NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DATA VISUALIZATION OF ISR AND C2 ASSETS ACROSS MULTIPLE DOMAINS FOR BATTLESPACE AWARENESS

by

Brian T. Real
James E. Ellis Jr.

September 2009

Thesis Advisor: John Osmundson
Thesis Co-Advisor: Daniel Dolk

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 2009	Master's Thesis

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Data Visualization of ISR and C2 Assets Across Multiple Domains for Battlespace Awareness	
6. AUTHOR(S) Brian T. Real, USMC and James E. Ellis Jr., USN	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited	

13. ABSTRACT (maximum 200 words)

In this thesis, we have developed a prototype application that is capable of providing ISR situational awareness to C2 nodes at the Joint Task Force (JTF) level and below. The prototype application is also capable of providing information that will allow joint intelligence planners to plan ISR operations more efficiently, including allocation of intelligence-gathering platforms and sensors, and processing, exploitation, and dissemination (PED) assets to information requests.

14. SUBJECT TERMS	15. NUMBER OF PAGES
ISR Situational Awareness, Information Request	85
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

DATA VISUALIZATION OF ISR AND C2 ASSETS ACROSS MULTIPLE DOMAINS FOR BATTLESPACE AWARENESS

Brian T. Real
Captain, United States Marine Corp.
B.S., Park University, 2002

James E. Ellis Jr.
Lieutenant Commander, United States Navy
B.S., University of North Florida, 1994

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 2009

Authors: Brian T. Real

James E. Ellis Jr.

Approved by: John Osmundson
Thesis Advisor

Daniel Dolk
Co-Advisor

Dan C. Boger
Chairman, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In this thesis, we have developed a prototype application that is capable of providing ISR situational awareness to C2 nodes at the Joint Task Force (JTF) level and below. The prototype application is intended to increase the JTF's level of visibility on the information request process related ISR activities. The application also demonstrates the capability of providing information that will allow joint intelligence planners to plan ISR operations more efficiently, including allocation of intelligence-gathering platforms and sensors, and processing, exploitation, and dissemination (PED) assets to information requests.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

We thank our families and friends for their unconditional support and understanding they provided during the preparation of this thesis.

Additionally we thank the faculty and staff of the Naval Postgraduate School, in particular, Dr. John Osmundson and Dr. Daniel Dolk, who supported us as thesis advisors.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The draft Distributed Common Ground/Surface Systems (DCGS) Concept of Operations [1] states:

> The warfighter's ability to maintain situation awareness of ISR (intelligence, surveillance and reconnaissance) operations is a significant factor in his ability to exercise C2.

The document further states:

> DCGS and warfighter C2 systems have been developed independently and are frequently incompatible. As a result, synchronization of ISR and operations planning, and real-time support to operational events is lacking. DCGS must improve product and system interfaces with C2 structures and procedures.

In this thesis, we have developed a prototype application that is capable of providing ISR situational awareness to C2 nodes at the Joint Task Force (JTF) level and below. The prototype application is also capable of providing information that will allow joint intelligence planners to more efficiently plan ISR operations, including allocation of intelligence-gathering platforms and sensors, and processing, exploitation and dissemination (PED) assets to information requests.

An example ISR scenario has been analyzed, and 15 events have been identified in ISR. The process starts with the submission of an information request (IR), through the approval cycle, to allocation to a collection asset, allocation to a processing, exploitation and dissemination (PED) node, to delivery to the original requestor. The 15

events are points at which information must be gathered to provide ISR situational awareness.

The prototype application has been architected and the software designed to replicate the ISR process, as well as the DCGS physical and functional architecture. Multiple access databases have been created to provide 15 record sets that correspond to the 15 points at which information needs to be captured. Each record set is populated with data representative of the data that would be available from DCGS systems. Each record set is then converted into individual XML documents, which are merged into combined XML documents. The combined documents can be parsed, for example, into data items that relate to IR status or ISR asset status. The parsed data is converted into XHTML format, made available on the Web, and displayed to the operational user. Varieties of alternative displays have been developed and are discussed in the thesis.

This prototype application strongly suggests that the problem of interfacing DCGS and C2 nodes and ISR nodes can be solved with a relatively simple application. Before embarking on development of an operational application, however, some further research needs to be done, to include:

- Analysis of whether data is, or can be operationally captured, at each of the 15 event points in DCGS;
- Analysis of whether captured data can be operationally converted into the proper XML format at each of the event points;
- Analysis of whether system security requirements can be met with the prototype architecture and design;

- Analysis of the scalability, reliability, and maintainability of a system based on the prototype architecture and design.

THIS PAGE INTENTIONALLY LEFT BLANK

I. PROBLEM DEFINITION

A. DCGS BACKGROUND

The Department of Defense is continually looking for methods of developing the Distributed Common Ground/Surface Systems (DCGS) Intelligence, Surveillance and Reconnaissance (ISR) tasking, processing, exploitation and management process (TPED). An architectural framework is under construction, and interoperability standards and new technologies are being studied. The goal is to determine how best to support commanders and war-fighters by offering common services or structures to help synchronize, discover, visualize, and coordinate ISR-related efforts. This thrust is an attempt to connect the war-fighter to the right information at the right time, and allow for the self-synchronization of ISR-related efforts by improving visibility of processes and actions of heterogeneous units.

DCGS seeks to provide applications and services that allow these heterogeneous units visibility of each other's actions and information, fostering the development of a collaborative atmosphere. The expectation is that this collaborative atmosphere will allow units to operate more efficiently, since any unit can see what other units have done with regard to specific ISR-related activities. By sharing information, units can better coordinate to avoid double-tasking missions already being undertaken by other units.

This thesis addresses the development of a prototype application intended for use by units at the Joint Task

Force (JTF) level and below. The prototype application is intended to illustrate a system that can increase the JTF's overall level of visibility on the information request process and related ISR activities. Development of the prototype application will help further determine requirements and issues concerning ISR situational awareness at the JTF level.

Additional benefits of the prototype are the facilitation of more efficient planning of ISR operations by joint intelligence planners, including the allocation of intelligence-gathering platforms and sensors, and processing, exploitation and dissemination (PED) assets to information requests.

Currently, commanders at the JTF level and below submit information requests and receive information products that fulfill those requests. There is little to no visibility on the process in between. Having a visualization of the process can help the commander predict when the information product will be available and improve his timeline for related decisions. Additionally, commanders often share areas of influence and can request information about similar areas of interest. Currently, one commander cannot see what other commanders are requesting, which results in each individual request being fulfilled and double-tasking assets for similar missions. By providing a collaborative environment and allowing commanders to visualize each other's actions, units with similar goals can self-synchronize action and allow for the more efficient use of available assets.

B. PROPOSED SOLUTION

In this thesis, we provide a method that develops an XML-based application using a 15-step conceptual framework that is based on an information request use case scenario. We cover the development of a simple schema, the application of that schema in collecting events at necessary information points, and the fusion of the information collected to provide relevant reports that provide the visualization of the process. The reports display the steps in the IR process and capture the steps in a history, providing a quick visualization of the request's progress. Aggregated reports provide visualization of the current progress of multiple requests. The aggregation of all requests provides a visualization of how and where the system is under the most strain. ISR managers and commanders can use the points of contact displayed in the reports and visualizations that provide a basis for self-synchronizing actions and collaboration. Although the focus of this work is data visualization, the addition of metrics at the information points could be used by ISR managers as a decision support tool when tracking information requests and IR product fulfillment. The prototype visualization tool can show an ISR manager the current step of an information request in the product development phase. The addition of metrics that show the average time the producing node takes to create the product, or which node in a series has the expertise to produce the product more efficiently, could move this visualization tool to the level of a data-driven decision support tool.

C. METHODOLOGY

This system was developed using a spiral incremental method of which this is the completion of the first increment. Interviews were conducted with personnel from the Joint Systems Baseline Assessment office to elucidate initial requirements. A use case scenario was used to develop the process and system methodology. This system is envisioned to be a part of a larger ISR Situational Awareness application.

D. SCOPE

The original scope of this thesis was to determine the requirements of a fully functional ISR Situation Awareness system. Once development had started the problem was found to be larger than anticipated so the scope was narrowed to provide a working prototype of a related activity, the Information Request process. Reports have been developed for tracking individual requests and related activities as well as aggregate reports for units and system wide activity. This is not a fully functional operational system but a visualization tool so all security related issues are not covered comprehensively. During development, certain data control issues were encountered and are discussed later under the section concerning XSLT Server Side Transformations.

E. DOCUMENT STRUCTURE

Chapter II of this document covers the basic composition of the JTF and factors that detract from the JTF’s ability to develop a collaborative environment. The

initial system requirements are also discussed and compared to the requirements of a federated database management system. In Chapter III, we discuss the selected use case and the 15-step process derived from the use case. In this chapter, we also discuss the prototype’s architecture and software design considerations. In Chapter IV, certain important code snippets are explained and the visualization reports are described in detail. In Chapter V, there is a discussion on Joint Intelligence process management and alternative design views are presented. Chapter VI finishes with conclusions, recommendations, and future and related work.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND INFORMATION

In this chapter background information on the composition of a Joint Task Force are presented, the Information Request process is discussed and comparisons to a Federated Database Management System are examined to determine initial system requirements. The intention is to provide a reader who is not a domain expert the information needed to better understand the issues discussed in the remaining chapters.

A. JOINT TASK FORCE DEFINITION

A Joint Task Force is a temporary activity that is established or developed to accomplish certain operational objectives, military operations, or to support a specific situation. When the purpose for the JTF has been achieved, or when the joint task force is no longer required, it is dissolved by the commander or the authorizing official who constituted its inception. A Joint Task Force can be established by a Combatant, an establishing authority such as the Secretary of Defense, a Subordinate Unified Commander, or the commander of an already established Joint Task Force. A Joint Task Force may be established by geographic area or functional basis. The Joint Task Force will normally be assigned a Joint Operational Area. The JTF may be comprised of many components or service functions. Figure 1 depicts possible configurations of the JTF [3].

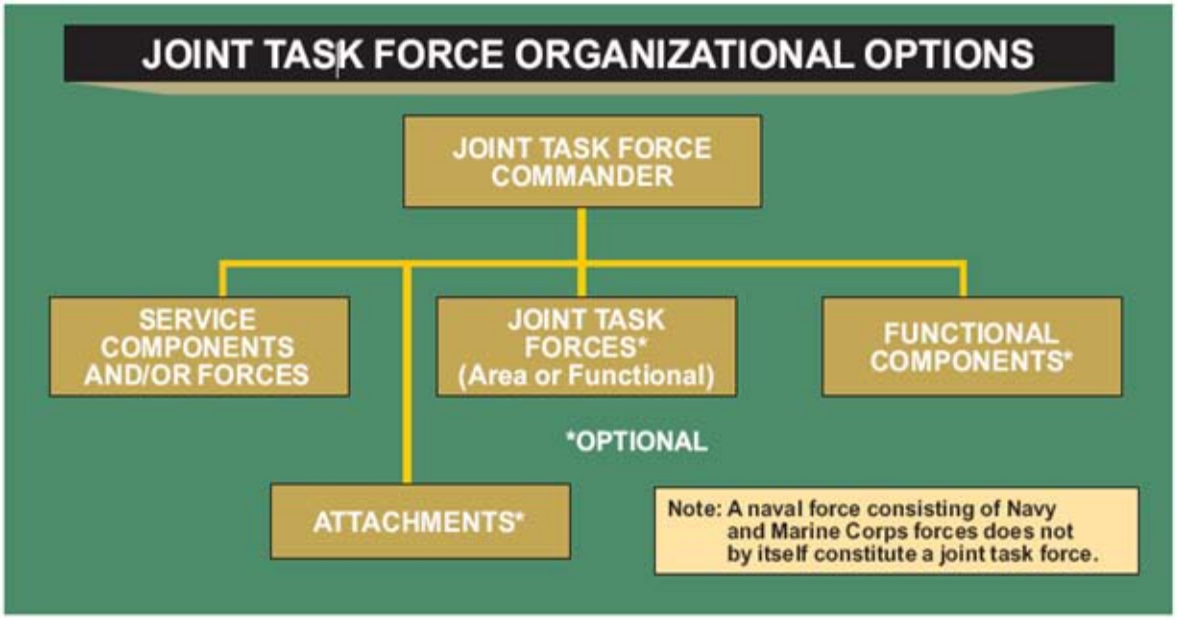


Figure 1. Joint Task Force Organizational Options. (From [3])

From this description, we can see that a JTF may consist of many options. Some JTF's may be larger than others; again, the components and size will reflect the scope of the assigned mission. In a coalition environment, a JTF may contain components from all of the US service components as well as forces from foreign militaries. The Commander Joint Task Force (CJTF) usually organizes the JTF based on his Concept of Operations (CONOPS). Since JTF's are designed and instituted to accomplish an assigned mission, any tool needed to support information requirements' visualization and tracking must have the ability to be rapidly deployable, and highly scalable, and partition-able in order to control data access.

B. THE ISR-RELATED ACTIVITY FOR STUDY

One ISR-related activity that is a sufficient candidate for study in a prototype application is the information request/intelligence product development process. An application that tracks and monitors this process would be useful in any JTF scenario, regardless of assigned mission. Interviews with personnel from the Joint Systems Baseline Assessment office reveal the lack of an established ability within JTF units to visualize the efforts of other units in the informational request process [4].

This lack of visualization results in a lack of synchronization of ISR efforts, which, in turn, contributes to an unnecessarily high level of uncertainty under which adjacent commanders must make decisions. Currently, a commander will submit an information request and receive an intelligence product that addresses the question(s) presented to some level of detail. It is assumed that the commander requires the information in order to make a decision, perform some action or decrease some amount of uncertainty. Thus, it would be useful to the commander to have the ability to track the progress of the request in order to be able to determine or project when the information product will be available for use. It would also be useful to JTF planners and mid-level unit commanders to track the aggregate totals of information requests and their location within the fulfillment process. Also, data visualization of these activities could assist in the process of managing assets related to the information request process.

