Statement of Work
and
Technical Document

Group [4]
Laura Friedman
Sophia Gannon
Denise Goetz
Matthew Schaupp
Jack Wilson

Contents

1. Business Problem

The current development environment within the company is resulting in significant financial loss. First, there is no industry compliant system for development that provides the security to share code and communicate within the organization. All methods in-house are not secure and do not have any form of version control, resulting in duplicate and bug-ridden code. Second, due to the sensitive nature of the company's coding projects, all source code must be stored within the company's internal technology infrastructure. Using a public or cloud-based solution is not an option. Third, the company is losing potential business because it does not currently comply with the mandated Sarbanes-Oxley (SOX) standards. The estimated lost revenue for these three factors is approximately $5 million.

Our vision is to have an in-house system that enforces SOX compliance, significantly reduces losses due to non-compliance, and increases revenue. The cost to develop the new, compliant, in-house system is $500K. However, the projected increase of $5 million in revenue from additional business and the non-tangible benefits of client trust and compliance with federal law far exceed the cost. Thus, the development of this system would be a strategic investment that would add significant business value.

2. Risk Analysis

ID	Risk Description	Consequence/ Impact	Probability Score	Impact Score	Action/ Mitigation
1	Changes of project expectations from client. (scope creep)	Customer add/change project definition causing rework, project delay and project cost increases.	Medium	High	Detailed business requirements with customer sign-off. Project manager will manage customer expectations and keep project definition within original requirements. Additional features will be addressed as an enhancement project.
2	New system does not meet SOX requirements.	Non-compliance prevents business with SOX clients resulting in lost revenue. Also, financial penalties would be imposed. System would be vulnerable to theft and attack.	Low	High	Obtain current SOX compliance requirements. Incorporate these requirements as "must do" items from a hardware, software and procedural perspective. Have legal department review to ensure we are meeting SOX requirements.
3	System limitations do not accommodate the proposed system.	The new system will not work and requires upgrading to new hardware and software This will cause the project to go over budget for capital expenditures. Project delays would occur because of the unplanned system upgrades.	Medium	High	Determine accurate system requirements for running the new software. Get current inventory of technology infrastructure. Plan upgrades for minimal impact. Negotiate and coordinate adequate vendor support for new technology infrastructure and

					system rollout.
4	Failure to meet milestones or key deliverables.	Delays in project schedule. Cost of overtime to recover time loss or missed project delivery date.	Medium	Medium	Maintain communication throughout the project. Weekly status reports. Reprioritize deliverables as needed to meet deadlines.
5	Employees are resistant to new system.	A risk of a company audit if the employees do not adopt the new system/process. Slow workflow until all employees are up to speed.	High	Low	Managers promote new system and communicate expectations for use. Training needs to be provided. Establish mentors for added training support. Communicate cut-off dates for old system and migration to new.
6	Disaster occurs.	Floods, tornadoes, fires or disasters in general happen. For example, if a tornado hits the building, employees can be hurt, and/or hardware can be damaged. All work at this location will stop.	Low	High	Disaster/Recovery is out of scope for this project. Defer to your Disaster/Recovery Plan.

3. Proposed Project

- Implement private server using HTTPS, SSH, and VPN
- Expand the existing JMessenger application to provide a robust, secure, database-driven, version control system.
- Provide a storage area for projects where users connect securely to the database to access source code files
- Implement security to only allow files to be saved on the server. No local hard drive storage will be permitted.
- Design the database to maintain version control
- Allow users to save to and from source code repositories
- Generate reports for users and management to provide tracking for SOX compliance and version control
- Offsite access via the VPN

4. Discussion of Chosen SDLC

In order to have a successful development process, we will be implementing a Feature Driven Development (FDD) Methodology. This approach is a combination of Waterfall and Agile approaches to development. This approach was selected because we can rollout the application development platform in 3 phases, helping the users adapt to the new system more quickly, while also enabling the project development team an opportunity to correct problems as they arise and incorporate them into the next phase of features released.

The first three phases - planning, analysis, and design - will be complete in one iteration per phase for the entire project. The implementation phase will be repeated based on the number of features included in each iteration of the phase. It is anticipated that there will be three implementation phases to complete the development of all features.

After the completion of each implementation phase, a Joint Application Development (JAD) session will be conducted with a team of key stakeholders and developers to ensure the system is being developed in a manner that will provide Business Process Improvement (BPI) and to mitigate risks such as scope creep. In addition, at the conclusion of each iteration there will be a review of the overall analysis and design to include any additional required features that were not defined in the initial three phases of the SDLC. Each resulting version at the completion of an implementation phase will be one step closer to the desired system until completion.

Finally, to ensure user confidence in the new system we will keep each version in testing and out of a production environment until the final version is completed. Thorough testing will be conducted to ensure quality and assurance of standards and measure.

Total project duration is 9 months as follows:

Develop Overall Model phase – 1.5 months

- Major Tasks: technical, organizational, and economic feasibility studies; network infrastructure design; software and hardware specifications.
- Major Deliverable: Formal presentation recommending go/no go decision, network infrastructure design document, infrastructure installation.

Build a Features List phase – 1 month

- Major Tasks: End user requirements and method of analysis, with all features defined
- Major Deliverable: Application proposal with list of grouped features & system design document.

Plan by Feature phase – 2 months

- Major Tasks: Define feature sets and classes.
- Major Deliverable: Development plan with class owners and feature set owners

Design & Build by Feature phase – Iteration 1 – 1.5 months

- Features Included: Check-in one source code file; check-out one source code file. Version control logging. Send and receive messages to other developers and team members.
- Major Tasks: install new system/network hardware and software, build application software, build database, develop test plan, test the application, refine and resolve problems from testing.
- Major Deliverable: New version control software system.

Design & Build by Feature phase – Iteration 2 - 1.5 months

- Features Included: Check-in multiple source code files, check-out multiple source code files. Waitlist for files checked-out. Notify next on waitlist when file(s) are checked in.
- Major Tasks: build application software, build database, develop test plan, test the application, refine and resolve problems from testing.
- Major Deliverable: New version control software system.

Design & Build by Feature phase – Iteration 3 - 1.5 months

- Features Included: Set time-limit for checked-out files and lockout the ability to check-in files if the time-limit is exceeded. Generate auditing reports for management and key stakeholders and team members. HTTPS, SSH, and VPN.
- Major Tasks: build application software, build database, develop test plan, test the application, refine and resolve problems from testing. Implement HTTPS, SSH, and VPN.
- Major Deliverable: New version control software system.

5. Team Composition

Roles and Members

The team is composed of the following 11 roles and 16 members:

A. 1 - Project Manager
B. 1 - Systems Analyst
C. 1 - Business Analyst
D. 1 - Network Engineer/Architect
E. 3 - Developers
F. 1 - Database Administrator (DBA)
G. 1 - Technical Support Specialist(s)
H. 1 - Systems Tester
I. 1 - UX Tester
J. 1 - Corporate Lawyer
K. 4 - User Acceptance Testers

Most positions will be filled by existing employees because our company already develops software for external clients and is prepared to handle the majority of this project's staffing needs. New positions will be contract-to-hire, and are:

1 – Developer
1 – Technical Support Specialist
2 – User Acceptance Testers

Team Skills

Cross-functional

1. Management
2. Testing skills/Troubleshooting
3. Rapid Application Development cycles
4. UML Design
5. Business/Financial Analysis
6. User Interface Experience (UIX) Design
7. Java & SQL Programming
8. Network Design and Architecture
9. Software Design
10. Network Security

Soft Skills

1. Ability to meet project deadlines.
2. Ability to Work Under Pressure

3. Time Management
4. Self-motivation
5. Conflict Resolution
6. Adaptability
7. Creativity
8. Communication
9. Decision-Making
10. Organizational Skills
11. Detail Oriented

Overall

- Project manager knows how to take control and lead others effectively.
- Each team member has a set of abilities that ensure the success of the project.
- Team members are confident and interact comfortably with people and groups.
- The team is flexible enough to address issues as they arise with skill and diplomacy.

