD.

## Security Details

TBD.

# Technical Specifications – SMP Library

## Technical Specification 1: *SMP Protocol PUT Packet Format*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Producer Requirements: 2.2.1 Requirement 1: Send Message Feature

## Specification Details

An SMP server processes SMP PUT messages.

The format of an SMP PUT packet is defined as follows:

**Packet Field:** SMP Version

**Packet Field Datatype:** String

**Packet Field Value:** Version\_1\_0

**Packet Field:** Packet Type (PDU (Protocol Data Unit))

**Packet Field Datatype:** String

**Packet Field Value:** PutMessage

**Packet Field:** Message Priority

**Packet Field Datatype:** String

**Packet Field Values:** LOWEST\_PRIORITY, MEDIUM\_PRIORITY, HIGH\_PRIORITY

**Packet Field:** Message Date and Time

**Packet Field Datatype:** String

**Packet Field:** Message Content

**Packet Field Datatype:** String

## Security Details

If the packet is unencrypted, the message content can be read by a network packet analyzer (packet sniffer). It’s recommended that the message content is encrypted. The encryption method is TBD.

## Technical Specification 1: *SMP Protocol GET Packet Format*

## Associated Requirement

SMP Software Requirements Specification

SMP Client Consumer Requirements: 2.3.1 Requirement 1: Get Message Feature

## Specification Details

An SMP server processes SMP GET messages.

The format of an SMP GET packet is defined as follows:

**Packet Field:** SMP Version

**Packet Field Datatype:** String

**Packet Field Value:** Version\_1\_0

**Packet Field:** Packet Type (PDU (Protocol Data Unit))

**Packet Field Datatype:** String

**Packet Field Value:** GetMessage

**Packet Field:** Message Priority

**Packet Field Datatype:** String

**Packet Field Values:** LOWEST\_PRIORITY, MEDIUM\_PRIORITY, HIGH\_PRIORITY

## Security Details

If the packet is unencrypted, the message content can be read by a network packet analyzer (packet sniffer). It’s recommended that the message content is encrypted. The encryption method is TBD.

## Technical Specification 2: *Name of Technical Specification*

## Associated Requirement

TBD.

## Specification Details

TBD.

## Security Details

TBD.

## Technical Specification 2: *Name of Technical Specification*

## Associated Requirement

TBD.

## Specification Details

TBD.

## Security Details

TBD.

# Technical Specifications – SMP Server Data Store

## Technical Specification 1: *Data Store Type*

## Associated Requirement

SMP Software Requirements Specification

SMP Server Requirements: 2.1.1 Requirement 1: Application Start Feature

## Specification Details

The SMP server application stores the SMP messages in an ASCII text file. The datastore is designed to implement the behavior of a message queue. A queue is a first-in, first-out data structure. Messages are sent to the SMP server and inserted into the queue in the order they’re received. The messages are retrieved from the queue in a first-in, first-out fashion. In other words, the first message inserted in the queue is the first message retrieved from the queue.

## Security Details

TBD.

## Technical Specification 2: *Name of Technical Specification*

## Associated Requirement

TBD.

## Specification Details

TBD.

## Security Details

TBD.

# User Interfaces – GUI Version

## SMP Server Interface

## Start Server Button

The Start Server button starts the SMP Server. An SMP message consists of a priority level, the message body, and the current date and time. The server adds the message to a text file. There is one text file for each of the priority levels. The text files act as queues. Which text file the server writes to depends on the priority selected.

## Show Messages Button

Depending on which option button is selected, the Show Messages button should read in records from the appropriate file and display the message date, message priority, and the message content for each message in the file and list the messages in the Messages window.

Graphical user interface

Description automatically generated

## Message Producer Client Interface

## Send Message Button

The SMP Message Producer client program is designed to send an SMP PUT request to the server. When the Send Message button is clicked, an SMP message is sent to the server. The message sent to the server consists of the message date, message priority and the message content. The server adds the record to a file associated with the message priority.

Graphical user interface, application

Description automatically generated

## Message Consumer Client Interface

## Get Message Button

The SMP Message Consumer client program is designed to send an SMP GET request to the server to retrieve the next message. When the Get Message button is clicked, an SMP message is retrieved from the server. The server sends back a message. Which message is sent back from the server depends on the priority selected. The message sent back to the client consists of the message date, message priority and the message content. The server deletes the record from the file after the client acknowledges that it has received the message.

Graphical user interface, text, application

Description automatically generated

# User Interfaces – CLUI Version

## SMP Message Producer Client Interface

Using TCP/IP and the Sockets API (Application Programming Interface), the SMP Message Producer client program is designed to send messages to a server. A message consists of the current date and time, a priority level, and the message body.

A screenshot of a computer

Description automatically generated with medium confidence

## SMP Message Server Interface

The SMP Message Server program is designed to listen for SMP requests sent to it by SMP Producer and SMP Consumer clients. An SMP message consists of the current date and time, a priority level, and the message body. The server adds the SMP message to a text file. There is one text file for each of the priority levels. The text files act as queues. Which text file the server writes to depends on the priority selected.

Text

Description automatically generated

## SMP Message Consumer Client Interface

The SMP Message Consumer client program is designed to send an SMP request to the server to retrieve the next message. The server sends back a message. Which message is sent back from the server depends on the priority selected. The message sent back to the client consists of the message body and message date. The server deletes the record from the file after the client acknowledges that it has received the message.

Text

Description automatically generated

## SMP Server Admin Interface

The SMP Server Admin reads in records from the appropriate file and displays the message date, the message priority, and the message.

TBD

# Technical Considerations

Describes the issues that need to be addressed before implementing a software solution.

## Dependencies

Describe any dependencies.

## Assumptions

Describe any assumptions that may be wrong.

## General Constraints

Describe any constraints that could have an impact on the implementation of the software.

## Guidelines

Describe any guidelines for the implementation of the software.

## Development Methods

Describe the software development method that will be used.

# Software Configuration

# Software Build Instructions

## Build Tools

## Compiler Command Line

## Linker Command Line

# Program Execution

## Command-Line Arguments

Project→ Properties → Run/Debug Settings → (Select Project) → Edit → Arguments

# Package Installer

TBD

# Access and Security

## Server Access

# Appendix

## Acronyms

* + GCC – GNU Compiler Collection

## Engineering Terms

### Embedded Processing

Some terms related to embedded processing

# Notes

# Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Hours | Team Member | Comments |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |