CS 411W Lab III

Prototype Test Plan/Procedure
For
AIR Tracker

Prepared by: Rahil Patel, Green Team

Date: 04/27/2009

Table of Contents

List of Figures

List of Tables

1 Objectives

AIR Tracker is a quality assurance (QA) system with intentions to ultimately reduce the amount of mishandled bags. It provides real-time tracking of bags throughout the entire Ground-level Routing Process (GRP), which includes the critical areas surrounding transfers. It alerts the airport baggage handling staff when bags get off track. Lastly, it provides a historical summary of alerts to assist the airport in finding problem areas within the baggage handling system. AIR Tracker is customized, built and targeted for airports of any size.

Although quite different from the real world product (RWP), the prototype will still be able to complete a myriad of objectives. The objectives will be completed in the order of the process flow itself, starting with the RuBee scanner and ending with reports.

First, to prove that a scanner can grab data from a tag and send it to the simulated AIR Tracker application, the RuBee demonstration kit will be used. Then, to prove that AIR Tracker can query the correct data from the airport database and store it in the AIR Tracker database, the database tables will be checked. Next, to prove alerts can be generated, a bag will be sent off course and the AIR Tracker database will be checked to see if an alert has been inserted. Then, to prove that an alert can be cleared, the bag that went off course will be placed in its correct path, and the AIR Tracker database will be checked to see if the alert status has changed. Lastly, the data from the GRP simulation will have to be sent to the virtual AIR Tracker database, which then can be queried to create a report.

These objectives are turned into specific requirements, and each requirement will be tested. This will make it easier to determine if the prototype is successful or not.

2 References

Patel, Rahil. (2009). *Lab I – AIR Tracker Product Description*. Chesapeake, VA: Author.

Patel, Rahil. (2009). *Lab II – Prototype/Product Specification for AIR Tracker*. Chesapeake, VA: Author.

3 Test Plan

The following sections of the test plan will discuss the types of tests to be performed, the testing schedule, reporting procedures, resource requirements, the testing environment, and team member responsibilities.

3.1 Testing Approach

Functionality of the AIR Tracker prototype will be verified through a combination of component and system tests. The major functional components of the prototype are shown in Figure 1.

[This space is intentionally left blank]

