SOFTWARE PROJECT MANAGEMENT PLAN

IEEE 1058

Project: Java Air

Team: Avian Limited
Project Manager: Steve Jia
Advanced Software Engineering Fall Semester 2016
Dr. Ruijian Zhang

Steve Jia
ssjia@pnw.edu

Contents

Approvals

Title	**Signature**	**Date**
Project Manager / Author	Steve Jia	2016-11-06

Revision History

Version	**Date**	**Author**	**Description**
1.0	2016-09-20	S. Jia	Document Creation
1.0.1	2016-09-21	S. Jia	Added index items for 5.1, 5.2, and 5.3; Added main contents for sections 1, 2, 3, and 4.

1.0.2	2016-09-29	S. Jia	Finished section 5;
			Finished section 6;
			Added content for section 7;

1.0.3	2016-10-15	S. Jia	Added the Java Air schedule
1.1	2016-11-02	S. Jia	Added Risk Prioritization table
1.2	2016-11-03	S. Jia	Added addition information for risks (section 5.4); Added Function Point Analysis table; Added process model information;
1.3	2016-11-06	S. Jia	Updated Risk Analysis table

1. Overview

1.1. Project Summary

This project is organized to produce a software application for Java Air's customers and employees. The overall plan for this product is planning, design, implementation, and testing. This product will be developed in several stages, with each stage following review and demonstration of each version. The early versions of the product will mainly be basic graphical user interfaces and animations, and with advances in development, later versions of the product will have database support and more advanced graphical user interfaces and animations, as well as error checking and logging functionalities.

1.2. Evolution of the SPMP

This document will be maintained on a weekly basis by the project manager and leader, Steve Jia. Contents of this document will be improved and maintained based on the development progress of the project team. It is subject to configuration management by means of the Software Configuration Management Plan (SCMP). It is the project manager's responsibility to submit this document as a Configuration Item (CI), and to keep it up to date. This Software Project Management Plan (SPMP) mainly follows the format of IEEE 1058.1-1998.

2. References

[IEEE] The applicable IEEE standards are published in "IEEE Standards Collection," 1997 Edition.

[Braude] The principle source of textbook reference material is *Software Engineering: An Object-Oriented Perspective* by E. Braude (Wiley, 2000).

3. Definitions

CI = Configuration Item

CMMI = Capability Maturity Model Integration

IEEE = Institute of Electrical and Electronics Engineers

QA = Quality Assurnace

SEI = Software Engineering Institute

SCMP = Software Configuration Management Plan

SPMP = Software Project Management Plan (this document)

SRS = Software Requirements Specification

SDD = Software Design Document

SQAP = Software Quality Assurance Plan

SVVP = Software Verification and Validation Plan

STP = Software Test Plan

UD = User Documentation

WBS = Work Breakdown Structure

U/PD = User/Product Director

PM = Project Manager

RE = Requirement Engineer

SA = Software Architect

IE = Integration Engineer

TE = Testing Engineer

CD = Code Developer

PNW = Purdue University Northwest

4. Project Organization

4.1. External Interfaces

The project team will interface with the following individuals and organizations: Dr. Ruijian Zhang (U/PD), for technical and standards direction, as well as requirements and specifications. The project team may also interface with members of PNW for tips and ideas.

4.2. Internal Structure

Figure SPMP.1 shows the organization of the Java Air project within Avian Limited.

The project will be organized as a team of peers with designated roles. The roles are project manager, requirement engineer, software architect, integration engineer, testing engineer, and code developer. Each project team member has another role besides the code developer role.

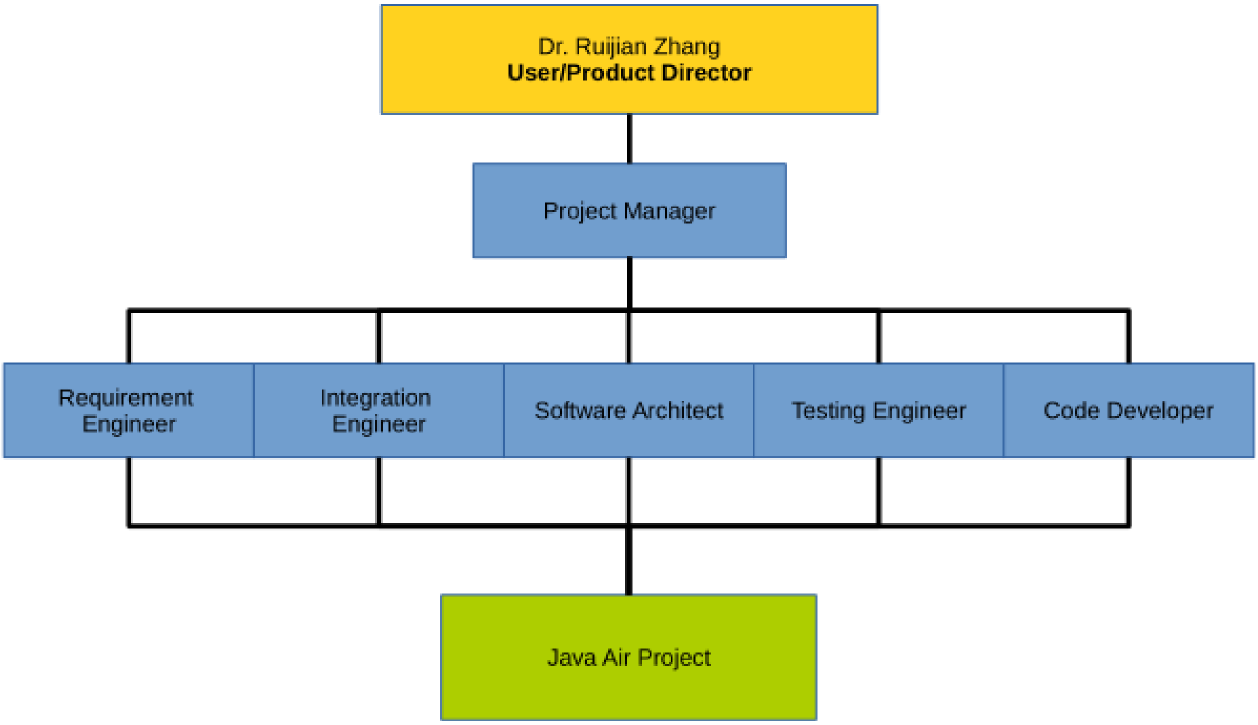


Figure SPMP. 1: Organization of Avian Limited

4.3.Roles and Responsibilities

The responsibilities of the participants in the project are show in Table SPMP.1.

Members	*Requirement Engineer*	*Integration Engineer*	*Project Manager*	*Testing Engineer*	*Software Engineer*	*Code Developer*
Liaison Responsibility	U/PD, PM	U/PD, PM	U/PD, All Members	PM	PM, RE	PM, TE
Document Responsibility	SRS	SCMP & UD	SPMP	SQAP, SVVP & STP	SDD	Code Base

Table SPMP. 1: Avian Limited Members, Roles, and Responsibilities

Each role has its own set of responsibilities.

- The project manager will be responsible for the "health" and progress of the project as well as of each team member. The project manager will also be responsible for creating and maintaining the scope of the project, the schedule of the project, and task assignments of the project. The project manager is also responsible for creating and maintaining the Software Project Management Plan.
- The requirement engineers will be responsible for establishing the requirements and specifications for the software product. The requirement engineers will meet with team members to discuss requirements and then review them with the U/PD. The requirement engineers are also responsible for creating and maintaining the Software Requirements Specification. Because the SRS will be completed early on during the project, the requirement engineers will be asked to assist other team members with their documentation, especially that of the testing engineers'.
- The software architect will be working closely with the requirement engineers to translate the SRS into functional specifications and UML diagrams. The software architect will also list different software classes and units with their respective inputs and outputs. The software architect is also responsible for creating and maintaining the Software Design Document.
- The integration engineer is primarily responsible for designing the user interfaces, integrate the interfaces with the code-base, and performing product demonstrations to the team and to the U/PD. The integration engineer is also responsible for creating and maintaining the Software Configuration Management Plan as well as the Software User Documentation.
- The testing engineers are primarily responsible for reviewing and testing of the software product. The testing engineers will establish test plans and record test results. They will also report debugs and issues to the code developers. The requirement engineers are also responsible for creating and maintaining the Software Quality Assurance Plan, the Software Verification and Validation Plan, and the Software Testing Plan. Due to Matt (TE)'s expertise in database, he will be designing and implementing the database for the product. Matt will also take primary responsibility for creating SQL queries.
- The code developers will be responsible for creating the code base and implementing the software interfaces that were designed. They will also perform functioning point analysis on each unit, and also perform bugs and issues fixes on the code-base.

Being Responsible for a document includes the following:

- Making sure that the document is created on time
- Having the project manager identify the writer(s) of the document
- Keeping the document up-to-date throughout the project life cycle

5. Managerial Process Plan

5.1. Project Start-Up Plan

5.1.1. Estimation Plan

Due to inexperience of most of the team members, it's hard to estimate this project's cost based on previous projects or experiences. Therefore, estimation of this project's duration and cost will be made first based on the Work Breakdown Structure, the Network Diagram, and will be updated when the high-level requirements and the detailed specifications and designs are established.

The work breakdown structure (Figure SPMP.2; next page) was created based on a team activity lead by the project manager. During this activity, team members worked together first to identify project deliverables, then establish steps and items that each team member will do in order to complete this project. All the work packages (tasks) are then compiled together to create a Function Point analysis (Figure SPMP.3) to illustrate the estimated duration and cost of the project.

Based on the rough estimation using Function Point analysis and COCOMO I formulas, I estimated the project would cost between 6.71 and 8.89 person-month and would take between 5.15 and 5.73 months to complete. This estimation is longer than the required project duration, therefore the project team needs to be more time efficient and put in more effort in order to complete this project on-time.

5.1.2. Staffing Plan

The roles will be filled as follows.

Members	*CAO, Yuwei*	*EXRLEBEN, Amy*	*JIA, Steve*	*MOSCATEL, Matt*	*QI, Guoyu*	*WU, Xu*	*ZHANG, Rui*
Role	Requirement Engineer, Code Developer	Integration Engineer, Code Developer	Project Manager, Code Developer	Testing Engineer, Code Developer	Requirement Engineer, Code Developer	Testing Engineer, Code Developer	Software Architect, Code Developer

