Software Engineering- CSC 4350

Spring 2017

An encryption and decryption system for message communication

ADEPT

Amani Konduru

Benjamin Garber (**D**aniel)

Edward Bull

Paul David Utesch

Team

3/24/2017

Table of Contents

I. Requirements Traceability Matrix (RTM)

Entry #	System Specification text	Type	Build	Use Case Name	Category
1	The Adept Mail Server shall store user e-mails in a database.	SW	1	Send Email, Edit Emails	Server
2	The Adept Mail Server shall move user e-mails between user-designated mailboxes upon an authenticated request from that user.	SW	2	Edit Folders	Server
3	The Adept Mail Server shall delete user-designated e-mails from its database upon an authenticated request from that user.	SW	2	Delete Emails	Server
4	The Adept Mail Server shall serve user data when authenticated requests are received from the Adept Mail Client via a minimally compliant IMAP protocol.	SW	1	Serve Updates	Server
5	The Adept Mail Server shall send user emails from other Adept Mail Servers upon an authenticated request from that user via a minimally compliant SMTP protocol.	SW	2	Send Email, Send External Email	Server
6	The Adept Mail Server shall receive user emails from other Adept Mail Servers via a minimally compliant SMTP protocol.	SW	2	Receive Email	Server
7	The Adept Mail Server shall encrypt all incoming and outgoing connections using the TLS 1.2 standard.	SW	1	Receive Email, Send External Email, Serve Updates, Edit Emails, Edit Folders, Authenticate	Server
8	The Adept Mail Server shall support multiple concurrent connections.		1	Receive Email, Serve Updates, Edit Emails, Edit folders	Server
9	The Adept Mail Client shall request user email data from the Adept Mail Server via a minimally compliant IMAP protocol.	SW	1	Request Update	Client
10	The Adept Mail Client shall store user email data locally in a local database.	SW	1	Request Update	Client

11	The Adept Mail Client shall send user emails to the Adept Mail Server via a minimally compliant SMTP protocol.	SW	1	Send Email	Client
12	The Adept Mail Client shall provide a graphical user interface to allow users to generate requests and view their emails.	SW	2	Authenticate, view Email, Manage Emails, Manage Folders	Client
13	The Adept Mail Client shall require local authentication from any user before executing local requests.	SW	1	Authenticate	Client
14	The Adept Mail Client shall provide remote authentication to the Adept Mail Server prior to executing any requests.	SW	1	Manage Emails, Manage Folders	Client
15	The Adept Mail Client shall locally encrypt and decrypt the subject and body of every email it sends and receives, respectively, using symmetric-key block encryption based on a user provided password.	SW	2	Request Update	Client

II. Test Cases

Test-case Identifier	ServerConnectivity
Feature	RTM: 4, 5, 7, 8
Feature Pass/Fail Criteria	The test passes if each request receives a properly formed protocol compliant response.
Means of Control	IMAP, SMTP direct connections
Data	A series of protocol specific commands (both well-formed and mal-formed) and their expected protocol-specific responses.
Test Procedure	Using an automated script, the tester will create a series of concurrent connections to the server on both its IMAP and SMTP interfaces. Once each connection is established, a series of commands will be sent and the responses logged in a flat file. The script will then compare each command with its expected response. The tester will verify.
Special Requirements	An authenticated account on the test server.

Test-case Identifier	SendFunctionality
Feature	RTM: 1, 5, 7, 10, 11, 12, 13, 14, 15
Feature Pass/Fail Criteria	The test passes if emails and their associated mailboxes are sent from the client, received by the server, and stored in the database.
Means of Control	Client UI
Data	Placeholder content and names will be used for both the email and mailboxes.
Test Procedure	Using an authenticated account, the tester will use the client UI to send an e-mail to another authenticated account on the same server.
Special Requirements	Two authenticated accounts on the test server.

Test-case Identifier	ModifyFunctionality
Feature	RTM: 1, 2, 3, 7, 10, 12, 13, 14, 15
Feature Pass/Fail Criteria	The test passes if the e-mail is moved from one mailbox to another mailbox, and then deleted.
Means of Control	Client UI
Data	Placeholder content and names will be used for both the email and mailboxes.
Test Procedure	Using an authenticated account, the tester will use the client UI to move an email from one mailbox to another mailbox. The tester will confirm the move, then delete the e-mail. The tester will confirm the deletion.
Special Requirements	An authenticated account on the test server.