6. Quality & Assurance Standards and Measures

The project will be implemented utilizing ISO/IEC 9126 standards. To ensure quality software, comprehensive and standardized testing will take place at the conclusion of each Design & Build phase as part of the analysis process of the next Design & Build phase. This will ensure all requirements are met and errors are discovered and resolved throughout the development process.

Unit Testing

This stage of testing is focused on the smallest unit of the program which is generally a class. Each class will be tested using multiple test cases through a "black-box"1 and "white-box"1 approach to ensure that developer interpretation is not a factor in test results. Test cases will be comprehensive, testing not only expected inputs and outputs but also variations and deviations.

Integration & System Testing

Integration testing will begin during the second iteration of the Design & Build phase and will focus on whether the classes successfully work together without producing errors. This level of testing will be conducted using the four standard approaches: user interface, use-case, interaction, and system interface. Interactive debugging will also be used throughout the testing process.

Systems testing will also begin in the second iteration of the Design & Build phase. The focus is the overall performance of the system, and the scope of systems testing is aimed at ensuring that the system meets requirements, is secure, is usable, and is reliable.

User Acceptance Testing

The final level of testing is focused on the users of the system to ensure usability in a mock production environment through the use of mock data (alpha testing). In this controlled and closely monitored testing environment, we will verify the system meets all demands and functions according to all requirements.

Test Plan

The following test plan incorporates the three levels of testing: unit, integration & system, and acceptance. All test plans will be created under the oversight of the Project Manager.

Test Stage	**Type**	**Developed By**	**Completed By**
Unit	Black and White box tests	Network Engr/Architect DBA Developers	Network Engr/Architect DBA Developers Systems Tester
Integration & Systems	User interface, use-case, requirements, security, performance, reliability, system interface	Business Analyst Systems Analyst Developers Network Engr/Architect DBA	UX Tester Systems Tester
User Acceptance	Usability (UX)	Business Analyst Systems Analyst	User Acceptance Testers

7. Project Management Plan

It is critical to meet the government's standards defined in the Sarbanes-Oxley (SOX) Act. We must secure our proprietary and financial data by maintaining records of all transactions between developers. Our team will obtain current SOX compliance requirements and include our legal department to ensure we are meeting SOX requirements.

Another key objective is to create a program version control system and repository for software development and team communication. Security requirements of SOX mandate we install private secure servers using HTTPS, SSH and a VPN for out of network communications. An inventory of current hardware, software and networking versions will help determine necessary upgrades for the proposed solution.

To mitigate risks, the project manager will get detailed written expectations of business requirements and manage customer expectations by staying within the original set of requirements and avoiding scope creep. The project manager is responsible for communicating status, meeting project deadlines and re-prioritizing when necessary.

Implementing the new system will have many steps. Users will be trained on the new system and mentors will be added for additional support. Cut-off dates for the old system will be communicated well in advance.

The following is a proposed schedule which plans specific milestones, deliverables, and time allotments for each phase of the system. ***Some milestones and tasks within each phase are overlapping.*** The Project Manager will be involved and/or oversee all tasks.

Develop Overall Model phase – Duration 33 Days – 1.5 Calendar Months		
Milestones	**Team Members**	**Time Allotment (days)**
Organizational Feasibility Study	Project Manager Business Analyst Human Resource Manager	5
Technical Feasibility Study	Project Manager Systems Analyst	5
Economic Feasibility Study	Project Manager Business Analyst	5
Go/No-Go Presentation	Project Manager	2
Hire Team Members	Human Resource Manager Project Manager	5

Network Infrastructure Design	Project Manager Systems Analyst Network Engr/Architect	5
Hardware Specifications	Project Manager Systems Analyst Network Engr/Architect Business Analyst	4
Software Specifications	Project Manager Systems Analyst Software Developers Business Analyst	4
System/Network Infrastructure Installation	Network Engr/Architect DB Administrator Project Manager Software Developers	10

Build a Feature List phase – 40 Days – 2 Calendar Months		
Milestones	**Team Members**	**Time Allotment (days)**
End User Requirements Analysis	Project Manager Systems Analyst UX Designer Software Developers	5
End User Diagrams and Visuals	Project Manager Systems Analyst UX Designer Software Developers	10
Application Proposal	Project Manager Systems Analyst UX Designer	3
Application Software Design	Project Manager Systems Analyst Software Developers	8
Database Design	Project Manager Database Administrator Systems Analyst	5
File System Design	Project Manager Database Administrator	2

	Systems Analyst	
System Design Document	Project Manager Systems Analyst	4
Quality Assurance & Test Plan	Project Manager Systems Tester UX Tester Systems Analyst	3

Plan by Feature phase - 1 Month		
Milestones	**Team Members**	**Time Allotment (days)**
Plan for Design & Build by Feature phase – Iteration 1	Project Manager Business Analyst UX Designer	6
Plan for Design & Build by Feature phase – Iteration 2	Project Manager Systems Analyst UX Designer	6
Plan for Design & Build by Feature phase – Iteration 3	Project Manager Systems Analyst UX Designer	6
Go/No-Go Presentation	Project Manager	2

Design & Build by Feature phase – Iteration 1 – 1.5 Months		
Milestones	**Team Members**	**Time Allotment (days)**
Build Database ● Initial database design	DB Administrator Developers	5
Design & Build Application ● Check-in source code file ● Check-out source code file ● Version control logging ● Send/Receive messages	Systems Analyst UX Designer Developers	13
Develop Test Plan	Project Manager Systems Tester Systems Analyst UX Tester	2

Application Testing	Project Manager Systems Tester Systems Analyst UX Tester	3
Refine and Resolve Issues	Project Manager Systems Analyst Developers DB Administrator	5
Version Control Software System Release 1	Project Manager	2

Design & Build by Feature phase – Iteration 2 – 1.5 Months		
Milestones	**Team Members**	**Time Allotment (days)**
Build Database ● Additional tables for phase 2 features	DB Administrator Developers	5
Design & Build Application ● Check-in multiple files ● Check-out multiple files ● Waitlist for file checkout ● Notify waitlist	Systems Analyst UX Designer Developers	13
Develop Test Plan	Project Manager Systems Tester Systems Analyst UX Tester	2
Application Testing	Project Manager Systems Tester Systems Analyst UX Tester	3
Refine and Resolve Issues	Project Manager Systems Analyst Developers DB Administrator	5
Version Control Software System Release 2	Project Manager	2

Design & Build by Feature phase – Iteration 3 – 1.5 Months		
Milestones	**Team Members**	**Time Allotment (days)**

Build Database ● Additional tables for phase 3 features	DB Administrator Developers	5
Design & Build Application ● Set time-limit for checked-out files ● System lockout for exceeding time-limit ● Generate auditing reports ● Implement HTTPS and SSH ● Implement VPN	Systems Analyst UX Designer Developers	13
Develop Test Plan	Project Manager Systems Tester Systems Analyst UX Tester	2
Application Testing	Project Manager Systems Tester Systems Analyst UX Tester	3
Refine and Resolve Issues	Project Manager Systems Analyst Developers DB Administrator	5
Version Control Software System Release 3	Project Manager	2

8. Works Cited

Addison-Hewitt Associates. (2003). *A Guide to the Sarbanes-Oxley Act.* Retrieved from http://www.soxlaw.com/

Ambler, Scott W. (n.d.). *Feature Driven Development (FDD) and Agile Modeling.* Retrieved from http://agilemodeling.com/essays/fdd.htm

Dennis, Alan, Wixom, Barbara, & Tegarden, David. (2012). *Systems Analysis and Design with UML, 4th Edition.* John Wiley & Sons, Inc.