Table SPMP. 2: Avian Limited Roles Assignment

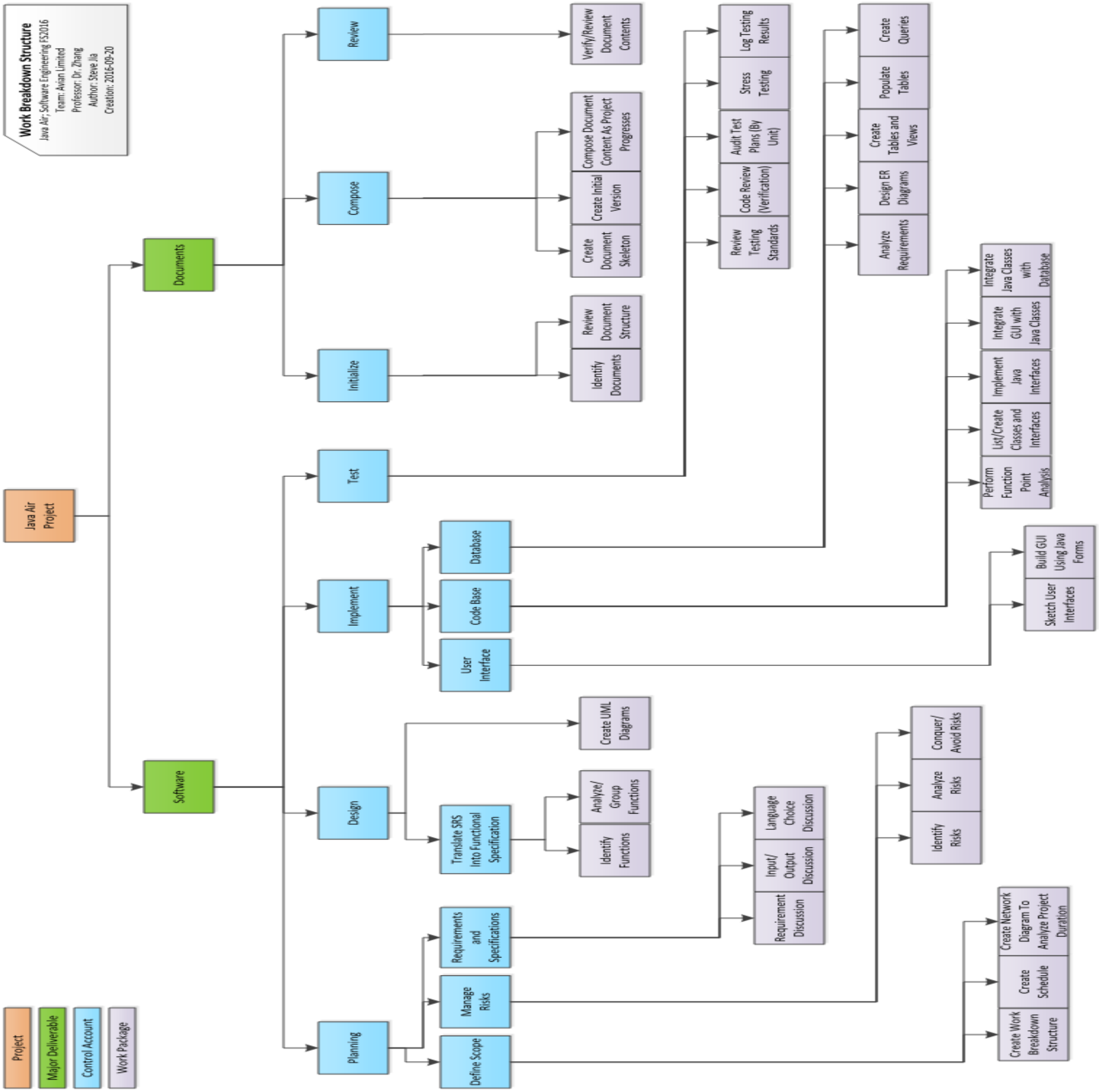
5.1.3. Resource Acquisition Plan

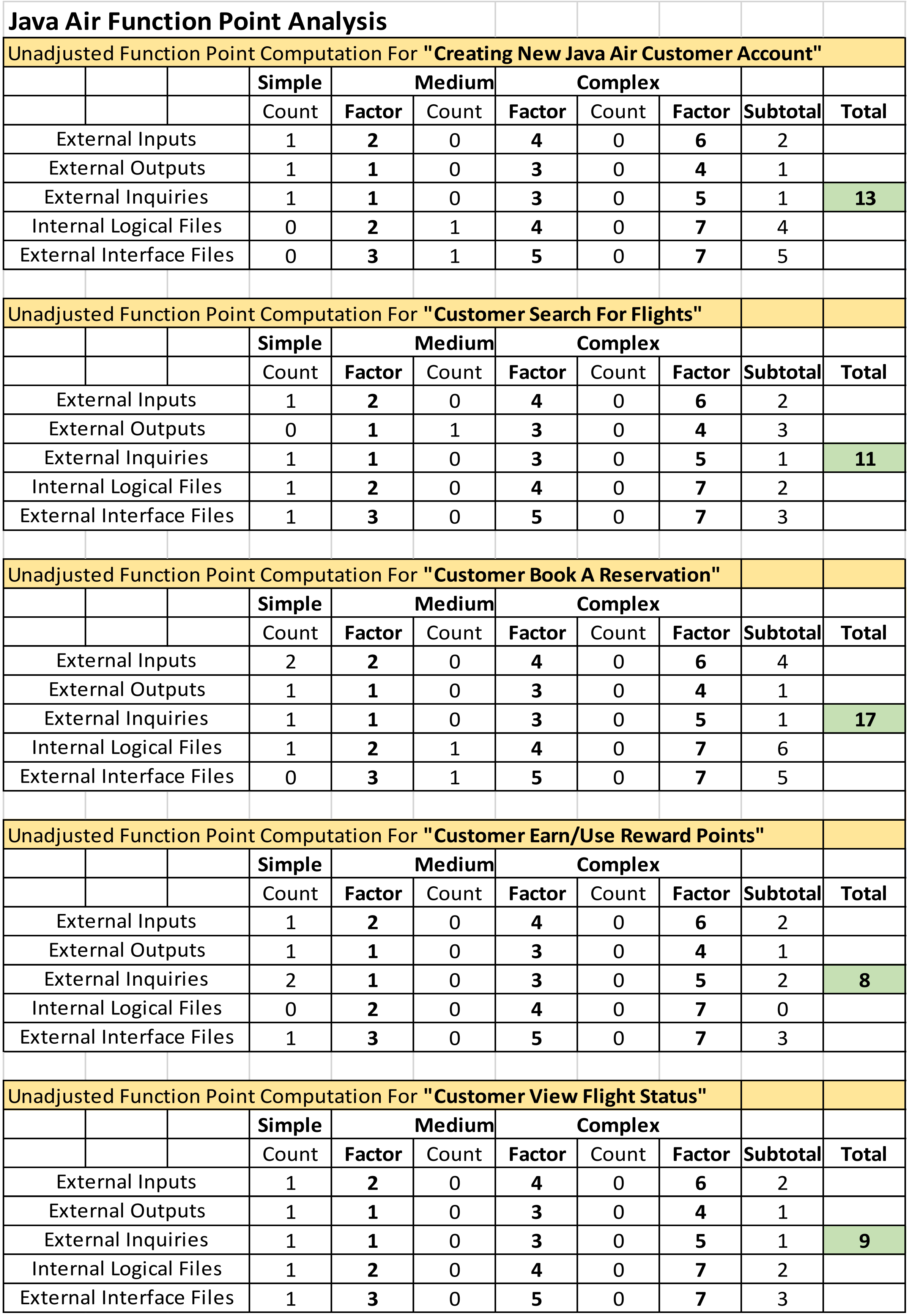
Each team member will use his/her person computer as well as university (PNW) computers to work on the project. Software tools used for this project will be primarily free-ware, but each team member can use their personally-licensed software if desired.

5.1.4. Project Staff Training Plan

All team members whose are not proficient in Java should study the PowerPoint slides from the Object-Oriented Programming Design course (Purdue University Calumet, Fall Semester 2015) that are located on the shared Google Drive folder. Team members should also refer to the ***Oracle Java Online Tutorials*** (https://docs.oracle.com/javase/tutorial/) for guidance and official documentations. Team members should also increase their Java programming skills by working on practice questions that are provided by ***CodingBat*** (http://codingbat.com/java).

Figure SPMP. 2: Java Air Project Work Breakdown Structure





Java Air Function Point Analysis

Unadjusted Function Point Computation For **"Creating New Java Air Customer Account"**

	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	1	**2**	0	**4**	0	**6**	2	
External Outputs	1	**1**	0	**3**	0	**4**	1	
External Inquiries	1	**1**	0	**3**	0	**5**	1	**13**
Internal Logical Files	0	**2**	1	**4**	0	**7**	4	
External Interface Files	0	**3**	1	**5**	0	**7**	5	

Unadjusted Function Point Computation For **"Customer Search For Flights"**

	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	1	**2**	0	**4**	0	**6**	2	
External Outputs	0	**1**	1	**3**	0	**4**	3	
External Inquiries	1	**1**	0	**3**	0	**5**	1	**11**
Internal Logical Files	1	**2**	0	**4**	0	**7**	2	
External Interface Files	1	**3**	0	**5**	0	**7**	3	

Unadjusted Function Point Computation For **"Customer Book A Reservation"**

	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	2	**2**	0	**4**	0	**6**	4	
External Outputs	1	**1**	0	**3**	0	**4**	1	
External Inquiries	1	**1**	0	**3**	0	**5**	1	**17**
Internal Logical Files	1	**2**	1	**4**	0	**7**	6	
External Interface Files	0	**3**	1	**5**	0	**7**	5	

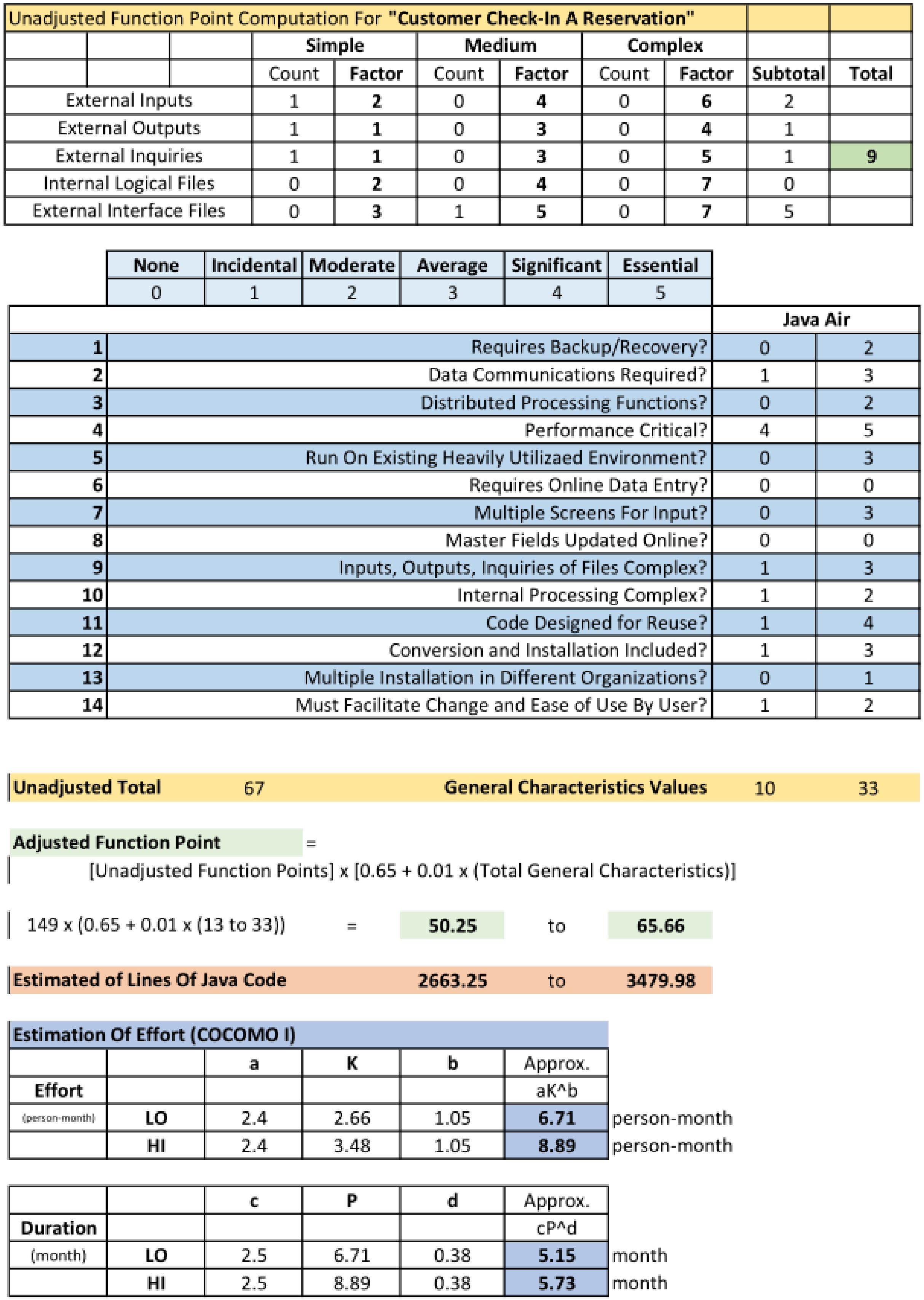
Unadjusted Function Point Computation For **"Customer Earn/Use Reward Points"**

	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	1	**2**	0	**4**	0	**6**	2	
External Outputs	1	**1**	0	**3**	0	**4**	1	
External Inquiries	2	**1**	0	**3**	0	**5**	2	**8**
Internal Logical Files	0	**2**	0	**4**	0	**7**	0	
External Interface Files	1	**3**	0	**5**	0	**7**	3	

Unadjusted Function Point Computation For **"Customer View Flight Status"**

	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	1	**2**	0	**4**	0	**6**	2	
External Outputs	1	**1**	0	**3**	0	**4**	1	
External Inquiries	1	**1**	0	**3**	0	**5**	1	**9**
Internal Logical Files	1	**2**	0	**4**	0	**7**	2	
External Interface Files	1	**3**	0	**5**	0	**7**	3	

Figure SPMP. 3 - Java Air Function Point Analysis



Unadjusted Function Point Computation For **"Customer Check-In A Reservation"**								
	Simple		**Medium**		**Complex**			
	Count	**Factor**	Count	**Factor**	Count	**Factor**	**Subtotal**	**Total**
External Inputs	1	**2**	0	**4**	0	**6**	2	
External Outputs	1	**1**	0	**3**	0	**4**	1	
External Inquiries	1	**1**	0	**3**	0	**5**	1	**9**
Internal Logical Files	0	**2**	0	**4**	0	**7**	0	
External Interface Files	0	**3**	1	**5**	0	**7**	5	

None	**Incidental**	**Moderate**	**Average**	**Significant**	**Essential**
0	1	2	3	4	5

		Java Air	
1	Requires Backup/Recovery?	0	2
2	Data Communications Required?	1	3
3	Distributed Processing Functions?	0	2
4	Performance Critical?	4	5
5	Run On Existing Heavily Utilizaed Environment?	0	3
6	Requires Online Data Entry?	0	0
7	Multiple Screens For Input?	0	3
8	Master Fields Updated Online?	0	0
9	Inputs, Outputs, Inquiries of Files Complex?	1	3
10	Internal Processing Complex?	1	2
11	Code Designed for Reuse?	1	4
12	Conversion and Installation Included?	1	3
13	Multiple Installation in Different Organizations?	0	1
14	Must Facilitate Change and Ease of Use By User?	1	2

Unadjusted Total 67 **General Characteristics Values** 10 33

Adjusted Function Point =

[Unadjusted Function Points] x [0.65 + 0.01 x (Total General Characteristics)]

149 x (0.65 + 0.01 x (13 to 33)) = **50.25** to **65.66**

Estimated of Lines Of Java Code **2663.25** to **3479.98**

Estimation Of Effort (COCOMO I)						
		a	**K**	**b**	Approx.	
Effort					aK^b	
(person-month)	**LO**	2.4	2.66	1.05	**6.71**	person-month
	HI	2.4	3.48	1.05	**8.89**	person-month

		c	**P**	**d**	Approx.	
Duration					cP^d	
(month)	**LO**	2.5	6.71	0.38	**5.15**	month
	HI	2.5	8.89	0.38	**5.73**	month

5.2. Work Plan

5.2.1. Work Activities

The work on this project will be divided into configuration management, quality assurance (including testing), requirements analysis, design, and implementation. The project roles and responsibilities are show in Table SPMP.1 as well as a detailed list on page 5.

5.2.2. Schedule Allocation

The detailed schedule is shown in Figure SPMP.4. Please refer to the Software Quality Assurance Plan (SQAP) for the schedule of quality activities.

5.2.3. Resource Allocation

The work breakdown structure in Figure## shows the bottom line of person-months available each month.

Currently design has not completed, so task assignment is not available. Names will be added once design and configurations have been determined.

5.2.4. Budget Allocation

The Java Air project does not have a dollar-budget, therefore the budget for this project has been determined by the Project Manager to allow an average of 4 hours per engineer per week. Therefore, in a typical week, 28 hours are budgeted for the project. The monthly engineer-hour budget takes in consideration for the Labor Day National Holiday, PNW Fall Break and Thanksgiving Break.

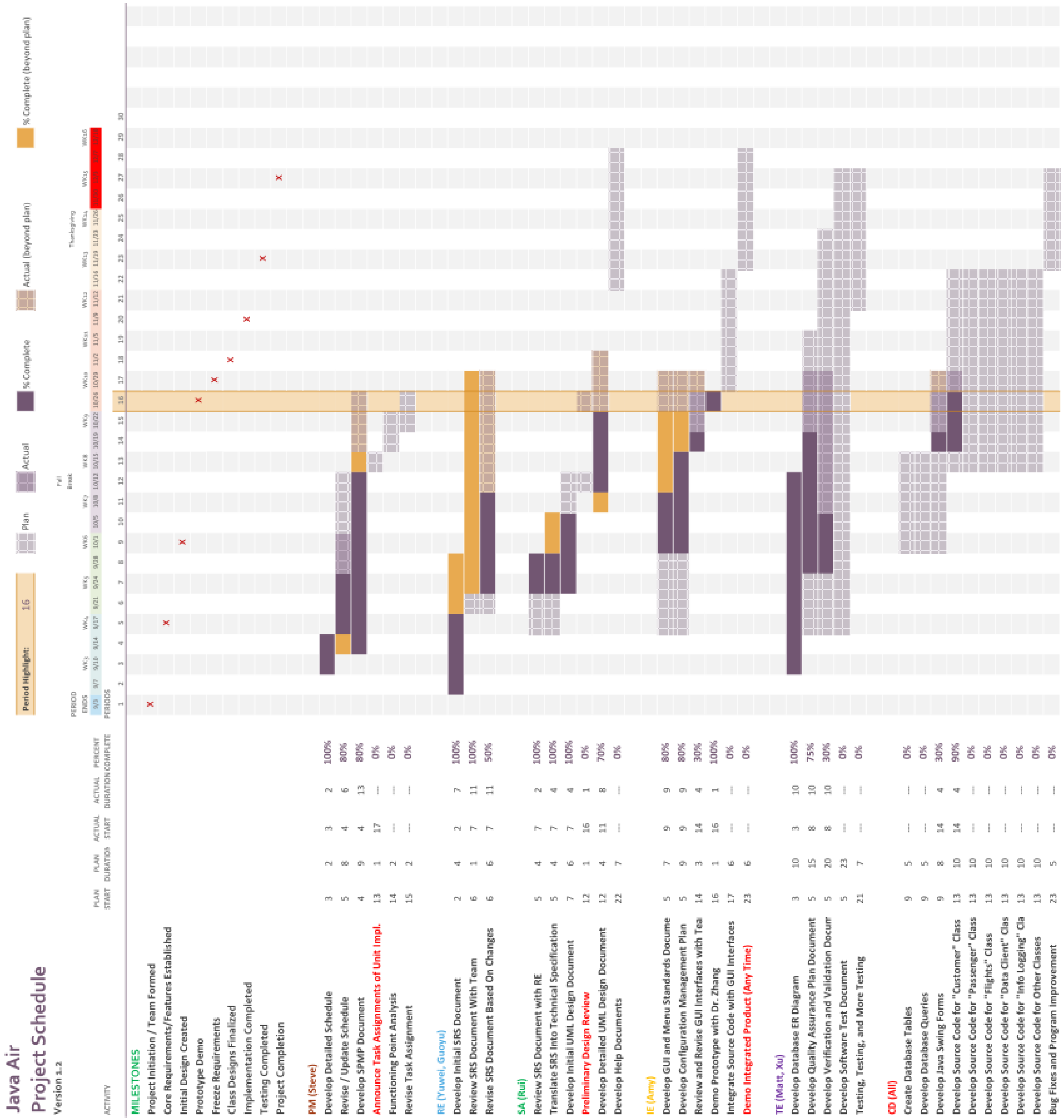
Table SPMP. 3: Java Air Budget Allocation