Test-case Identifier	ExternalSendFunctionality
Feature	RTM: 1, 5, 6, 7, 10, 11, 12, 13, 14, 15
Feature Pass/Fail Criteria	The test passes if the e-mail is sent and stored from the first server to the second server.
Means of Control	Client UI
Data	Placeholder content and names will be used for both the email and mailboxes.
Test Procedure	Using two test servers, the tester will use the client UI to send an e-mail to the first test server. The e-mail address will not be present on the first test server, so it will be forwarded to the second. The tester will then log in with the target account on the second server and confirm the e-mail was received.
Special Requirements	An authenticated account on test server 1. An authenticated account on test server 2.

Test-case Identifier	UpdateFunctionality
Feature	RTM: 1, 7, 9, 10, 12, 13, 14, 15
Feature Pass/Fail Criteria	The test passes if the server receives the update request from the client, sends back e-mail data, and the client updates its local database with that data.
Means of Control	Client UI
Data	Placeholder content and names will be used for both the email and mailboxes.
Test Procedure	Using an authenticated account, the tester will use the client UI to request a mailbox update from the server. Once the update is complete, the tester will verify that all data has transferred correctly.
Special Requirements	An authenticated account on the test server.

III. Construction Cost Model (COCOMO)

Please enter the size in **SLOC**: 5000

Software Development Mode

The software project needs to be described with one of three development modes. These modes range from the familiar to the ambitious, tightly constrained development projects.

- Choose **Organic** for relatively small teams developing software in a highly familiar, in-house environment.
- Choose **Semi-Detached** when the team members have some experience related to some aspects of the system under development but not others and the team is composed of experienced and inexperience people.
- Choose **Embedded** if the project must operate within a strongly coupled complex of hardware, software, regulations, and operational procedures, such as real-time systems.

Choose one: ◉ **Organic** ○ **Semi-Detached** ○ **Embedded**

Software Development Cost Drivers

There are four categories of cost drivers that are found significant performance predictors for a software development project. Each category has several cost drivers. Each driver has a possible rating from Very Low (VL) to Extra High (XH). The ratings are used in the model to adjust the baseline development effort estimation up or down.

For **HELP** on each cost driver, select the driver name.

Product Attributes

○ VL ○ L ○ N ◉ H ○ VH ○ XH : Required Reliability
○ VL ○ L ○ N ○ H ◉ VH ○ XH : Database Size
○ VL ○ L ○ N ◉ H ○ VH ○ XH : Product Complexity

Computer Attributes

○ VL ○ L ○ N ○ H ◉ VH ○ XH : Execution Time Constraint
○ VL ○ L ○ N ○ H ◉ VH ○ XH : Main Storage Constraint
○ VL ○ L ○ N ○ H ◉ VH ○ XH : Virtual Machine Volatility
○ VL ○ L ◉ N ○ H ○ VH ○ XH : Computer Turnaround Time

Personnel Attributes

○ VL ○ L ◉ N ○ H ○ VH ○ XH : Analyst Capability
○ VL ○ L ○ N ◉ H ○ VH ○ XH : Applications Experience
○ VL ○ L ◉ N ○ H ○ VH ○ XH : Programmer Capability
○ VL ○ L ○ N ◉ H ○ VH ○ XH : Virtual Machine Experience
○ VL ○ L ◉ N ○ H ○ VH ○ XH : Programming Language Experience

Project Attributes

○ VL ○ L ◉ N ○ H ○ VH ○ XH : Modern Programming Practices
○ VL ◉ L ○ N ○ H ○ VH ○ XH : Use of Software Tools
○ VL ◉ L ○ N ○ H ○ VH ○ XH : Required Development Schedule

COCOMO II - Constructive Cost Model

Model(s): COCOMO
Monte Carlo Risk: Off
Auto Calculate: Off

Software Size Sizing Method: Source Lines of Code

	SLOC	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	5000						
Reused	500	0	0	50	5		
Modified	500	20	20	50	5	50	0