International Standards Organization. (March 2011). *ISO/IEC 25010:2011.* Retrieved from https://www.iso.org/standard/35733.html

Miles, Russ & Hamilton, Kim. (25 April 2006). *Learning UML 2.0: A Pragmatic Introduction to UML.* O'Reilly Media, Inc.

Oracle. (2010). *Sun Java System Application Server Enterprise Edition 8.2 Deployment Planning Guide.* Retrieved from https://docs.oracle.com/cd/E19900-01/819-4741/abfch/index.html

Tutorials Point. (n.d.). *Software Testing - ISO Standards.* Retrieved from https://www.tutorialspoint.com/software_testing/software_testing_iso_standards.htm

Appendix A —Technical Problem Description

Part 1: Current Operable Features of JMessenger

Use Case ID / Link:	1	**Use Case Name:**	Send Message
Description	Developer needs to send a message to another user.		
Primary Actor or Persona	Developer		
Pre-Conditions	Developer computers must have JMessenger installed. Recipients must be signed in to receive a message.		
Post-Conditions	Message is sent to desired user.		
Triggers	Developer wants to communicate with another user.		
Main/Happy Path	Developer launches JMessenger. Types in a message to desired user and clicks the Send button. The message gets sent and shows up in the recipient's JMessenger panel.		
Alternative / Exception Flows	Developer sends a message and message is not received due to recipient not being logged into JMessenger.		
Use Case Frequency	The frequency would be on average, 10 messages per Developer per day.		
Non-functional Requirements	UI is easy to understand, which means almost no training is needed.		

Use Case ID / Link:	2	**Use Case Name:**	Receive Message
Description	Developer receives a message via JMessenger		
Primary Actor or Persona	Developer		
Pre-Conditions	Developers computer must have JMessenger installed. For the recipient to receive the message they need to be signed in.		
Post-Conditions	Developer clicks the receive button. Message is received from another user.		
Triggers	Another user sends a message to the Developer.		
Main/Happy Path	Developer clicks the receive button. The message is received in the JMessenger Panel of the receiving end.		
Alternative / Exception Flows	A message bound for the recipient is not received due to recipient not being logged into JMessenger.		
Use Case Frequency	The frequency would be on average, 10 messages per Developer per day.		
Non-functional Requirements	UI is clearly easy to understand, which means almost no training is needed.		

Part 2: Proposed Enhanced Features of JMessenger

Use Case ID / Link:	3	**Use Case Name:**	Send Source Code to Repository
Description	Developer is ready to send/upload source code to repository.		
Primary Actor or Persona	Developer		
Pre-Conditions	Developer needs to have a user id for JMessenger on the server. Developer would need to have files to share with the repository.		
Post-Conditions	The developer would get a message stating the files were received and be able to see the newer versions on the repository.		
Triggers	The developer needs to share files with the secure repository.		
Main/Happy Path	Developer is ready to send/upload a program to the repository. They sign into JMessenger on the secure server, which validates they are a developer. From JMessenger on their PC they select the files to be uploaded and then click send. The program(s) get uploaded to JMessenger on the secure server which logs the transaction and any changes to the program(s). A message appears stating the files were successfully uploaded.		
Alternative / Exception Flows	1) Developer is denied access due to invalid password. 2) Developer denied access due to maintenance on JMessenger or the secure server. 3) Developer is denied access due to not being on this development team. 4) Upload files are scanned for viruses and can fail to upload due to carrying a virus.		
Use Case Frequency	Each Developer uploads on average 2 times a day. There are 102 developers corporate wide.		
Non-functional Requirements	The upload response time for the typical file transfer would be (depending on network bandwidth) $T_{response}$ = **n/r -** T_{think} where ● **n** is the number of concurrent users ● **r** is the number requests per second the server receives ● T_{think} is the average think time (in seconds) $T_{response}$ = **n/r -** T_{think} = (5000/ 1000) - 3 sec. = 5 - 3 sec. Therefore, the response time is two seconds.		