Month Number	**Allocation (hours)**
1 - September	105
2 – October	112
3 – November	101.5
4 – December	28
Total	346.5

5.3.Control Plan

In the initiation and planning phase of the Java Air project, the entire team will meet every Saturday evening from 5:00 p.m. to 6:00 p. m. and whenever a meeting is needed. Unscheduled meetings can take place after class, or any time team members are available. The project manager will schedule extra meetings with the team members after each class on Mondays and Wednesdays.

During the requirement analysis and design phases of the project, the requirement engineers, the software architect, and the project manager will meet Tuesday afternoons to review specifications and design documents.



Java Air
Project Schedule
Version 1.2

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
MILESTONES					
Project Initiation / Team Formed					
Core Requirements/Features Established					
Initial Design Created					
Prototype Demo					
Freeze Requirements					
Class Designs Finalized					
Implementation Completed					
Testing Completed					
Project Completion					
PM (Steve)					
Develop Detailed Schedule	3	2	3	2	100%
Revise / Update Schedule	5	8	4	6	80%
Develop SPMP Document	4	9	4	13	80%
Announce Task Assignments of Unit Impl.	13	1	17	---	0%
Functioning Point Analysis	14	2	---	---	0%
Revise Task Assignment	15	2	---	---	0%
RE (Yuwei, Guoyu)					
Develop Initial SRS Document	2	4	2	7	100%
Review SRS Document With Team	6	1	7	11	100%
Revise SRS Document Based On Changes	6	6	7	11	50%
SA (Rui)					
Review SRS Document with RE	5	4	7	2	100%
Translate SRS Into Technical Specification	5	4	7	4	100%
Develop Initial UML Design Document	7	6	7	4	100%
Preliminary Design Review	12	1	16	1	0%
Develop Detailed UML Design Document	12	4	11	8	70%
Develop Help Documents	22	7	---	---	0%
IE (Amy)					
Develop GUI and Menu Standards Docume	5	7	9	9	80%
Develop Configuration Management Plan	5	9	9	9	80%
Review and Revise GUI Interfaces with Tea	14	3	14	4	30%
Demo Prototype with Dr. Zhang	16	1	16	1	100%
Integrate Source Code with GUI Interfaces	17	6	---	---	0%
Demo Integrated Product (Any Time)	23	6	---	---	0%
TE (Matt, Xu)					
Develop Database ER Diagram	3	10	3	10	100%
Develop Quality Assurance Plan Document	5	15	8	10	75%
Develop Verification and Validation Docum	5	20	8	10	30%
Develop Software Test Document	5	23	---	---	0%
Testing, Testing, and More Testing	21	7	---	---	0%
CD (All)					
Create Database Tables	9	5	---	---	0%
Develop Database Queries	9	5	---	---	0%
Develop Java Swing Forms	9	8	14	4	30%
Develop Source Code for "Customer" Class	13	10	14	4	90%
Develop Source Code for "Passenger" Class	13	10	---	---	0%
Develop Source Code for "Flights" Class	13	10	---	---	0%
Develop Source Code for "Data Client" Clas	13	10	---	---	0%
Develop Source Code for "Info Logging" Cla	13	10	---	---	0%
Develop Source Code for Other Classes	13	10	---	---	0%
Bug Fixes and Program Improvement	23	5	---	---	0%

Figure SPMP. 4 - Java Air Detailed Schedule

For the First week of October when the Project Manager is unavailable, Testing Engineer Matt will be the substitute Project Manager. The substitute PM will coordinate meetings with the rest of the team members and communicate meetings to the Project Manager.

5.3.1. Requirements Control Plan

The requirement engineers (Cao, Qi) will report to the project leader on the status of the Software Requirements Specification in-person or in writing each Monday.

5.3.2. Schedule Control Plan

The project manager (Jia) will report to the team on the status of the schedule in-person and/or in writing each Wednesday or Saturday.

5.3.3. Budget Control Plan

The project leader will report to the U/PD on the status of the budget in writing in each meeting minutes.

5.3.4. Quality Control Plan

The testing engineers (Moscatel, Wu) will provide written reports to the project manager and carbon copy rest of the team members.

5.3.5. Reporting Plan

The written reports referred to in this section (5.3) will be via e-mail.

5.3.6. Metrics Collection Plan

Please see section 5 of the Software Quality Assurance Plan.

