Usability Evaluation of an Industrial Infrastructure Security Command and Control Interface

Brian Bobo
Iowa State University
bjbobo@iastate.edu

Shaun Broyhill
Iowa State University
broyhill@iastate.edu

Colleen Dolphin
Iowa State University
dolphin@iastate.edu

Carl Thunberg
Iowa State University
thunberg@iastate.edu

SUMMARY

We conducted a usability evaluation of an industrial infrastructure security command and control interface. Our evaluation included expert reviews and live user testing.

INTRODUCTION

Falcon Reach is a software product that has an operational concept to provide a security surveillance situational awareness interface for a large industrial site. The system includes a wide array of sensors, cameras, and devices to provide full support for a command center. We conducted a usability study of a beta version of the software that excludes any hardware sensors.

The focus of this usability study is to provide a qualitative evaluation versus a quantitative evaluation. Since the end user group for this product will contain a small user base, it would not have been cost effective to test using a large participant group. Effective qualitative results were achieved through careful participant recruitment and investing in wide task coverage [Lindgaard and Chattratichart 2007].

Our usability study focused on one primary research question: What improvements can be made to the Falcon Reach software interface that will allow the end user to more effectively interact with the system?

SAMPLE POPULATION

The Falcon Reach application has a very specific set of goals. It is not intended for the general population and was created for a discrete purpose, namely to provide command and control of a security infrastructure for a large industrial facility located in a non-secure environment. The main rule regarding test users is that they should be as representative as possible of the intended users of the system [Nielsen 1993]. The criteria which were applied to the participant selection included their experience with similar applications, their availability, and their voluntarism.

Since the Falcon Reach application is a new design, it was not possible to recruit participants who fit the exact mold of an intended user. Instead it was decided to select previous users of similar software applications. The Falcon Reach interface can be viewed as "mission software," which can be defined as a human-machine interface which employs elements of a system intended to perform a military-type mission. A perfect example would be the operator console and software suite installed upon an aircraft, which is intended to manipulate the intelligence cameras and radios used to communicate with ground parties (not air traffic control). Therefore, experience with mission software was a discriminator for participants.

Schedule and resource issues often drive the execution of usability evaluations in industry [Lazar et. al. 2010]. Such was the case for the Falcon Reach user test. No funding was available to compensate the participants, so they needed to volunteer their time. Out of a pool of seven qualified participants, only three were willing to support the effort without a "charge number" to put on their timecard. In addition to being willing to work on their own time, the availability of personnel was also a limiting factor in the number of participants

selected. Only two of the three volunteers were able to participate during the abbreviated calendar window required for the evaluation.

METHODOLOGY

Our evaluation of the Falcon Reach interface was accomplished in a segmented fashion, in order to combine both formative (task-level) and summative (problem discovery) testing methods [Lewis 2006]. Two evaluation approaches were chosen to provide the maximum amount of constructive feedback for the designers. The first was a heuristic inspection conducted by two of the team members. The second was a usability evaluation conducted by a third team member, using conventional task performance to assess usability, to include a questionnaire to rate participant satisfaction with the interface design.

Heuristic Evaluation

Two team members conducted heuristic evaluations, since the number of evaluators directly correlates to the number usability problems likely to be found. The evaluations were conducted separately, so that the probability of a given evaluator identifying a specific problem would remain independent of whether the other evaluator identified the same issue [Nielsen and Landauer 1993]. Independent evaluations also ensure that neither evaluation is biased by the other.

The team members were able to conduct their evaluations remotely using Falcon Reach screen shots provided by our on-site team member. The on-site team member also provided audio and video captures of himself walking through specific task scenarios.

As a basis for our evaluations, we used the heuristics originally proposed by Nielsen and Molich [1990], and later further refined by Nielsen. These heuristics are fully described in Appendix I. After each independent evaluation was completed, the evaluations were merged, and an impact rating was then assigned to each discovered interface problem.

We used a task-completion impact schema to prioritize usability issues as either high, medium, or low-impact, as follows:

- *High-impact:* Would significantly impact the user's ability to complete a task.
- *Medium-impact:* Would slow down or confuse the user, but might not completely prevent task completion.
- *Low-impact*: Might annoy the user, but would not present an obstacle to task completion.

As expected, the independent heuristic evaluations identified many similar interface issues, as well as some dissimilar issues. Several of these same issues were later corroborated by users during live testing.

Usability Testing

A usability evaluation of the Falcon Reach interface was the initial intent for the research project and required a significant amount of time and effort. Usability testing requires a lot of advance planning in order to glean meaningful data which can then be presented to the designers for their consideration toward incorporation [Lazar et. al. 2010]. In order to follow a concrete plan based upon previous research, the following stages of usability testing were followed [Rubin and Chisnell 2008]:

1. Develop the test plan
2. Set up the test environment
3. Find and select participants
4. Prepare test material
5. Conduct the test sessions
6. Debrief the participants
7. Analyze data and observations
8. Report findings and recommendations

Test Plan Development

The development of the Falcon Reach test plan was conducted in the overarching context of a project plan, which was created to present to the project's sponsor for approval. The purpose was briefly stated as to provide feedback to the developers who were working on the software's first version. Project objectives were identified as improved usability, design, and end-user satisfaction. A fourth objective was to bolster the operational concept for the designers, who had very little insight into the overall objectives for the Falcon Reach system. Assumptions which applied to the project were enumerated, which included being granted access to the interface developers to set up the test environment and to provide context. Constraints such as having access to a small participant pool were also documented. The project plan also described the boundaries for the research, the summary milestones, and the projected

resource requirements. Requirements and tasks were identified, as well as the individual responsibilities of the team members. Having a plan in place, the team turned its focus to the test environment.

Fortunately a mock-up of a Falcon Reach control center was being used by the software development team, and access was granted to the principal investigator for the usability evaluation.

Testing Environment

The centerpiece of the test setup was a laptop which connected to the Falcon Reach application on a separate server. The application consisted of four windows (Appendix II):

- A video representation of what would be displayed from an electro-optical/infrared turret-based camera that in real life would be mounted on an aerostat.
- A map application which displayed "own location" as well as various points of interest.
- A Points window which listed points of interest along with their details.
- An Alerts window which provided two purposes: an information-only chat methodology as well as a means to create alerts for all subscribers to the system.

Participant Selection

As mentioned previously, access to appropriate usability test participants was very limited. According to Nielsen, the main rule regarding test users is that they should be as representative as possible of the intended users of the system [Nielsen 1993]. The principal investigator for the usability evaluation inquired within his work environment for individuals who had experience with "mission" software, were available during the two week time frame according to the project plan, and were willing volunteers. Two participants met these criteria and were chosen.

Test Material

Test material was then created in anticipation of the actual task evaluations which began with the creation of a thorough prebrief to be delivered at the beginning of the test session. A script was created to reduce any variation between participant performances. The prebrief consisted of many elements. An overall description of the project's purpose was provided. Next, a graphic which depicted the Concept of Operations (CONOPS) behind the Falcon Reach system was presented. It was made clear to the participant that it was the interface which was under evaluation, not the participant's individual performance. Elements of disclosure commonly found in an Institutional Review Board (IRB) consent form were shared, and the participant was asked to verbally approve the use of their voice on the audio-visual recording. The overall methodology for the evaluation was discussed, followed by a brief description of the Falcon Reach interface to include the human-machine interface (HMI) input devices (laptop, mouse, and hand controller for simulated camera panning and zooming). The prebrief also explained the usage of the "Think Aloud Protocol" [Lazar et. al. 2010], stressing that it was encouraged to be employed during task execution, but not at the expense of expeditious task completion.

The task list was also prepared, which stemmed from three operational scenarios which were created as a subset of the overarching Falcon Reach CONOPS. Cue cards were also prepared to be read by the participant prior to their recording: "Participant 1—Set 1," "Participant 1—Set 2," etc.

Prior to an actual test session with a participant, a pilot run was made with one of the developers to ensure that the tasks as written were sufficiently straight-forward so that minimal interaction would be required from the investigator during task performance. During execution there were some aspects of task scenarios which confused the pilot participant, resulting in refinement of the task cards created for the actual test subjects.

User Test Sessions

After a participant was prebriefed they were asked if they had any questions, which were answered. A screen capture program was employed to not only capture the laptop's screen, but also to record the participant's voice using a headset. Once the participant was ready, the recording was begun, followed by their reading of the appropriate cue card. They were also instructed to read the title of each scenario: "Scenario 1: Set My Location on Map and Follow Aircraft, "Scenario 2: Create a Point at Target, Mark as Suspect, Move," etc. Each scenario card was handed to the participant one at a time, and once they had completed the card the last task line had them

record "End Scenario." Both participants were comfortable "thinking aloud" during task performance. In the event that they were not, the investigator had a cue card with the words "Please Think Aloud" printed on it.

The investigator remained a silent observer during the execution of tasks and only provided assistance when the participant was clearly "stumped" and was unable to move forward with task performance and had become frustrated.

After the first set of three scenarios were completed, the participant was given a 15 minute break which was followed by providing them 15 minutes of "free play" with the interface to explore its capabilities. After this period of time, the same exact set of three scenarios was executed.

At the conclusion of the task execution sequences, the participant was provided with a questionnaire. The designers also had an opportunity to briefly discuss with the participant their impressions after working with the interface.

User Questionnaire

Appendix III shows the usability evaluation participant questionnaire used in this study. A color-coded analysis of participant responses to the questionnaire can be found in Appendix IV.

There were two participants, User 1 and User 2, who performed a usability test on the Falcon Reach software. When they were done completing all of the tasks, they were asked to fill out a questionnaire. Both participants scored their computer experience as high, and that they had over 200 hours of mission software experience. User 1 had some past experience with a similar software application, whereas User 2 did not. Throughout the rest of the questionnaire, participants used a Likert scale ranging from "Strongly Agree", "Agree", "Undecided", "Disagree", and "Strongly Disagree" to express how successful they perceived their task completion. Out of 23 questions, there were eight that the participants scored the same—all of those scores were rated "Agree". There were five questions that were only differentiated by "Strongly Agree" and "Agree" between the two users.

User 1 only had three "Disagree" answers and one "Undecided". He had issues with displaying only "suspect" targets on the map and finding it easier to create a point through the point list compared to the map window. User 1 was undecided as to whether or not the Falcon Reach software was intuitive to use from the start.

User 1 commented that the problems he had setting "My Location" were his own fault—that he overlooked the button and he thought it was clearly labeled. He mentioned that the "suspect" affiliation was unfamiliar and that "Show Marked" was not an intuitive label for "Show Suspect." He also said that when creating alerts, pulling a geographic position from the map was a different sequence than creating an alert from the map (red button). He also found inconsistencies between entering a point coordinate on the map versus the point window. The point window did not offer options to pull from the map, which was different from creating an alert dialog, which does. And finally, when trying to move a point his instinct was to look for a drag and drop once the point was selected. He did not notice the option in the right-click menu, so he thought the only way to move was manual text entry.

User 2, who had less experience with the software, said he agreed that the software was intuitive to use from the start, but rated more things as problematic. For example, User 2 had five "Undecided" responses, two "Strongly Disagree" responses, and one "Disagree" response. Like User 1, he also had issues with designating the target point as "suspect" and displaying only "suspect" targets on map. He was also unsure about how adding a map label was easy to do; if he found the pop-up hint useful when hovering over a button; and if it was easier to create a point through the point list compared to the map window. He disagreed with the clarity of the fields required when creating a point, and his greatest challenge was updating the alert message and the priority of the alert.

User 2 commented that there were inconsistencies between editing points and alerts. You can edit a point, but not an alert—one of his main difficulties with the software. He also wanted to provide an edit feature as a right-click pull down option, and he did not think the fields that were required for alerts were very clear.

ANALYSIS

To analyze the recordings of each user, several methods were used. The first test included recording

the amount of time required to perform tasks by each user. However, this proved to be quite difficult through the review of the screen capture and audio files. One issue included determining accurate start and stop points for each task. For most tasks, the users effectively vocalized a start point for the task, but that was not the case for all tasks. Also, when the user vocalized the start point by reading the task, two different scenarios occurred: either they would wait until finalizing the reading of the task header to start the task, or they would start the task while reading the task.

A second issue that occurred when attempting to record the timing of each task was that it was difficult to determine when the participant had successfully completed the task. Occasionally it would appear as if the participant would complete a task, yet be unaware of whether the task was actually complete or not. Continued mouse movement and verbal cues would indicate that the user was unaware of whether the task was complete even though it actually was.

Therefore, since it was difficult to establish a set start or end point for tasks, it was determined that this data would provide little benefit due to its inherent inaccuracy. In the future, usability study protocols should be developed to establish specified start and stop points, and timing should be performed as the tasks are completed versus attempting to establish timing after the screen capture has occurred.

The next procedure used to analyze the screen captures was to identify specific tasks or areas that appeared to create difficulty for both test users. There were two primary areas that increased the difficulty for users: the previously mentioned unawareness of task or subtask completion, and the alert system.

When looking again at the situational awareness of the user being incapable of identifying whether the task was complete or not, a count was performed on the specific number of times this was identified. For the first user, it appeared as if this occurred on four separate occasions, whereas for the second user it appeared to occur six times. These observations were not based on overall task completion alone, but also included subtasks within the primary tasks.

The main point at which users appeared to be unaware of task completion was when icons on the map would change colors to indicate completion. Since the icons were relatively small in relation to the overall screen space, it would be easy to overlook the changes in icon color. At several points the user would comment verbally about whether a task was complete or not. Typically this occurred at the point of one these icon changes.

The second area that presented the most difficulty for users involved the alert tasks. Several limitations of the software application were presented, which included the inability to elevate or add additional data to alerts. Users consistently associated alerts with other items in the application, and expected to have the ability to modify the alert. This caused the task completion time and number of errors to increase significantly during the alert portion.

TEAM MECHANICS

Working as a group online with new classmates can pose many challenges. Throughout the semester, the Shneiderman group experienced issues that we were able to work through and resolve. This, in turn, improved our process of working together.

Our locations stretched between Minnesota, Colorado, Los Angeles, and Mexico City, so we needed to be mindful of finding a time to meet every week. We took into account the time change, work schedules, and other commitments and set up weekly group meetings. We met every Wednesday via teleconference to discuss our progress on projects and bring up any existing problems or questions.

Another challenge that we faced was that the software used for our final project was only available in Carl's location, so that's where the study took place. In order to provide the rest of the group with information and data to analyze, Carl used Camtasia to record the software in use. He first recorded himself using the software and guided us through several scenarios. In addition to the recordings, he also took screenshots of the software in different stages to use as another reference. For the usability test, Carl used Camtasia to record the participants completing each task, which also included their verbal feedback. Pictures of the participants in the work area were also provided so the rest of the group could see the setup.

Deciding how to best communicate with each other was also a concern. There were many options to choose from on Blackboard, so our group had to

decide which method was the most efficient way for us. Otherwise, messages would get lost if the group members did not check all of the various places on Blackboard.

When emailing through the Group area of Blackboard, the address list was hidden and prevented the "Reply All" option, so another message thread would have to be started, which could create confusion. Instead, we decided to email directly from our Iowa State email addresses to avoid this problem. We found that we could only add or edit content on the primary wiki pages versus the comments area, so we posted most of our discussions using the wiki. We also made use of the File Exchange on Blackboard to keep track of all project documents, rather than emailing. This helped keep our materials organized.

Throughout all of these challenges, the Shneiderman group worked very well together. We were able to overcome problems by communicating with each other and keeping the other members up-to-date. Each person was dedicated to bringing forth their best effort, and making sure the knowledge gained in class was implemented into our collaborative work.

FINDINGS AND CONCLUSION

We needed to answer two key questions: What improvements can be made to the Falcon Reach software interface that will allow the end user to more effectively interact with the system, and whether this usability study effectively provided an answer to this question.

We demonstrated that since this study was a qualitative versus quantitative study, it was extremely important to create effective task scenarios, heuristic inspections, and usability evaluations. The combination of these three items proved effective in helping to answer the primary question for this paper. The heuristic inspection and screen capture data did demonstrate several key areas for improvement that can be passed on to the developers. These include investigating the limitations of the alert system, as well as establishing better defined task completion awareness.

This analysis was compared to the user evaluation forms that helped rate the user's satisfaction of the Falcon Reach project. The evaluation verified many of the difficulties that were apparent in the screen capture analysis, and provided further validation of the need to make this application capable of higher efficiency.

Therefore, it is clear that this usability study did effectively answer the core question of providing insight into improvements that would prove to increase the efficiency and effectiveness of interacting with the Falcon Reach software.

In conducting this usability study, we also were provided insight into further methods and protocols that should be included in future studies on the Falcon Reach project should the evaluation team continue to have the geographic difficulties that persisted in this usability study. These include the availability of software and markers for timekeeping analysis.

REFERENCES

Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hocheiser. 2010. *Research Methods in Human-Computer Interaction.* Wiley, Chichester, West Sussex, UK.

James R. Lewis. 2006. Sample Sizes for Usability Tests: Mostly Math, Not Magic. *Interactions* 13, 6, 29-33.

Gitte Lindgaard and Jarinee Chattratichart. 2007. Usability Testing: What Have We Overlooked? *Proc. ACM CHI'07 Conf.* (San Jose, CA, 28 April-3 May), 1415-1424.

Jakob Nielsen and Rolf Molich. 1990. Heuristic Evaluation of User Interfaces. *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1-5 April), 249-256.

Jakob Nielsen. 1993. *Usability Engineering.* Academic Press, Cambridge, MA.

Jakob Nielsen and Thomas K. Landauer. 1993. A Mathematical Model of the Finding of Usability Problems. *Proc. INTERCHI'93 Conf.* (Amsterdam, NL, 24-29 April), 206-213.

Jeffrey Rubin and Dana Chisnell. 2008. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests.* Wiley, Indianapolis, IN.

Appendix I: Nielsen's Usability Heuristics

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

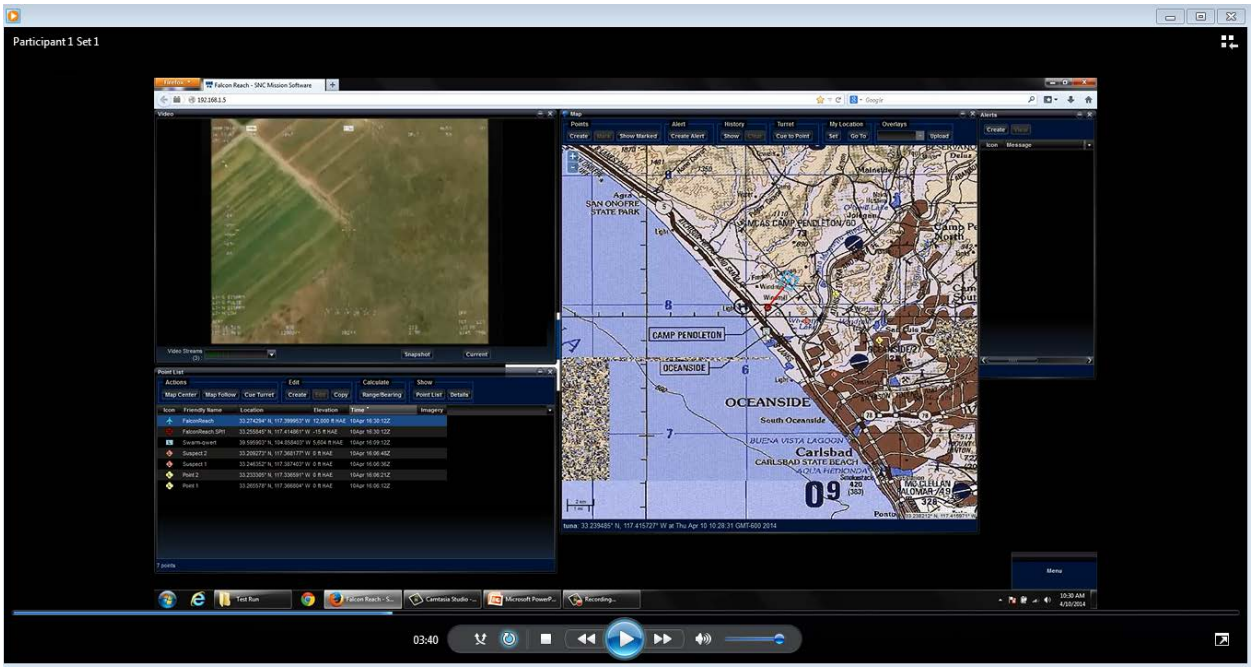
Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Appendix II: Falcon Reach Four Windows for Test Setup



Appendix III: Falcon Reach Usability Evaluation Participant Questionnaire

FALCON REACH HMI USABILITY EVALUATION PARTICIPANT QUESTIONNAIRE

1. Computer experience (circle one): LOW MEDIUM HIGH
2. Hours of mission software experience (circle one): NONE 0-50 51-100 100-200 >200
3. Do you have previous experience with this version of software or it's pedigree (circle one)? YES NO

   If yes, explain:

Now that you have used the Falcon Reach software, please place a check to indicate the rating that best represents your opinion about completing the tasks listed below.	Strongly Agree	Agree	Undecided	Disagree	Strongly Disagree
1. I had no problem setting a point for "My Location" on the map.	❒	❒	❒	❒	❒
2. Centering the map on "My Location" was easy to do.	❒	❒	❒	❒	❒
3. I could easily find and view the details for "My Location."	❒	❒	❒	❒	❒
4. After identifying and zooming in on a target in the Sensor Video, I was able to easily create a point at that target location on the map.	❒	❒	❒	❒	❒
5. Designating the target point as "suspect" was easy to do.	❒	❒	❒	❒	❒
6. I could easily find out to display only "suspect" targets on the map.	❒	❒	❒	❒	❒
7. Switching between a hand controller and mouse did not take long to do.	❒	❒	❒	❒	❒
8. It is clear which tasks require a hand controller and which ones require a mouse.	❒	❒	❒	❒	❒
9. I had no problem creating an "Alert" and entering the alert message.	❒	❒	❒	❒	❒
10. It was easy to locate the "Alert" I created on the map.	❒	❒	❒	❒	❒
11. I could easily update the alert message.	❒	❒	❒	❒	❒
12. I could easily update the priority of the alert.	❒	❒	❒	❒	❒
13. Adding a map label was easy to do.	❒	❒	❒	❒	❒
14. When hovering over a button, I found the pop-up hint useful.	❒	❒	❒	❒	❒
15. The icons placed on the map were easy to see on top of the map graphic.	❒	❒	❒	❒	❒
16. I found the icon shapes helpful and easy to identify.	❒	❒	❒	❒	❒
17. It was clear what fields were required when creating a point, alert, etc.	❒	❒	❒	❒	❒
18. When setting "My Location" on the map, I knew the difference between the GPS and Map Click options.	❒	❒	❒	❒	❒
19. When creating a point, I know how to enter in the longitude and latitude manually, if necessary.	❒	❒	❒	❒	❒
20. I like having the option of creating a point through the point list and through the map window.	❒	❒	❒	❒	❒
21. It is easier to create a point through the point list compared to the map window.	❒	❒	❒	❒	❒
23. The Falcon Reach software was intuitive to use from the start.	❒	❒	❒	❒	❒
24. The Falcon Reach software was easy to learn when exploring it on my own.	❒	❒	❒	❒	❒