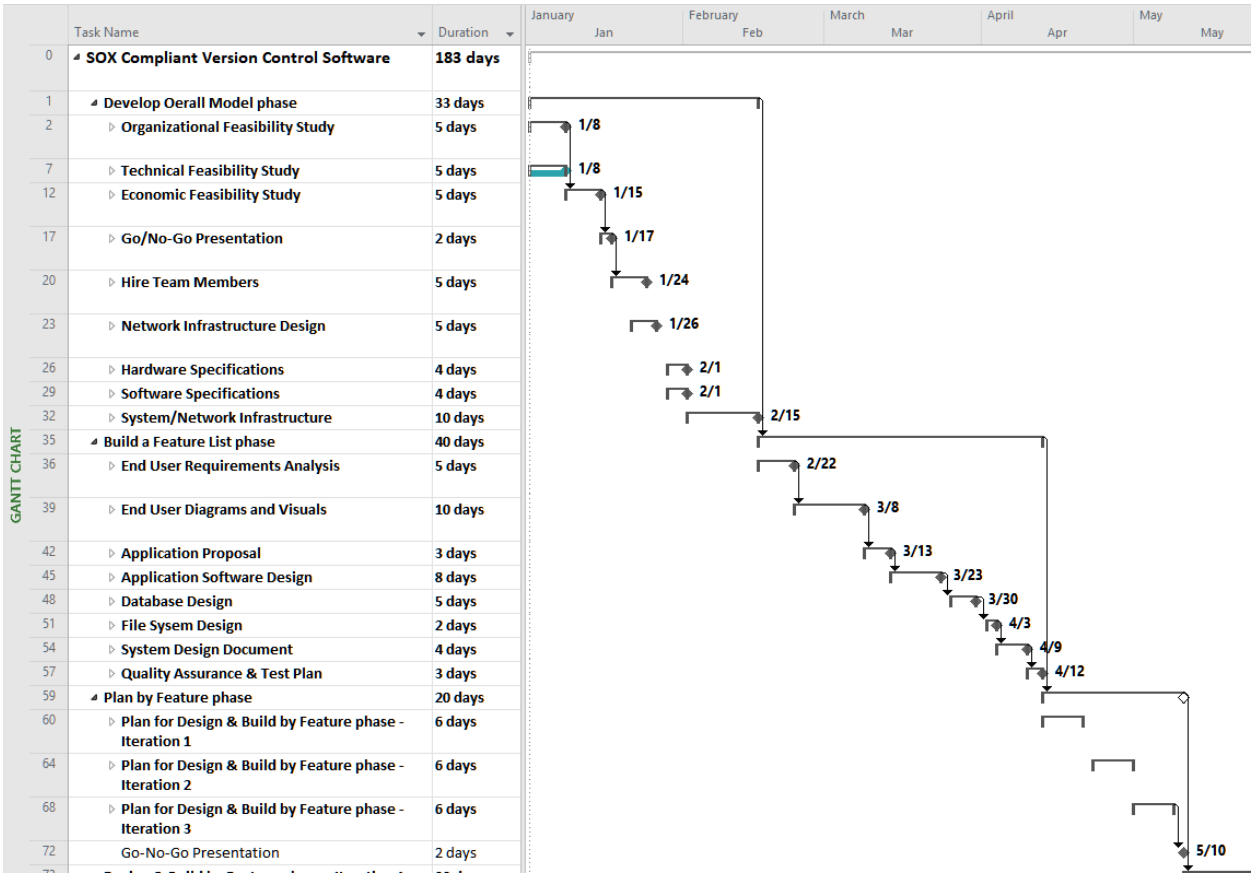
Use Case ID / Link:	4	**Use Case Name:**	Receive Source Code from Repository
Description	Developer needs to retrieve an updated version of the source code from the repository.		
Primary Actor or Persona	Developer		
Pre-Conditions	Developer needs to have a user id for JMessenger on the server and be part of the project development team.		
Post-Conditions	The developer receives a message stating the latest version of files were transferred from the repository to their work area.		
Triggers	The developer has been given an assignment and needs to retrieve the updated files from the repository.		
Main/Happy Path	Developer needs to make a change to a program. They sign into JMessenger on the secure server and it validates they are a developer. They select the project to sync files with and the server transmits the files to the individual's JMessenger environment. Developers receive the latest version of the source code. A log of checked out programs is kept with all corresponding information.		
Alternative / Exception Flows	1. Developer is denied access due to invalid password. 2. Developer denied access due to maintenance on JMessenger or the secure server. 3. Developer is denied access due to not being on the development team.		
Use Case Frequency	Developer may retrieve from the repository 1-2 times a day.		
Non-functional Requirements	The upload response time for the typical file transfer would be (depending on network bandwidth) $T_{response}$ = **n/r -** T_{think} where ● **n** is the number of concurrent users ● **r** is the number requests per second the server receives ● T_{think} is the average think time (in seconds) $T_{response}$ = **n/r -** T_{think} = (5000/ 1000) - 3 sec. = 5 - 3 sec. Therefore, the response time is two seconds.		

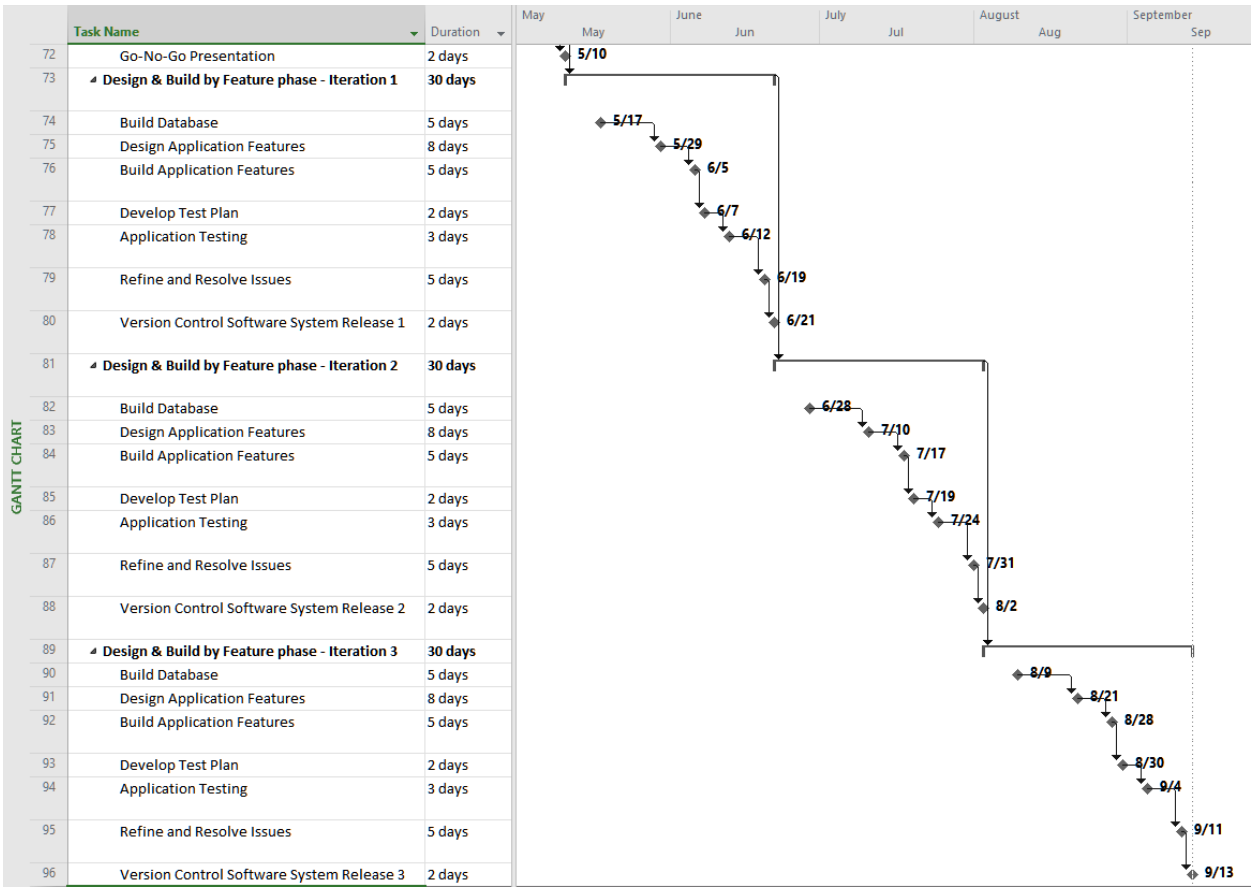
Use Case ID / Link:	5	**Use Case Name:**	Remotely Send Source Code to Repository
Description	Developer is ready to send/upload source code to repository remotely from work.		
Primary Actor or Persona	Developer		
Pre-Conditions	Developer would need to sign into a Very Secure Network (VPN) to get into the company server. Developer needs to have a user id for JMessenger on the server. Developer would need to have files to share with the repository.		
Post-Conditions	The developer would get a message stating the files were transferred from the repository to their workstation. The latest versions would be loaded.		
Triggers	The developer needs to share files with the secure repository off work location.		
Main/Happy Path	Developer is ready to send/upload a program to the repository. They sign into the (VPN) to get to the corporate server. They sign into JMessenger on the secure server, which validates they are a developer on the project team. From JMessenger, they select the files to be uploaded and then click send. The files(s) get uploaded to JMessenger on the secure server which logs the transaction and any changes to the file(s). A message appears stating the files were successfully uploaded.		
Alternative / Exception Flows	1. Developer can not access VPN. 2. Developer denied access due to invalid password. 3. Developer denied access due to maintenance on JMessenger or the secure server. 4. Developer denied access due to not being on this development team. 5. Upload files scanned for viruses and can fail to upload due to carrying a virus.		
Use Case Frequency	If on a client site away from headquarters, developers will need the ability to upload files. Developers may upload work from home if working from home.		
Non-functional Requirements	For security reasons, work can only be done on shared folders through the VPN.		