5.4. Risk Management Plan

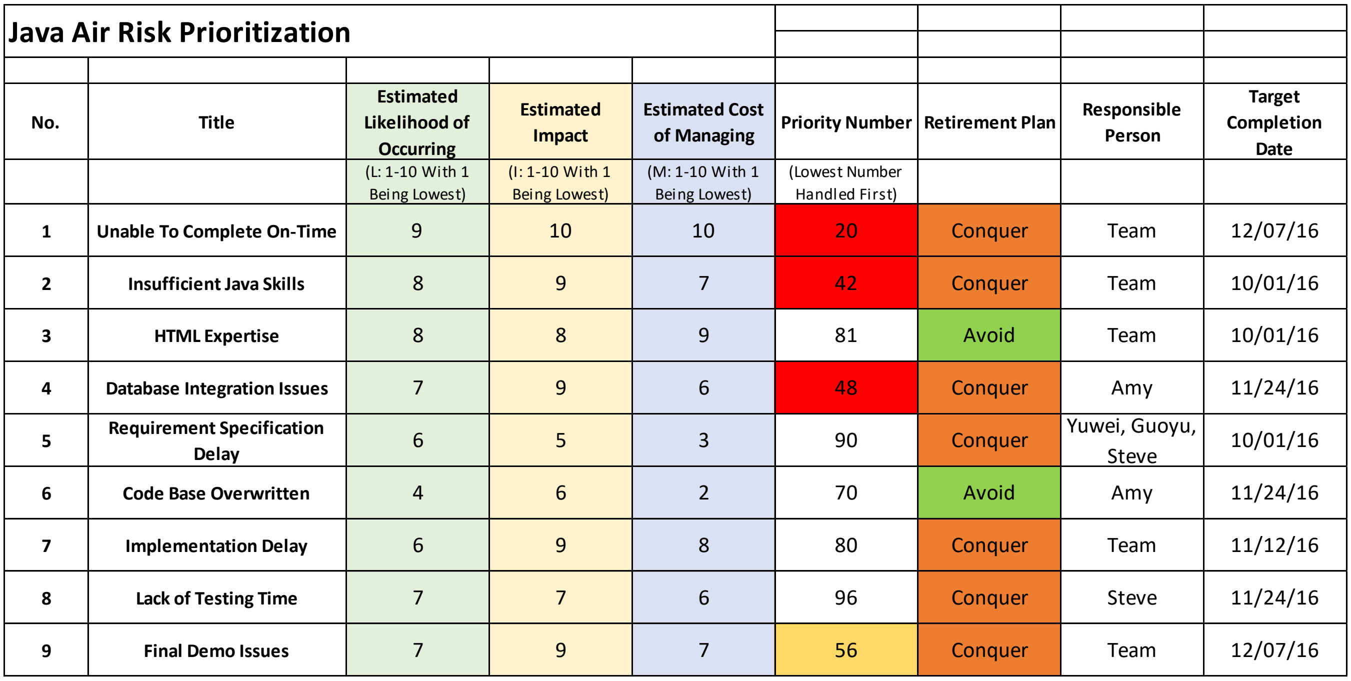
Table SPMP.4 shows a format for risk reporting and retirement. Starting from the design phase, each team meeting will have an agenda item for risk identification brainstorming and reporting on risks that have been identified.

5.4.1. Risk #1: Java Expertise

Based on the language-choice decision table, Java is the most suitable language choice for the Avian Limited team members for the Java Air project. However, four out of seven team members identified themselves as having little or no knowledge at all about Java and object-oriented programming.

Using Java to implement the Java Air code base is crucial, and most of the team members are fluent in Java is a critical issue that has very high impact. By interviewing the team members that lack Java programming experiences, the project manager found that they lack programming experience in general.

Because of this, risk retirement by avoidance is not a viable option, since these team members do not know how to code in any other major programming languages. Therefore, this risk must be conquered. The project manager will provide guidance materials on the Java programming language, reference to the official Oracle Java documentations, and URL to a Java coding-exercise website.



Java Air Risk Prioritization

No.	Title	Estimated Likelihood of Occurring	Estimated Impact	Estimated Cost of Managing	Priority Number	Retirement Plan	Responsible Person	Target Completion Date
		(L: 1-10 With 1 Being Lowest)	(I: 1-10 With 1 Being Lowest)	(M: 1-10 With 1 Being Lowest)	(Lowest Number Handled First)			
1	Unable To Complete On-Time	9	10	10	20	Conquer	Team	12/07/16
2	Insufficient Java Skills	8	9	7	42	Conquer	Team	10/01/16
3	HTML Expertise	8	8	9	81	Avoid	Team	10/01/16
4	Database Integration Issues	7	9	6	48	Conquer	Amy	11/24/16
5	Requirement Specification Delay	6	5	3	90	Conquer	Yuwei, Guoyu, Steve	10/01/16
6	Code Base Overwritten	4	6	2	70	Avoid	Amy	11/24/16
7	Implementation Delay	6	9	8	80	Conquer	Team	11/12/16
8	Lack of Testing Time	7	7	6	96	Conquer	Steve	11/24/16
9	Final Demo Issues	7	9	7	56	Conquer	Team	12/07/16

Table SPMP. 4 Java Air Risk Analysis

5.4.2. Risk #2: HTML Expertise

Originally, the team thought having a website as the graphical user interface, however, after interview, all the team members are identified as “little or no experience in HTML and Javascript”. This issue can cause project delays and software performance issues later on.

After communicating with the U/PD and the team members, the project manager decided to retire this risk by avoidance. The team will use Java JFrames/Swing Forms, and an IDE like Netbeans, to create the graphical user interface.

5.4.3. Risk #3: Database Integration Issues

After preliminary discussion, the team decided to use a database to store the project’s data and act as the back-end for the user interface and Java classes. However, most of the databases require a server, which was not achievable.

The back-end is very important to the project. The database can store all the project information and software configurations, and it provides easy access to data to the user interface. Also, one of the most important things is to have this database offline, running locally with the rest of the software product using SQL queries to communicate.

The project manager researched possible products and found SQLite, which should satisfy all the requirements. The team retired this risk by conquest.

5.4.4. Risk #4: Requirement Specification Delay

The Software Requirements and Specification is a very important piece of document, and it must be completed in a short amount of time to give more time to the rest of the project phases. To prevent project delays caused by an incomplete SRS, the project manager will work closely with the requirement engineers to establish the contents and to make sure the content is of great quality. The project manager will check on the progress of the document frequently to make sure the SRS is completed on-time.