C. PATHOLOGIES

The current lack of visualization can be attributed directly to the composition and nature of the JTF itself. As stated above, a Joint Task Force, by its definition, is a temporary organization that is task organized to accomplish a specific mission. A JTF is composed of various organizations, each with its own organic assets and systems for operation. They are task organized for the mission by the guidance of the JTF commander. The composition of the process to submit and fulfill information requests within a JTF will be influenced by two immediate factors that are not fully predictable: the units that compose the Joint Task Force and the direction of the Joint Task Force Commander on the composition and responsibilities of those units [5]. Therefore, a process to submit and fulfill requests will emerge, but how it will emerge and the attendant responsibilities of the participants cannot be fully predicted. Adding to the complexity is the fact that most military information systems in use today were developed as service-centric entities whose data are not fully compatible or interchangeable with the same types of information systems developed by other services for similar purposes. A familiar scenario is the formation of a JTF whose data formats become dictated by the service that brings the most assets to the organization. Any application that supports the information request process should be flexible enough to allow commanders to dictate any important information requirements. Alternatively, it should be supported by a schema simple enough to gather information that would be

useful to any JTF, while remaining easily accomplished by the various units supporting the effort.

D. INITIAL REQUIREMENTS

After reviewing the initial background information, we can identify parallels between the JTF situation and a federated database management approach. The application must take information contained in multiple information stores and combine it for the complete fulfillment of system queries of related information. Under a Federated DBMS scenario, a variety of large and small databases is used by several sections that, overall, comprise all the information of the one entity. In both situations, we are faced with a distributed database (or information stores) scenario. As such, our application will have similar initial requirements to those of a Federated Database Management System (DBMS). The main requirements of a Federated DBMS are listed below, and each is discussed individually.

Federated DBMS requirements:

"1. The user should be able to access a number of heterogeneous databases as if accessing a single database" [6]. This requirement has to do with distributed transparency. The user should be able to retrieve all relevant information from all of the entities' data stores as though they were accessing a local data store. For the user, the experience should be transparent to the process under which the application operates under normal operating circumstances.

2. “The user should be able to access any database using a familiar data model and language” [5]. In order for the application to be quickly and easily adapted, the application should be easy for the user to operate. The user should not be burdened with the logic needed to access and retrieve the information from the heterogeneous information stores.

3. “Federated DBMS should not require any significant changes to existing database systems or applications” [5]. The JTF will most likely be composed of different units, each with their own service centric organic assets. Since the JTF is temporary in nature, it would be beneficial to allow these units to continue using their own organic assets as much as possible.

4. “The system should accommodate the addition of new databases to the network” [5]. The fluid nature of the JTF supports this requirement. Mission focus can be expanded or narrowed depending on the situation. An application architecture must be fluid to allow for the addition and deletion of information stores as the composition of the JTF changes.

5. “The user should be able to access the databases for both retrieval and updates” [5]. This is an area where our study does not agree with the Federated DBMS scenario. Applications do not need the ability to change the original data stores. We are specifically interested in the information contained within these stores. Data can be visualized through predetermined queries and drill down menus, without having to allow direct access to all users to

the original information stores or databases. This will be demonstrated later in the prototype.

6. “Performance of Federated DBMS should be comparable to that of homogeneous distributed systems” [5]. This requirement is in agreement with the needs of ISR situational awareness. The application should operate in real time or near real time in order to provide the most relevant information that other units can see and on which self-synchronization efforts can be based.

Other issues:

7. Other issues that must be considered are “global concurrency control, global deadlock handling and global semantic integrity enforcement” [5]. Since the project involves combining data from multiple sources, there is a need to ensure that only one version of the data source is used to develop the application’s displays and reports. This is required to ensure consistency and allow accurate synchronization of efforts based on the displays and reports produced. Since we are not focusing on allowing user access to change the original data stores, the issue of global concurrency control is less important than that of data quality. We instead focus on the accuracy of the data being supplied to the application, and apply a mechanism to allow the individual user to judge data accuracy. Global deadlock handling is another potential application issue. Deadlocks occur when one program is waiting to complete a transaction based on the actions or locks placed on a data item by the actions of another program. If the programs are co-dependent, a deadlock can occur as each waits for the other to release its locks. When dealing with this issue, two

things must occur: The deadlock must be detectable and a procedure must be invoked to disrupt the deadlock. In our application, the ability to update the sources supplying the data will not exist. Any issues related to deadlock handling can be avoided by allowing only the fusion and viewing of data, rather than the ability to update the original databases. The data supplied to the application will have to be designed according to a global schema to ensure correct formatting. To help ensure that the information supplied to the application is semantically correct, a mechanism will need to be developed to check and ensure that the data being supplied meets the requirements of a supplied schema.

Now that we have reviewed the problem and some background information on DCGS, we next generate a use case scenario depicting the information request process. We also cover the process of fulfilling the request and describe some the prototype’s architectural considerations.

III. USE CASE AND PROTOTYPE ARCHITECTURE

In this chapter, we describe a subject use case, derive the IR process from the use case, develop a prototype architecture and discuss selected important design considerations. The use case involved in this study was taken from a joint intelligence exercise conducted on 22 May 2006. It is an example of the information request process that was taken from the OV-6c——an Operation Event Trace diagram——of a facility seizure exercise. The event trace diagram is presented in Figure 2 [6]. The diagram in Figure 2 provides a visual description of the operation and the steps involved, from the request generation to the information product's delivery. The various unit entities that must interact are listed horizontally across the top of the diagram under the thread description. The time sequence of the unit interactions are listed vertically, from top to bottom, along the left side of Figure 2. Various actions are depicted as arrows throughout the middle of the diagram.

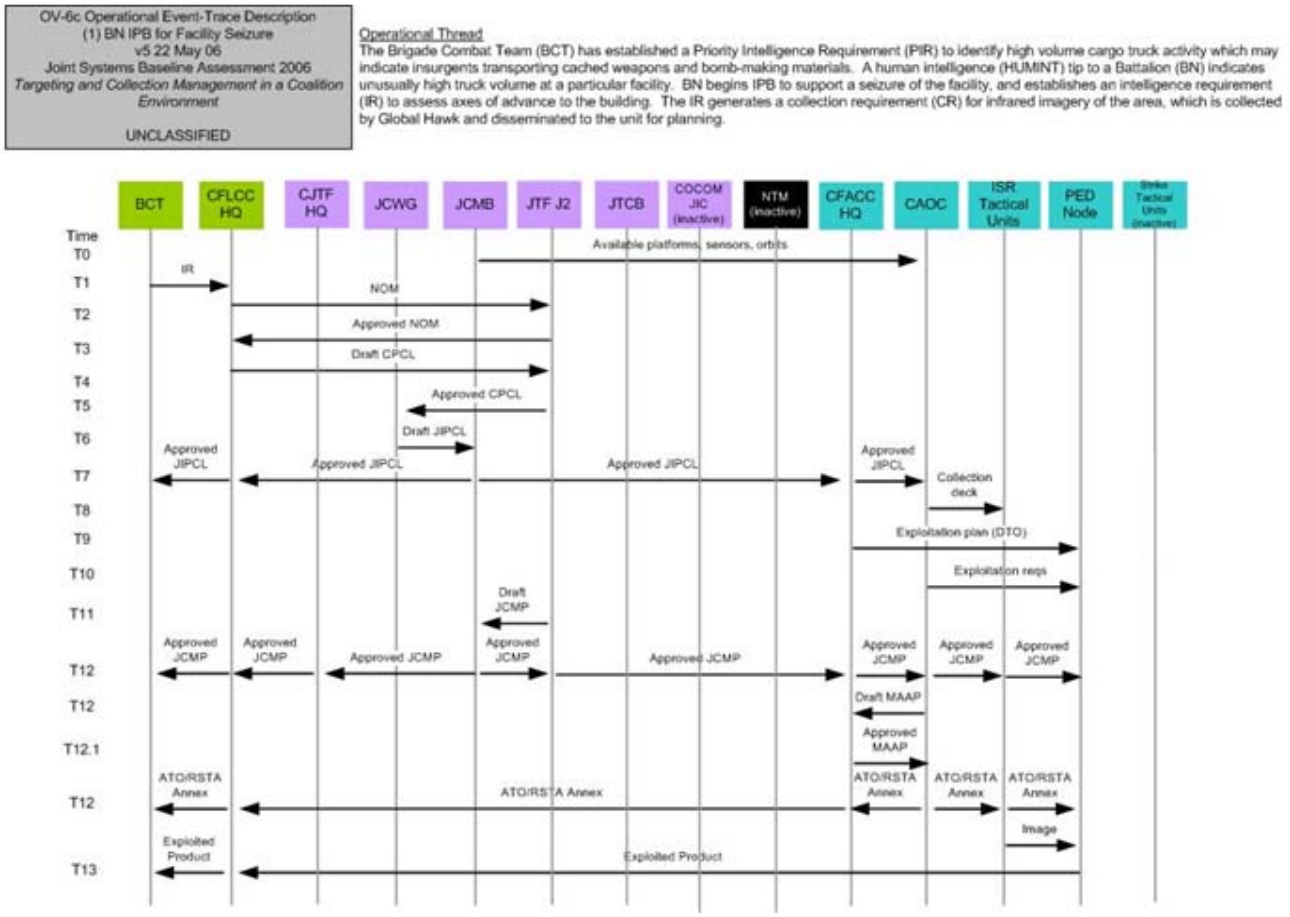


Figure 2. OV-6c Operational Event Trace Description (From [6])

A. SCENARIO DESCRIPTION

Scenario Description: The Brigade Combat Team (BCT) has established a Priority Intelligence Requirement (PIR) to identify high-volume cargo truck activity that may indicate insurgents transporting cached weapons and bomb-making materials. A human intelligence (HUMINT) tip to a Battalion indicates unusually high truck volume at a particular facility. BN begins IPB to support a seizure of the facility and establishes an intelligence requirement (IR) to assess axes of advance to the building. The IR generates a

collection requirement (CR) for infrared imagery of the area, which is collected by Global Hawk and disseminated to the unit for planning.

This use case offers a good scenario for study as it demonstrates all the steps in the process from the beginning, with the generation of the Information Request, to the end, with the delivery of the product, which satisfies the original request.

B. PROCESS DEFINED FROM DIAGRAM

From the study of this use case diagram, we can generate a list of the steps involved in the information request process. The steps represent the actions that need to be taken when an information request is generated to produce an intelligence product. These steps also translate into points where information should be captured. If we link all the events through a common thread that traces a particular document number, then we begin to visualize the history of actions performed and any future actions required to satisfy the information request.

In the context of developing this prototype, 15 actions have been identified as steps in the information request process. These 15 steps can be categorized into three sub-processes, which are listed below:

C. IR REQUEST TO INTELLIGENCE PRODUCT DEVELOPMENT PROCESS

Sub Process 1 IR Approval:

1. IR Creation
2. IR Submission
3. IR Approval

4. IR Validation

5. Determination of an IR Deadline

15. Information product approval by the original requestor

Sub Process 2 IR Collections:

6. Prioritization of the IR

7. Development of the collection plan

8. Assignment of the IR to a collection mission

9. Assignment of a collection mission to a collection asset

10. Execution of the collection mission

11. Transfer of the collected data relevant to the IR

Sub Process 3 IR Processing, Exploitation and Dissemination:

12. Transferred data is processed into a useable form

13. Processed data is exploited by an analyst

14. Information product is developed to answer the IR

By recording simple events at each step, the details of the actions taken at that step can be captured. The aggregation of these simple events provides a history of the information request, which can be used to visualize the status of the request. An aggregation of all the requests can provide data to the middle and upper echelon users who can visualize the status of all requests and the level of effort required at each defined area.

D. PROTOTYPE ARCHITECTURE

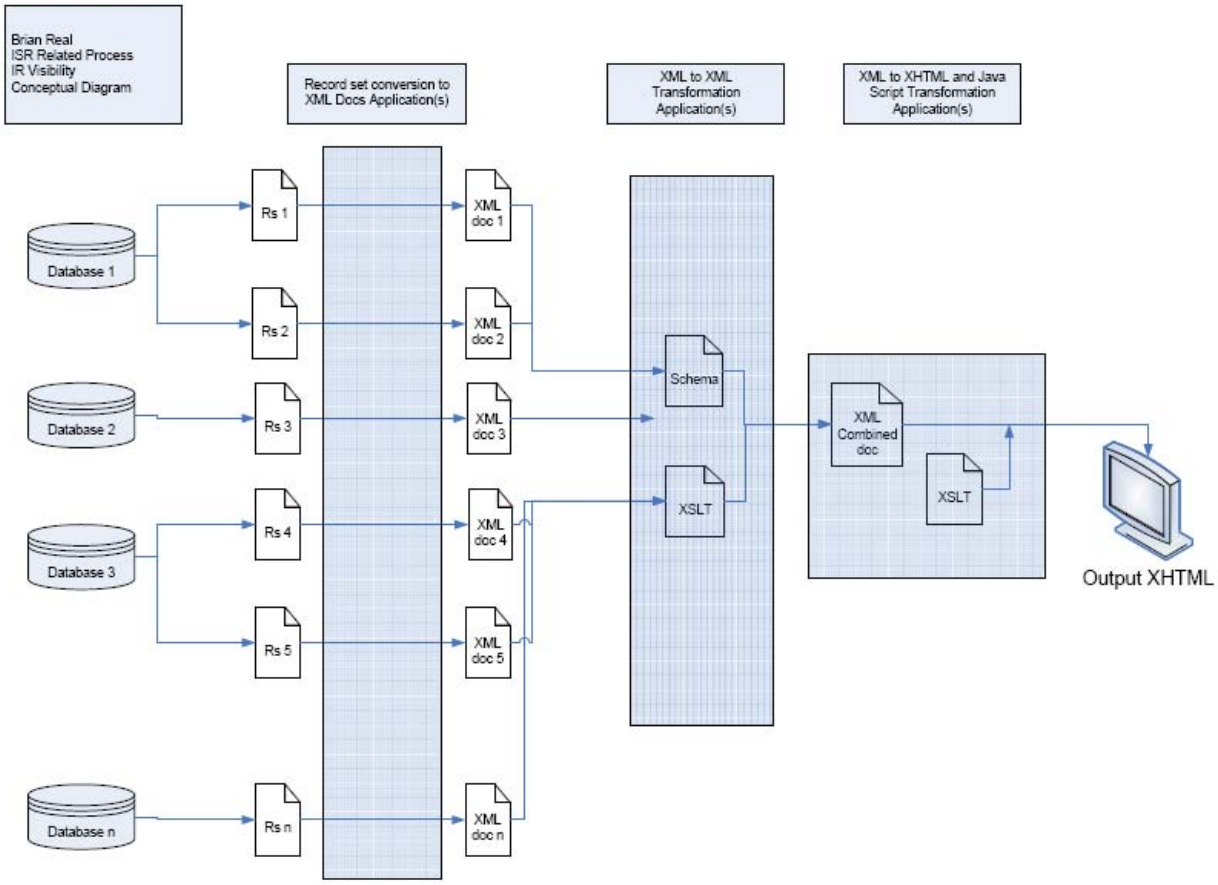


Figure 3. Conceptual Diagram

Figure 3 illustrates the prototype's architecture. Information stored in separate databases is collected through relevant, predetermined queries to form record sets of that data to be used within the application. The prototype has three separate databases from which fifteen queries are developed. These queries match the fifteen information points explained above. The record sets are converted to XML documents to become interoperable information feeds that are stored in a data repository. The separate feeds are combined into one data source XML

document and checked for proper formatting through the application of a an XML schema document. The combined and formatted data is also grouped and sorted for more efficient querying and retrieval. This data is then persisted in the data repository in the resulting combined document. The resulting combined document is transformed to provide reports and information visualization components for the three levels of displays for the end users. A top-level display depicts the overall activity of the system. A mid-level display is used for unit managers to track multiple information requests. The individual display provides visualization on a single information request and ISR activities related to that request. These activities include links to more in-depth information, which provides the user with a means to visualize status and links for contact information. This also provides the user with the means to connect with other personnel involved with the information request fulfillment process.