Software Scale Drivers

Precedentedness	Nominal	Architecture / Risk Resolution	Low	Process Maturity	High
Development Flexibility	Very High	Team Cohesion	High		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	Low	Analyst Capability	High	Time Constraint	Nominal
Data Base Size	Low	Programmer Capability	High	Storage Constraint	Nominal
Product Complexity	High	Personnel Continuity	Low	Platform Volatility	Low
Developed for Reusability	Nominal	Application Experience	Nominal	**Project**	
Documentation Match to Lifecycle Needs	High	Platform Experience	High	Use of Software Tools	High
		Language and Toolset Experience	High	Multisite Development	Nominal
				Required Development Schedule	High

Maintenance: Off

Software Labor Rates
Cost per Person-Month (Dollars): 0
Calculate

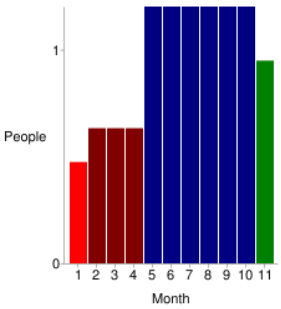
Results

Software Development (Elaboration and Construction)

Effort = 10.1 Person-months
Schedule = 10.2 Months
Cost = $0

Total Equivalent Size = 5270 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	0.6	1.3	0.5	$0
Elaboration	2.4	3.8	0.6	$0
Construction	7.7	6.4	1.2	$0
Transition	1.2	1.3	0.9	$0

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.3	0.8	0.2
Environment/CM	0.1	0.2	0.4	0.1
Requirements	0.2	0.4	0.6	0.0
Design	0.1	0.9	1.2	0.0
Implementation	0.0	0.3	2.6	0.2
Assessment	0.0	0.2	1.8	0.3
Deployment	0.0	0.1	0.2	0.4

Your output file is http://csse.usc.edu/tools/data/COCOMO_March_25_2017_06_50_29_414519.txt

IV. Updated WSD

Name	Role
Amani Konduru	Project manager and tester, Document handler, Java Programmer, Database Structure (PostgresSQL schemas)
Benjamin Garber (Daniel)	Developer and programmer, Servers, Java Programmer
Edward Bull	Programmer and tester, Java Programmer, GUI tester
Paul David Utesch	Developer and programmer, Java Programmer

V. Gantt Chart

Create V1 Stories	**S-01001**	**Ed Bull,akonduru**	**Done**	**1**
Document 2 - Requirements Elicitation	S-01017	akonduru	Done	1
Learn how to use VersionOne	S-01022	Ed Bull,bgarber,akonduru	Done	1
Set up group Github	S-01018	Ed Bull	Done	1
Plan Server Database And Write CreateTables SQL Script	S-01019	Putesch,akonduru	Future	1
Set up PostgresSQL test server	S-01009	akonduru	In Progress	1
Implement Test Database	S-01020	akonduru	Future	1
Server ServerController	S-01002	Ed Bull	In Progress	1
Server SmtpServer	S-01003	Ed Bull	In Progress	1
Server ImapServer	S-01004	Ed Bull	In Progress	1
Server SmtpConnection	S-01005	Ed Bull	In Progress	1
Server ImapConnection	S-01006	Ed Bull	In Progress	1
Server CmdProcessor	S-01007	bgarber	In Progress	1
Server QueryGenerator	S-01008		Future	1
Server SmtpClient	S-01021		Future	1
Document 2 Title Page	S-01037	akonduru	Done	1
Document 2 Problem Statements	S-01038	Ed Bull	Done	1
Document 2 RTM	S-01039	Ed Bull	Done	1
Document 2 WSD	S-01040	akonduru	Done	1
Document 2 Gantt	S-01041	akonduru	Done	1
Document 2 Dictionary	S-01042	Ed Bull,bgarber	Done	1
Set up PostgresSQL test server	S-01043	akonduru	In Progress	2
Implement Test Database	S-01044	akonduru	Future	2
Server ServerController	S-01045	Ed Bull	In Progress	2
Server SmtpServer	S-01046	Ed Bull	In Progress	2
Server ImapServer	S-01047	Ed Bull	In Progress	2
Server SmtpConnection	S-01048	Ed Bull	In Progress	2