5.4.5. Risk #5: Code Base Overwritten

Once the code base implementation begins, every Avian Limited team member will be using git to implement their changes. During this phase, there is a chance that a team member would accidentally overwrite the working code base and generate errors in the code or cause the code base to unable to be compiled.

The avoid this issue, the integration engineer will develop a thorough Configuration Management plan, guide every team member to make sure they understand how to use git, and monitor changes made to the remote repository as well as keeping a backup of a working code base.

5.4.6. Risk #6: Implementation Delay

The implementation phase and testing phase of the project are towards the end of the school semester. In order to deliver a product of quality, the testing engineers need plenty of time to find and fix bugs that are in the code base. If implementation takes too long, it would shorten the testing time.

The team must conquer this risk. Each member of the team will implement the code base as much as they can in order to meet the implementation phase deadline.

5.4.7. Risk #7: Lack of Testing Time

Similar to section 5.4.6, the project team will conquer this risk. The project manager will make sure that the testing engineers will have enough time to conduct testing, and the project manager will allocate additional personnel resources towards the testing phase.

5.4.8. Risk #8: Final Demo Issues

Many issues might occur during the final acceptance test. To conquer the risks that might occur, the team will test the code base thoroughly to rid of any bugs. The team will also figure out a way to generate an easy-to-use executable for Windows systems so that the program can be tested on different PCs before the final demo.

5.4.9. Risk #9 Unable to Complete Project On-Time

After estimating the duration and effort of this project using Function Point Analysis. The project manager realized that very likely the project team will not be able to complete the project in the short project duration set by the U/PD.

In order to conquer this risk, the whole team needs to make sure all the deadlines set by the project manager are met on-time or earlier, and all team members must put in extra effort to complete this project. The project manager will focus on the core requirements and functionalities to complete first, and then focus on less critical requirements. The project manager will also inform the U/PD of the project progress, and manage his expectations towards the final product.

5.5. Project Closeout Plan

The Java Air software product will not be maintained beyond December, 2016. At the end of the project, when the software testing is completed and all the deliverables are reviewed, the project manager will deliver the items to the U/PD and also submit a final project report and a team performance and participation document.

6. Technical Process Plan

6.1. Process Model

The Java Air project will be executed using a Unified process development process. During the inception phase, the project manager will form the project team, assign roles, obtain a list of deliverables, and establish a project scope. The project manager will create a schedule, identify the project milestones, and identify as many risks as possible.

Towards the later of the inception phase, the requirement engineers will start to gather requirements and start to compose the Software Requirements and Specification. At the beginning of the elaboration phase, while the requirement engineers are working on the SRS, the rest of the team members should start to compose their document(s) of responsibility. During the elaboration phase, the integration engineer should have established a good configuration plan and start to guide team members with using git. The software architect should start to review the SRS and start to work on the UML design and the Software Design Document. Towards the end of the elaboration phase, the requirement engineers will finalize the SRS, and the software architect should have a usable version of the detailed design for the team to review and start to implement.

The GUI will be the first code base components to be implemented during the construction phase. As more code base are being implemented, the software architect will make small necessary changes, and the testing engineers should complete the Software Quality Assurance Plan, Software Verification and Validation Plan, as well as the Software Testing Plan. When most of the code base is implemented and the testing has started, the project will move into the transition phase.

Throughout the project, the project manager will monitor the progress and health of the project to make sure the project is finished on time and of good quality.

6.2. Methods, Tools, and Techniques

The Java Air project will use UML for design and will be implemented in Java.

For planning and documentation, Microsoft Word, Microsoft Excel, and Microsoft Visio will be used. If team members need free versions of similar software, they can use LibreOffice products.

For design, StarUML will be used to sketch diagrams and document classes, reference variables, methods, relationships, and hierarchy.

For implementation, Java will be the language of choice. Team members can use IDEs such as Netbeans and Eclipse for development.

Github.com, git, and tortoiseGit will be used for version control.

6.3. Infrastructure Plan

The Java Air software product will require a computer with at least a single CPU core and at least 2 GB of memory. The computer must be running a Windows 7 or later operating system. The computer might also need Java installed. The computer should also have a mouse and keyboard in order to operate the software product.

6.4. Product Acceptance Plan

The U/PD and the project manager will finalize acceptance criteria prior to the last six weeks of the project's end. The final demo will be held prior or on December 7th, 2016. The U/PD will give test cases for the project team to demonstrate and the final product will be free of bugs. The project manager will also give a summary of the project to the U/PD during the final acceptance test.

7. Supporting Process Plan

Notes to the integration engineer: the project manager created configuration-item numbers for the documents. The CI number will be JAD (Java Air Documents) followed by a dash, and then by a single digit from 1 to 8.

7.1. Configuration Management Plan

The Java Air Configuration Management Plan, configuration item ***JAD-2***, will be created and maintained by the integration engineer.

7.2. Verification and Validation Plan

The Java Air Software Verification and Validation Plan, configuration item ***JAD-6***, will be created and maintained by the testing engineers.

7.3. Documentation Plan

The Java Air Software User Documentation Plan, configuration item ***JAD-8***, will be created and maintained by the integration engineer.

7.4. Quality Assurance Plan

The Java Air Software Quality Assurance Plan, configuration item ***JAD-5***, will be created and maintained by the testing engineers.

7.5. Testing Plan

The Java Air Software Testing Plan, configuration item ***JAD-7***, will be created and maintained by the testing engineers.

7.6. Requirements and Specification Plan

The Java Air Software Requirements and Specification, configuration item ***JAD-3***, will be created and maintained by the requirement engineers.

7.7. Design Plan

The Java Air Software Design Plan, configuration item ***JAD-4***, will be created and maintained by the software architect.

8. Additional Plans

8.1. Communication Plan

The Java Air project team will communicate effectively in-person during regularly held team meetings, and communicate via e-mail remotely.

Each team member must be informed on items of their responsibility. For example, the integration engineer must be informed on changes that will be made to the code base repository before she can approve and finalize the changes. The project manager must be informed on all items, even small meetings and discussions held being individual team members.

Each team member must communicate the information to necessary team members as soon as possible.

The project manager will compile information and communicate it to the UP/D regularly.