Use Case ID / Link:	6	**Use Case Name:**	Remotely Receive Source Code from Repository
Description	Developer needs to retrieve an updated version of the source code from the repository.		
Primary Actor or Persona	Developer		
Pre-Conditions	Developer would need to sign into a Virtual Private Network (VPN) to get into the company server. Developer needs to have a JMessenger user id on the server and be part of the project development team.		
Post-Conditions	The developer would get a message stating the files were transferred from the repository to their workstation. The latest versions would be loaded.		
Triggers	The developer has been given an assignment and needs to retrieve updated files from the repository.		
Main/Happy Path	While off-site, a developer needs to make a change to a program. They sign into a VPN and access the secure server. Once they have access, they sign into JMessenger on the secure server, which validates they are a developer. They select the project to sync files with and the server transmits the files to the individual's JMessenger environment. Essentially they receive the latest version of the updated files. A log is kept of checked out programs.		
Alternative / Exception Flows	1. Developer can not access the VPN. 2. Developer is denied access due to invalid password. 3. Developer denied access due to maintenance on JMessenger or the secure server. 4. Developer is denied access due to not being on this development team.		
Use Case Frequency	If offsite at a client site or working from home, developers may need the ability to upload source code.		
Non-functional Requirements	For security reasons, work can only be done on shared folders through the VPN.		

Appendix B: Project Management Detail

Gantt Chart




GANTT CHART
Task Name
Duration
May
June
July
August
September
72 Go-No-Go Presentation 2 days
73 Design & Build by Feature phase - Iteration 1 30 days
74 Build Database 5 days
75 Design Application Features 8 days
76 Build Application Features 5 days
77 Develop Test Plan 2 days
78 Application Testing 3 days
79 Refine and Resolve Issues 5 days
80 Version Control Software System Release 1 2 days
81 Design & Build by Feature phase - Iteration 2 30 days
82 Build Database 5 days
83 Design Application Features 8 days
84 Build Application Features 5 days
85 Develop Test Plan 2 days
86 Application Testing 3 days
87 Refine and Resolve Issues 5 days
88 Version Control Software System Release 2 2 days
89 Design & Build by Feature phase - Iteration 3 30 days
90 Build Database 5 days
91 Design Application Features 8 days
92 Build Application Features 5 days
93 Develop Test Plan 2 days
94 Application Testing 3 days
95 Refine and Resolve Issues 5 days
96 Version Control Software System Release 3 2 days
5/10
5/17
5/29
6/5
6/7
6/12
6/19
6/21
6/28
7/10
7/17
7/19
7/24
7/31
8/2
8/9
8/21
8/28
8/30
9/4
9/11
9/13

Task List

SOX Compliant Version Control Software

ID	Task Name	Duration	Start	Finish	Predecess
0	**SOX Compliant Version Control Software**	**183 days**	**Tue 1/2/18**	**Thu 9/13/18**	
1	**Develop Oerall Model phase**	**33 days**	**Tue 1/2/18**	**Thu 2/15/18**	
2	**Organizational Feasibility Study**	**5 days**	**Tue 1/2/18**	**Mon 1/8/18**	
3	Identify organizational structure	2 days	Tue 1/2/18	Wed 1/3/18	
4	Define job descriptions	1 day	Thu 1/4/18	Thu 1/4/18	3
5	Identify salary and benefit costs	1 day	Fri 1/5/18	Fri 1/5/18	4
6	Complete Feasibility Study document	1 day	Mon 1/8/18	Mon 1/8/18	5
7	**Technical Feasibility Study**	**5 days**	**Tue 1/2/18**	**Mon 1/8/18**	
8	Identify required technology	2 days	Tue 1/2/18	Wed 1/3/18	
9	Current technology inventory	1 day	Thu 1/4/18	Thu 1/4/18	8
10	Identify new technology cost	1 day	Fri 1/5/18	Fri 1/5/18	9
11	Complete Technical Feasibility Study document	1 day	Mon 1/8/18	Mon 1/8/18	10
12	**Economic Feasibility Study**	**5 days**	**Tue 1/9/18**	**Mon 1/15/18**	**2**
13	Identify project costs	2 days	Tue 1/9/18	Wed 1/10/18	
14	Identify projected revenue	1 day	Thu 1/11/18	Thu 1/11/18	13
15	Identify projected expense reductions	1 day	Fri 1/12/18	Fri 1/12/18	14
16	Complete Economic Feasibility Study document	1 day	Mon 1/15/18	Mon 1/15/18	15
17	**Go/No-Go Presentation**	**2 days**	**Tue 1/16/18**	**Wed 1/17/18**	**12**
18	Prepare Presentation	1.5 days	Tue 1/16/18	Wed 1/17/18	
19	Presentation Meeting	0.5 days	Wed 1/17/18	Wed 1/17/18	18
20	**Hire Team Members**	**5 days**	**Thu 1/18/18**	**Wed 1/24/18**	**17**
21	Conduct Interviews	4 days	Thu 1/18/18	Tue 1/23/18	
22	Hire Candidates	4 days	Fri 1/19/18	Wed 1/24/18	21FS-3 d
23	**Network Infrastructure Design**	**5 days**	**Mon 1/22/18**	**Fri 1/26/18**	
24	Define Network & Hardware Infrastructure	3 days	Mon 1/22/18	Wed 1/24/18	22FS-3 d
25	Technology Infrastructure Plan	2 days	Thu 1/25/18	Fri 1/26/18	24
26	**Hardware Specifications**	**4 days**	**Mon 1/29/18**	**Thu 2/1/18**	

