6A.3 INTEGRATING NAWIPS INTO THE NEW NWS SERVICE ORIENTED ARCHITECTURE

Steve Schotz*[1], Jason P. Tuell[2], Scott Jacobs[1], David Plummer[1], Stephen Gilbert[1], Ronla Henry[2]
[1]NOAA / NWS / NCEP / NCO, Camp Springs, Maryland
[2]NOAA / NWS / OST, Silver Spring, Maryland

1. INTRODUCTION

AWIPS-II is a multi-phase project to develop a robust and extensible system to support the entire NWS enterprise mission needs. (See Tuell et al., 2008, for an overview of the entire AWIPS evolution project). The first phase of this project is migration of the existing AWIPS functionality for the Weather Forecast Offices (WFOs) and River Forecast Centers (RFCs) to the AWIPS-II architecture. AWIPS-II employs a modern Service Oriented Architecture (SOA) and is written primarily in the Java programming language. The migration is being conducted by Raytheon Technical Services Company (RTS) under the direction of the NWS Office of Science and Technology (OST) Systems Engineering Center (SEC) and is expected to be complete in 2009.

Another phase of AWIPS-II is to migrate the National Centers for Environmental Prediction (NCEP) AWIPS, hereafter referred to as NAWIPS, functionality to AWIPS-II. NAWIPS was developed to address NCEP unique requirements that are not available in the WFO/RFC AWIPS. AWIPS-II provides the opportunity to migrate NAWIPS functionality to a common architecture. This migration provides numerous benefits to the NWS including improving the collaborative forecast process among NWS components and the opportunity for the entire NWS to utilize common services provided by the AWIPS-II modern architecture.

The purpose of this paper is two-fold. First, the current NAWIPS system is described including major functionality, design goals and architecture. Second, a strategy to migrate NAWIPS functionality to AWIPS-II is presented including schedule, approach and major challenges.

2. NAWIPS BACKGROUND

NAWIPS includes a wide spectrum of meteorological applications and support functions for the visualization of meteorological data and generation of products. Used primarily to support NCEP operational forecast production, the system has been in use since the early 1990s. NAWIPS is developed and supported by NCEP Central Operations.

* *Corresponding author address:* Steve Schotz, NOAA / NWS / NCEP / NCO / Systems Integration Branch, 5200 Auth Road, Room 302, Camp Springs, MD 20746; email: Steve.Schotz@noaa.gov.

NAWIPS utilizes the General Meteorological Package (GEMPAK) to provide many of its core functions. GEMPAK was originally developed at NASA Goddard in the 1980's to support mesoscale meteorological research and later the university community for academic purposes.

The NAWIPS system is currently used by NCEP forecasters at the following forecast centers:

- Aviation Weather Center
- Climate Prediction Center
- Hydrometeorological Prediction Center
- Ocean Prediction Center
- Storm Prediction Center
- Tropical Prediction Center

The numerical model developers in the Environmental Modeling Center also visualize the output from both parallel and production models during development.

In addition, NAWIPS is run on the NCEP Central Computing System (CCS) super-computer to generate graphical products primarily from the forecast model output. These products are made available to the general meteorological community and the public on the NCEP web pages and public data servers.

Other users include the NWS Alaska and Pacific Regions for their National Center-like functions and select RFCs to support their quantitative precipitation forecast functions. Additionally, NAWIPS is distributed to the University Corporation for Atmospheric Research (UCAR) Unidata Program which provides NAWIPS to the university community (including about 200 universities) for academic and research purposes. Unidata also makes NAWIPS available to private industry users to support their commercial functions.

NAWIPS is used at the NCEP service centers to support the forecast process in the creation of various products including watches, warnings, advisories and guidance products. These products, in graphical, gridded and text formats, are distributed to NWS local WFOs; the aviation, marine, and general meteorological communities; and the public. From the early 1990s to the present, the NCEP service centers have undergone considerable modernization in coordination with the NWS, thus requiring continuing enhancements to the Information Technology (IT). Over the years, NAWIPS has been operationally supported on various UNIX and Linux workstation configurations, including single and multi-monitor systems, from various workstation vendors including HP, SGI,

IBM, Sun, and Dell and on the NCEP super-computer.

NAWIPS capabilities can be broadly divided into two categories: data visualization and integrated product generation. Data visualization is defined to be capabilities necessary to ingest, decode, perform calculations and display meteorological data. The following provide an overview of visualization abilities of NAWIPS. Critical elements to support NCEP unique mission requirements are highlighted.