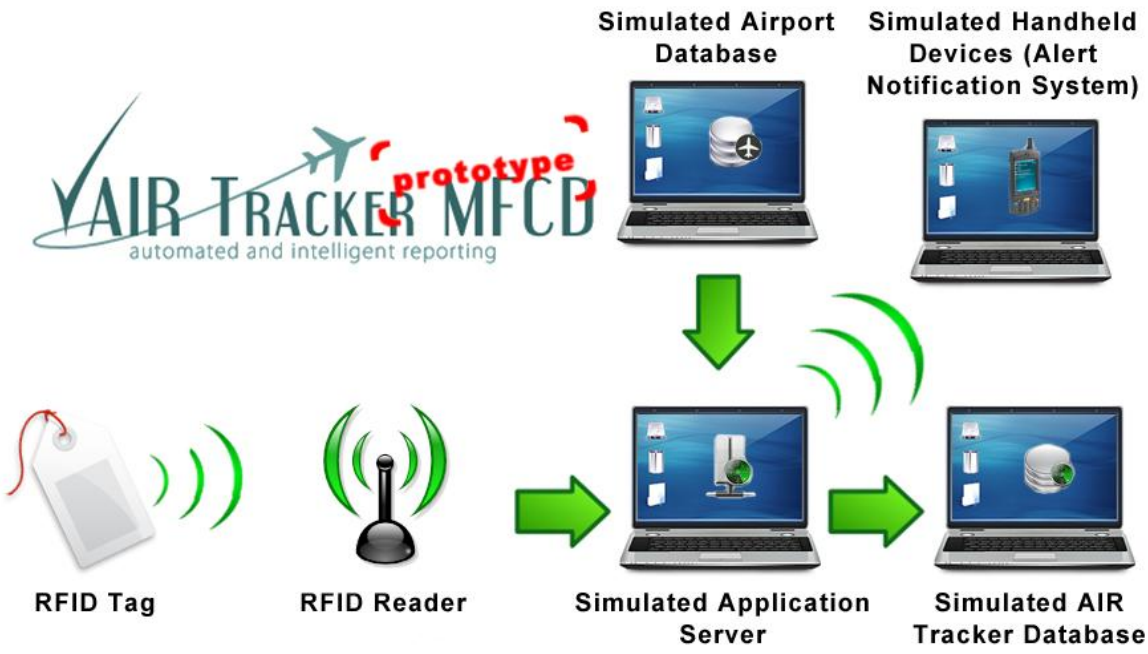

Figure 1. Prototype major functional component diagram

Component tests will be run on the RuBee hardware. This is to ensure that the hardware is capable of the reading. The system tests compose the rest of the tests and will analyze most of the AIR Tracker components together. This includes the AIR Tracker database functionality, AIR Tracker alert generator functionality, AIR Tracker alert clearing functionality, AIR Tracker alert clearing functionality, and the Archival and report functionality.

The tests will make use of real and simulated data. The RuBee demonstration will extract real data from real tags. For the rest of the plan, simulated data will be used. The simulated data is located in a flat file and will be used to populate the airport database.

All tests will be conducted in a controlled classroom environment using the ODU network and hardware. Verification of user interface tests will be completed through observation and by monitoring back-end logs to demonstrate the proper transfer of information through the system. Performance tests will produce reports which summarize the actual results and compare them to the requirement benchmarks.

3.2 Identification of Tests

Table 1 lists the test cases, categorizes them by component, and provides a short description of each. Each test case validates the functionality of its component, and each component is required for the prototype to operate. For more details on a specific test case, see section 5.1.

Category	Description	Test Case #	Description
1	RuBee reader input processing	1.1	RuBee™ Reader Processing
		1.2	RuBee™ Reader Processing Performance
2	AIR Tracker database functionality	2.1	AIR Tracker Database Functionality
		2.2	AIR Tracker and Airport Database Integrity
		2.3	MySQL Database Software Performance
3	AIR Tracker alert generator functionality	3.1	Routing Simulation Creation Functionality
		3.2	Routing Simulation Functionality
4	AIR Tracker alert clearing functionality	4.1	Alert Generator-to-Database Interface Functionality
		4.2	AIR Tracker Alert Generator Functionality (Internal)
		4.3	AIR Tracker Alert Generator Functionality (External)
5	Archival and report functionality	5.1	Alert Generator-to-Handheld Alert System Interface Functionality
		5.2	Handheld Alert System GUI Functionality
		5.3	Historical Reports of Alerts Functionality

Table 1. AIR Tracker prototype test cases by category

3.3 Test Schedule

Green Group has a total of 45 minutes to demonstrate the prototype. The hardware will be prepared during the feasibility presentation, which will use the first 15 minutes. This consists of connecting the laptop to the projector, the RuBee receiver to the laptop, and the antenna to the receiver. The remaining 30 minutes are broken down in Table 2.

Start Time	Duration	Test Objective	Test Event
0:15	3	RuBee reader input processing	1.1, 1.2
0:18	3	AIR Tracker and airport database functionality	2.1, 2.2, 2.3
0:21	7	AIR Tracker alert generator functionality	3.1, 3.2, 3.3, 3.4, 3.5
0:28	7	Handheld alert system functionality	4.1, 4.2, 4.3, 4.4
0:35	3	archival and report functionality	5.1, 5.2
0:42	3	Historical reports of alerts functionality	6.1

Table 2. AIR Tracker prototype demonstration schedule

3.4 Fault Reporting and Data Recording

During the demonstration, the test cases will be checked off by the panelists as the group demonstrates each case. When the demonstration is finished, the group will gather the results and determine whether the group passed or failed the test cases according to the panelist's checks. At the end of the demonstration, the panelists will report any faults verbally and a teammate will record them. Listed in Table 3 are the main functions of the AIR Tracker prototype and how data is recorded for each one, as well as how each fault is reported.

Functionality	Values	How Values are Recorded	Fault Reporting
RuBee reader input processing	Tag ID and timestamp	Outputted to a text file	Visual identification of text file
AIR Tracker and airport databases	All values according to schemas	Stored in the database	Visual identification of tables
Alert Generator	None (GUI)	AIR Tracker alert generator functionality	Visual identification of lack of alerts
Alert Clearing	None (GUI)	AIR Tracker alert clearing functionality	Visual identification of lack of clearing
Archival and Reporting	Alert table values in the AIR Tracker database	Stored in the database	Visual identification of the alerts table

Table 3. Data reporting and fault reporting

3.5 Resource Requirements

All AIR Tracker software will be run on a laptop with two MySQL databases pre-installed. The laptop must have MySQL, Qt, Microsoft PowerPoint, and a web browser installed, in order for the AIR Tracker software to run. Table 4 has detailed descriptions of each resource required during the test.

Resource Name	Description
Laptop PC	Laptop should include a USB or a RS 232 communication port and an operating system of Windows XP, Mac or Ubuntu.
USB-to-Serial adapter	This adapter is needed if the laptop does not have a serial port.
RuBee demo kit	The kit includes tags, a receiver, antennas, a demo kit manual, accessories, and RuBee™ Finder Software
AIR Tracker prototype software	The software includes the simulation, both databases, and all interfaces described in Lab II.
MySQL	MySQL must be installed, because both databases use it.
Qt	Qt is required to display the GUIs.
Web Browser	A web browser is needed to view the databases via PHP script.
Microsoft PowerPoint	Microsoft PowerPoint is used during the feasibility presentation
Projector	A projector is essential for the feasibility presentation

Table 4. AIR Tracker Prototype Test Resource Requirements

3.6 Test Environment

The prototype demonstration will take place in the conference room (E&CS 3316) on the ODU campus. Figure 2 shows the layout of this room.

[This space is intentionally left blank]

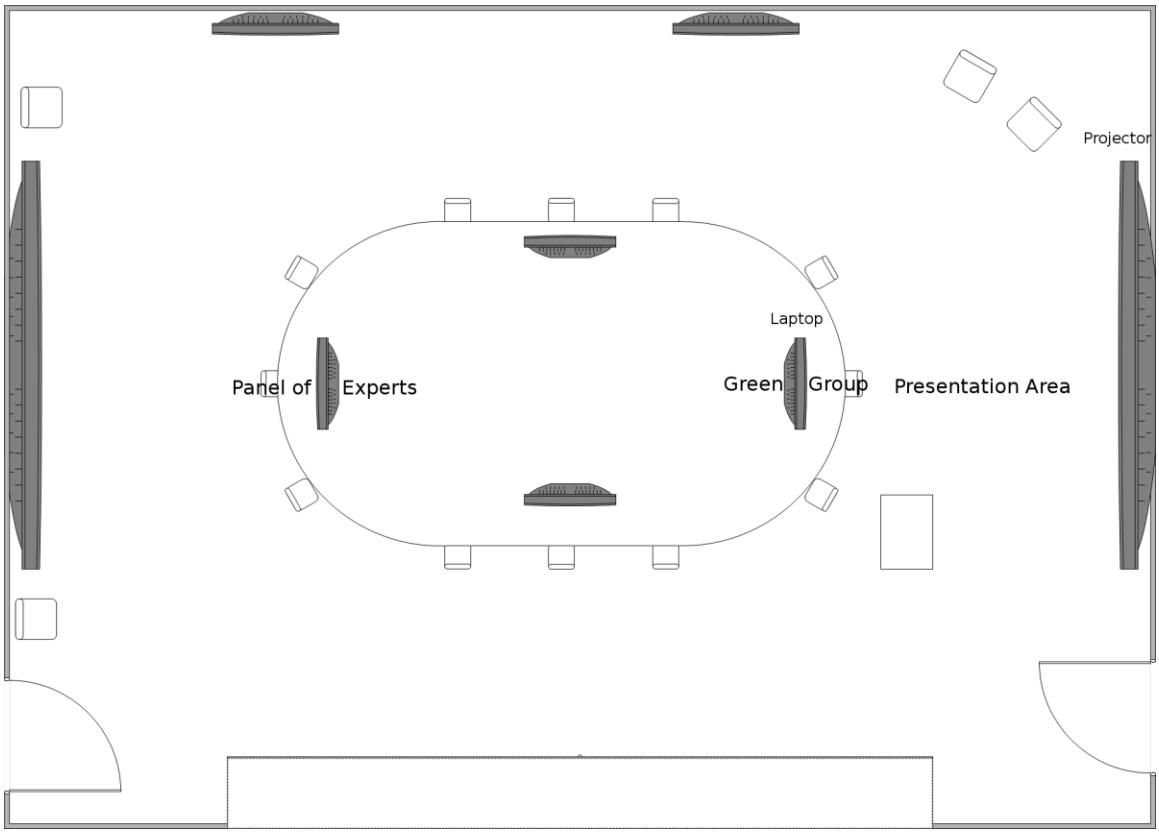

Figure 2. Conference Room Layout

The room supplies a projector with laptop connectivity. The laptop containing the AIR Tracker prototype will be plugged into this projector. The prototype includes both databases and the AIR Tracker application. The laptop also contains the PowerPoint presentation. Lastly, the RuBee demonstration kit will be connected to the laptop via USB to serial adapter.

The panel will be on one side of the table, leaving the Green Group on the other side. The team member presenting at the time will be near the projector screen with another teammate at the laptop, controlling the AIR Tracker prototype the entire time. Two more teammates will be needed to set up and demonstrate the RuBee kit.

4 Test Responsibilities

Jeremey Sellen will be primary speaker for the AIR Tracker prototype demonstration. Joel Elixson will be assisting Jeremey in running the presentation by controlling the laptop while Jeremey speaks. The GUIs faithfully created by Ashley Casper will be demonstrated by Neil Monday. All hardware demonstrations will be the responsibility of Rahil Patel. Table 5 shows the different responsibilities for each team member.

Team Member	Responsibility
Joel Elixson	Presentation Assistant
Neil Monday	GUI
Rahil Patel	Hardware
Jeremey Sellen	Presenter

Table 5. Demonstration Responsibilities and Assignment

5 Test Procedures

AIR Tracker, Inc. has prepared test procedures to ensure test cases are performed in an efficient manner. The test cases are organized into categories that span the scope of the whole project. Descriptions, procedures, and expected results are elaborated in Section 5.1.

5.1 Test Case Name and Identifier

To ensure that all tests are run effectively, tables 5-1 through 5-13 have been generated. Each table lists the category, the purpose, the actual activity, the expected results, as well as a place to annotate whether the test was successful or not. These should be used in verifying the atomic success of the prototype.

Test Category ID: 1	**Description:** RuBee Reader Processing		
Test Case: 1.1	**Purpose:** This test demonstrates correct operability of the RuBee Reader and Finder software		
Specification: 3.1.1.1	Set-Up RuBee Reader Hardware		
Specification: 3.1.1.2	Set-Up RuBee Finder Software		
Test Level:	Component		
Test Type:	Functional		
Setup Conditions:	RuBee™ demo kit required; RuBee™ Reader components connected and RuBee Finder software installed		
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Place each of four RuBee tags within range of ranger antenna		Each tag will be seen in the Finder software when in range of the antenna.
2	Remove each of four RuBee tags from the ranger antenna's field		Each tag will not be seen in the Finder software when not in range of the antenna.

Table 6. RuBee Reader Processing

Test Category ID: 1	**Description:** RuBee™ Reader Processing Performance		
Test Case: 1.2	**Purpose:** This test demonstrates the performance of the RuBee Reader and Finder software		
Specification: 3.2.1	Evaluate performance of the RuBee reader and Finder Software		
Test Level:	Component		
Test Type:	Performance		
Setup Conditions:	Test 1.1 is completed successfully		
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Insert all four tags into the RuBee reader's field		The Finder software registers all four tags simultaneously
2	Incrementally move each tag one foot away from the RuBee reader		The RuBee reader should no longer register the presence of the tag after six to twelve feet (depending on obstructions)
3	Insert a tag into the RuBee reader's field and then remove the tag after the Finder software has recognized it		The Finder software should show the absence of the tag within five seconds of removal

Table 7. RuBee Reader Processing Performance

Test Category ID: 2	**Description:** AIR Tracker Database Functionality		
Test Case: 2.1.1	**Purpose:** Ensure that the databases have been properly installed, created, and populated with test entries.		
Specification: 3.1.2.1	Install MySQL database software and create the AIR Tracker and airport database schemas		
Specification: 3.1.2.2	Populate airport and AIR Tracker databases		
Test Level:	Component		
Test Type:	Functional		
Setup Conditions:	MySQL v.5.1 database software installed; phpMyAdmin v.5.2.5 software installed; AIR Tracker and airport schemas created		
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Populate airport database using a flat file and parametric values from the AIR Tracker simulation		Database is populated with a number of entries equal to the simulation parameters input by the user in the application. A comparison will be made between the flat file, the application and the database
2	Utilize the AIR Tracker application and the airport database to populate the AIR Tracker database		Database is populated with a number of entries equal to the simulation parameters input by the user

Table 8. AIR Tracker Database Functionality

Test Category ID: 2	**Description:** AIR Tracker and Airport Database Integrity		
Test Case: 2.1.2	**Purpose:** Verify content of the databases is within acceptable ranges and duplicate entries are not present		
Specification: 3.1.2.2	Populate airport and AIR Tracker databases		
Test Level:	Component		
Test Type:	Functional		
Setup Conditions:	MySQL v.5.1 database software installed; phpMyAdmin v.5.2.5 software installed; AIR Tracker and airport schemas created		
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Utilize MySQL constraints and attribute types to constrict the types of values accepted by each database table		Attributes are within reasonable bounds and duplicate entries are not present upon examination of the contents of the database

Table 9. AIR Tracker and Airport Database Integrity

Test Category ID:	**Description:** MySQL Database Software Performance		
Test Case: 2.2	**Purpose:** Verify that the MySQL database software is capable of performing queries in a time frame that will not perturb any of the running simulations		
Specification: 3.2.2	Evaluate the performance of the MySQL database software		
Test Level:	Component		
Test Type:	Performance		
Setup Conditions:	Tests 2.1.1 and 2.1.2 are successfully completed		
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Send a queue of 10 queries to each database while utilizing the mysqladmin tool to determine the approximate time for completion		Each database should complete all queries in less than one second or read 10,000 rows per second. Confirmed with the Glib timer utility
2	Send a queue of 10 random insertions and 10 random updates to each database while utilizing the mysqladmin tool to determine the approximate time for completion		Each database should complete all insertions and updates in less than one second
3	Delete the 10 random insertions from each database while utilizing the mysqladmin tool to determine the approximate time for completion		All entries should be deleted after 30 seconds

Table 10. MySQL Database Software Performance

[This space is intentionally left blank]

Test Category ID: 3		**Description:** Routing Simulation Creation Functionality	
Test Case: 3.1		**Purpose:** Verify the internal and external routing simulations correctly generate and load simulation assets	
Specification: 3.1.4.1		The routing simulations are generated from the user's input	
Specification: 3.1.4.2		Simulation data structures are easily accessed for verification of asset creation	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Test 2.2 is completed successfully	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Initiate the internal routing simulation		The user is prompted for several simulation parameters, and each parameter specifies its minimum and maximum bounds
2	Input parameter values		The program indicates both the internal and external simulations have been successfully created
3	Request the program print all simulated objects and parameters to a log file		All objects have been successfully loaded with respect to the user's input, and randomly generated objects are different from previous simulations