Page 1

SOX Compliant Version Control Software

ID	Task Name	Duration	Start	Finish	Predecess
27	Define Hardware Required	2 days	Mon 1/29/18	Tue 1/30/18	25
28	Purchase Technology	2 days	Wed 1/31/18	Thu 2/1/18	27
29	**Software Specifications**	**4 days**	**Mon 1/29/18**	**Thu 2/1/18**	
30	Define Software Required	2 days	Mon 1/29/18	Tue 1/30/18	25
31	Purchase Software	2 days	Wed 1/31/18	Thu 2/1/18	30
32	**System/Network Infrastructure**	**10 days**	**Fri 2/2/18**	**Thu 2/15/18**	
33	Install New Hardware	5 days	Fri 2/2/18	Thu 2/8/18	31
34	Install New Software	5 days	Fri 2/9/18	Thu 2/15/18	33
35	**Build a Feature List phase**	**40 days**	**Fri 2/16/18**	**Thu 4/12/18**	**1**
36	**End User Requirements Analysis**	**5 days**	**Fri 2/16/18**	**Thu 2/22/18**	
37	Define User Requirements	4 days	Fri 2/16/18	Wed 2/21/18	
38	User Requirements Document	1 day	Thu 2/22/18	Thu 2/22/18	37
39	**End User Diagrams and Visuals**	**10 days**	**Fri 2/23/18**	**Thu 3/8/18**	**36**
40	Define User Diagrams & Visuals	7 days	Fri 2/23/18	Mon 3/5/18	
41	UML & User Diagrams & Visuals Document	3 days	Tue 3/6/18	Thu 3/8/18	40
42	**Application Proposal**	**3 days**	**Fri 3/9/18**	**Tue 3/13/18**	**39**
43	Applicationn Proposal Document	2 days	Fri 3/9/18	Mon 3/12/18	
44	Go/No-Go Presentation	1 day	Tue 3/13/18	Tue 3/13/18	43
45	**Application Software Design**	**8 days**	**Wed 3/14/18**	**Fri 3/23/18**	**42**
46	Design Application	6 days	Wed 3/14/18	Wed 3/21/18	
47	Software Design Document	2 days	Thu 3/22/18	Fri 3/23/18	46
48	**Database Design**	**5 days**	**Mon 3/26/18**	**Fri 3/30/18**	**45**
49	Define Database Requirements	4 days	Mon 3/26/18	Thu 3/29/18	
50	Database Schema	1 day	Fri 3/30/18	Fri 3/30/18	49
51	**File Sysem Design**	**2 days**	**Mon 4/2/18**	**Tue 4/3/18**	**48**
52	Define Log File Formats	1 day	Mon 4/2/18	Mon 4/2/18	
53	Log File Design Document	1 day	Tue 4/3/18	Tue 4/3/18	52
54	**System Design Document**	**4 days**	**Wed 4/4/18**	**Mon 4/9/18**	**51**

Page 2

SOX Compliant Version Control Software

ID	Task Name	Duration	Start	Finish	Predecess
55	Integrate User & System Design Documents	3 days	Wed 4/4/18	Fri 4/6/18	
56	Presentation	1 day	Mon 4/9/18	Mon 4/9/18	55
57	**Quality Assurance & Test Plan**	**3 days**	**Tue 4/10/18**	**Thu 4/12/18**	**54**
58	Develop Systems & Integration Test Plan	3 days	Tue 4/10/18	Thu 4/12/18	
59	**Plan by Feature phase**	**20 days**	**Fri 4/13/18**	**Thu 5/10/18**	**35**
60	**Plan for Design & Build by Feature phase - Iteration 1**	**6 days**	**Fri 4/13/18**	**Fri 4/20/18**	
61	UX Design for Features	2.5 days	Fri 4/13/18	Tue 4/17/18	
62	Systems Design for Features	2.5 days	Tue 4/17/18	Thu 4/19/18	61
63	Design Document - Iteration 1	1 day	Fri 4/20/18	Fri 4/20/18	62
64	**Plan for Design & Build by Feature phase - Iteration 2**	**6 days**	**Mon 4/23/18**	**Mon 4/30/18**	
65	UX Design for Features	2.5 days	Mon 4/23/18	Wed 4/25/18	60
66	Systems Design for Features	2.5 days	Wed 4/25/18	Fri 4/27/18	65
67	Design Document - Iteration 2	1 day	Mon 4/30/18	Mon 4/30/18	66
68	**Plan for Design & Build by Feature phase - Iteration 3**	**6 days**	**Tue 5/1/18**	**Tue 5/8/18**	
69	UX Design for Features	2.5 days	Tue 5/1/18	Thu 5/3/18	64
70	Systems Design for Features	2.5 days	Thu 5/3/18	Mon 5/7/18	69
71	Design Document - Iteration 3	1 day	Tue 5/8/18	Tue 5/8/18	70
72	Go-No-Go Presentation	2 days	Wed 5/9/18	Thu 5/10/18	68
73	**Design & Build by Feature phase - Iteration 1**	**30 days**	**Fri 5/11/18**	**Thu 6/21/18**	**59**
74	Build Database	5 days	Fri 5/11/18	Thu 5/17/18	
75	Design Application Features	8 days	Fri 5/18/18	Tue 5/29/18	74
76	Build Application Features	5 days	Wed 5/30/18	Tue 6/5/18	75
77	Develop Test Plan	2 days	Wed 6/6/18	Thu 6/7/18	76
78	Application Testing	3 days	Fri 6/8/18	Tue 6/12/18	77
79	Refine and Resolve Issues	5 days	Wed 6/13/18	Tue 6/19/18	78