E. DESIGN CONSIDERATIONS

1. Software Languages

XML 1.0 is used to structure and store the data. XSD, the XML schema language, is used to ensure semantic data integrity in the application.

XSLT 1.0 and 2.0 are both used in the prototype application to transform XML documents into other formats. The prototype transforms XML, using XSLT to a re-formatted XML document when the separate files are combined into one source XML document, and transforms XML to XHTML to produce

the reports and displays for the three user levels. XSLT is also used to generate JavaScript to produce the mission in progress display.

XPath 1.0 and 2.0 are both used in support of XSLT operations. XPath is used to identify XML elements and attributes and perform functions during XSLT transformations.

Javascript and VBscript are both used to produce some user desktop functionality and displays.

Cascading Style Sheets 1.0 is used for page presentation in the Web application.

Access databases were used as the data stores from which the XML feeds are built.

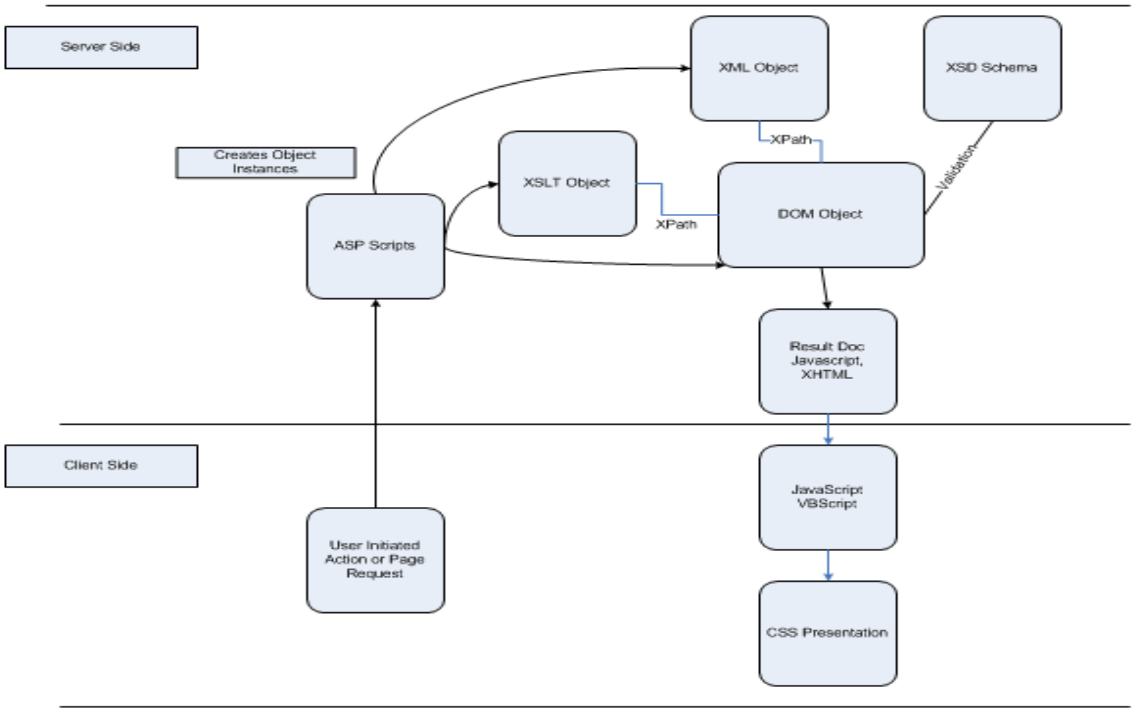


Figure 4. Software Component Interactions

2. Prototype Non-Functional Software Requirements

ASP 3.0 is used in the prototype for server side scripting due to the developer's experience in the language. Other server side scripting languages were not used or evaluated.

3. Server Side XSLT Transformations

In the prototype, XSLT transformations are used to convert data to other formats within the application, and all transformations take place using server side processing. These transformations can take place on the server or on the client, and each option has its own advantages. By processing XSLT transformations on the client, the user experiences a richer interface, like those of desktop applications, rather than a Web site. Client side transformations also allow the use of asynchronous JavaScript and XML (AJAX) applications, which can be updated with new data source information, when the XML file is changed, without having to reload the current page for the user.

Despite the advantages of client side transformations, server side transformations were chosen for the superior benefits they produce. The environment in which the application is envisioned to function is one of a wide variety of units, and possibly some Non-Governmental Organizations (NGOs), each with its own organic computer assets. By conducting transformations on the server, a cross browser solution is immediately achieved. As the XSLT transformations that produce the Web pages are executed, the

resulting code is produced in XHTML, which is readable in all current browsers. This process is depicted in Figure 5.

Although strict security issues were not studied, transformations conducted on the server allow for control of the data within the application. The transformation programs are passed filter parameters that filter out the data not needed for the client. The mid-level unit display reflects this principle, as only IRs belonging to the unit are displayed. Although not covered in this prototype, similar user levels can be developed, and data not appropriate for certain users can be filtered out to ensure access control of the data. If the transformations were to be conducted using client-side processing, all the data would have to be sent to the client, and a client-side application would be used to filter the displayed items on the client. Once all the data is sent to the client, control of the data could potentially be compromised by those who were not intended to have access to it. Finally, the transformations are conducted on the server due to the processing power required on some XML documents. It is assumed that desktops or laptops used in the various JTF units will not be of a consistent configuration across all the units comprising the JTF. It is also assumed that a server would provide transformed result pages faster and more consistently than user components. See also the discussion on DOM parsing.

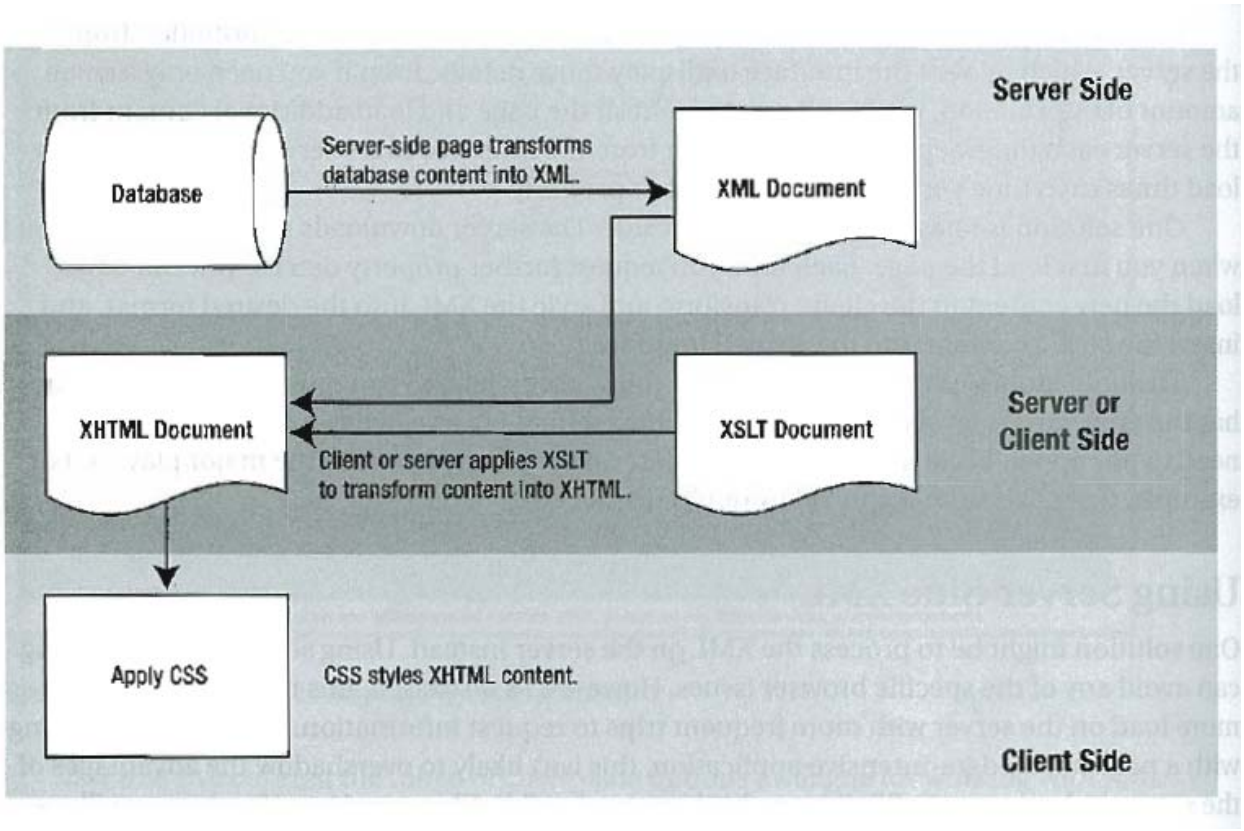


Figure 5. Depicting the XML to XHTML transformation Process (From [8])

4. Validating XML Using XML Schema Language

In order to control global semantic integrity, the prototype uses the XML schema language to validate the XML documents used in the application. XML documents are termed valid when they conform to the pre-defined structure dictated by the rules contained in either its assigned Document Type Definition (DTD) or Schema [9]. If a document is not valid, a parser will fail to process the document. Thus, invalid data is not processed into the reports and displays. The XML schema language offers the greatest flexibility and control over the rules that control what constitutes valid structures in the XML documents.

DTDs offer very limited data typing support [10]. In DTDs, XML elements and attributes mainly consist of Parsed Character Data (PCDATA) or simply Character Data strings (CDATA), which lacks the control over the data that can be achieved with the XML Schema language. The XML Schema Language allows for the specification of data types acceptable in the XML document structure. By using data types along with patterns and regular expressions [11], [12], much greater control is achieved, and data that are more specific can be structured into the application. The "Event_Contact" element definition in the Combined Events.XSD schema located in Appendix A offers a good example of the control that can be established over allowable data. A snippet of the schema is contained in Figure 6. Our prototype will only accept e-mail addresses in this field with a .mil extension. In order for the "Event_Contact" element to be deemed valid, an @ sign must be detected and a .mil extension must be present. Our schema also allows only uppercase letters, lowercase letters or numbers in the e-mail user name and mail group extension. This type of control is not possible using DTDs.

```
- <xs:element name="Event_Contact">
  - <xs:simpleType>
    - <xs:restriction base="xs:string">
        <xs:pattern value="[0-9a-zA-z]+@[0-9a-zA-z]+.mil" />
        <!-- this means something@something.mil  -->
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
```

Figure 6. Snippet of the schema

F. DOCUMENT OBJECT MODEL PARSING

Parsing refers to the processing of an XML document. There are currently two main categories of XML processing. The Document Object Model (DOM) uses tree-based processing, and the Simple API for XML (SAX) uses event-based processing [13]. The prototype application uses DOM level 2 and the MSXML 4.0 and above processing. MSXML 4.0 was released in 2001, and this is the oldest version that will allow support for the XML Schema language. The choice of using the DOM model was based on the flexibility and ease of implementation. The DOM was developed by the World Wide Web Consortium (W3C). This organization is responsible for helping institute standards for Web programming languages through their Request for Comment (RFC) proposals. Although not enforceable, their RFC proposals have become the de-facto standards for the industry.

The DOM builds a result tree in memory based on the structure of the XML document. This tree structure is then manipulated by the API to traverse all the nodes in the result tree to perform functions and retrieve values. The prototype application makes use of this feature by testing child nodes for conditions and, if the conditions are met, the parent node is selected for some action. This action is not possible using the Simple API for XML (SAX). The disadvantage of using the DOM is the large memory representation needed to use the DOM functionality. Nodes cannot be manipulated in the DOM until the whole document has been read into memory and the result tree is built in

memory. If a user has a large XML document and is looking for one small piece of information, this approach becomes inefficient.

SAX was developed to address this problem. SAX operates on event-based parameters. It searches through the document until the event is reached. Once nodes have been passed, they are not saved in memory, so parent nodes cannot be easily retrieved once a child node that meets the event parameters has been reached. SAX has been compared to watching a train pass a location. As the railcars pass by, they are lost and the stream cannot be reversed [14]. SAX also requires the installation of a separate processor to run the SAX API while support for the DOM has been built in to most servers and browsers. SAX is a more efficient way of retrieving small bits of information in a large XML document, but the DOM allows for more flexible use and retrieval of all the nodes in the document.

G. CONVERSION OF RECORD SETS TO XML

The prototype accomplishes the conversion of record sets to XML format through use a custom-built Active Server Page (ASP) script. The custom script was built to ease the process of transitioning the data to XML format and the storing of the data into a file repository. The custom script also allows for the addition of data quality indicators as data attributes. A representation of some of the results from the custom script is listed in Figure 6. Another method to covert information from an Access database to XML format is through the built-in functionality offered through the Access program [15]. The results of the built-in Access conversion to XML are listed in Figure 7.