Table 11. Routing Simulation Creation Functionality

[This space is intentionally left blank]

Test Category ID: 3	**Description:** Routing Simulation Functionality
Test Case: 3.2	**Purpose:** Verify the internal and external routing algorithms correctly route bags through each routing simulation
Specification: 3.1.3.1	Each bag moves through an internal routing simulation in accordance with its owner's itinerary
Specification: 3.1.3.2	Each bag moves through an external routing simulation in accordance with its owner's itinerary
Test Level:	Component
Test Type:	Functional
Setup Conditions:	Test 3.1 is successfully completed and get needed itinerary information from airport database

Test Case Activity		**Pass/Fail**	**Expected Result**
1	Generate several non-alert inducing paths for a small set of bags within the internal routing simulation		Each bag arrives at its destination loading bay in line with its owner's itinerary, and the result is visualized
2	Generate non-alert inducing states for a small set of bags within the cart module of the external routing simulation		Each bag arrives at the correct belt loader in line with its owner's itinerary, and the result is visualized
3	Generate non-alert inducing states for a small set of bags within the belt loader module of the external routing simulation		Each bag arrives at its destination aircraft in line with its owner's itinerary, and the result is visualized

Table 12. Routing Simulation Functionality

[This space is intentionally left blank]

Test Category ID: 4		**Description:** Alert Generator-to-Database Interface Functionality	
Test Case: 4.1		**Purpose:** Verify insertion of alerts into AIR Tracker database is successful	
Specification: 3.1.4.3		Alert information is inserted into the AIR Tracker database	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Connectivity to AIR Tracker database returns 'open' status	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Attempt to insert an entry into the AIR Tracker database and output the result to a file		Output to text record indicates alert insertion is successful

Table 13. Alert Generator-to-Database interface Functionality

Test Category ID: 4		**Description:** AIR Tracker Alert Generator Functionality (Internal)	
Test Case: 4.2.1		**Purpose:** Determine if AIR Tracker's alert generation algorithm works correctly with respect to a simulation of internal baggage routing (e.g. baggage routing along a pusher)	
Specification: 3.1.4.3.1		A bag is correctly or incorrectly routed throughout the internal structure of the simulated airport according to routing simulation parameters	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Internal routing simulation (AIR Tracker application on main computer) is running; Test 3.1 is successfully completed	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Generate a trajectory in line with the bag's expected path along the pusher		No alert is triggered or inserted into the AIR Tracker database, and no alert is visualized
2	Generate a trajectory out of line with the bag's expected path along the pusher		An alert is triggered and inserted into the AIR Tracker database, and an alert is visualized

Table 14. AIR Tracker Alert Generator Functionality (Internal)

Test Category ID: 4	**Description:** AIR Tracker Alert Generator Functionality (External)
Test Case: 4.2.2	**Purpose:** Determine if AIR Tracker's alert generation algorithm works correctly with respect to a simulation of external baggage routing
Specification: 3.1.4.3.2	A bag is correctly or incorrectly routed throughout the external structure of the simulated airport according to routing simulation parameters
Test Level:	Component
Test Type:	Functional
Setup Conditions:	External routing simulation (AIR Tracker application on main computer) is running; Test 3.1 is successfully completed

Test Case Activity		**Pass/Fail**	**Expected Result**
1	Instruct the simulation to "stall" a bag, leaving it in the loading bay		An alert is generated, stored in the AIR Tracker database, and sent to the Handheld Alert System GUI
2	Instruct the simulation to place a bag on a luggage cart		No alert is generated or stored in the AIR Tracker database
3	Instruct the simulation to remove or "drop" a bag from a luggage cart		An alert is generated, stored in the AIR Tracker database, and sent to the Handheld Alert System GUI
4	Instruct the simulation to place the bag onto a belt loader		No alert is generated or stored in the AIR Tracker database
5	Instruct the simulation to "stall" a bag, leaving it on the luggage cart		An alert is generated, stored in the AIR Tracker database, and sent to the Handheld Alert System GUI
6	Instruct the simulation to place the bag onto an incorrect belt loader		An alert is generated, stored in the AIR Tracker database, and sent to the Handheld Alert System GUI