To support the NCEP mission, NAWIPS is required to access a comprehensive set of global meteorological data including the following:

- Surface observations, e.g., METAR, Synoptic, ship and buoy reports, satellite derived winds
- Upper air data sets, e.g., rawindsonde, aircraft, pilot reports, and model generated soundings
- Imagery, e.g., radar, satellite.
- Grids generated from objectively analyzed surface and upper air data
- Model output, e.g., grids from NCEP global, and regional models, and other modeling centers including ECMWF, UKMET, etc., and Model Output Statistics (MOS)
- Lightning Data
- Forecaster generated text products and graphical products, e.g., watches, warnings, SIGMETs, etc.

The NCEP forecast mission encompasses a wide spectrum of geographic and temporal scales, from mesoscale to global, and from minutes to months, respectively. In addition, many of NCEP's forecast responsibilities are event-driven requiring different types of calculations and display options that cannot always be pre-defined. Therefore, the following requirements are critical:

- Flexible user-specified geographic and temporal scales
- User selectable map projections, e.g., Mercator, Lambert Conformal, Polar Stereographic, etc.
- User specified calculations on base data sets to create "on the fly" displays of derived parameters including:
    - Flexible grid diagnostics calculation tools
    - Surface and upper-air data calculation tools
    - Image processing calculations (such as cloud top temperature and height) and other scientific algorithms (such as Object Dvorak Techniques for tropical storm classification)
- Flexible user-selectable display options
- Flexible geo-referencing calculation and display functions

Integrated product generation is defined to be the creation and editing of graphical products overlaid on meteorological displays with all the visualization capabilities described in the previous section. Integrated product generation is critical to support the forecast process because meteorological data displays must be available to the forecaster as guidance or as a first guess for product creation. The following are additional critical product generation requirements currently satisfied by NAWIPS:

- Drawing and editing tools using meteorological objects such as fronts, weather symbols, watch boxes, jets, etc.
- Graphical objects earth referenced to properly navigate to user selected geographic areas and map projections
- Facility to import first-guess fields, e.g., model fields and forecaster generated fields.
- Object layering to support multi-component or multi-time concurrent product creation and editing.
- Product formats derived from graphical objects

3. NAWIPS DESIGN OBJECTIVES

NAWIPS customers represent a wide spectrum of the meteorological community from operations to research. The original customer base for GEMPAK in the 1980s was the NASA research community. Research requirements drove the architecture to support a high degree of flexibility. In the 1990's and 2000's, the NCEP service centers modernization drove later requirements, in particular the need to have a highly adaptive architecture that could support evolving user requirements while maintaining system robustness to support operational missions. The following provides critical design goals with brief supporting explanations and trade off discussions.

3.1. Maximize Functional Flexibility

As indicated in previous sections, the NAWIPS heritage, most particularly GEMPAK, was directed towards research requirements. Thus, functional flexibility was an early design goal. Functional flexibility drove the design of the following components:

- Generic calculation tools, in particular grid diagnostics and surface and upper-air parameter derivations from base data sets, are necessary. The grid diagnostics package is designed to allow user constructed calculations.
- Display capabilities including plotting attributes such as color, size, line type, line thickness, etc., are allowed to be user selectable as opposed to hard-coded l.
- Visualization and product generation functions are designed to operate over any

geographic region for supported map projections as opposed to fixed or pre-defined geographic scales.

Trade offs in adopting this design objective include the following:

- A lesser emphasis placed on the importance of performance. Flexible and generic calculation and display capabilities with minimal “hard coding” do not necessarily optimize performance.
- Initial development time is longer because developing more flexible and generic software requires more time to design and implement than developing less generic software. However, over the life cycle of the system it is likely that the initial development cost is recovered.

3.2. Maximize Code Re-use by Employing Functional Libraries

Layered functional libraries are designed to allow higher-level routines and applications within the architecture hierarchy to call the appropriate library routines. Additional design objectives in support of code re-use maximization include:

- Never write code more than once to perform a particular function.
- Functional libraries must include test programs to facilitate “black box” testing of individual library routines.
- The software must be robust to support operational mission requirements.
- The software must be centrally supported to minimize maintenance costs.

Trade offs include the following:

- Initial development time is longer in designing and developing functional libraries than developing software without code re-use considerations. However, the initial development cost is likely recovered as the software evolves. In addition, maintenance costs are lower over the life cycle of the system

3.3. Support Evolutionary Development for an Operational Environment

Supporting evolutionary development is critical in accounting for user feedback, less than well-defined requirements, and new requirements. As indicated earlier, the evolutionary development strategy has been crucial since the early 1990s to support NCEP modernization.