Appendix IV: Falcon Reach Usability Evaluation Questionnaire Analysis

	CODE
1	Strongly Agree
2	Agree
3	Undecided
4	Disagree
5	Strongly Disagree

	Section1	Participant 1	Participant 2
Q1	Computer Experience [Low/Medium/High]	High	High
Q2	Hours of mission software experience	>200	>200
Q3	Do you have previous experience with this version of software or it's pedigree? If yes, explain.	Yes: contributed significantly during first year of Burma software development	No

	Section 2	Participant 1	Participant 2
Q1	I had no problem setting a point for "My Location" on the map.	4	1
Q2	Centering the map on "My Location" was easy to do.	2	1
Q3	I could easily find and view the details for "My Location."	2	1
Q4	After identifying and zooming in on a target in the Sensor Video, I was able to easily create a point at that target location on map.	2	2
Q5	Designating the target point as "suspect" was easy to do.	2	3
Q6	I could easily find how to display only "suspect" targets on map.	4	3
Q7	Switching between a hand controller and mouse did not take long.	2	2
Q8	It is clear which tasks require a hand controller and which ones require a mouse	2	2
Q9	I had no problem creating an "Alert" and entering the alert message	2	2
Q10	It was easy to locate the "Alert" I created on the map.	2	2
Q11	I could easily update the alert message.	2	5
Q12	I could easily update the priority of the alert.	2	5
Q13	Adding a map label was easy to do.	2	3
Q14	When hovering over a button, I found the pop-up hint useful.	2	3
Q15	The icons placed on the map were easy to see on top of the map graphic.	1	2
Q16	I found the icon shapes helpful and easy to identify.	1	2
Q17	It was clear what fields were required when creating a point, alert, etc.	2	4
Q18	When setting "My Location" on the map, I knew the difference between the GPS and Map Click options.	2	1
Q19	When creating a point, I know how to enter in the longitude and latitude manually, if necessary.	2	2
Q20	I like having the option of creating a point through the point list and through the map window.	2	2
Q21	It is easier to create a point through the point list compared to the map window.	4	3
Q22	The Falcon Reach software was intuitive to use from the start.	3	2
Q23	The Falcon Reach software was easy to learn when exploring it on my own.	2	2
		Comments:	Comments:
		On scenario 1, Set My Location, my difficulty was my own fault. The button was clearly labeled and visible, I just overlooked it. "Suspect" affiliation was unfamiliar "Show Marked" was not intuitive label for "Show Suspect" On creating alerts, pulling geographic position from map was a different sequence than creating an alert from map (red button) The interface to enter a point coordinate is different between map and point window. Point window did not offer option to pull from map, which is different from create alert dialog, which does. For moving poin, my instinct was to look for a drag and drop once the poin was selected. I did not notice the option in the right-click menu, so I thought the only way to move was manual text entry.	There were inconsistencies between editing points and alerts. You can edit a point but not an alert. It would be greatly beneficial to be able to edit an alert to provide additional information as the alert changes status. Provide edit feature as a right click pull down option Not very clear on what fields were required for alerts.