The data elements displayed in the two figures will be explained in the next section. Although these screenshots were taken at different times and the data contained varies slightly in content, the structure in both examples is similar, with some subtle but important differences. The first difference to notice is the root elements. In the Access conversion example the root element is called <dataroot> and references the Microsoft Office Data Schema. This schema is constructed upon the data found within the table from which this Query was taken, whereas in the custom solution, an external schema can be specified. If invalid data existed in the database, the Access solution would build the schema to the invalid data, thus making it valid in that specific XML document. Our solution provides the ability to perform external control on the data allowed to enter the system. We also only need to change one schema to change the data requirements of all documents entering the system. We have no ability to change the name of the elements when exporting through the Microsoft solution. Each row in the Microsoft solution has been tagged as <IRCreationXMLTable> while, in the custom-built ASP solution, we changed the element name to <Event>. As part of our attempt to provide for the global concurrency control requirement, we want to add a time element documenting when the data was collected for each row. The Access solution automatically adds this time element, but only to the root element (dataroot). Since we will be later combining, grouping and sorting data from many sources, the original documents will be transformed into new documents. Thus, we need to add our time element to each row element so that each element will have an indication of data quality on its

own. In the custom-built solution, we can see the <Event> element has been given the attribute "collected" with a value of the server time when the query was converted. The custom-built ASP solution also has the ability to persist the converted query as an XML file automatically to the repository, whereas this would have to be done manually under the Access/Microsoft solution.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Documents xsi:noNamespaceSchemaLocation="http://www.brianreal.com/XML Holder/XSD Files/Combined_Events_Schema.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  - <Event collected="8/4/2009 2:58:48 PM">
      <Event_Reference>AA7001</Event_Reference>
      <Event_Label>Document Creation</Event_Label>
      <Event_Status>Completed</Event_Status>
      <Event_Date>2009-08-01</Event_Date>
      <Event_Time>07:00:00</Event_Time>
      <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
      <Event_Comments />
      <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
    </Event>
  - <Event collected="8/4/2009 2:58:48 PM">
      <Event_Reference>AA7002</Event_Reference>
      <Event_Label>Document Creation</Event_Label>
      <Event_Status>Completed</Event_Status>
      <Event_Date>2009-08-01</Event_Date>
      <Event_Time>07:10:00</Event_Time>
      <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
      <Event_Comments />
      <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
    </Event>
```

Figure 7. Data Quality Indicators as Data Attributes

```
<?xml version="1.0" encoding="UTF-8" ?>
- <dataroot xmlns:od="urn:schemas-microsoft-com:officedata" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="IRCreationXMLTable.xsd" generated="2009-08-08T11:34:23">
  - <IRCreationXMLTable>
      <Event_Reference>AA7001</Event_Reference>
      <Event_Label>Document Creation</Event_Label>
      <Event_Status>Completed</Event_Status>
      <Event_Date>2009-08-01</Event_Date>
      <Event_Time>07:00:00</Event_Time>
      <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
      <Event_Comments>Creation Comments would be here</Event_Comments>
      <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
    </IRCreationXMLTable>
  - <IRCreationXMLTable>
      <Event_Reference>AA7002</Event_Reference>
      <Event_Label>Document Creation</Event_Label>
      <Event_Status>Completed</Event_Status>
      <Event_Date>2009-08-01</Event_Date>
      <Event_Time>07:10:00</Event_Time>
      <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
      <Event_Comments>Creation Comments would be here</Event_Comments>
      <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
    </IRCreationXMLTable>
```

Figure 8. Access conversion to XML

In this section, an information request process use case was described, a process was derived from the use, and the prototype architecture and selected design considerations were discussed. In the next section, selected operational elements of the prototype and the application’s report configurations are discussed.

IV. SELECTED CODE AND REPORT EXPLANATIONS

In this section, we will cover some selected underlying code and explain how it operates to produce the reports used for visualization of ISR-related processes. Ultimately, we seek to present a global view of all the actions necessary to visualize the progress of one or more IR documents. We start with the basic building block of the system, the event, and then move through some selected code snippets of other system components. After that, the report displays will be described and explained in a walk-through fashion. What we are trying to accomplish is to tie together all the actions that must occur to produce an intelligence product when an information request is submitted. The actions that occur will be referred to as *Events* in the remainder of the chapter. We tie all these events together by using a code or a document number that remains unique throughout the system. Once all the events of a single document number have been linked, we use the information to visualize a history of the document. The aggregation of all the document histories in a unit provides a mid-level or unit-level view of all the information requests belonging to that unit. The aggregation of the document histories of all the units that comprise the JTF is used to provide the top-level view or a visualization of the IR documents active in the system.

A. THE EVENT ELEMENT

The event element is the basic building block, which captures basic information about the actions that have

occurred or still need to occur in fulfillment of an IR. In our example, we identified fifteen information points where we need to capture information on these actions. These actions translate into events: A unit creates an IR, an IR is proven to be a valid request that supports operations, or a collection mission is in progress that captures data required to fulfill the IR, and so on for each of the fifteen information points. It is assumed that information relating to these events can be captured by the operational intelligence personnel performing the work. The prototype uses fifteen of these record sets from three databases to simulate data repositories that may already exist. Figure 9 is an example of the basic information captured, which comprises a record set with the top record AA7001 underlined. Figure 10 is an example of an XML Event element that comprises an information feed developed from the same record, AA7001.

IRPrioritizedXMLTable

Event_Refer	Event_Label	Event_Statu	Event_Date	Event_Time	Event_Contact	Event_Comments
AA7001	Document Prioritized	Completed	2009-08-05	11:00:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7002	Document Prioritized	Completed	2009-08-05	11:10:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7003	Document Prioritized	Completed	2009-08-05	11:20:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7004	Document Prioritized	Completed	2009-08-05	11:30:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7005	Document Prioritized	Completed	2009-08-05	11:40:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7006	Document Prioritized	In Progress	2009-08-06	11:00:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7007	Document Prioritized	In Progress	2009-08-06	11:10:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
AA7008	Document Prioritized	In Progress	2009-08-06	11:20:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
BB7001	Document Prioritized	Completed	2009-08-05	11:00:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
BB7002	Document Prioritized	Completed	2009-08-05	11:10:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here
BB7003	Document Prioritized	Completed	2009-08-05	11:20:00	Prioritizer@S2Prioritizer.mil	S2 Prioritizer Comments here

Figure 9. An Example Record Set

```
- <Event collected="8/7/2009 2:48:15 AM">
    <Event_Reference>AA7001</Event_Reference>
    <Event_Label>Document Prioritized</Event_Label>
    <Event_Status>Completed</Event_Status>
    <Event_Date>2009-08-05</Event_Date>
    <Event_Time>11:00:00</Event_Time>
    <Event_Contact>Prioritizer@S2Prioritizer.mil</Event_Contact>
    <Event_Comments>S2 Prioritizer Comments here</Event_Comments>
    <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
  </Event>
```

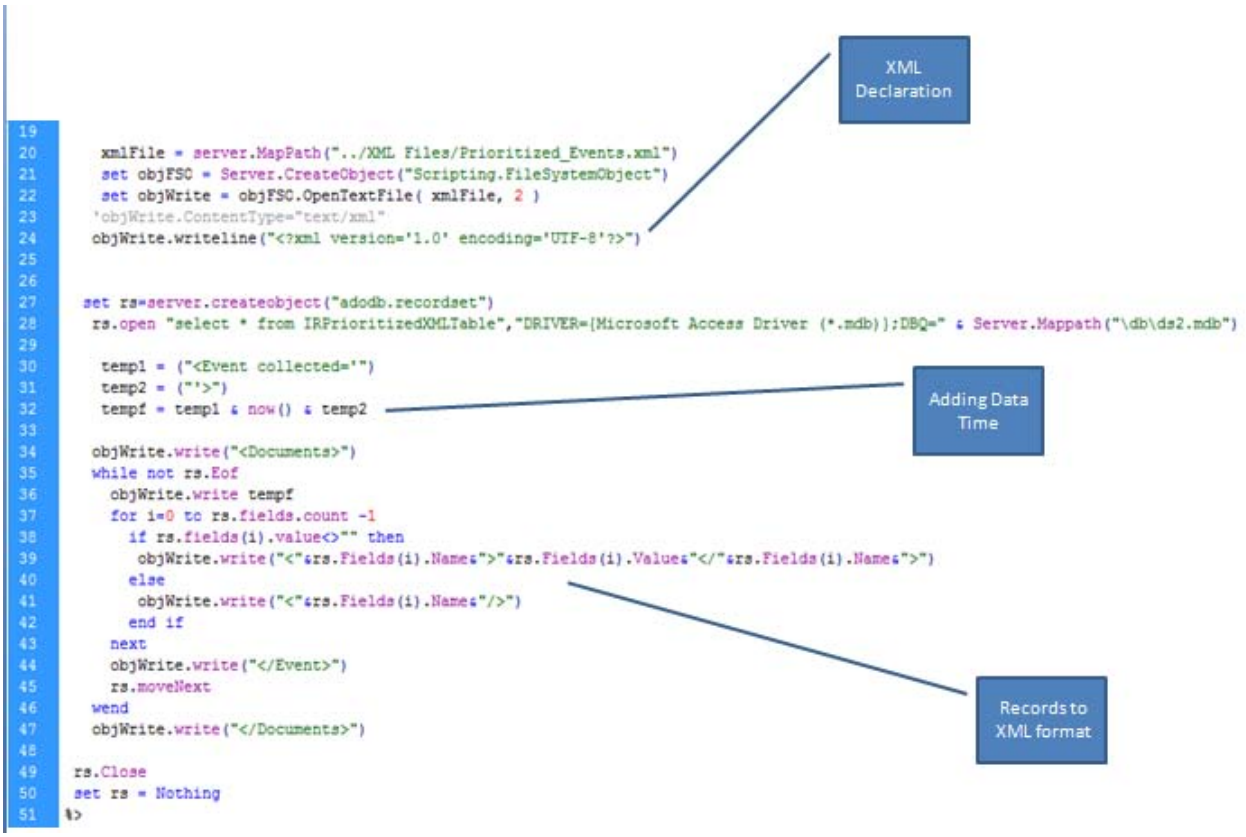
Figure 10. Record AA7001 converted to XML

This feed is built from the record set and simply tells us the status of the document at this information point, the time and date the action occurred, who conducted the action, their contact information, and any comments that were entered concerning the event. There is also a link included as a place mark to allow a means to retrieve more information on the event. In our example event, the JTF S-2 has completed prioritizing this information request. The Event_Info_Link could be used as a place mark to direct the user to the Joint Integrated Prioritized List to see where the request was placed on the list.

B. CONVERSION OF THE RECORD SET INTO XML INFORMATION FEED DOCUMENTS

The conversion of the record set into an XML information feed is achieved through the use of Visual Basic on Active Server Script (ASP) page. The source code to perform this program can be found in Appendix A under the RS to XML section. This program accomplishes three things. First, it conducts the query at the information point, then it adds a quality indicator, and finally, it writes the information to a file in the tagged XML markup format. The

program makes use of the ASP server Scripting method of the File System Object to make the information persistent. In order for this program to work, specific read/write access must be given to the destination file's properties. Figure 11 is a code snippet from the ASP script and shows how the information is written into well-formed XML. On line 24, the script writes the first line of the new XML document as the XML declaration. To capture the quality indicator as an attribute, a series of temporary variables are used in conjunction with the server's Now() function. This will insert the current server time, which the user can view in the reports to provide an indication of how current the data is being displayed. While there are still files in the record set, the program continues to loop through and convert each record, or row, into XML format by copying each Field or the column's name and adding the required angle brackets to create tags. The column's value is placed in between the created tags. If there is no value in the Field (column), the script creates an empty tag. The result is a file that meets the criteria of a well-formed XML document. As the file is created, it is written to the document repository.




```
<%
   xmlFile = server.MapPath("../XML Files/Prioritized_Events.xml")
   set objFSO = Server.CreateObject("Scripting.FileSystemObject")
   set objWrite = objFSO.OpenTextFile( xmlFile, 2 )
  'objWrite.ContentType="text/xml"
  objWrite.writeline("<?xml version='1.0' encoding='UTF-8'?>")


 set rs=server.createobject("adodb.recordset")
  rs.open "select * from IRPrioritizedXMLTable","DRIVER={Microsoft Access Driver (*.mdb)};DBQ=" & Server.Mappath("\db\ds2.mdb")

   temp1 = ("<Event collected='")
   temp2 = ("'>")
   tempf = temp1 & now() & temp2

  objWrite.write("<Documents>")
  while not rs.Eof
    objWrite.write tempf
    for i=0 to rs.fields.count -1
      if rs.fields(i).value<>"" then
       objWrite.write("<"&rs.Fields(i).Name&">"&rs.Fields(i).Value&"</"&rs.Fields(i).Name&">")
      else
       objWrite.write("<"&rs.Fields(i).Name&"/>")
      end if
    next
    objWrite.write("</Event>")
    rs.moveNext
  wend
  objWrite.write("</Documents>")

rs.Close
set rs = Nothing
%>
```

Figure 11. Code for Record Set Conversion to XML

C. COMBINING, GROUPING AND SORTING THE SEPARATE XML FEEDS

Once the separate XML feed files have been created, they are combined into one document through a series of XML to XML transformations using XSLT and ASP scripts. The sequence is started in the prototype by means of a button push. This is not optimal, as the user has to update his data sources manually. A better means would be to use an application running in the user's background, which would update the source file automatically through a timer function. The files should be combined and grouped to provide for the global concurrency control issue mentioned in Chapter II.

To provide consistent views across the system, there should only be one source of data. The program that begins the transformation process is located in Appendix A under the App pages Folder, Combine_and_Group_Function_Calls.asp. Two transformations are called. The first transformation accomplishes two things. First, it combines all the information feeds from the fifteen XML files and applies the schema to the result document. This transformation makes use of the XPath document() function. The separate pages with the various Event information feeds are passed to the XSLT document as parameters. After the first transformation function is complete, the second transformation deepens the structure of the result document. It creates groups and sorts all the events into the groups by use of the document number found in the Event_Reference tag. The second transformation makes the display of the document reports easier and slightly faster. Instead of having to search through every <Event> to find the correct <Event>s to display for a request, the second transformation allows the display programs to search through group elements called <History>. Figure 12 displays a snippet of the resulting combined grouped XML file. This file can viewed in Appendix A in the XML Files folder as GroupByDocNumber.xml. It is from this file that the three levels of displays are created.