Trade offs include the following:

- Rapid prototyping is limited because NAWIPS must be robust for operational use.

3.4. Maximize Hardware Independence

The software is designed to eliminate, to the extent practical, dependence on specific vendor hardware or associated vendor specific software utilities. Additional goals associated with this objective include:

- Hardware independence allows the software to evolve beyond near-term hardware solutions.
- The software is developed, compiled and tested concurrently on multiple platforms to enhance system robustness.
- Hardware independence provides users with the flexibility to buy the “best value” hardware platforms for their environment.
- Drawing primitives and other instructions are written in software. Necessary device dependent primitives and system services, e.g., UNIX calls, are isolated in the lowest level device drivers and system services libraries.

The trade offs in adopting this design objective includes:

- Performance is not necessarily optimized because vendor specific solutions are avoided. However, over the NAWIPS life cycle, hardware performance improvements likely compensate for avoiding using vendor dependent features.
- More initial development time is required to develop software functions in lieu of using vendor specific solutions. However, as NAWIPS is ported to new platforms, the amount of required new development is greatly reduced

3.5. Minimize External Software Dependence

The software architecture avoids the use of vendor supplied software packages such as Relational Data Base Management Systems (RDBMS), Geographic Information Systems (GIS) and Computer Aided Design (CAD) systems. Goals associated with this objective include:

- Avoidance of external software systems simplifies NAWIPS installation at user sites.
- It allows flexibility in the way the software is configured on hardware in support of the wide variety of NAWIPS user environments.
- It reduces NAWIPS user required costs, e.g., software licenses.

Technical issues that influence the avoidance of external software systems include:

- RDBMS systems are not well suited for efficient access to grid and image data

sets. In addition, relational operations are not particularly useful for interactive meteorological systems.

- GIS systems provide some capabilities, e.g., earth referencing of data sets. However, GIS capabilities would have to be carefully integrated with other NAWIPS functional components to be beneficial.
- CAD systems provide drawing tools. However, meteorological objects are not provided by vendors and would have to be developed making use of vendor specific primitives.

The avoidance of external software systems in addressing NAWIPS requirements has also led to a relatively simple NAWIPS architecture. Each NAWIPS application is a single threaded executable. NAWIPS libraries are directly linked to each application. The notable exceptions are GEMPAK display programs that interface with the GPLT and device driver sub-processes. This relatively simple architecture has the following advantages:

- It maximizes NAWIPS configuration flexibility. NAWIPS can run on a single workstation, server, or super computer. No particular hardware configuration is assumed by the architecture.
- The NAWIPS is relatively easy to configure and install.

The “simplicity’” of the NAWIPS architecture has a potential significant drawback. The architecture is not designed to communicate with non-NAWIPS applications at a functional level. As indicated above, NAWIPS makes use of linked functional libraries. Parameter exchanges among applications and library routines are accomplished using subroutine parameters and global structures that are internal to the NAWIPS architecture. Significant redesign of the NAWIPS architecture would be required to make its services and capabilities available to external applications if these applications are not linked directly to GEMPAK libraries.

4. NAWIPS ARCHITECTURE

The NAWIPS includes the GEMPAK software and a set of GUI programs. The GEMPAK software provides the N-AWIPS core capabilities including data decoding, data analysis, navigation, display, and product formatting.

The NAWIPS architecture is comprised of eight major functional subsystems. This section includes brief descriptions of each sub-system function, and a listing of select functional libraries, where applicable.

To satisfy the code re-use design objective, NAWIPS programs are built on a functional library hierarchy. Each library contains a set of subroutines that perform a related set of functions. Note that a library and its respective subroutines names are denoted by two or three character designators and will be referenced in this manner in this document. For example, the SF library deals with surface file data access. The following sections describe each subsystem function, and list associated libraries.

4.1. Main Programs

The GUI programs offer the user a graphical interface to the data access and plotting and graphical object drawing of the other functional subsystems. Their primary focus is to satisfy operational users. GUI programs include:

- NMAP the primary program used by operational forecasters to perform data visualization and integrated product generation.
- NWX a program to display text products by graphical selection of issuing station or centers
- NTRANS a program that display pre-generated metafiles for numerical model fields

The GEMPAK programs were developed to provide the maximum amount flexibility in user-specification. They provide a textual interface to specific GEMPAK functionality. GEMPAK programs are divided into functional groups based upon data types, i.e., surface and upper-air observations, and grid datasets denoted by SF, SN, GD in their program names, respectively. Within each data category, several programs perform different functions, e.g., listing data versus plotting or contouring data. There are also programs to perform objective analyses (OA) and general graphics utilities (GP).