Table 15. AIR Tracker Alert Generator Functionality (External)

Test Category ID: 5		**Description:** Alert Generator-to-Handheld Alert System Interface Functionality	
Test Case: 5.1		**Purpose:** Verify an alert is capable of being sent by AIR Tracker and received by the Handheld Alert System GUI	
Specification: 3.1.5		AIR Tracker sends information about triggered alerts to the Handheld Alert System where it is then visualized	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Tests 4.2.1 and 4.2.2 are successfully completed. AIR Tracker application is running on the main computer	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Send a generic alert to the Handheld Alert System		A generic alert is visualized, and the alert is logged

Table 16. Alert Generator-to-handheld Alert System Interface Functionality

Test Category ID: 5		**Description:** Handheld Alert System GUI Functionality	
Test Case: 5.2		**Purpose:** Verify the Handheld Alert System GUI is capable of clearing alerts manually and automatically	
Specification: 3.1.5.1		The Handheld Alert System GUI displays alerts on a simulated handheld device and provides alert clearing functionality	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Test 5.1 is successfully completed. AIR Tracker application is running on the main computer	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Select the manual clear option		The alert is no longer active or visualized
2	Do not select a clear option and force the simulation to recover the bag		The alert is automatically cleared after being recovered, and the alert is no longer visualized
3	Do not select a clear option and force the simulation to ignore the alert		The alert is automatically cleared after several seconds, sent to the master baggage handler, and no longer visualized

Table 17. Handheld Alert System GUI Functionality

Test Category ID: 6		**Description:** Historical Reports of Alerts Functionality	
Test Case: 6.1		**Purpose:** Verify the alert reporting feature is operable	
Specification: 3.1.5.2		The Master Baggage Handler is able to view a history of alerts and save selective histories as reports	
Test Level:		Component	
Test Type:		Functional	
Setup Conditions:		Test 5.2 is successfully completed. Database PHP interface is running	
Test Case Activity		**Pass/Fail**	**Expected Result**
1	Select a date and time range and alert type		A history of alerts using the selected parameters is visualized

Table 18. Historical Reports of Alerts Functionality

6 Traceability to Requirements

Each test case can be traced to a functional or performance requirement. The traceability matrix shown in Table 6 correlates the requirements to its associated test cases. All of the requirements have at least one 'X' in their row. This shows that the test cases completely cover and demonstrate all the requirements.

[This space is intentionally left blank]

Requirements		Test Cases													
Component	**Req ID**	1.1	1.2	2.1.1	2.1.2	2.2	3.1	3.2	4.1	4.2.1	4.2.2	5.1	5.2	5.3	
RuBee H/W Setup	3.1.1.1	**X**													
RuBee S/W Setup	3.1.1.2	**X**													
Rubee Reader Processing Performance	3.2.1		**X**												
Database Functionality	3.1.2.1			**X**											
	3.1.2.2			**X**	**X**										
MySQL DB Software Performance	3.2.2					**X**									
Routing Simulation Creation	3.1.4.1						**X**								
	3.1.4.2						**X**								
Routing Simulation Functionality	3.1.3.1							**X**							
	3.1.3.2							**X**							
Alert Generator to DB Interface	3.1.4.3								**X**						
AIR Tracker Alert Generator (Internal)	3.1.4.3.1									**X**					
AIR Tracker Alert Generator (External)	3.1.4.3.2										**X**				
Alert Generator to Handheld Alert System Interface	3.1.5											**X**			
Handheld Alert System GUI	3.1.5.1												**X**		
Historical Reports of Alerts	3.1.5.2													**X**	

Table 19. Traceability matrix for the AIR Tracker prototype