```
<?xml version="1.0" ?>
- <Documents>
  - <History DocumentNumber="AA7001">
    - <Event collected="8/7/2009 2:48:19 AM" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Event_Reference>AA7001</Event_Reference>
        <Event_Label>Document Creation</Event_Label>
        <Event_Status>Completed</Event_Status>
        <Event_Date>2009-08-01</Event_Date>
        <Event_Time>07:00:00</Event_Time>
        <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
        <Event_Comments />
        <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
      </Event>
    - <Event collected="8/7/2009 2:48:24 AM" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Event_Reference>AA7001</Event_Reference>
        <Event_Label>Document Submission</Event_Label>
        <Event_Status>Completed</Event_Status>
        <Event_Date>2009-08-02</Event_Date>
        <Event_Time>08:00:00</Event_Time>
        <Event_Contact>OriginalRequestor@UnitAA.mil</Event_Contact>
        <Event_Comments />
        <Event_Info_Link>Links/default_link.asp</Event_Info_Link>
      </Event>
```

Figure 12. GroupByDocNumber.xml snippet

D. DISPLAY REPORTS

A template was applied to the active pages to provide the prototype Web site a familiar navigation experience and allow the site to take on a familiar appearance to the user. The template provides a navigation bar at the top of the page and four sections that allow the page content to be organized. Figure 13 depicts the template composition. The top left section is used to provide a placeholder for addition navigation links and options to the user. Below this section is a left-side bar content area. This section is used to position links to the programs that update the system information. At the top, to the right of the addition navigation section, is the Search Bar area. XSLT transformations are used to provide this section with the XHTML used to produce the search options base on the information in the XML source file document. Below this section is the main content section where the three levels

of reports are displayed. The intent of the template is to provide the user with a Web site environment that offers familiar navigation and requires a low level of instruction to operate effectively.

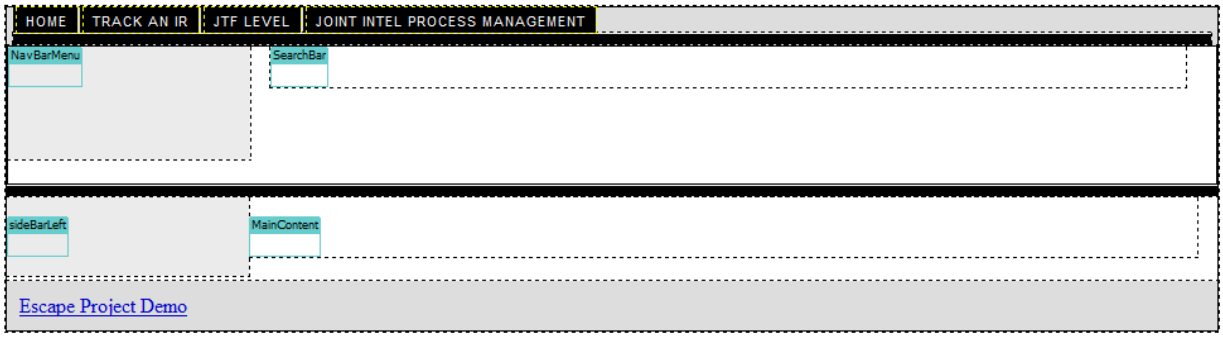


Figure 13. Template Layout

E. INDIVIDUAL DOCUMENT HISTORY REPORT

The individual document history report is intended to provide a quick visualization of the progress of an individual information request and provides a visual cue of the actions that remain to be accomplished to complete the IR. The report connects the various ISR-related activities that have occurred and those that need to be accomplished to the information request. A detailed section also provides some basic details of each action that occurred to the IR creating the history. The detailed section also allows for the supplementation of additional information that could be used to provide a system user with more details of the actions that have been taken or are in the process of occurring. These links and the contact information supplied could be employed by the system users to coordinate or perform self-synchronization actions. The report also

decreases some level of uncertainty as to the IR's status. Figure 14 offers a snapshot of a report for IR #XX7005.

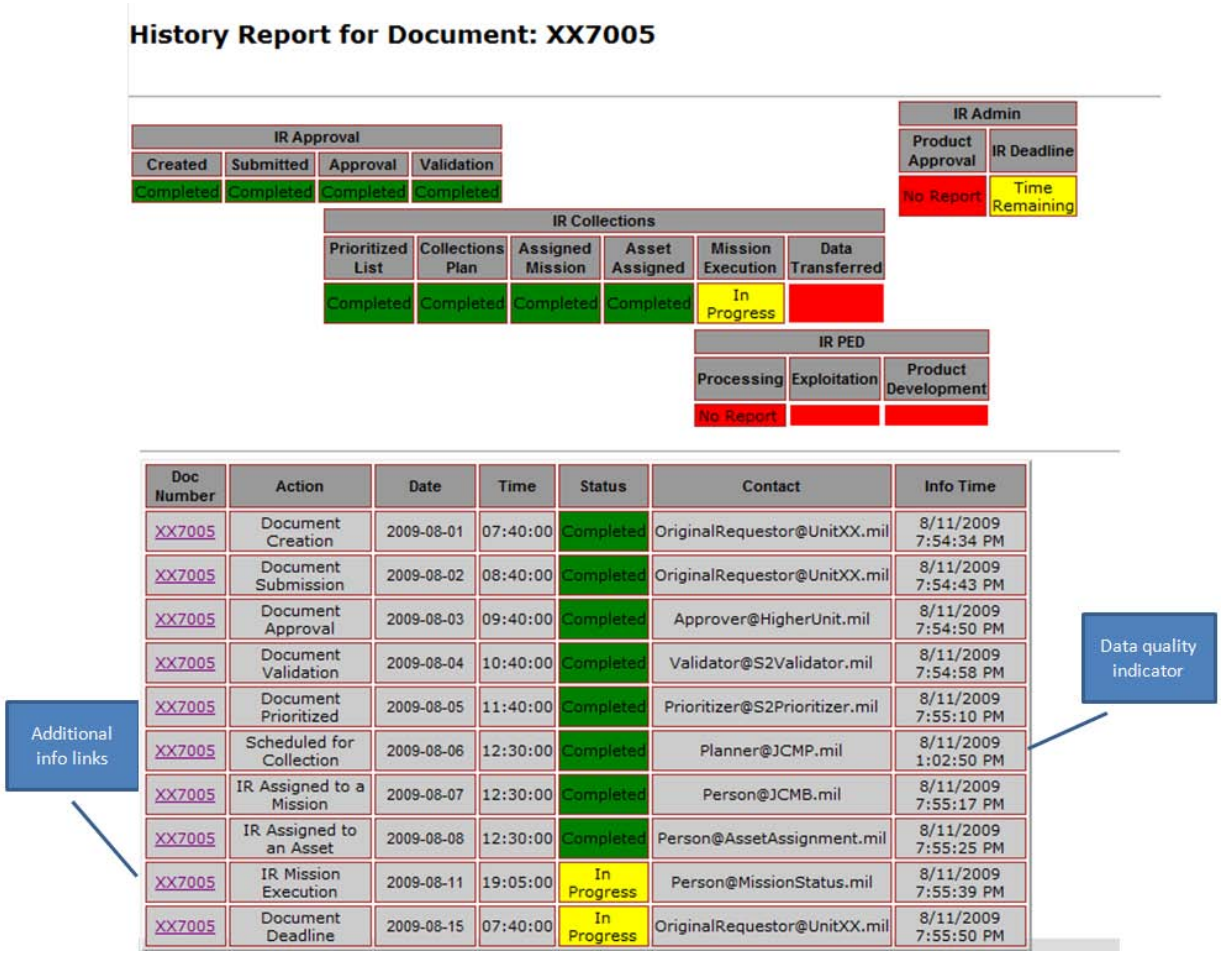


Figure 14. Individual Document History Display

The top of the report displays the IR number referred to as the document number identifying the report. The middle section provides a series of boxes, meant to be read to the right and down, which follows the steps needed to complete the request and develop an information product. The bottom section of the report provides the details of the events that have occurred or are in progress. It also provides contact information to the person who conducted the

event and the Doc Number column is used to attach the link to more information for each event in the list. To the right, the data quality indicator is displayed in the Info Time column. This is the time the information was removed from the database and converted to an information feed. We can see that data for the Scheduled for Collection event is older than the rest of the feeds. In this way, users can judge the value of the data being displayed. From this report, we can see that IR XX7005 has its collection mission currently in progress and that the product deadline is six days away. If a system user wanted to check on the mission's status, the link under the Doc Number column on the IR Mission Execution row can be used to link the user to the current updated mission feed. When the link is activated, a new window pops out and the linked feed is displayed. Figure 15 is the example display on this document.

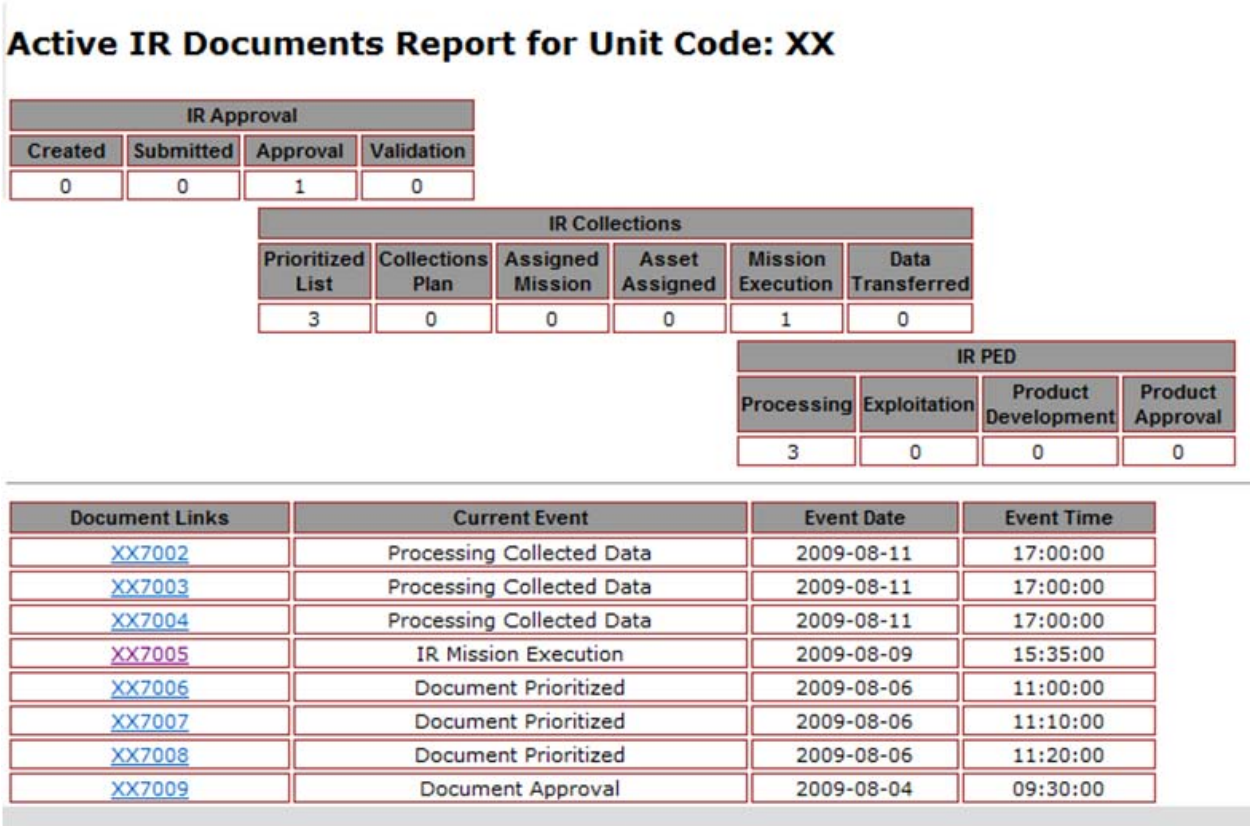
Figure 15. Mission in Progress Display

This display is produced from a notional XML feed whose information is transformed into Java Script instructions using XSLT and displayed using the Google Maps API. The Blue dot represents the collection asset's current location and the red dots are either collection points or the mission's start/stop point. When the user's mouse passes over a collection point, information is displayed in the same fashion as that shown for XX7005 on Figure 15. Using the link in the IR History Report, users can follow the progress of the collection mission. In the same manner, other links attached to the report's other events can provide additional information for coordination or self-synchronization of system users. For example, the Document Submission event link could lead to the details of the original request in some other repository. Analysts at a PED node could use the link to check the requirements of the information product they would need to produce to satisfy the IR.

F. THE MID-LEVEL UNIT REPORT

The aggregation of all the active individual document histories produces the mid-level report, intended to allow unit managers to track multiple information requests. The report has a similar format to the Individual Document History Report. The top of the report identifies the unit for which the report is constructed. Under the report title, a series of step boxes matching the information points identified in Chapter II are displayed. In each of these boxes, the number of active documents, or documents listed as "In Progress," is counted for the unit's code. At the bottom of the report, each active document is listed and

the active event is displayed. The Document links column provides a link to open the individual Document History report on each active document in the unit. Figure 16 is a sample mid-level report for the fictional unit XX. From this report, we can see where the active documents are in the process and when they were last acted upon. In this report, we see there are eight active documents including XX7005. This is the same document reviewed earlier. If a unit level manager needed to find the status of an individual document, he would simply click on the document link. This would cause another window to open with Individual Document History Report for that document to be produced. Unit managers can use the mid-level report to open and monitor the status of several documents at once. Managers can use the information contained in the individual reports for coordination purposes.



Active IR Documents Report for Unit Code: XX

IR Approval			
Created	Submitted	Approval	Validation
0	0	1	0

IR Collections					
Prioritized List	Collections Plan	Assigned Mission	Asset Assigned	Mission Execution	Data Transferred
3	0	0	0	1	0

IR PED			
Processing	Exploitation	Product Development	Product Approval
3	0	0	0

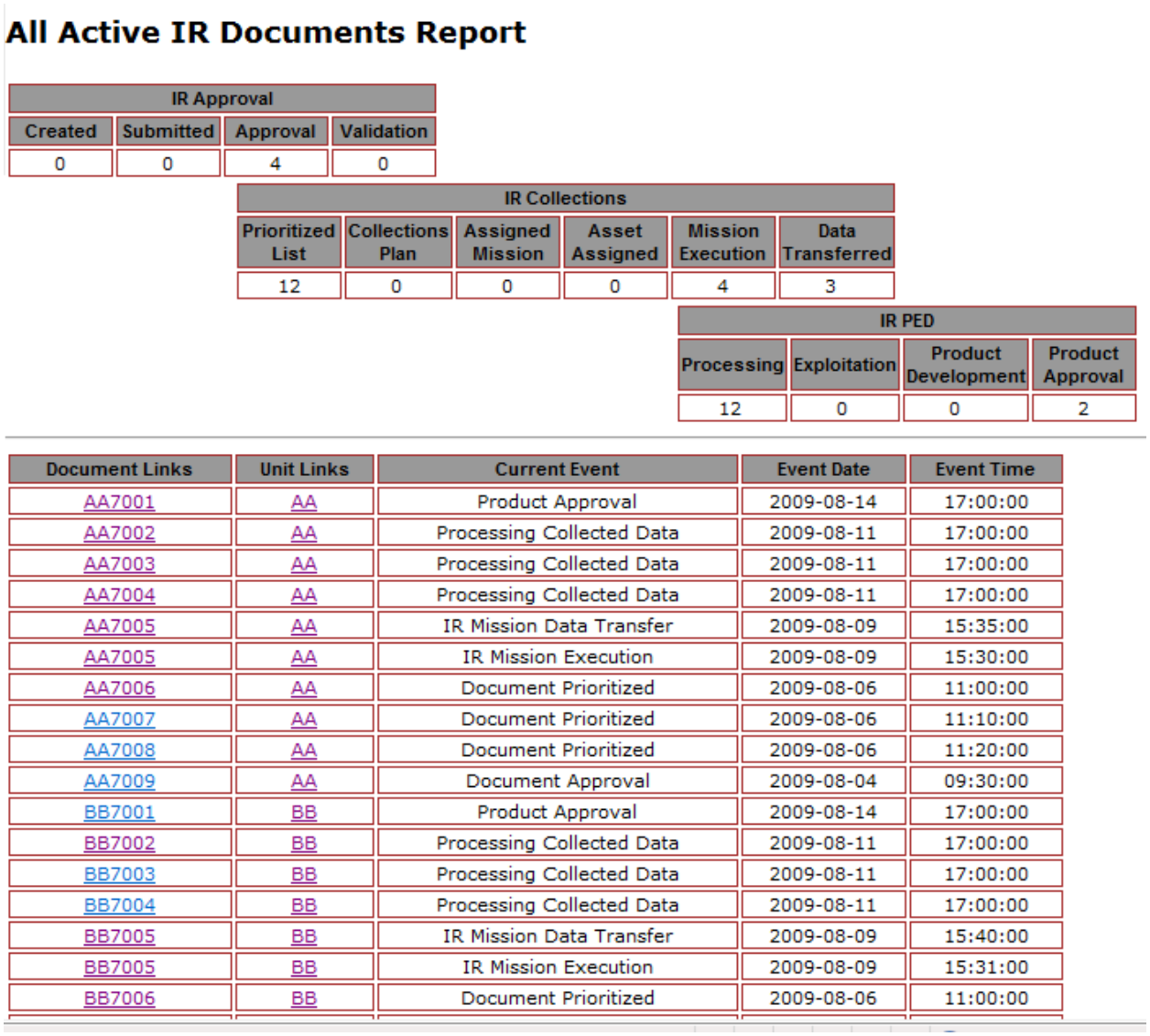
Document Links	Current Event	Event Date	Event Time
XX7002	Processing Collected Data	2009-08-11	17:00:00
XX7003	Processing Collected Data	2009-08-11	17:00:00
XX7004	Processing Collected Data	2009-08-11	17:00:00
XX7005	IR Mission Execution	2009-08-09	15:35:00
XX7006	Document Prioritized	2009-08-06	11:00:00
XX7007	Document Prioritized	2009-08-06	11:10:00
XX7008	Document Prioritized	2009-08-06	11:20:00
XX7009	Document Approval	2009-08-04	09:30:00

Figure 16. Mid-Level Unit Report Display

G. THE TOP-LEVEL DISPLAY

The top-level display is composed from the aggregation of all the active documents listed in the system. It is intended to be used to display the level of activity for all users in the system. It is also meant to be used by top-level managers to help visualize the level of activity at ISR-related nodes. To increase familiarity, the report's format is similar to that of the mid-level. The top-level report provides similar details to the mid-level report, such as the current active events of the documents being displayed. The report also provides links to both the mid-level reports and the ability for the user to go directly to

an individual document history report. In this manner, top-level managers can visualize how busy the various ISR-related node sections currently are, and can drill down into reports to the level of detail desired. Figure 17 is an example top-level report. Only a part of the documents listed is included in the diagram. We can see the information required to produce the information products of twelve information requests are being processed into a useable form. Top-level managers could use this information to plan the use of the analysts needed to produce the products.



All Active IR Documents Report

IR Approval			
Created	Submitted	Approval	Validation
0	0	4	0

IR Collections					
Prioritized List	Collections Plan	Assigned Mission	Asset Assigned	Mission Execution	Data Transferred
12	0	0	0	4	3

IR PED			
Processing	Exploitation	Product Development	Product Approval
12	0	0	2

Document Links	Unit Links	Current Event	Event Date	Event Time
AA7001	AA	Product Approval	2009-08-14	17:00:00
AA7002	AA	Processing Collected Data	2009-08-11	17:00:00
AA7003	AA	Processing Collected Data	2009-08-11	17:00:00
AA7004	AA	Processing Collected Data	2009-08-11	17:00:00
AA7005	AA	IR Mission Data Transfer	2009-08-09	15:35:00
AA7005	AA	IR Mission Execution	2009-08-09	15:30:00
AA7006	AA	Document Prioritized	2009-08-06	11:00:00
AA7007	AA	Document Prioritized	2009-08-06	11:10:00
AA7008	AA	Document Prioritized	2009-08-06	11:20:00
AA7009	AA	Document Approval	2009-08-04	09:30:00
BB7001	BB	Product Approval	2009-08-14	17:00:00
BB7002	BB	Processing Collected Data	2009-08-11	17:00:00
BB7003	BB	Processing Collected Data	2009-08-11	17:00:00
BB7004	BB	Processing Collected Data	2009-08-11	17:00:00
BB7005	BB	IR Mission Data Transfer	2009-08-09	15:40:00
BB7005	BB	IR Mission Execution	2009-08-09	15:31:00
BB7006	BB	Document Prioritized	2009-08-06	11:00:00

Figure 17. Top-Level Display of All Active Documents

H. THE SEARCH FEATURE

A separate search feature was added to the Web application to provide another means of switching between reports, instead of having to drill down the menus for information. The search feature allows users who know what they are looking for to get to the report they want more

quickly. Figure 18 is a screenshot of the Web application, partially displaying an Individual Document History report of IR #XX7005.

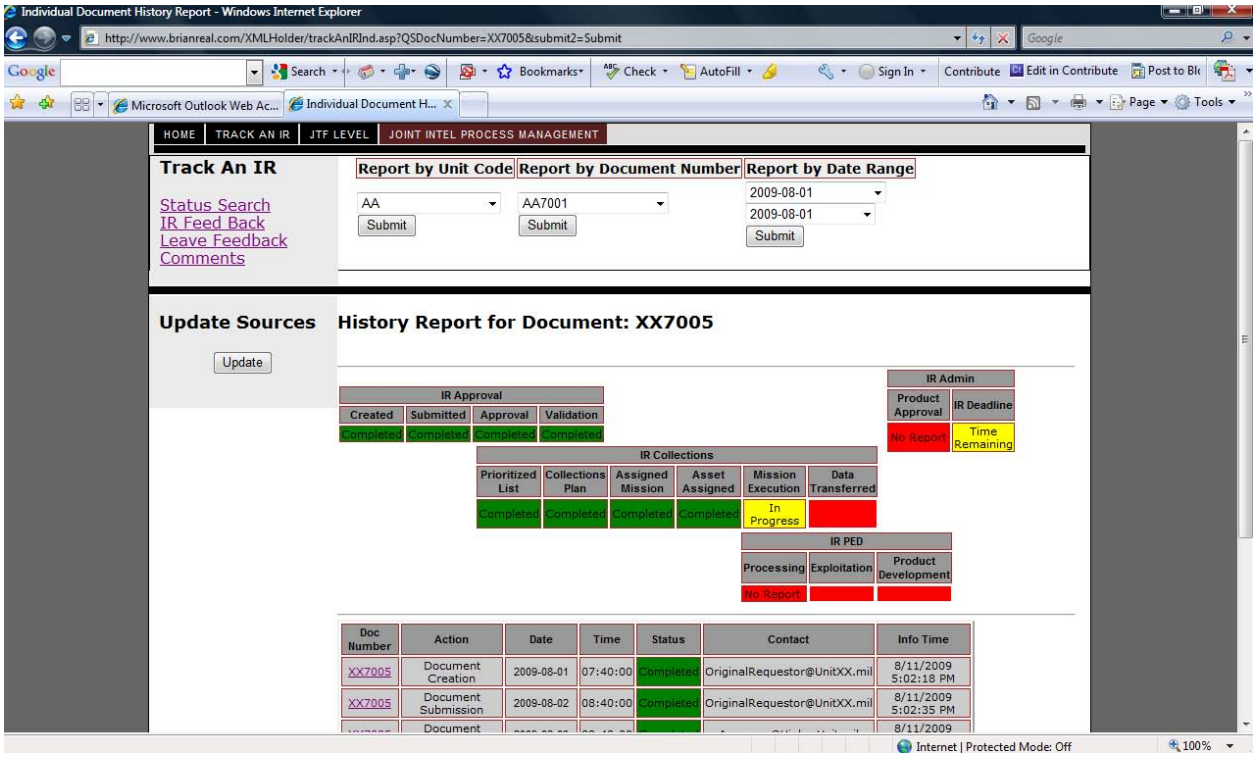


Figure 18. Screenshot of the Web Application

These reports are not meant to be the only representations possible when visualizing the IR and related ISR processes. In the next section, alternate methods of displaying the same data are discussed.

V. JOINT INTELLIGENCE PROCESS MANAGEMENT AND ALTERNATE DISPLAYS

This prototype was designed to give visibility to the information request process and the associated tasks. These tasks vary, from approving the information request, to assigning assets to the requests, to processing and exploiting the intelligence data once it is retrieved, to making sure the information gets back to the originator. The user requires information on the availability of the assets and where they are located. The prototype needs to tell the user who controls those assets and how are they being applied. Beyond the assets, the user needs to know what data is being collected, when and where the data is being collected, and the quality of the collected data.

A. JOINT INTELLIGENCE PROCESS MANAGEMENT

Under the “Joint Intelligence Process Management” tab of the prototype, there are two functions. “PED Node Info” is the first screen by default as shown in Figure 19. This screen allows the user to look at one or more PED node units at a time. The page has a search function at the top, which allows the user to narrow the choices viewed. A list of available commands is displayed in the left-hand column and updated once the submit button has been pushed. The user then selects one of the various commands listed in the left column, which displays the details about the PED node in the center display area. This area contains information about the PED node such as the unit’s name, the skill set, the manager’s estimate, the number of analysts, and the number

of IRs that are currently in work. The manager's estimate is a self-assessment from the PED node's manager as to how heavily loaded the unit is. Additionally, the center display contains a scrolling table listing the IRs that the PED node is currently working. The IR number in the table is also hyperlinked to retrieve additional information about the IR. Furthermore, the table provides the user with an estimated completion of when the PED node expects to finish the IR.

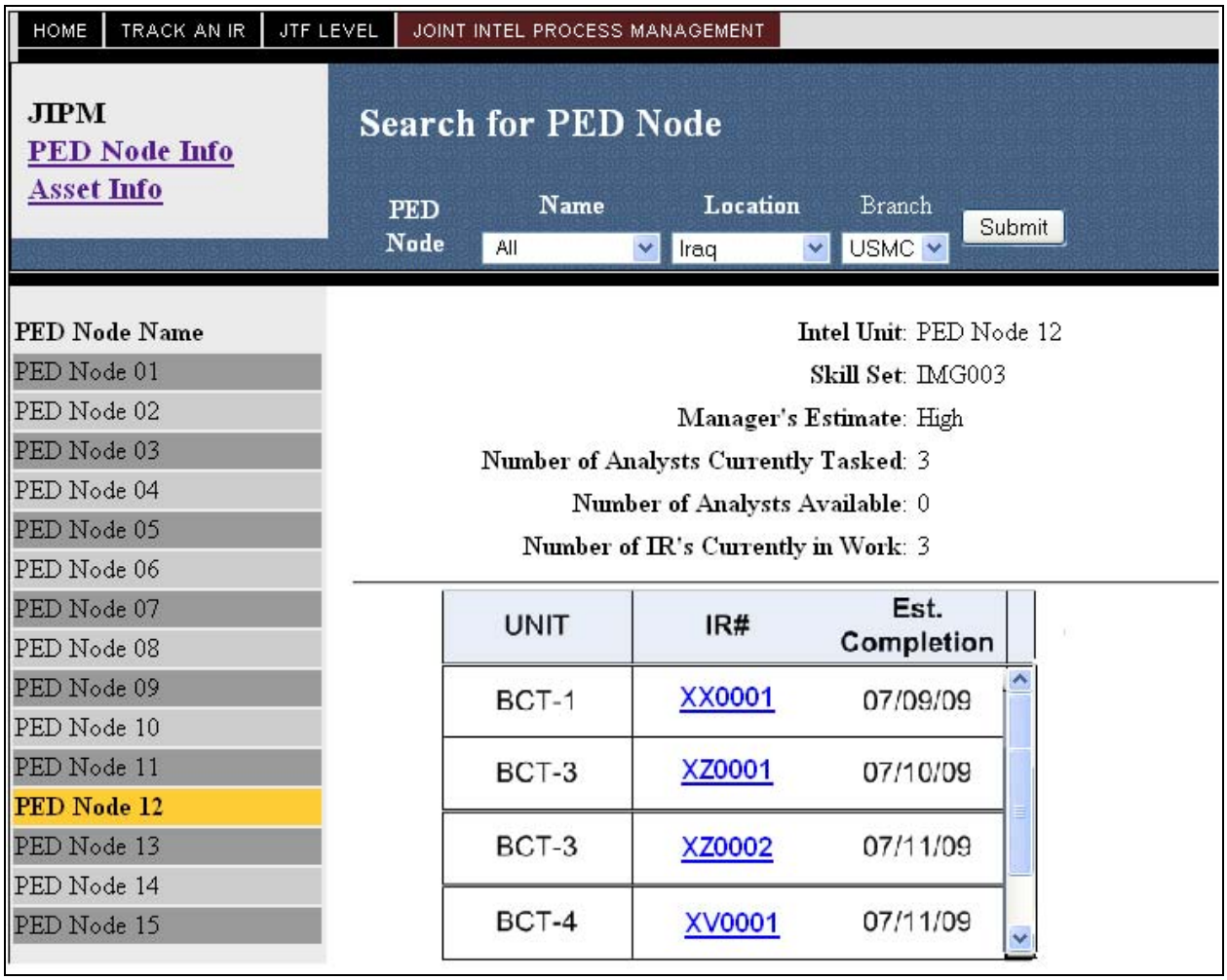


Figure 19. JIPM PED Node Information

B. ASSET INFORMATION

The second menu item associated with the "Joint Intel Process Management" tab is the "Asset Info" screen in Figure 20. The Asset Info screen serves multiple functions: It allows planners to see how an asset is currently utilized and with which IR it is associated.

Figure 20. JIPM Asset Info

C. SEARCH FOR ASSETS PROCESS

This page works in a similar fashion to the PED Node Info page. A user selects from the Search function a particular unit, location, or platform. The drop-down menus for the search function are dynamically populated from the database, ensuring that only correct information is entered in the search function. Pressing the submit button of the search function updates the list of unit names on the left-

hand column. The user then selects a unit name from the column on the left, which displays the details about that unit.

The results in the center provide information on each of the unit's assets. In particular, it provides the user with a myriad of information, including which unit owns the asset and who controls that unit. It specifies as to what type of asset, such as whether it is a Predator or a P-3, and how many of the assets the command owns. It also provides a summary status of those assets. Figure 20 indicates that this command has four fully mission-capable (FMC) P-3s. Additionally, the center displays a scrolling table with information on each asset. Each line contains a unique Bureau Number (BUNO), which is associated with only that asset. The table contains information about the asset's functional status as well as the current mission status associated with that asset. Any deviations or items that warrant a comment are displayed. The IR number currently associated with the asset is displayed with a hyperlink to get additional information about the IR. IR Data Location is another important field display, allowing the user who desires highly time-sensitive information to know exactly where the data is located. If the data cannot be obtained from the asset until after it returns to base, a Return To Base (RTB) time is also displayed.

D. ALTERNATIVE DISPLAYS

The prototype's current displays meet the aforementioned requirements. The displays were designed within the limits of the designer's capabilities. Every day, new graphic design techniques are developed and implemented

on the Internet. Knowledge of those design techniques is rapidly shared through discussion and chat rooms on the Internet. The intent of rest of this chapter is to present displays that are not included in the working prototype. These displays were either too difficult to design or outside the scope of time needed to complete them. However, it is important to understand that this prototype is not the only way to display IR Tracking information, and a myriad of alternate presentations are possible.

The first design presented is a variation on our current asset status display. This design is very similar with one exception: the use of radio buttons to choose or filter the data. The user views this page starting at the top and selecting items that limit what is displayed in the main display area. The page starts out by displaying everything, but as more and more boxes are checked, the more succinct the data displayed. In the example below, the user has selected the CENTCOM AOR. This selection dynamically alters what is available in the “JTF Level” box below. The user then selects which JTF unit(s) on which he/she would like information. Unit names are displayed vertically along the middle left side, which again provides move filtering functionality.

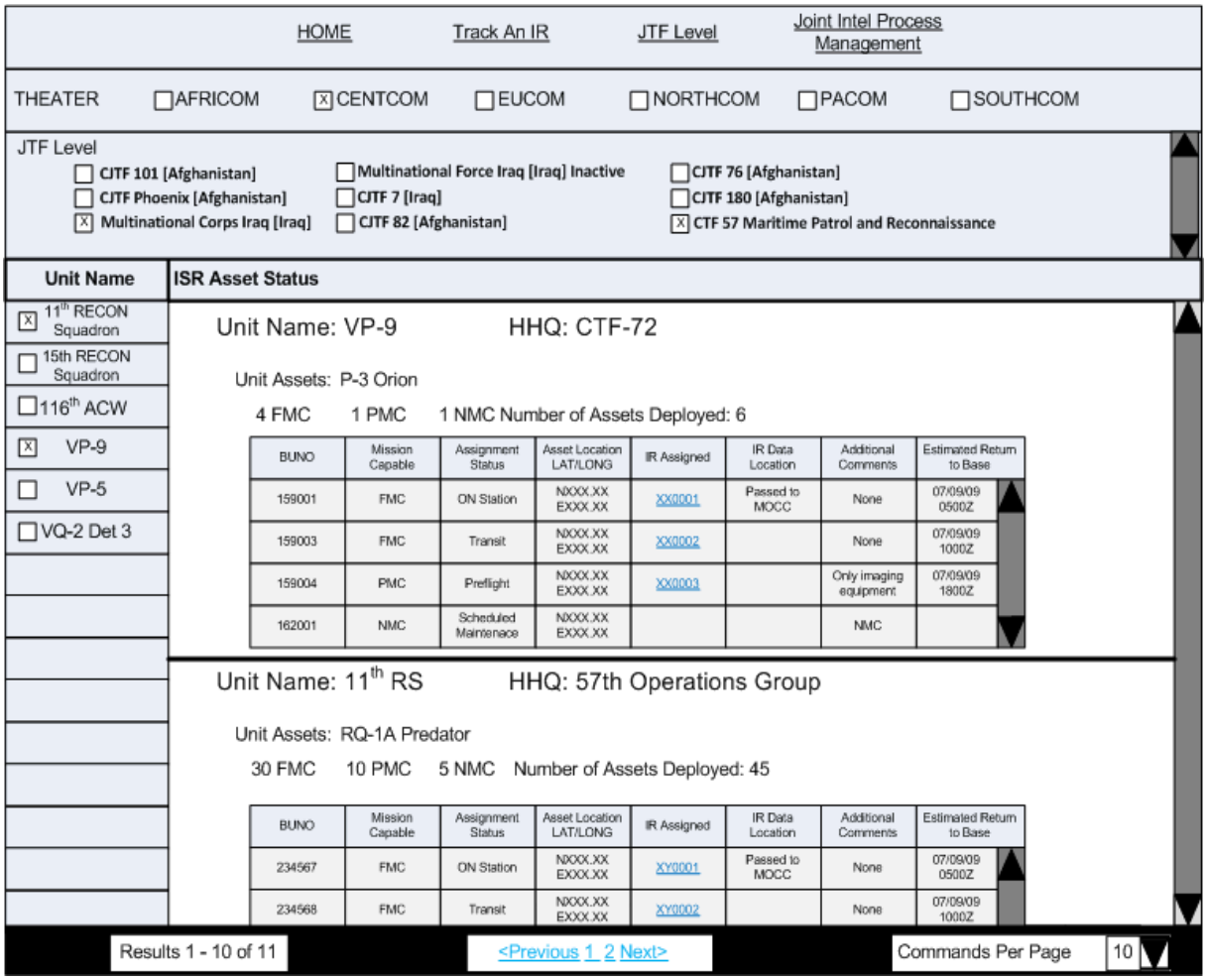


Figure 21. JIPM Asset Info Alternate

A selected unit will show up in the body of information. The user has additional means of how many commands can be displayed on any given page. This is done via a dropdown arrow that lets the user choose 5, 10, or 25 units per page. Page navigation is incorporated, allowing quick navigation through multiple pages.

Besides the need to know about the specific state of assets, certain users also need to inquire about the loading of PED nodes. Figure 22 provides a display that indicates each PED node's workload. In theater, this is very helpful

in the planning process. The PED Node Loading Display will add to the user's toolset in deciding which node has the appropriate skill and which node is under-tasked enough to complete an analysis.

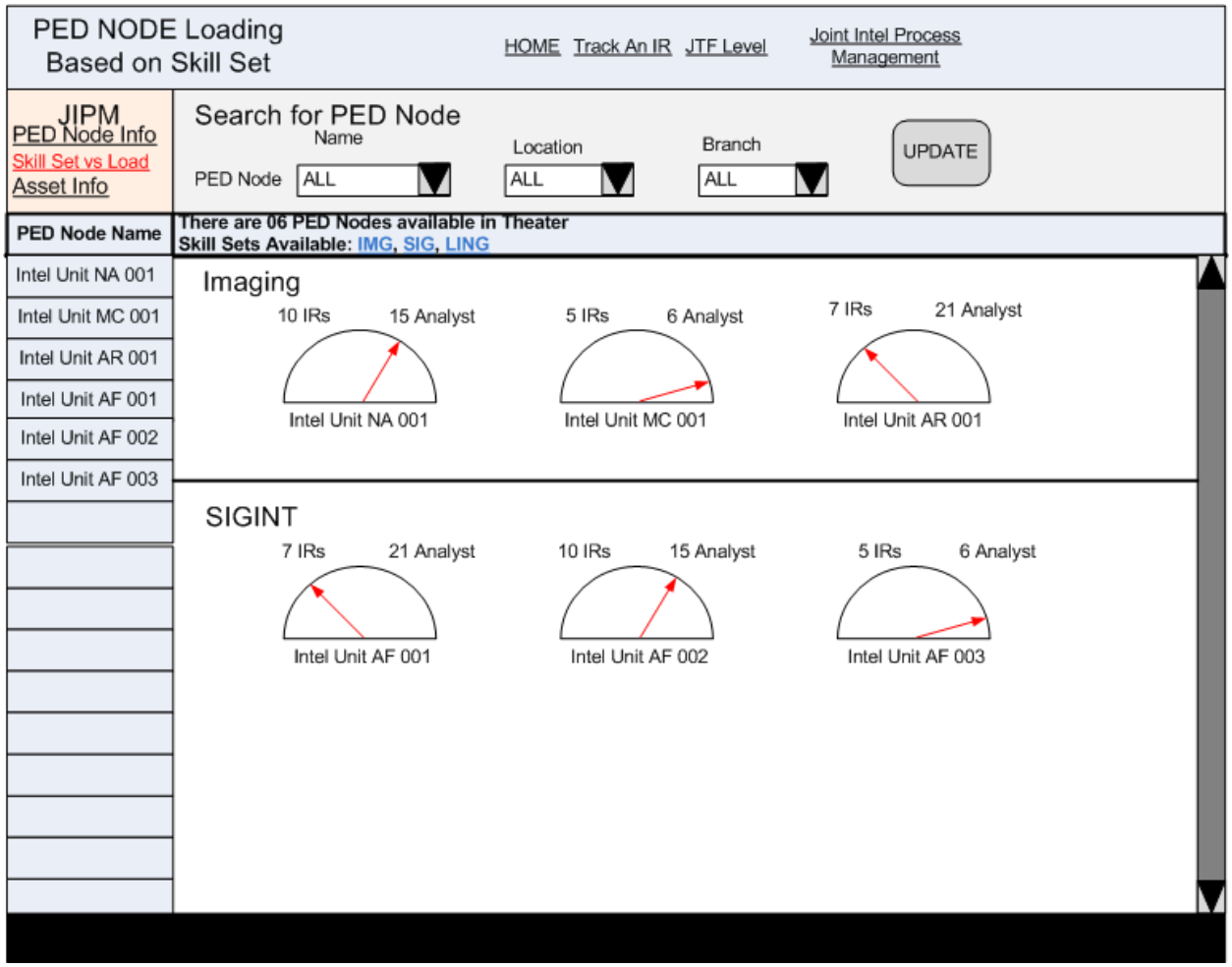


Figure 22. JIPM Skill Set vs. Load

In Figure 22, the user can quickly identify the PED nodes that appear to have heavy loading and which nodes have minimum loading. Using the display above, an intelligence manager needing a PED node qualified in imaging could quickly see that Intel Unit AR 001, having 21 analysts, and only seven IRs, would be the most likely place to send his

data. But, there are more questions that are not answered in this display. For example, why does this unit have 21 analysts, but is used to analyze only seven information requests? An information hyperlink off each node could provide more detailed information. Maybe the unit is preparing to deploy back to home and has slowly been reducing its workload.

Near real-time information about PED node status and asset status could help intelligence planners and managers improve the process of allocating IRs to platforms, sensors and processing nodes. Since all of the required information is captured by the ISR situational awareness system, one or more decision aids could be developed to help the planners optimize the planning process.

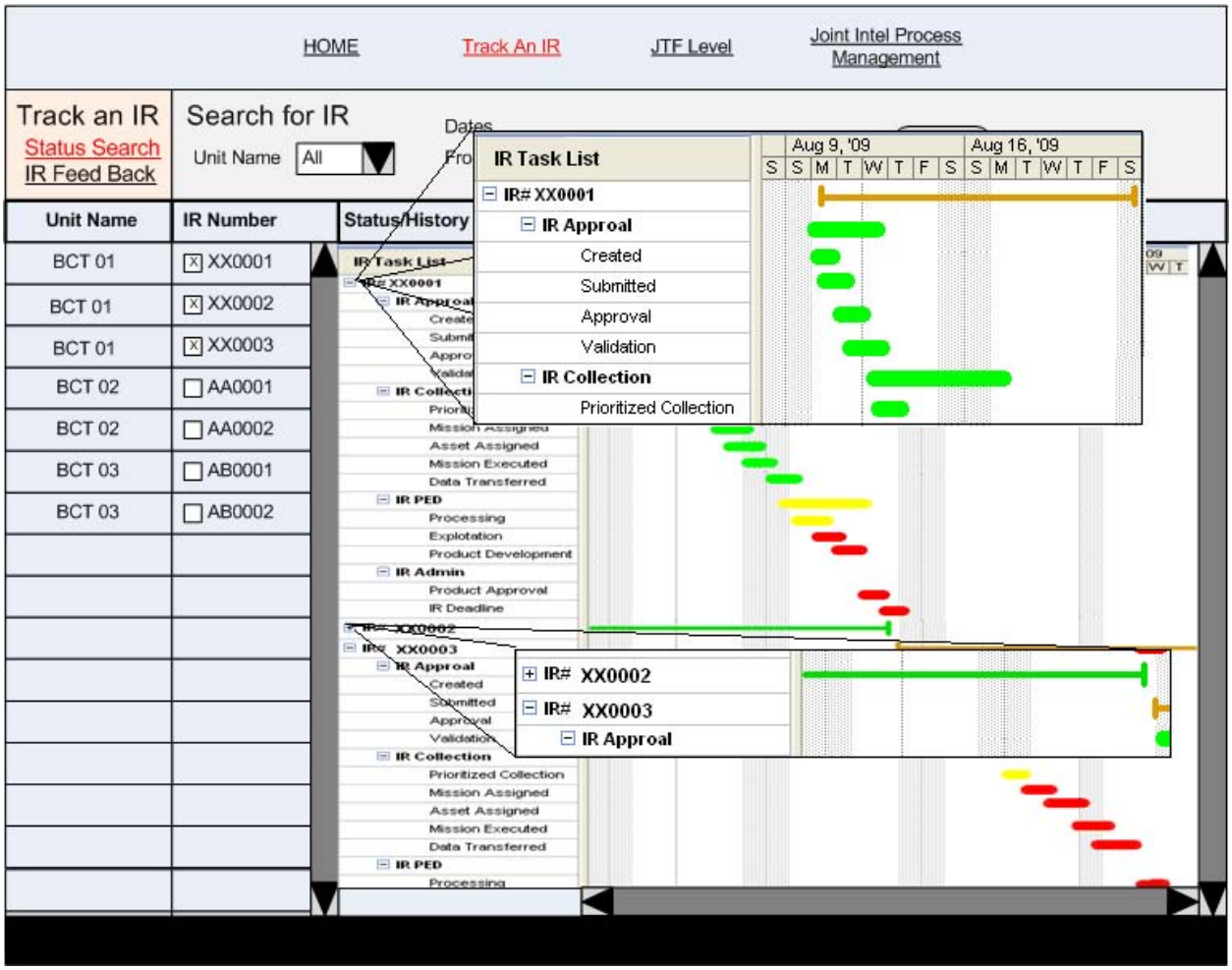


Figure 23. Track An IR Alternate Display with +/- Boxes

E. ALTERNATE TRACKING INFORMATION REQUEST WITH SLIDING CALENDAR

When looking through a list of Information Requests, a user wants to quickly determine the IR's status. Figure 23 does this in numerous ways. The page is similar to our prototype in that it uses color to indicate status–red for incomplete, yellow for in-progress, and green for complete. What is different is the use of the plus and minus boxes that can be selected to see more or fewer details on a particular IR. When a plus box is selected, any subordinate

information is displayed, including a color bar showing dates associated with events on a sliding calendar bar at the top. Using this sliding calendar, the user can determine when events have been completed, indicated in green; when event are in progress, indicated by the yellow; and when events are estimated to be completed, indicated by the red bar.

When a minus sign is selected, the row collapses, displaying a single bar associated with the entire IR. If the IR has been completed, it will have a green bar associated with it, as illustrated by IR #XX0002 in Figure 23. If the task is incomplete, it will have a yellow bar, as shown with IR# XX0001. Additionally, each IR number on each line would be hyperlinked to its history.

This expanding/collapsible viewing method allows the user to quickly gather the information desired, and not be overwhelmed with extra information.

F. ALTERNATE TRACKING INFORMATION REQUEST WITH INFORMATION BUBBLES

A good Web site display should quickly inform the user of information without having to read the details. In Figure 24, the user uses the search bar and the left column navigation bar to quickly limit the IRs displayed on the page. The use of color-coded bubbles allows the user to quickly determine the status of an IR [16]. Each IR has a main bubble followed by smaller bubbles representing each event.

Figure 24. Track An IR Alternate Bubble Display

The larger bubble is a quick summary of the IR's current status. In Figure 25, the bubble for the first IR indicates that a mission is being executed, whereas the bubble for the next IR indicates there is some sort of delay in the mission. The smaller bubbles list each step in the process. As each step is executed, the bubble's color turns from grey to green. If there is an issue with a step, the color changes to orange to visually indicate a problem. Each bubble lists date information. If the bubble is green, the date information indicates the event has occurred. If the bubble is grey, the date is an estimate of when the

event will occur. Additionally, each bubble will be hyperlinked to the history of each IR.

Figure 25. Track an IR Alternate Bubble Information (From [16])

G. ADDITIONAL DATA DISPLAYS

The purpose of an Information Request is to get needed data to a user so that a commander can complete a mission. The quicker a commander can get this information, the more time a commander has to plan his mission, resulting in better execution of missions.

One method for quickly getting the IR's data back to the originator is the use of an add-in like the Rockwell Collins Spot Beam. Spot Beam is an add-in that runs as a Falcon-View plug-in. Spot Beam displays ongoing sorties

with multiple UAV tracks on a map in real time. The user can then select a UAV track and the associated EO/IR sensor data. What is unique about this system is that the user can select a small portion of the data, and not the entire data stream. This allows users to get just what they want to see, reducing the amount of wasted bandwidth. Figure 26 displays a selected UAV track. The green portion of the track is the entire track. The yellow portion is within the green portion of the track, and is the portion of the video that will be transferred back to the user.

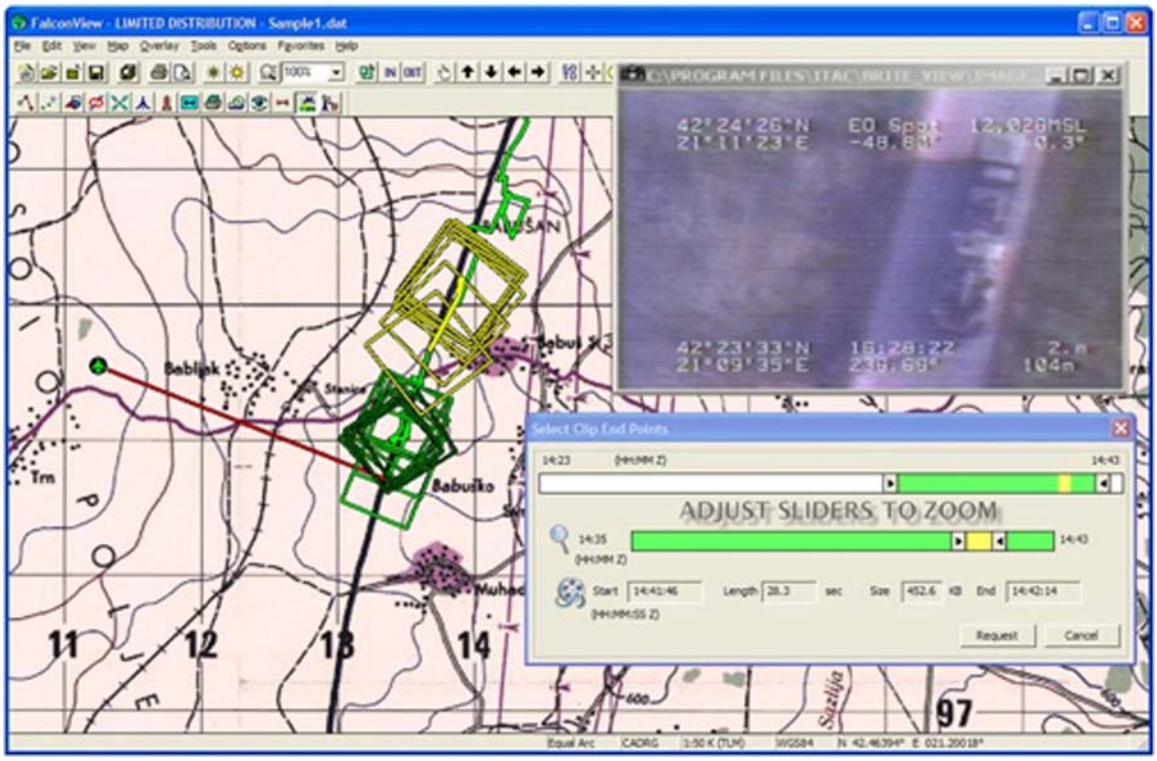


Figure 26. Spot Beam Software FalconView Plug-In (From [17]).

The user can adjust the length of the clip transmitted by dragging the sliders on the end of the yellow, as shown in Figure 25.

This program is produced by Rockwell Collins and only works on FalconView. This program or something similar could work as a Web browser plug-in with Google Maps. The user of this prototype would click on a link under Mission Execution from the History Report, such as displayed in Figure 14. This link would pop up another window with the current track of the IR and display the capabilities shown above. This would allow the user to get immediate feedback from information requests, giving the user more time in the planning process.

In summary, there are a wide range of options in presenting ISR situational awareness data to C2 nodes and to intelligence planners and managers. The prototype architecture provides the data to support these alternate graphical user interfaces, and final choices of GUI designs will depend on operational users' assessments. Additionally, the ability to capture and display relevant information in near real time may help improve, or perhaps automate, the intelligence planning process.

VI. CONCLUSIONS

A. SUMMARY

The prototype discussed in this thesis demonstrates the ability of a Web application to help visualize efforts and connect users to the right information at the right time. It also allows for self-synchronization efforts between users. Although not an exhaustive effort, the prototype proves the concept of providing visualization of ISR-related processes through the conversion, fusion, and manipulation of data using currently available XML-related open source methods.

B. CONCLUSIONS

An ISR visualization tool or related SA system would require the ability to fuse data from multiple sources and allow the user access to the information in a manner transparent to the user. The performance of the data access should be comparable to that of accessing a local source. The system will have to allow for the simple addition of supplementary information sources and display the information in a design that the user would easily find familiar. A Web application built using XML-based technologies offers a means of achieving these requirements. Using predetermined queries to produce record sets that form the base of the XML information feeds offers users the freedom of continuing to use existing systems for daily routines while supplying information to a greater ISR system with minimal invasive impact. The processing of the XML data and development of reports using server-side XSLT

transformations allows for the development of simple code that can be read on the most popular browsers in use today. The XML Schema language can be used to address the issue of global semantic integrity of the system data, and the use of combined source documents allows for global concurrency control of the data displayed in system reports. The system should display all the tasks and associated tasks needed to complete an ISR-related activity. Users should have the ability to drill down into these tasks to achieve the level of detail desired, or receive directing links to other systems containing the details desired, to allow for coordination and other self-synchronizing efforts.

C. RECOMMENDATIONS

1. All data elements should be marked with quality indicators at the most basic level possible. Since data will be combined and otherwise fused with other sources, the user requires an ability to judge the quality of the data being displayed. It is upon this data that the user will base his coordinating and self-synchronizing efforts, and an indication of the data quality will give him the justification with which to base those efforts.

2. The schema application should occur as low as possible in the information development chain. In the prototype, the schema application occurred in the first transformation when the different sources were being combined. The schema application should be applied before the fusion and preferably right after the conversion of the record set to XML format. During use of the prototype, it was found that errant data could be entered into the system and not discovered until the data was being fused. This

caused the operation to fail and the combined document was not achieved. By checking the data separately and before it is fused, bad data will be kept out of the system, and the rest of the data sources can still be combined to provide at least partial visibility.

3. Both event-based and tree-based types of XML processors will have a place in any ISR-related system using XML-based technologies. Event-based processors bring the advantage of speed when searching large documents for small amounts of information, and tree-based processors offer the flexibility needed for queries that are more complicated. The design of system tasks should take into account these abilities to maximize the benefits of each type of processor and provide the highest level of responsiveness to the user.

4. Server-side transformations are needed to assure control of the data. Commonly referenced data intended for all users can be processed on the client, but any data sensitive to user levels cannot. Once a data source is sent to the client for processing, control over the data source can be compromised.

D. FUTURE WORK

More work is needed to determine all the details necessary for a fully functional ISR SA system. The prototype provides a framework to which the more specific details can be added. Once development had started, the problem was found to be larger than anticipated, so the scope of the effort was narrowed to provide a working prototype of a related activity, the IR Process. More specific details related to ISR SA, such as asset ownership,

current location, and capability status, along with time estimates of node-level activities, would provide a greater and more desired level of detail and Situational Awareness [18]. The prototype shows completed activities, the current IR process activity, and the remaining activities. Adding the average time of each activity would provide the user with a better ability to judge the time remaining until the requested information would become available. The addition of a skill set rating by available C2 Nodes could also help move this visualization tool to the level of a data driven decision support tool for use by ISR managers. Also, the ability of current systems to provide the information portrayed in the prototype will need to be determined.

E. RELATED WORK

A study is needed to determine the requirements of an SA system that allows for the dynamic reallocation of ISR assets.

A study is needed on the use of Web services to supply the information feeds, the ability to merge data provided by these services, and a performance comparison of combined Web service feeds versus those generated from record sets.

A study of the Encrypted XML Interchange effort, or the value of using encrypted XML documents to provide security rather than a network encrypted approach, would be beneficial to the design of a system using XML documents for information sources.

LIST OF REFERENCES

[1] DCGS Joint Concept of Operations, version 1.0, United States Joint Forces Command, JTC-I/DI-I, Norfolk, VA, September 30, 2008.

[2] United States. Joint Forces Command. DCGS Joint Concept of Operations. Draft, pp. 4-13, 30 September 2008.

[3] United States. Joint Publication 3-33. Joint Task Force Headquarters. pp. I-1, February 16, 2007.

[4] LtCol Sovada, Personal Interview. November 17, 2008.

[5] United States. Joint Publication 3-33. Joint Task Force Headquarters, pp. I-4 & II-6, II-10, February 16, 2007.

[6] Magdi Kamel, Nabil Kamel, "Federated database management system: Requirements, issues and solutions." Computer Communications, vol. 15, Issue 4, May 1992, pp. 270-278.

[7] OV-6c Operational Event Trace Description, Empire Challenge, May 22, 2006, Joint Systems Baseline Assessment, 2006.

[8] Sas Jacobs, Beginning XML with DOM and Ajax from Novice to Professional New York, APRESS, 2006, p. 118.

[9] Mark Baartse et al., Professional ASP XML, Birmingham, UK: WROX Press, 2000 p. 16.

[10] David Hunter et al., Beginning XML 4th ed Indianapolis, IN: Wiley Publishing, 2007, p. 142.

[11] Michael Kay, XSLT 2.0 3rd ed Indianapolis, IN: Wiley Publishing, 2004, pp. 493-522.

[12] Michael Kay, XPath 2.0 Indianapolis, IN: Wiley Publishing, 2004, pp. 447-457.

[13] Sas Jacobs, Beginning XML with DOM and Ajax from Novice to Professional, New York: APRESS, 2006, p. 17.

[14] David Hunter et al. Beginning XML 4th ed Indianapolis, IN: Wiley Publishing, 2007, p. 484.

[15] Procedures for Exporting Access data as XML, http://office.microsoft.com/en-us/access/HP030912931033.aspx (accessed August 17, 2009).

[16] Thomas Baekdal, (W)UPS - Package Tracking Usability - Articles - Baekdal.com. February 19, 2006. http://www.baekdal.com/articles/Usability/package-tracking-usability/ (accessed August 17, 2009).

[17] Rockwell Collins, Display System: Spot Beam. http://www.rockwellcollins.com/ecat/gs/Display_System.html (accessed August 17, 2009).

[18] John Osmundson, White Paper. “C-2-DCGS Situational Awareness.” GSOIS, Naval Postgraduate School, Monterey CA, March 2008.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California