The GEMPAK programs have an advantage over the GUI programs in that they can be called from scripts with user input provided either interactively or via scripts. For example, the programs can be executed from scripts to create automatic graphical output in the super computer environment.

4.2. Integrated Graphical Product Generation (PGEN)

This subsystem is used by NMAP. It allows the user to draw and edit geo-referenced, meteorological objects in the display window. The objects can be overlaid on any NMAP data display. The Vector Graphics File (VGF) is used to store product generation graphics and associated product information.

The product generation palette is divided into controls, actions, classes and objects. The actions may act in either an “object” or “group” mode. In “object” mode, actions are performed on individual objects. In “group” mode, actions are performed on groups of objects only.

The user can draw many different classes of objects. Some objects are drawing primitives, such as lines, circles, text, symbols and markers. Others are meteorological in nature, such as SIGMETS, AIRMETS, severe weather watches, jet streams and fronts. All objects use the same editing tools, as appropriate for the type of object. That is, objects that are multi-point use the same tools to select, modify or delete the objects. Similar tools are used for single-point objects.

4.3. Decoders and Encoders

This subsystem contains the functions and programs to convert data from one format to another. Decoders read observation and model data from standardized data formats (e.g., METAR, Upper Air, GRIB, BUFR, etc.) and write data to standard GEMPAK surface, upper air and grid data files. Other decoders create standard ASCII files from products and data types not well suited to standard GEMPAK files.

Encoders convert from GEMPAK files to standardized formats. For example, GRIB messages are created from GEMPAK grid data files. There are also encoders to create BUFR messages from NAWIPS Vector Graphics (VG) files containing Aviation Weather parameters.

4.4. Coordinate Transformations

NAWIPS has several coordinate systems for plotting and drawing. The coordinate transformation sub-system transforms a list of X/Y points from any of the coordinate systems into any other system. The coordinate systems are arranged in a hierarchy. The D coordinate system (Device) is at the bottom while the G system (Grid) is at the top. It is advantageous for application programs to use the highest possible coordinate system that fits the application.

When objects are plotted, their locations are in units of one of the coordinate systems. Every plot command identifies a coordinate system. When plotted items exceed the boundaries of the system in which they are plotted, the items are clipped. The following is a brief description of each coordinate system.

- D coordinates are the native coordinates of the currently selected graphics device. Every device is different. For example, if a device is a 1024 by 1280 display screen, the coordinate units are screen pixels. The diagonal corners of the coordinate system are (0, 0) and (1024, 1280). Vertex (0, 0) may be any corner as defined by the device.
- N coordinates are normalized device coordinates. N coordinate limits range from 0 to 1 along the longest dimension and from 0 to a value less than or equal to 1 along the other axis. The limits are determined as a linear scaling of the device coordinates.
- Correction for the pixel aspect ratio is included in the transformation from D to N coordinates. Vertex (0, 0) is the lower left corner of the plot space. When a view region or margins are set, two clipped versions of the N coordinate system result:
  - V coordinates, view coordinates, are the same as normalized coordinates, but the clipping boundaries correspond to the view region.
  - P coordinates, plot coordinates, are the same as normalized coordinates, but the clipping boundaries correspond to the region inside the margins.
- M coordinates are the Earth or graph coordinates, depending on the plotting mode (map or graph). Clipping boundaries are the same as for plot coordinates.
  - When plotting in map mode, M coordinates are the Earth's latitude/longitude system. Map projections are set to define a system that transforms any point on the Earth into a (possibly infinite) plane, a sub-region that is scaled to the normalized coordinate system and clipped as needed.
  - When plotting in graph mode, M coordinates are those coordinates defined at the time that the graph is initially set. For example, if both graph axes are defined to range between 0 and 100, then a point at (50, 50) is plotted in the center of the graph while a point at (120, 300) is clipped.
- G coordinates are grid coordinates. The X and Y in this coordinate system are the rows and columns of a grid. Using the inverse of the projection functions, grid point coordinates can be translated into Earth coordinates. The grid projection is independent of the map coordinate projection; the grid can be displayed on a projection different from the one native to the grid. Clipping is the same as for plot coordinates.

In addition to the D, N, V, P, M, and G coordinate systems, there are additional coordinate systems used internally by the coordinate transformation sub-system. For example, the plane into which M coordinates are transformed defines an L coordinate system that is linear. Since the L system is different for each map projection, application programmers never use it.

4.5. Drawing Tools

The Drawing Tools allow the programmer to call device independent attribute setting and

drawing functions. These device independent functions are then translated into device specific commands. In this way, the graphical output of the programs can be made to be nearly identical no matter the specific device driver.

4.6. Scientific Calculations

The Scientific Calculations subsystem contains a comprehensive set of tools to perform on the fly computations on meteorological base data sets. The subsystem includes two sets of libraries described in the following sections.

The grid diagnostics includes a powerful set of operators that perform calculations on grids. They include a suite of pre-programmed diagnostic functions for calculating standard meteorological quantities. In addition, user-specified diagnostics can be constructed by combining the pre-programmed diagnostics with differential, algebraic, or trigonometric operators. The grid diagnostics operators can be combined, when executing the programs using a Reverse Polish Notation (RPN)-like syntax. The grid diagnostics libraries manage and perform grid calculations on scalar and vector grids.

The parameter computations contain functions to compute meteorological parameters from surface and upper air data sets.

Parameter computation libraries perform the following types of computations:

- Unit conversions, e.g., Celsius to Fahrenheit
- Meteorological calculations from base data, e.g., relative humidity from temperature and dew point
- Meteorological character translations, e.g., cloud cover character string from numerical coded value

4.7. Data Access

The GEMPAK Data Management (DM) file format stores grid and station data in such a way as to maximize performance. Unlike other data formats designed for other purposes, such as GRIB (a data transmission format), the GEMPAK DM format was designed to meet the needs of the interactive user. The DM functions used to access the GEMPAK data files allow for data independent reading and writing and interpretation of the data for use in plotting and listing.

The GEMPAK Vector Graphics file format is used to store and retrieve the drawing objects for product generation.

For images, GEMPAK accepts both McIDAS and NIDS format for display and analysis.

4.8. Foundation Tools

The foundation libraries contain generic functions and routines that are utilized across the entire system. The functionalities in this subsystem include: string manipulations, time calculations, file and table access, operating system specific functions and geographic calculations.

5. MIGRATION STRATEGY

5.1. The Schedule

Planning is underway for the migration of NAWIPS into the AWIPS-II SOA. This planning will continue throughout Fiscal Year (FY) 2008. Execution of the plan will occur during FY2009 through FY2010 with OT&E and deployment to the users expected in FY2011.

5.2. The Team

The planning, development, testing and deployment of the migrated NAWIPS system are the main responsibility of the current NAWIPS developers. This work will be done in collaboration with the OST AWIPS-II team and RTS.

5.3. The Tasks

Planning activities include gap analyses of software, hardware, networking and data flow. These four analyses will result in individual assessments of need, providing the foundation for the actual migration strategy and plan. Standard project management techniques and theory are being applied to all migration planning, scheduling, training and risk analyses, as well as the migration itself.

Gap analysis requires the mapping of current NAWIPS functionality to AWIPS-II functionality. Where NAWIPS functionality exceeds the AWIPS-II capability, the NAWIPS team must extend the functionality of AWIPS-II. With respect to software functionality gaps, there are several potential implementation approaches that will be assessed by NAWIPS team. These include writing plug-ins, extending existing services, wrapping existing NAWIPS functionality or some combination of approaches. The analysis must also take into account any hardware, network or data discrepancies. For example, the NCEP forecasters have a need to display experimental runs of the numerical models. Since these data sets are not distributed to the general user community, the NAWIPS team needs to plan for development that will include ingesting this data into the AWIPS-II database.

During FY08 prototyping is being conducted to get a better sense of what will be involved in implementing the current NAWIPS functionality within the new AWIPS-II architecture and to gain experience using AWIPS-II services and development tools. For example, the team was able to write an ADE plug-in that decodes storm information from the Tropical Cyclone Forecast Advisory (TCM) text product. This product is issued

by the National Hurricane Center every six hours on current tropical systems. The prototype decoder extracts advisory identifying information such as the storm name, type of storm, basin, advisory number and issue time. In addition, the current and forecast storm positions and extents of the radial winds greater than 64, 50, and 34 knots are decoded, and all information is stored in the PostgreSQL database tables created and maintained by the ADE.

Raytheon has also provided training on creating plug-ins for their visualization tool CAVE, which allowed us the opportunity to see how we could request and display the tropical storm forecast data we had just decoded and stored. By modifying a sample JavaScript program, we were able to request data from a specific forecast advisory from the ADE database and receive it in CAVE using functionality already in the AWIPS-II system. Once we had the storm position and wind forecasts, a little more programming effort was involved to create the plug-in that actually displayed the data. We took advantage of the GeoTools GIS toolkit, included in AWIPS-II, to perform the geospatial calculations used to determine the geographic extent of the tropical cyclone winds.

6. CHALLENGES

There are numerous challenges facing the development team. These challenges include updating skill sets, resource contention and technical issues. These challenges and some mitigation strategies are described in the following paragraphs.

The NAWIPS developers must become familiar the AWIPS-II architecture, development environment tools, and services. Since all of the software in the current NAWIPS system is written in FORTRAN and C using X and Motif libraries for GUI support, the use of Java and Java support tools are new to the developers. Also SOA is a new concept for the NAWIPS team. Mitigation preparation includes providing training in these technologies. For example, the development team is attending training classes on SOA and Java. Time has been allotted in the schedule for training, accordingly. In addition, as mentioned previously, some time has been allocated in FY08 for limited prototype development activities for the NAWIPS team to gain experience in the AWIPS-II environment.

To allow for an efficient migration of NAWIPS, a moratorium on changes to the existing system will be instituted. This moratorium will impact NAWIPS customers who are accustomed to receiving functionality updates on a quarterly basis. The team realizes that bug fixes and table updates may be necessary during the moratorium period. These fixes will have to be accounted for in the planning of migration activities.

In addition to migration activities, NCEP is also planning to move to a new building in FY09. Many of the same people involved in the NAWIPS migration will be called upon to assist with the move planning, testing and execution. Moving activities will be factored into the migration plan.

There are several technical challenges in migrating the current NAWIPS functionality to the new architecture. Some significant challenges are described here.

First, the execution of GEMPAK applications in batch mode is used heavily to support pre and post-processing requirements in the forecaster environment as well as the creation of automated products on the CCS. This functionality must be preserved in the AWIPS-II environment. While the batch processing on the workstations and servers may work well under the AWIPS-II architecture, the unique environment of the super computer is likely to pose a significant challenge.

Another challenge facing the NCEP and outside users is the proliferation of locally developed applications. The developers at the individual sites will have to work closely with the migration development team to ensure these applications continue to work under the new system. The migration team is planning on conducting training sessions for the local developers to help mitigate any risks.

Similar to the locally developed applications, the user community has developed numerous UNIX shell scripts to execute the GEMPAK applications, on workstations and the super computer. Another goal of the migration team is to minimize the impact on these scripts. In order to accomplish this goal, there must be an interface to applications similar to the current GEMPAK interface.

The team will strive to make the user interface to the GUI programs, e.g., NMAP2, match the current system to minimize user impact. This goal may not be possible in all aspects of the graphical interface. In those cases where the interface must change, the migration team plans on providing documentation and conducting any necessary training for the users.

Another challenge is to preserve the appearance of NAWIPS output products. Again, the team will work toward having no differences in any center products. However, in those situations where changes are unavoidable, the team will work closely with the appropriate users to ensure as much similarity as possible.

NAWPS users outside NCEP, e.g., the university community, pose another challenge. This class of user will most likely not have access to an AWIPS-II database. Therefore, the NAWIPS migration will have to take into consideration how these users will access their data sets.

The AWIPS-II plans call for deliveries of the RTS-developed code to occur through June 2009 Thus, the deliveries span a large part of the NAWIPS migration period. Some aspects of the

AWIPS-II architecture might change as functionality is developed. To mitigate this risk, members of the NAWIPS group are working closely with OST by for example participating on the AWIPS-II Independent Verification and Validation team, and attending meetings and training classes hosted by RTS.

7. CONCLUSION

The migration of NAWIPS into the AWIPS-II SOA environment presents both challenges and opportunities. NAWIPS has many unique features that must be migrated to the new environment to satisfy user requirements. The opportunity exists to add this rich set of features to AWIPS-II for all users. The challenges to the migration, while significant, are not insurmountable. The migration team is developing an approach that attempts to minimize the impact to the users and minimize risk.

The views expressed are those of the authors and do not necessarily represent those of the National Weather Service.

8. REFERENCES

Tuell, J.P, R. K. Henry, J. D. Lawson, F. P. Griffith, 2008: The Evolution of AWIPS – Progress and Future Plans and Status. *24th Conf. on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology*, New Orleans, LA,, , Amer. Meteor. Soc., 6A.1