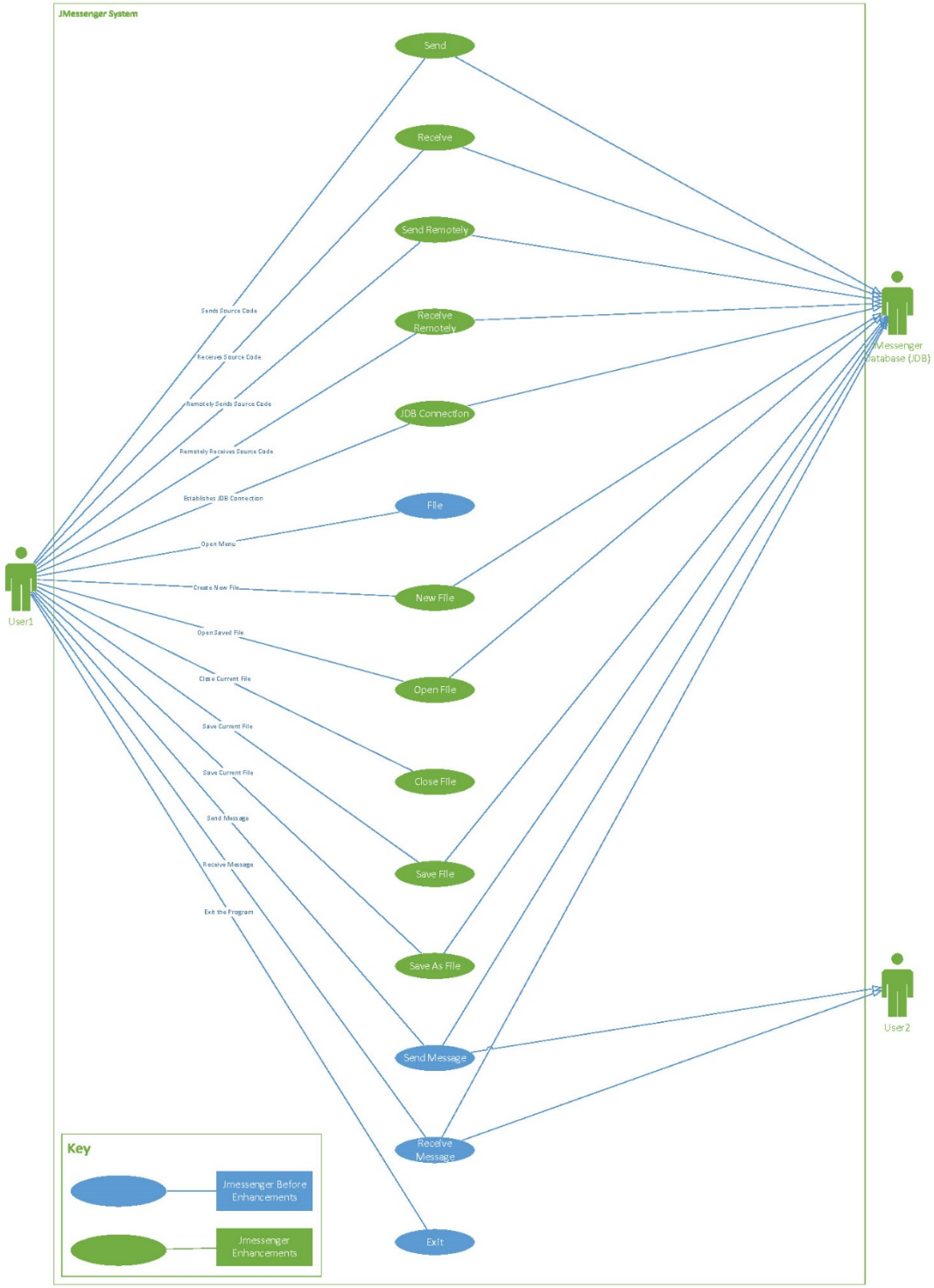
Page 3

SOX Compliant Version Control Software

ID	Task Name	Duration	Start	Finish	Predecess
80	Version Control Software System Release 1	2 days	Wed 6/20/18	Thu 6/21/18	79
81	**Design & Build by Feature phase - Iteration 2**	**30 days**	**Fri 6/22/18**	**Thu 8/2/18**	**73**
82	Build Database	5 days	Fri 6/22/18	Thu 6/28/18	
83	Design Application Features	8 days	Fri 6/29/18	Tue 7/10/18	82
84	Build Application Features	5 days	Wed 7/11/18	Tue 7/17/18	83
85	Develop Test Plan	2 days	Wed 7/18/18	Thu 7/19/18	84
86	Application Testing	3 days	Fri 7/20/18	Tue 7/24/18	85
87	Refine and Resolve Issues	5 days	Wed 7/25/18	Tue 7/31/18	86
88	Version Control Software System Release 2	2 days	Wed 8/1/18	Thu 8/2/18	87
89	**Design & Build by Feature phase - Iteration 2**	**30 days**	**Fri 8/3/18**	**Thu 9/13/18**	**81**
90	Build Database	5 days	Fri 8/3/18	Thu 8/9/18	
91	Design Application Features	8 days	Fri 8/10/18	Tue 8/21/18	90
92	Build Application Features	5 days	Wed 8/22/18	Tue 8/28/18	91
93	Develop Test Plan	2 days	Wed 8/29/18	Thu 8/30/18	92
94	Application Testing	3 days	Fri 8/31/18	Tue 9/4/18	93
95	Refine and Resolve Issues	5 days	Wed 9/5/18	Tue 9/11/18	94
96	Version Control Software System Release 2	2 days	Wed 9/12/18	Thu 9/13/18	95