Server ImapConnection	S-01049	Ed Bull	In Progress	2
Server CmdProcessor	S-01050	bgarber	In Progress	2
Server QueryGenerator	S-01051		Future	2
Server SmtpClient	S-01052		Future	2
Collate Document 3	S-01024	akonduru	In Progress	2
Document 3 Title Page	S-01025	akonduru	Accepted	2
Document 3 RTM (5 columns)	S-01027	Ed Bull	Accepted	2
Document 3 Use Cases and Int. Diagrams	S-01028	Ed Bull	Done	2
Document 3 Function Point Analysis	S-01029	bgarber	Done	2
Document 3 Database To Be Used	S-01030	akonduru	Future	2
Document 3 Updated WSD	S-01031	akonduru	Done	2
Document 3 Updated Gantt Chart	S-01032	Ed Bull	In Progress	2
Document 3 Dictionary	S-01033	Ed Bull	Accepted	2
Document 3 Use Cases Rationale	S-01034	Ed Bull	Done	2
Document 3 horizontal prototype	S-01035	Putesch	In Progress	2
Document 4 #2	S-01063	akonduru	Accepted	2
Document 4 #4	S-01064	Putesch	Accepted	2
Document 4 #5	S-01065	Ed Bull	Accepted	2
Document 4 #9	S-01066	Ed Bull,Putesch,akonduru	Accepted	2
Document 4	S-01067	Ed Bull,akonduru	Accepted	2
Document 6 #3	S-01068		In Progress	2
Document 6 #4	S-01069		In Progress	2
Document 6 #5	S-01070	bgarber	In Progress	2
Document 6 #9	S-01071		In Progress	2

VI. Glossary

1. **Encryption:** the process of converting data into a code, to prevent unauthorized access. Encryption is the process of transforming data into an unreadable, encrypted form. The transformation is done using one of several cryptographic algorithms that leverage computationally difficult mathematical problems to make reversing the transformation difficult if not effectively impossible

2. **Symmetric Encryption**: Symmetric Encryption uses a key or set of keys to both encrypt and decrypt data. If data is to be shared between two parties, they must both have the key or keys to decrypt or encrypt the data.

3. **Asymmetric Encryption:** Asymmetric Encryption, also known as Public Key Encryption, is a type of encryption where anyone in possession of a public key can encrypt a message. That message can then only be decrypted with a private key. This method is often used for identity authentication because it is computationally expensive. Once authentication is completed, communications will then often transition into symmetric encryption after generating a symmetric encryption key.

4. **End to End Encryptio**n: Only the communicating users can read the messages.

5. **SSL/TLS (Secure Sockets Layer / Transport Layer Protocol):** TLS and the now- deprecated SSL it is based on are network security protocols meant to secure client-server connections using both symmetric encryption for data transfer and asymmetric encryption for identity authentication. While there are many options that can be set in an SSL/TLS session, the foundation of the protocols lie in using encryption to authenticate the identities of the connected parties and to secure the privacy of the data transferred between them.

6. **Server:** a server program awaits and fulfills requests from client programs, which may be running in the same or different computers.

7. **Client:** requesting program or user.

8. **Socket:** Is one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to.

9. **SMTP protocol:** Simple Mail Transfer Protocol. It is an Internet standard for electronic mail (email) transmission. SMTP was first defined by RFC 821 and updated in RFC 5321.

10. **IMAP protocol:** Internet Message Access Protocol. Itis an Internet standard protocol used by e-mail clients to retrieve e-mail messages from a mail server over a TCP/IP connection. IMAP is defined by RFC 3501.

11. **TCP/IP :** IP (Internet Protocol) is the basic communication language or protocol of the ozInternet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). TCP (Transmission Control Protocol) is layered on top of IP to provide certain network control and data validation features for many internet communications.

VII. Rational for Test Cases

Test Case Rationale

We will need a test case for each requirement in the RTM. Our test cases cover the following requirements.

ServerConnectivity: 4, 5, 7, 8

SendFunctionality: 1, 5, 7, 10, 11, 12, 13, 14, 15

ModifyFunctionality: 1, 2, 3, 7, 10, 12, 13, 14, 15

ExternalSendFunctionality: 1, 5, 6, 7, 10, 11, 12, 13, 14, 15

UpdateFunctionality: 1, 7, 9, 10, 12, 13, 14, 15

The set of these requirements is the set of the RTM {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}. If these tests complete, we can say with confidence that we have tested each use case in our system.