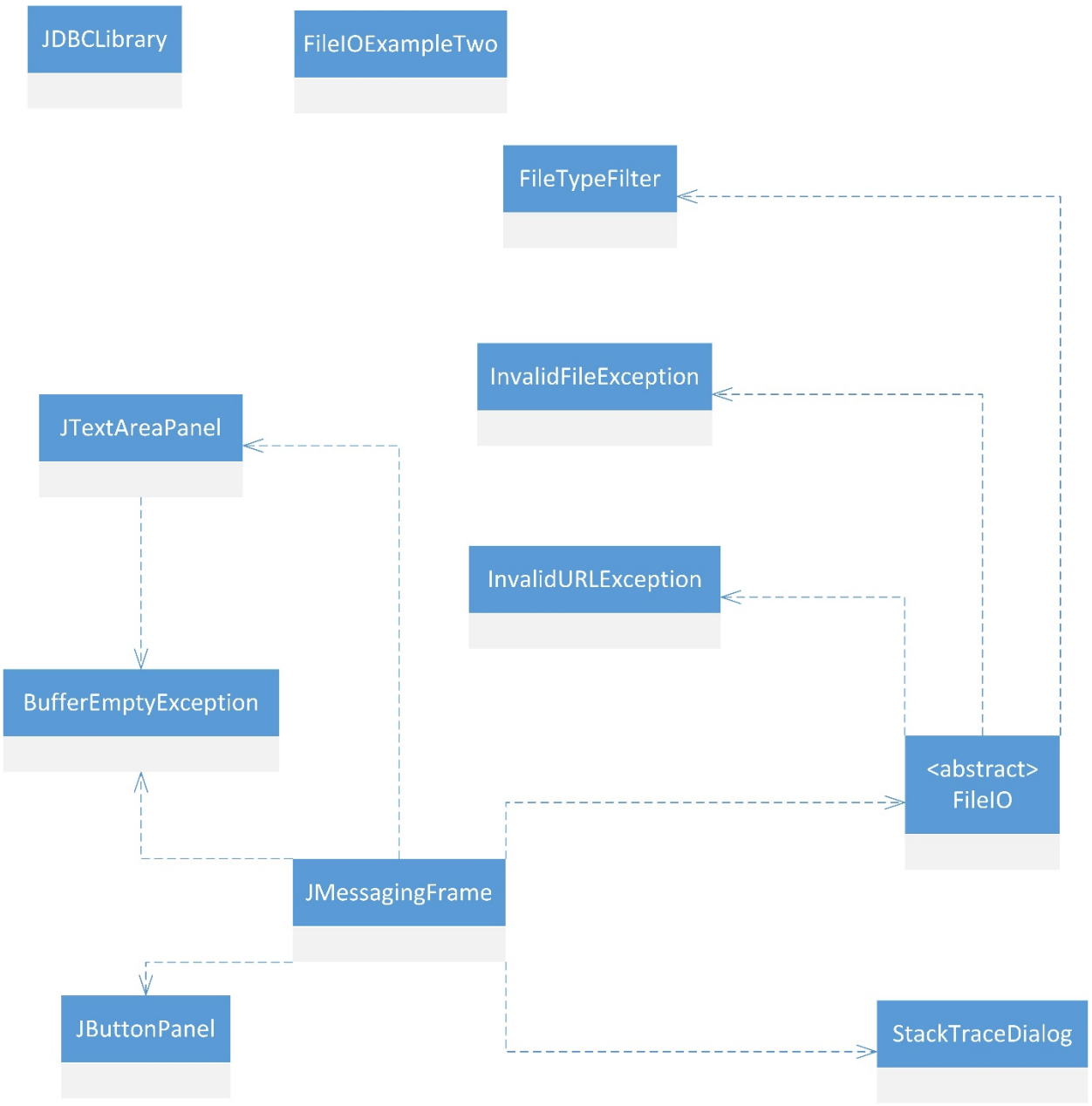
Page 4

Appendix C: UML System Use Case of Current Code & Proposed Features



Appendix D: UML Class Diagram of Existing Code

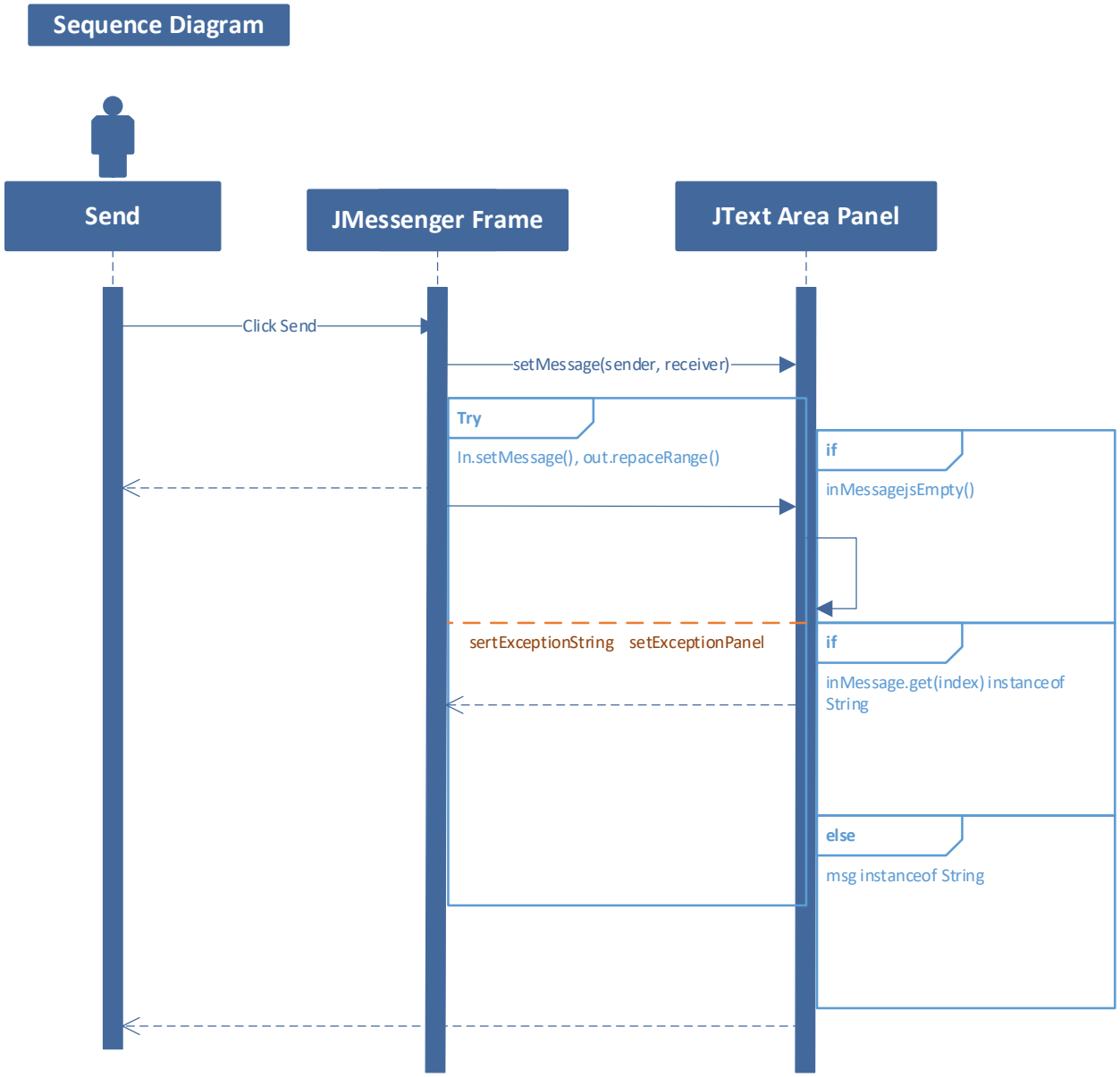
This is a basic diagram based on the existing code, prior to enhancements:



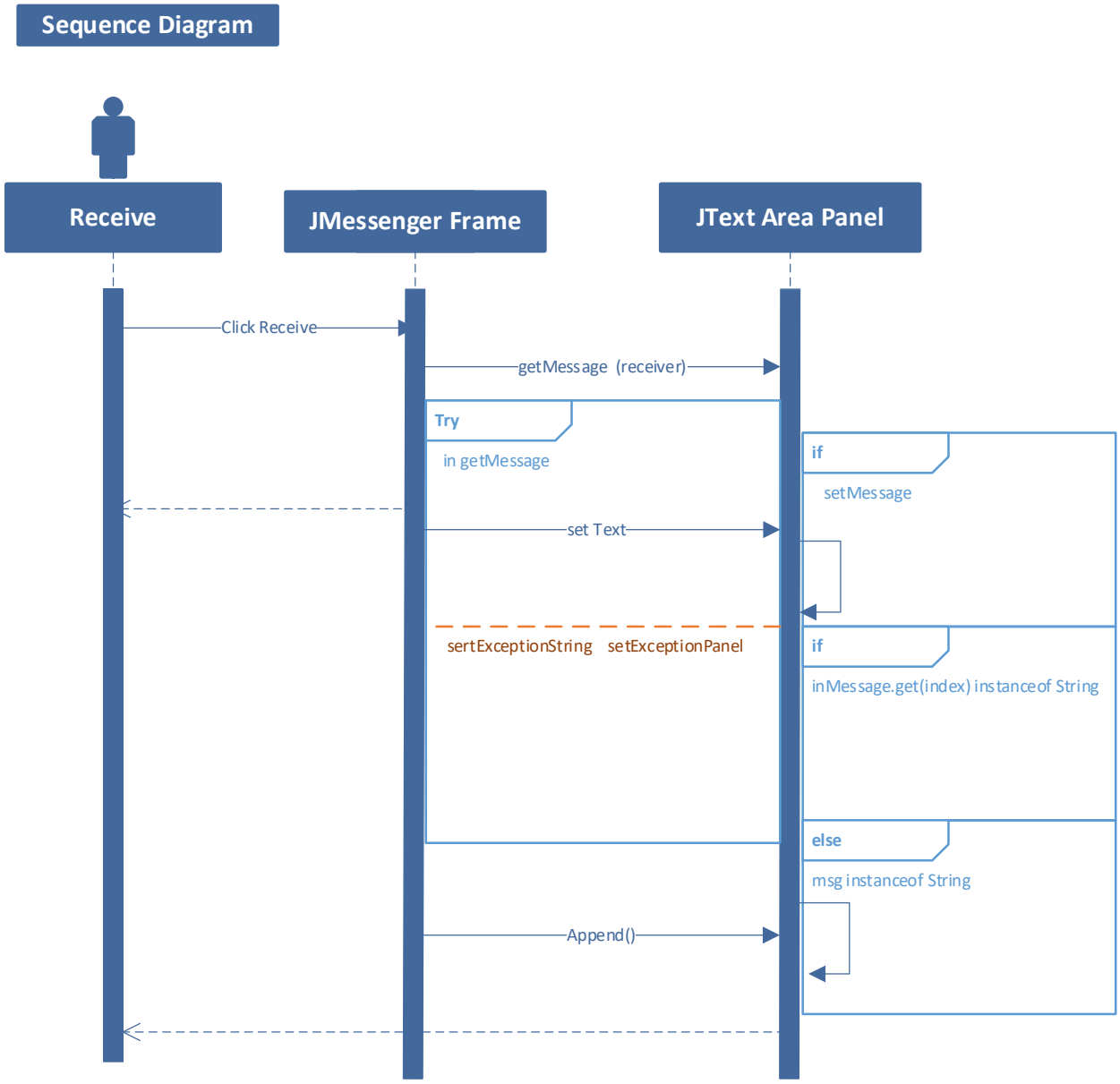
Appendix E: UML Sequence Diagrams of Existing Code

These are basic diagrams based on the existing code, prior to enhancements:

Send Sequence Diagram



Receive Sequence Diagram



Appendix F: UML State Diagram of Existing Code

This is a basic diagram based on the existing code, prior to enhancements: