

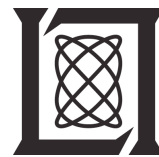
**Project Report
ATC-412**

An Applications Architecture to Support FAA Wake Turbulence Mitigation Systems Development and Deployment

Vaibhav K. Andleigh
David A. Clark

9 October 2013

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



Prepared for the Federal Aviation Administration,
Washington, D.C. 20591

This document is available to the public through
the National Technical Information Service,
Springfield, Virginia 22161

This document is disseminated under the sponsorship of the Department of Transportation, Federal Aviation Administration, in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

1. Report No. ATC-412		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle An Applications Architecture to Support FAA Wake Turbulence Mitigation Systems Development and Deployment			5. Report Date 9 October 2013		
			6. Performing Organization Code		
7. Author(s) Vaibhav K. Andleigh and David A. Clark, MIT Lincoln Laboratory			8. Performing Organization Report No. ATC-412		
9. Performing Organization Name and Address MIT Lincoln Laboratory 244 Wood Street Lexington, MA 02420-9108			10. Work Unit No. (TRAIS)		
			11. Contract or Grant No. FA8721-05-C-0002		
12. Sponsoring Agency Name and Address Department of Transportation Federal Aviation Administration 800 Independence Ave., S.W. Washington, DC 20591			13. Type of Report and Period Covered Project Report		
			14. Sponsoring Agency Code		
15. Supplementary Notes This report is based on studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology, under Air Force Contract FA8721-05-C-0002.					
16. Abstract The Wake Turbulence Program within the FAA is considering a number of new procedures for safely reducing wake vortex spacing requirements between aircraft. One category of procedures investigates wind-dependent procedures, i.e., procedures that can be applied when wind conditions are expected to transport the wake from a lead aircraft away from the path of a trailing aircraft. MIT Lincoln Laboratory developed a Wind Forecast Algorithm (WFA) to determine when conditions allow these wind-dependent procedures to be available to traffic managers. The baseline WFA is used within the Wake Turbulence Mitigation for Departures (WTMD) system, which establishes spacing procedures for departures on closely spaced parallel runways. A number of new procedures are also under consideration, each of which will require a modification and/or expansion of the baseline WFA. With time, the volume and number of disparate data sources used in the development process has steadily increased to the point where the existing development environment had become cumbersome and inadequate. As a result, through support of the FAA Wake Turbulence Program, MIT LL has undergone a complete overhaul of the computer processing and storage architecture used for WFA development. This will serve two main purposes. First, it will greatly expedite the development process, which is highly iterative and requires increasingly large volumes of data. Second, an updated architecture design will allow for an expeditious transition of developmental systems into the operational environment within FAA's NextGen framework. A key focus of this report describes how the new design is sufficiently compatible and flexible to serve within this anticipated FAA framework. The unified application architecture and infrastructure being designed and implemented will support continuing development, playback requirements, and real-time deployments. This architecture is composed of several application components including a wind data extract-transform-load (ETL) application, the WFA algorithm, and a display interface to accommodate both the development process and for potential use within the FAA operational environment. The Wind-ETL application component acquires, processes, and archives wind data from a variety of NOAA-based hourly forecasts and airport-vicinity weather measurement equipment. This wind data is ingested by the WFA, which computes and disseminates its availability predictions to the WTMx Display application component, which archives these predictions and also allows for presentation to the airport tower supervisor via the WTMx display user interface decision support tool. This architecture is designed to be flexible to accepting new weather data feeds, scalable to the high bandwidth and processing and storage capabilities required, provide sufficient automation and self-healing capabilities, and portable to allow its introduction into alternate facility sites and its integration into other FAA software systems.					
17. Key Words			18. Distribution Statement This document is available to the public through the National Technical Information Service, Springfield, VA 22161.		
19. Security Classif. (of this report) Unclassified		20. Security Classif. (of this page) Unclassified		21. No. of Pages 61	22. Price

This page intentionally left blank.

ABSTRACT

The Wake Turbulence Program within the Federal Aviation Administration (FAA) is considering a number of new procedures for safely reducing wake vortex spacing requirements between aircraft. One category of procedures investigates wind-dependent procedures, i.e., procedures that can be applied when wind conditions are expected to transport the wake from a lead aircraft away from the path of a trailing aircraft. MIT Lincoln Laboratory developed a Wind Forecast Algorithm (WFA) to determine when conditions allow these wind-dependent procedures to be available to traffic managers.

The baseline WFA is used within the Wake Turbulence Mitigation for Departures (WTMD) system, which establishes spacing procedures for departures on closely spaced parallel runways. A number of new procedures are also under consideration, each of which will require a modification and/or expansion of the baseline WFA. With time, the volume and number of disparate data sources used in the development process has steadily increased to the point where the existing development environment had become cumbersome and inadequate.

As a result, through support of the FAA Wake Turbulence Program, MIT Lincoln Laboratory has undergone a complete overhaul of the computer processing and storage architecture used for WFA development. This will serve two main purposes. First, it will greatly expedite the development process, which is highly iterative and requires increasingly large volumes of data. Second, an updated architecture design will allow for an expeditious transition of developmental systems into the operational environment within FAA's NextGen framework. A key focus of this report describes how the new design is sufficiently compatible and flexible to serve within this anticipated FAA framework.

The unified application architecture and infrastructure being designed and implemented will support continuing development, playback requirements, and real-time deployments. This architecture is composed of several application components including a wind data extract-transform-load (ETL) application, the WFA algorithm, and a display interface to accommodate both the development process and for potential use within the FAA operational environment. The Wind-ETL application component acquires, processes, and archives wind data from a variety of NOAA-based hourly forecasts and airport-vicinity weather measurement equipment. This wind data is ingested by the WFA, which computes and disseminates its availability predictions to the WTMx Display application component, which archives these predictions and also allows for presentation to the airport tower supervisor via the WTMx display user interface decision support tool. This architecture is designed to be flexible to accepting new weather data feeds, scalable to the high bandwidth and processing and storage capabilities required, provide sufficient automation and self-healing capabilities, and portable to allow its introduction into alternate facility sites and its integration into other FAA software systems.

This page intentionally left blank.

TABLE OF CONTENTS

	Page
Abstract	iii
List of Illustrations	vii
List of Tables	xi
1. INTRODUCTION	1
2. DESIGN REQUIREMENTS	3
2.1 Backdrop to Current Status	3
2.2 Meeting Future Data Requirements	4
2.3 Meeting Future Client Needs	5
2.4 Future Application Requirements	15
3. WEATHER DATA DESCRIPTIONS	17
3.1 Overview of Data	17
3.2 Forecast Model Data	18
3.3 Airport Surface Observation System Data Feeds	23
3.4 Aircraft Wind Data	24
3.5 LIDAR Data	25
3.6 Data Source Summary	25
4. DATA SYSTEM DESIGN	27
4.1 Data Acquisition	27
4.2 Data Processing	28
4.3 Data Archiving	31
4.4 Weather Data Feed Infrastructure	31
5. APPLICATION ARCHITECTURE	35
5.1 Review of Requirements	35
5.2 Wake Turbulence Application Architecture (WTAA)	35
6. SUMMARY AND FUTURE WORK	57

TABLE OF CONTENTS
(Continued)

	Page
Glossary	59
References	61

LIST OF ILLUSTRATIONS

Figure No.		Page
1	Application usage modes for the WTMx suite of applications.	6
2	Overview of the purpose of business process modeling and the benefits it offers.	7
3	WTMx concept of operations in which the WFA algorithm acquires weather data and airport configuration data to generate availability outlook predictions that are presented to the tower controller to aid in decisions in response to pilot requests for scheduling a preferred runway slot.	8
4	Business process flows for the WFA application with regards to weather data processing, WFA algorithm operation, and presentation via the WTMx display.	9
5	Business process flows for the WFA applications in Analysis Mode, which supports over 50 airports and can run using very long durations of weather data.	11
6	Business process flows for the WFA application in Playback Mode which spans shorter durations of weather data at a single airport.	13
7	Business process flows for the WFA application in Real-Time Mode, which uses real-time data feeds for a single airport.	14
8	Weather data is acquired from NOAA-based hourly forecast models as well as real-time, airport-vicinity weather measurement equipment.	17
9	Each grid point in a single NOAA forecast contains dozens of different weather products, of which only a subset are of use to WFA.	18
10	Weather forecasts generated by several NOAA products provide wind forecast data from zero up to 24 hours into the future, updated every hour.	19
11	Weather forecast products by NOAA provide weather data at latitude and longitude resolution resulting in up to millions of data grid points per forecast.	20
12	Weather forecast products by NOAA provide weather data at up to 50 levels of altitude resolution in either pressure level (P) or natural level (N) coordinates.	21

LIST OF ILLUSTRATIONS (Continued)

Figure No.		Page
13	Weather data processing sequence starting from the acquired data and its filtering to specific airport wind data.	29
14	The WTMx application architecture emerges as composed of the Wind-ETL and WTMx Display application components, in addition to the WFA algorithm itself and databases.	36
15	High-level wake turbulence applications architecture illustrating data flows between key role players. This schematic uses WTMA as the example mitigation procedure.	38
16	Detailed WTMx application architecture illustrating the data flows amongst the WTMx application components.	39
17	Data flows amongst the Wind-ETL related application components in the WTMx application architecture.	40
18	Progress on the Wind-ETL related components, in which green arrowheads indicate it is working and red stop buttons indicated not yet functional.	41
19	Display concept of the WTMA Display user interface component.	43
20	Data flows among the WTMx Display-related application components for each of the three application modes.	45
21	Current progress on the WTMx Display-related application components, in which green arrowheads indicate it is working and red stop buttons indicate not yet functional.	47
22	Database technology selection is driven by the data flow rates and archive sizes of each of the three databases introduced as part of the WTMx data architecture.	48
23	Enterprise integration in the WTMx data architecture, in which wind data and predictions data in each message broker topic can be independently originate with one or more publishers (on the left) shared with a variety of subscribers (on	

LIST OF ILLUSTRATIONS (Continued)

Figure No.		Page
	the right). The use of Enterprise Service Bus (ESB) adapters permit applications to connect to the message broker.	52
24	WTMx data architecture supports three deployment topologies including independent airport deployments, deployments using a shared wind data aggregator, or deployments with a centralized aggregator and WFA, which is piped to each airport site-based display (specific airport site references are strictly notional).	53

This page intentionally left blank.

LIST OF TABLES

Table No.		Page
I	Summary of Data Acquisition and Archiving Requirements	28

This page intentionally left blank.

1. INTRODUCTION

Over the past decade, the FAA Wake Turbulence Program has investigated a number of procedural changes for safely reducing aircraft spacing imposed to avoid hazards associated with wake vortices generated from aircraft lift (Tittsworth et al., 2012). Many of the existing wake spacing standards were established decades ago. Since that time, there has been an enormous increase in the understanding of wake vortex generation, behavior, and decay, and in the impact of wake vortices on specific aircraft types. Efforts to develop new procedural changes have been made possible through these research advances.

Procedural changes that have recently been implemented under the direction of the Wake Turbulence Program, and additional changes under development, can be broadly categorized as weather-independent and weather-dependent. Weather-independent solutions include reduced aircraft spacing for specific configurations of closely spaced parallel runways (CSPR), and more sophisticated re-categorization of aircraft based on observed wake impact rather than simply by weight. Weather-dependent solutions allow for a safe reduction in spacing only under specific weather conditions.

The initial weather-dependent wake turbulence avoidance solution is the Wake Turbulence Mitigation for Departures (WTMD) system, which allows for a safe reduction of spacing for departures on CSPR. Existing wake procedures require a wait of up to 3 minutes for a departing aircraft following departure of a Heavy aircraft or B757 on the adjacent CSPR. The WTMD procedure safely eliminates this wait on the upwind runway during persistent crosswinds up to 1000 feet above ground level (AGL), which prevent transport of wakes from the leading aircraft on the downwind runway from entering the path of the trailing departure aircraft. WTMD requires a Wind Forecast Algorithm (WFA) to determine when favorable crosswind conditions are expected to persist to allow this procedure. The baseline WFA was developed by MIT Lincoln Laboratory, and a prototype WTMD system is currently undergoing operational evaluation. Within the FAA Wake Turbulence Program, analogous wind dependent solutions are also being investigated to address a host of other procedures, which includes arrivals on CSPR (Wake Turbulence Mitigation for Arrivals, or WTMA), arrivals and departures on single runways (WTM-SR), time paired departures (TPD), paired approaches (PA), and en route operations (ER). The baseline WFA will require adaptation and modification in order to meet the specific needs and requirements of each of these applications.

Development of the baseline WFA by MIT Lincoln Laboratory required extensive performance analysis to test the sensitivity and reliability of the algorithm. A multitude of disjoint data sources (both observations and numerical weather model output) was required both as potential input to the algorithm, and as validation to measure performance. These data sources continued to grow both in type and volume as development continued, with an increased strain on the capacity for data storage, efficiency and retrieval, and flexibility for processing and testing. It became apparent that an improved architecture for data acquisition, storage, retrieval, and processing was necessary in order to effectively meet the demands for the continuing development for the various planned applications under consideration. This report

describes the major architecture overhaul that has been implemented as an investment toward meeting these demands.

In designing the architecture, other considerations were taken into account beyond storage and retrieval efficiency. These include designing a data and processing capability that will expedite the development process in terms of serving various iterative analyses, and supporting a demonstration capability that includes playback functionality. Also taken into consideration was compatibility with potential use as a Decision Support Tool (DST) component within the existing FAA platform framework, and the future framework anticipated as part of the next generation FAA system (NextGen).

2. DESIGN REQUIREMENTS

2.1 BACKDROP TO CURRENT STATUS

Much of the early work at MIT Lincoln Laboratory focused on the algorithmic development of a WFA for a WTMD feasibility prototype, driven by detailed analysis of benefits and safety in the context of airport-specific weather and runway geometry configurations. This proof-of-concept prototype that established the baseline WFA was delivered to the FAA in 2009. Both a MATLAB version (for offline bulk analysis) and C-based version (for real-time demonstration) of the WFA were created as part of the development process. Subsequently, this baseline WFA has been used as a core component for ongoing adaptation of the WFA to support WTMA. A rudimentary WTMA display was also developed in MATLAB to aggregate and display key wind and WTMA availability outlook information that may ultimately be suitable for viewing to support air traffic management decisions.

During the early research and development efforts of WTMD and WTMA, many of the Lincoln Laboratory supporting activities were performed in a rapid prototyping, ad-hoc manner intended to maximize flexibility and minimize development time. Current (at the time) and legacy weather forecast model output data were acquired and supported, including multiple resolution (40- and 13-km) Rapid Update Cycle (RUC) model and Automated Surface Observing System (ASOS) data, using manual data acquisition scripting processes such as MATLAB and Perl scripts. Data were manually archived to a cartridge-style storage system.

While this approach was sufficient for the data throughput and performance needs in early development, the proliferation of potential wake turbulence mitigation techniques being investigated (collectively referred to in this document as WTMx), coupled with an expansion in the number and size, as well as increased performance and latency requirements of data acquisition and processing, has made the current set of manual processes insufficient for the current and growing needs with regard to labor resources and system requirements.

These data feed expansions include the official deployment by the National Ocean and Atmospheric Administration (NOAA) in May 2012 of a new operational weather model for serving FAA weather forecast requirements, the Rapid Refresh Model, referred to as RAP, that has replaced the previous generation RUC model. Additional expansions include further development (in various forms) using NOAA's now-experimental High Resolution Rapid Refresh (HRRR) model (representing about a 20-fold increase in data output), as well as algorithm research requiring the acquisition and integration of aircraft weather observations and other ground-based weather feeds (e.g., multiple source light detection and ranging (LIDAR) data). The perishability of some data at its source, sometimes as short as a few hours in the case of national model output data, requires near 24-hour acquisition and computational processes to avoid data gaps and provide timely access of processed data for algorithm benefits and safety analysis.

Over time, the manual processes could no longer scale with the exponentially increasing data sizes and processing times with reduced allowances for latency.

In addition, the WFA prototype design does not currently handle these new wind data feeds, and the algorithm science associated with developing the integrated wind predictions in WFA may also change based on future analyses of these incoming wind data sets. Finally, the first-look WTMA display was built specifically for research purposes, and was never meant for deployment to an actual field site.

The inherent fragility of the legacy approach can also lead to significant hardship in recovering from short duration equipment or network outages, as well as considerable extra effort in integrating from a disparate range of weather data feeds that differ in data format, code/script behavior, and inconsistent handling of data outlier and single point data omissions.

In short, the early positive results of the benefits and safety analysis during the development of WTMD led to a significant expansion of WTMx development scope along with its data and application system requirements, and it simply outgrew its incipient processes and structure. The rapid expansion of new weather data feeds affords an opportunity to redesign these processes and structure in a manner that can overcome these limitations and scale well into the future with enhanced capability and automation.

2.2 MEETING FUTURE DATA REQUIREMENTS

To overcome these obstacles, an entirely new approach is presented here for developing a scalable architecture. The new data architecture is designed to handle the increased workload associated with the ever increasing number and diversity of weather data feeds and increased data processing and archiving requirements with reduced latency.

The new data architecture is optimized for acquiring, processing, and archiving large, high-resolution wind data sets in varying formats from a variety of sources. Furthermore, this new approach provides more consistent data acquisition and handling in an automated fashion while simultaneously improving accessibility of pre- and post-processed data over long time frames asynchronously. Finally, these new processes are designed such that they can be easily integrated into other systems in adherence to modern network-centric, enterprise integration architectures espoused by both Lincoln Laboratory and the FAA. Recent efforts within the FAA, and government systems in general, have shown a propensity toward integrating closed piped airport software and hardware to provide unified situational awareness to their clients and associated decision support tools by promoting data sharing among disparate systems as a means of improving the accessibility, utility, and timeliness of the data being shared. The improved availability of these data among collaborating parties enables air traffic controllers and airlines and pilots to operate in a more strategic and coordinated fashion for improved efficiency and safety of flight operations.

Enterprise system integration is anticipated to occur at many different levels within the evolving FAA architecture, and will occur in the form of data sharing, application portability, and application integration. For instance, this architecture affords the capability to share not only WFA guidance and outlook predictions, but also related and supporting data products as well. Through the use of enterprise messaging, these data can be freely shared and distributed within a single site or across multiple disparate sites independent of operating system or software programming language through the use of self-describing, open data formats and open standards. The WTMx display application being developed uses a service-oriented architecture (SOA) based on open web formats, consisting of a set of loosely coupled software modules operating in a distributed computing environment, and can be deployed as a stand-alone application or embedded into other FAA applications using well known integration techniques once the deployment plans are formulated. While no WTMx testing framework has been formally specified at present, the data sharing capability and application portability features of the WTMx data architecture described here will also facilitate future test framework integration by pre-wiring access points for data exchange.

Another issue to take into consideration relates to the WTMx deployment approach that will be used during the test evaluation phase and subsequent operational phase, specifically about whether to run the bulk of WTMx data acquisition and processing at a single site, or independently across multiple sites, or even across multiple sites in a primary/backup-failover application topography. While no formal deployment decision has been made in this regard at this time, the WTMx architecture proposed here is agnostic to whatever deployment strategy is selected, and can accommodate any or all of them. This flexibility is enabled by the use of a component design in this architecture incorporating sequential data flows and its incorporation of multiple data buses at key data sharing access points along multiple stages in these data flows. As the program level deployment strategies become more definitive, the appropriate data connectors can be included as part of this architecture to transfer and share data as required, and facilitate deployment integration. This will be discussed in significantly more detail throughout Section 5.2.

2.3 MEETING FUTURE CLIENT NEEDS

To further assess the data processing and data infrastructure requirements, it is beneficial to identify how the WTMx suite of applications will be used to support the WTMx program going forward.

The new wind data infrastructure is designed to support development of wake mitigation algorithm development and testing through all of its *modes of use* including:

Analysis Mode (also known as research and development mode) using wind data over long time ranges spanning multiple years for the purposes of examining, testing, and validating algorithm performance and safety verifications over suitably long time periods. Included is cross-comparison of performance across the transition boundary of new operational models as they are introduced, such as what was required when the RAP model replaced RUC.

Playback Mode using wind data over several days but running in real-time or “fast time” to demonstrate the algorithm in action and its capabilities and operational workflow to interested parties.

Deploy Mode using real-time wind data feeds and running in real-time at an airport as part of planned operational tests and operational deployments, consistent with current and future anticipated system environments.

These three usage modes are described in more detail in Figure 1.

-
- **Analysis Mode**
 - to **train & evaluate** algorithm(s) at **select airports** over **large expanses of time**
 - performance **less stringent** given **flexible deadlines**
 - **Playback Mode**
 - to **illustrate results** of **WFA** and **real-time display** on **stored input data** over **shorter periods of time** at **select airports**
 - **Real-Time Mode**
 - to **prototype deployable** **WFA** and **real-time display** operating on **single airport**
 - require **high performance** to ensure **safety of WTMx guidance**

Figure 1: Application usage modes for the WTMx suite of applications.

Finally, the application infrastructure must be designed to be sufficiently flexible and portable such that it can be easily adapted to ingest alternate wind data feeds in other environments or organizations, and to allow the wake mitigation prediction displays to be embeddable in other existing and future software systems with minimal modification.

To support all three modes required for algorithm development, the WTMx application architecture is designed to offer a distinct application mode for each of the development modes identified above. The

next few sub-sections of this report explore each of these application modes as it pertains to application architecture.

2.3.1 Business Process Overview

Having characterized the modes of use of the WTMx application suite, it is now useful to examine the series of tasks that these applications must perform to satisfy the client needs. The set of tasks required to achieve a specific service for the customer defines a *business process*, and any system can be described as a set of processes operating in unison to collectively achieve the system objectives. This abstraction of the duties of a system forms the basis for Business Process Modeling (BPM), which is a useful system design technique for identifying and modeling the interactions between the client and the technologies that compose the (usually) complex system being designed, as presented in Figure 2. Over time, as the modeling efforts progress, yielding ever increasing detail, a clearer representation of the structure and behavior of the system, its underlying subsystems, and its components emerges, along with a specification of key interaction points with the client user(s).

Business Process = set of tasks that must be performed to achieve a specific service for a customer

Objective

- to **model** how the **users** and **technology** within a system will **interact** to achieve specific **business goals**

Benefits

- gain clearer **definition** of the **roles** of **users & technology** within a system, and **workflows**
- **identify** and **specify** the high-level **subsystems** and **components necessary** to achieve these goals
- leads to **development** of high-level **architecture** that **meets all of these goals** in a manner **acceptable to the customer (user)**

Figure 2: Overview of the purpose of business process modeling and the benefits it offers.

A general concept of operations (CONOPS) illustrating the inclusion of the WFA algorithm as part of an operational WTMx system and procedures for scheduling airport runway operations is shown in

Figure 3. Though formal CONOPS have yet to be developed beyond WTMD, a general concept is presented here in the context of a rationale for architecture design. During WTMx procedures, the wind data feed infrastructure and airport configuration data are fed as input to the WFA algorithm platform, which provides the wind situational awareness and availability outlook in the form of some graphical user interface (GUI) display to the tower controller. Depending on the favorability of the outlook, the tower controller can use the WFA predictions to efficiently schedule aircraft to use existing runways in the most efficient manner without compromising aircraft safety. The controller can then communicate these runway slot schedules to the pilots associated with these aircraft and direct them accordingly.

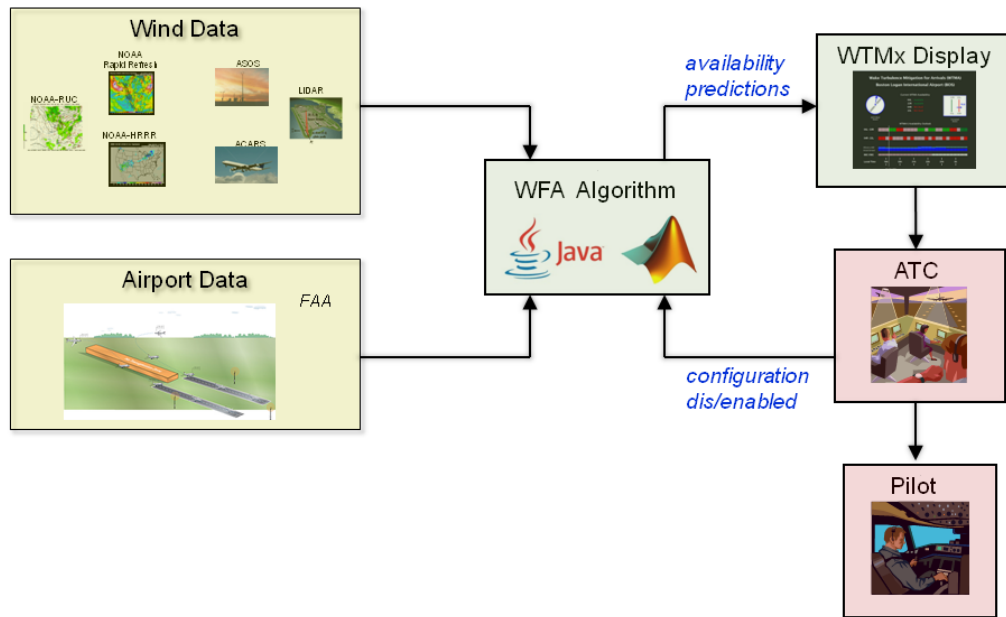


Figure 3: WTMx concept of operations in which the WFA algorithm acquires weather data and airport configuration data to generate availability outlook predictions that are presented to the tower controller to aid in decisions in response to pilot requests for scheduling a preferred runway slot.

The diagram in Figure 3 in effect visually encapsulates the principal *business process model* for applying any WTMx procedure(s) into airport operations. While specific details in the implementation and deployment of a particular WTMx procedure may vary from one procedure to another or by airport, the core business process model will be common to all, and by extension, the data architecture supporting them. By abstracting the wake turbulence data architecture in this manner, it is anticipated that this architecture will be able to support existing and ongoing analysis efforts on WTMD and WTMA, as well as extensions to other proposed WTMx systems and the adoption of additional wind data feeds as they become available.

An expansion of the BPM abstraction in Figure 3 leads to the specification of the key business process tasks and data flows required to support the WTMx system. The infrastructure components required to generate this archetype are modeled in the form of a workflow processes flowchart in Figure 4, which illustrates the data acquisition, processing, and information presentation operations required for a WTMx system design.

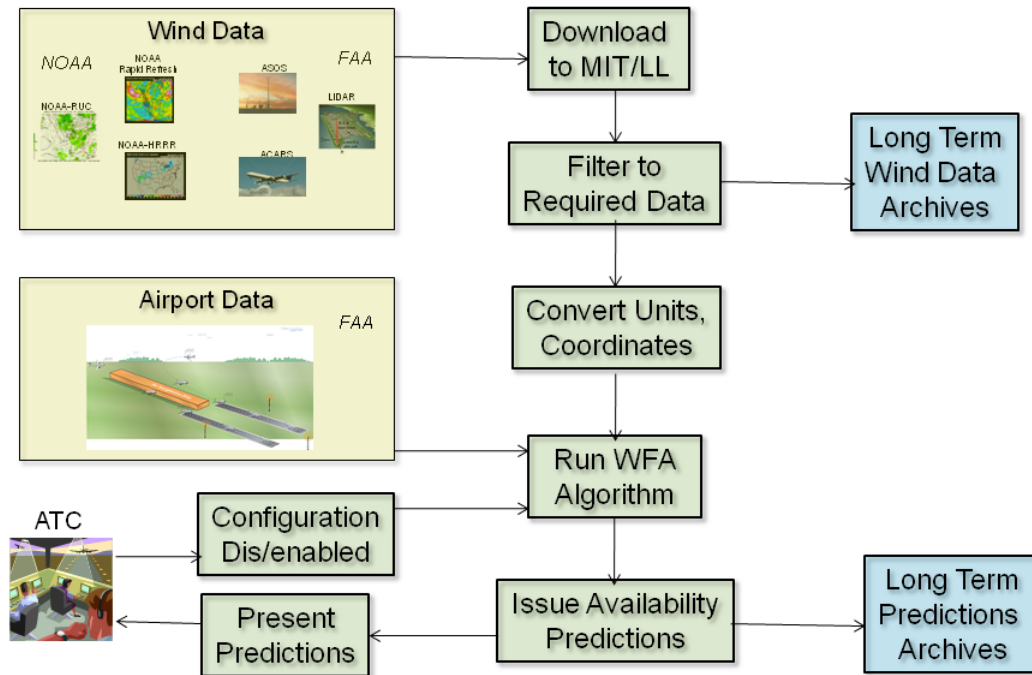


Figure 4: Business process flows for the WFA application with regards to weather data processing, WFA algorithm operation, and presentation via the WTMx display.

In Figure 4, the workflow processes begin with the acquisition of wind data to be archived, processed, and filtered into a more manageable size (which is also archived). This filtered data serves as input to the WFA algorithm along with airport data and current airport configuration information. The WFA algorithm then issues its crosswind favorability prediction, which is archived and simultaneously presented to the tower controller via the WTMx user interface. Whenever the airport configuration changes (e.g., WTMx is enabled or disabled, or perhaps due to a change in active runways), this change information is relayed back to WFA so that its input decks can be appropriately configured. In effect, Figure 4 is a more detailed description of the business process model diagram shown in Figure 3. This enhanced diagram more explicitly defines:

- the archiving of unfiltered and processed weather data at various stages,

- WFA prediction archiving, and
- the circular interaction between the tower controller and the WTMx user interface via the presentation of wind favorability outlook to the controller and airport configuration changes to the WFA algorithm.

With this enhanced business process model diagram, the overall workflow requirements of the WTMx application infrastructure can begin to take shape.

To further aid in defining the BPM specification, a more detailed understanding of how this infrastructure will be used is helpful. This infrastructure must support the WFA in each usage mode of algorithm development discussed in Figure 1: analysis mode, playback mode, and real-time mode. Having defined the fundamental WTMx business process model in Figure 4, the key components and data flows can be analyzed for each usage mode by defining the constituent tasks that must be performed, the data size and latency requirements for each task, and any additional requirements and constraints (also known as a business process rule or BPR) on a particular task or data flow. The next few subsections will develop the BPM specification in more detail for each of these modes.

2.3.2 Business Process Rules: Analysis Mode

The analysis (or research) mode of wake turbulence mitigation presents some unique and very demanding requirements on the wake turbulence application architecture because of the long duration and wide array of data required for research analyses. Figure 5 augments the schematic shown in Figure 4 by adding the timeframes associated with each application component. The weather data are acquired hourly and processed and archived, spanning a rolling three-year period which provides an adequately long duration of time for algorithm performance testing, without overwhelming the data storage systems. The three year rolling window allows for the most two recent calendar years of data to be available at all times.

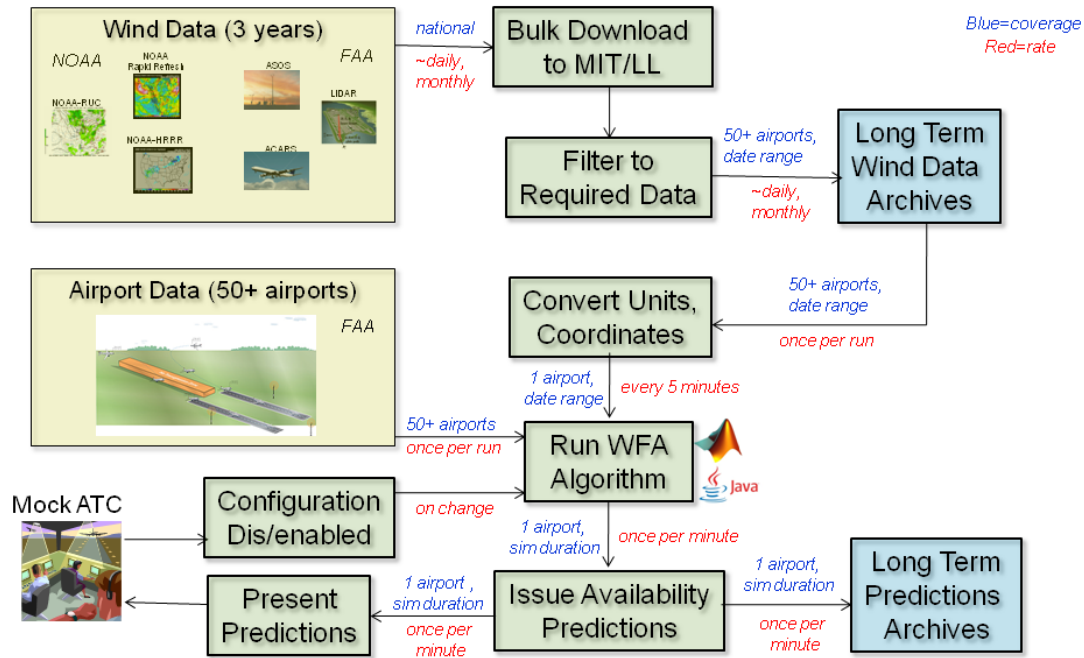


Figure 5: Business process flows for the WFA applications in Analysis Mode, which supports over 50 airports and can run using very long durations of weather data.

From the perspective of infrastructure design, the rolling three-year archive requirement imposes a significant challenge in terms of data storage space relative to the systems initially available. The majority of the space used, over 80%, is directly associated with archiving the filtered and processed high resolution forecast model data. The specifications for a new data storage system were computed precisely based on this three year rolling archival need required by the planned WFA development and testing.

The availability predictions generated from these spans of weather data, along with intermediate data useful for evaluating algorithm accuracy and efficiency, must also be archived over the same three-year span. While the storage needs for these predictions and intermediate data are not substantial, these data must be stored in an online database so as to be readily available for display and analysis. For research studies covering long spans of time, this can lead to challenges in performance of structured queries of the data, although the response time of these queries has some flexibility.

The analysis mode of WFA application also spans over more than fifty airports of potential interest for WFA analyses. The rationale for selecting these airports is based on commercial airport operations volume, runway configuration, and sponsor or collaborator interest, and this usage mode must support analyses at any of these airports for the same reasons. The CPU processing power required for processing the weather data generally scales linearly with the number of airports; however, the postprocessed online

airport-vicinity wind data files are fairly negligible in size. It should be noted that the number of airports to be supported could rise in the future, as has been the case historically on this project. Recently, other collaborating research organizations have expressed interest in wind data from several additional airports beyond the current 50+ airports of interest as part of WTMx benefits analysis and studies in progress, and the airport count could increase even further in the future.

Current computing infrastructure is capable of handling the current CPU workload for all the existing weather data feeds with the exception of the highest resolution forecast model feed. For processing this feed, the use of legacy software scripts and existing computing hardware has shown to be insufficient to process this data source in any reasonable time relative to the rate at which it is being acquired. Although recent efforts focused on modernizing this data processing helped to improve the computational efficiency of the data processing logic so that existing server systems may still provide sufficient computational ability, the data has not been processed in real-time presently due to insufficient data storage space. It is currently being archived offline in its original source form until a new data storage volume can be prepared for real-time data processing. The data storage requirements are cataloged in Section 4.1, and specifications of the data storage volume equipment acquired to address this need are described in detail in Section 4.3. As of now, while the data processing load can be handled by existing CPU systems, the highest resolution forecast model data processing has been hampered until the data processing software can be migrated to the new data storage volume recently installed and made available.

An additional requirement for the analysis mode of this architecture is its ability to inter-operate with the existing MATLAB code base to the extent possible. This will enable the timely transfer of advances in algorithm development yielding from rapid prototyping in the MATLAB WFA logic to this planned architecture, while also facilitating analyses over large data expanses that exceed the performance abilities of the MATLAB environment by using the Java-based WFA.

2.3.3 Business Process Rules: Playback Mode

The playback (or demo) mode applies more intermediate data storage and computational requirements on the application architecture. The playback mode of the application architecture will be composed of a two step approach:

- (1) *Preprocessing* – first the weather data are filtered, processed, and archived in the days or months prior to a demonstration requiring playback capability.
- (2) *Playback* – next, at the time of the playback, the WFA algorithm will query the requested range of airport-specific wind data and issue real-time (or “fast time”) wake mitigation procedure availability predictions for that airport.

This two-step approach is superimposed on the business process rules diagram in Figure 6. The advantage of this two-step approach is that it enables the playback mode architecture to operate on only the airport-specific files without any need to access or operate on raw source weather data within an

unfamiliar facility as part of routine demonstrations for sponsors and collaborators. While the playback mode will be required to support over 50 airports, only one airport will be demonstrated at a time.

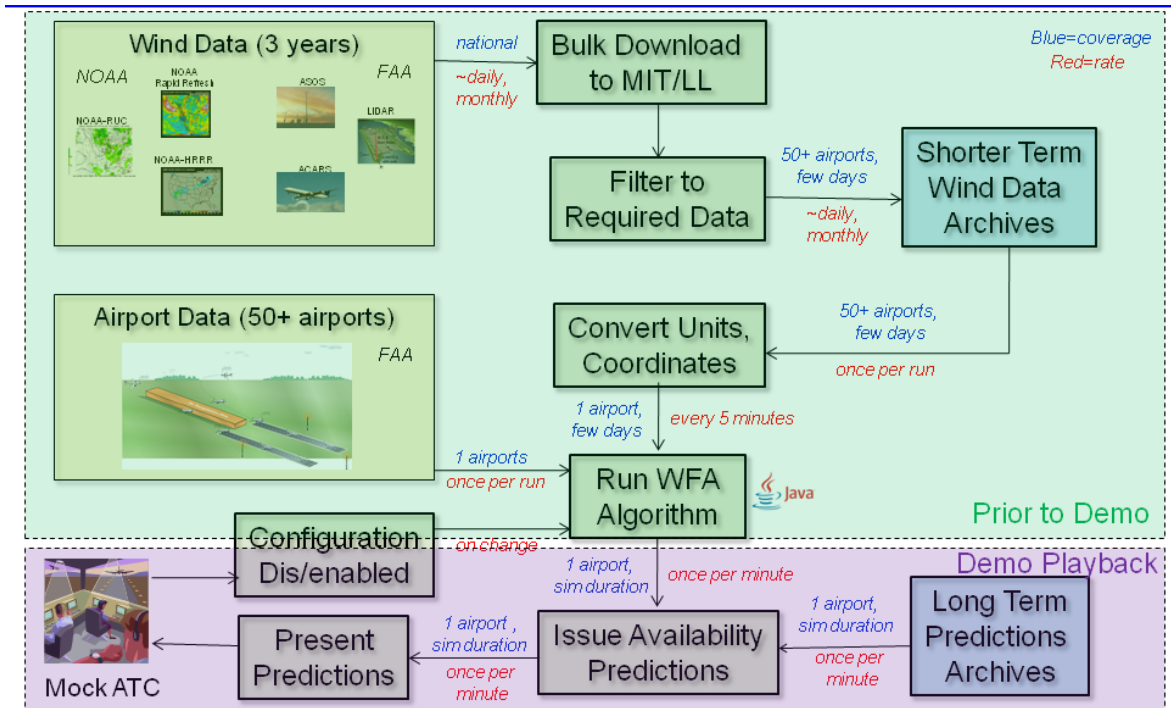


Figure 6: Business process flows for the WFA application in Playback Mode, which spans shorter durations of weather data at a single airport.

Because of the flexibility offered by acquiring and processing the weather data well in advance of the playback demonstration, this affords a minimal computational requirement since slower computation hardware may be used by initiating the computation in advance. The playback mode also does not require any additional wind data infrastructure relative to the analysis mode infrastructure requirements since the playback mode will only operate on a subset of the full wind data set required by the analysis mode.

The algorithm demonstration part of the playback mode will be more demanding in computational power and database store and query capability. For the playback demonstration, the algorithm will be required to operate in real-time (or fast time), and the predictions generated will be stored into the database. A database is required to permit the required “step forward and backward in time” playback capability required for demonstrations. Since a prediction data set is generated every minute, the database storage limits will scale linearly with the duration of the time simulated in the demonstration.

2.3.4 Business Process Rules: Real-Time Mode

The WTMx application architecture is designed to support algorithm operation in a real-time environment for the purpose of 1) supporting an operational prototype, and 2) to the extent possible, compatibility with the anticipated NextGen platform environment. This design is based on some fundamental assumptions regarding a generalized WTMx operating environment. The following paragraphs describe general assumptions regarding a generalized WTMx operating environment. The following paragraphs describe general assumptions regarding concept of operations, and the supporting requirements of the design for accommodating this concept.

The real-time mode of the algorithm drives the most challenging computational requirements on the WTMx application architecture. In the real-time mode, the application architecture must acquire and process all real-time weather data feeds, compute availability outlook every minute for a *single* airport, archive these predictions, and present them to the controller in real-time (as shown in Figure 7).

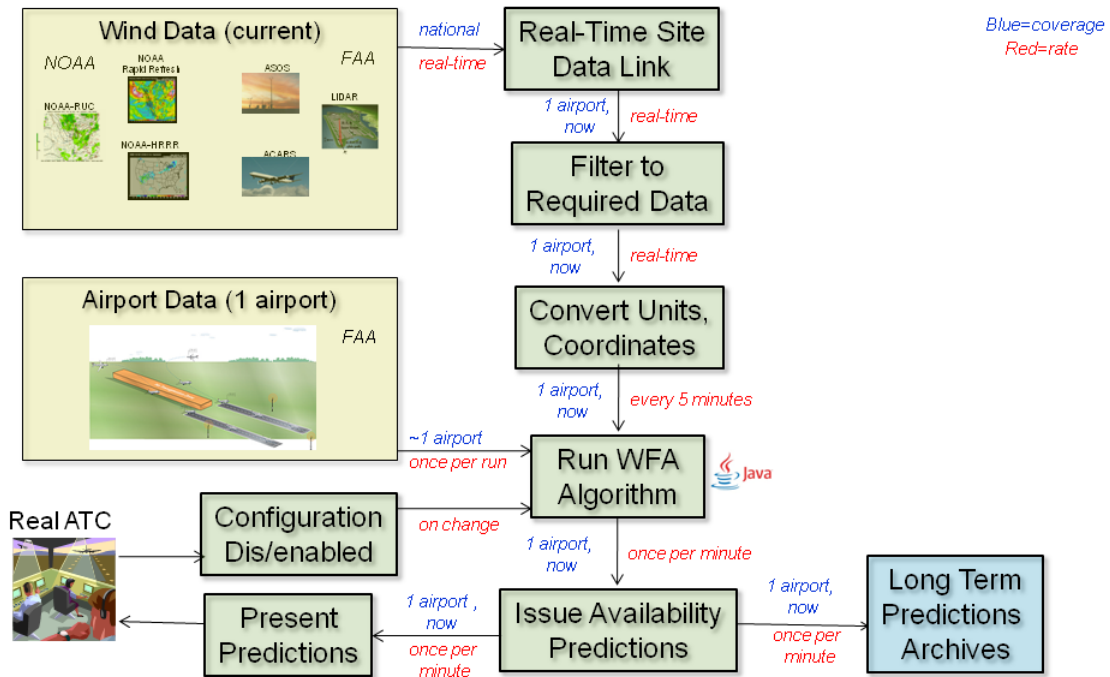


Figure 7: Business process flows for the WFA application in Real-Time Mode, which uses real-time data feeds for a single airport.

Because of the short duration of validity of the availability outlook prediction computed every minute and its inherent dependence on the timeliness of the weather data provided as input, it is critical that the WTMx application architecture is optimized to perform the necessary data transformations and

computations as rapidly as possible. Failure to meet these strict data and prediction delivery requirements will result in conservative nonavailable predictions by the algorithm. Finally, it is noted that the data storage requirements for the real-time mode of the application architecture are fairly small since only prediction information is archived.

2.4 FUTURE APPLICATION REQUIREMENTS

WTMD is the current wind-dependent solution being evaluated operationally. Within the FAA Wake Turbulence Program, analogous wind dependent solutions are also being investigated to address a host of other procedures, which includes arrivals on CSPR WTMA, arrivals and departures on WTM-SR, TPD, PA, and ER. The baseline WFA established for WTMD will require adaptation and modification in order to meet the specific needs and requirements of each of these applications. It is important that the architecture be designed to accommodate adaption and testing of components within each of these solutions.

Several key application components are slated as new concepts or as significant redesign of existing applications. These include:

- **Wind Data Warehouse** — this component provides a platform-independent application programming interface (API) and database implementation that accepts requests for spatial and temporal ranges of specific types of wind data and returns them to the requestor.
- **Wind Forecast Algorithm (WFA)** — while an existing WFA implementation exists in MATLAB and C form, the potential for additional wind data sources coupled with significant limitations in the ability to integrate this implementation into other systems necessitates a modernization of this application component.
- **WFA Display** — this application component will be newly created as a deployable, easy-to-integrate hardware/software application GUI display to support development and testing, and potentially to furnish in airport towers for tower supervisors and/or controllers based on, and to replace, the existing developmental MATLAB prototype(s).

These new application components must be designed to be future-proof to new wind data streams as they become available, scalable to handling a large number of streams and increasing data resolution requirements, and easily portable so that they may be integrated into other FAA systems and readily deployed both within Lincoln Laboratory and at external sites. These applications should take advantage of modern technology and software architectures while simultaneously maximizing reliability.

This page intentionally left blank.

3. WEATHER DATA DESCRIPTIONS

To help specify and evaluate the weather data system design, a detailed enumeration of the variety of wind data streams and an accounting of disk storage requirements and bandwidth needs is necessary. The following subsections will describe all of the wind data types and sources in detail, with a special focus on the amount of data processing, such as data point filtering, that is required in handling each data source.

3.1 OVERVIEW OF DATA

The development and operational deployment of wind forecast algorithms within WTMx systems depend on a multitude of wind data sources. Wind data may consist of actual wind measurements or output from numerical weather prediction model analysis and forecast fields. They may originate from sensor or model sources operated and maintained by government agencies, such as the NOAA and the FAA, other government-sponsored contractors, or private industry. The coverage areas and resolution of the data can vary from one source to another. Finally, the latency associated with the availability of current and future wind data will vary, as will the perishability of the data. A preview comparison of the differing features of these weather data feeds is illustrated in Figure 8, and will be the subject of the next few subsections in significantly more detail.

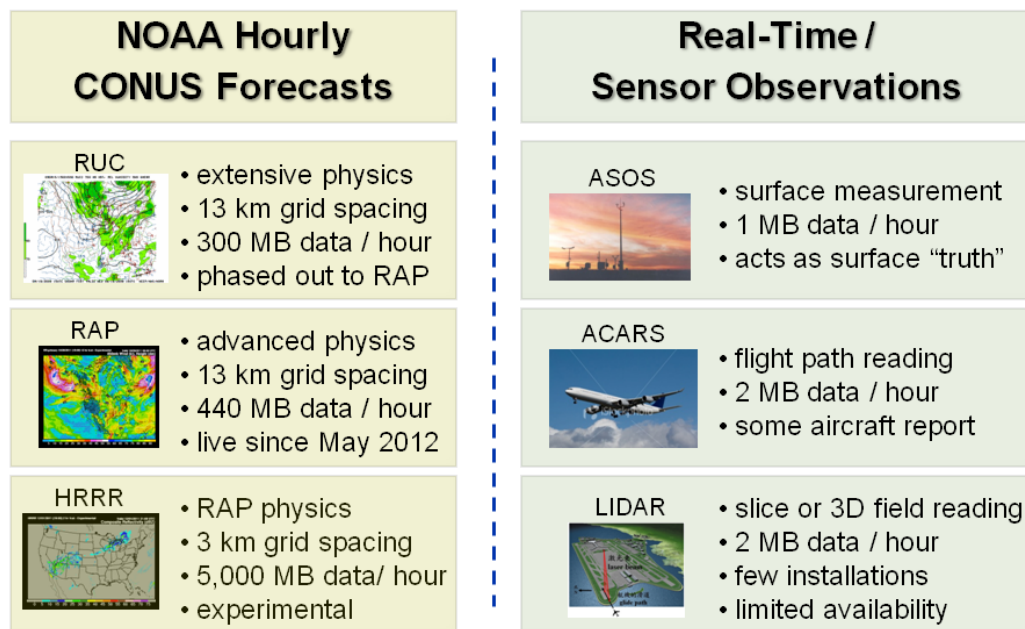


Figure 8: Weather data is acquired from NOAA-based hourly forecast models as well as real-time, airport-vicinity weather measurement equipment.

3.2 FORECAST MODEL DATA

The NOAA weather forecasts models applicable for use in WFA include the RUC, RAP, and HRRR forecast models. RUC was the operational model when WFA model development began, RAP is the current operational model, and HRRR is the current experimental model that is expected to become operational in coming years. All three are required to be archived in order to maintain continuity within the development process. These forecast models represent different levels of breadth and maturity, and have a number of key differences that define to their suitability to various WTMx applications. To assess this suitability, it is worthwhile to first characterize these forecast models in more detail.

The RUC, RAP, and HRRR forecast model predictions use input data from a wide array of weather radar and weather instruments coupled with custom specialized physics and weather models and historical forecasts to compute a broad range of dozens of weather products across the country (see Figure 9).

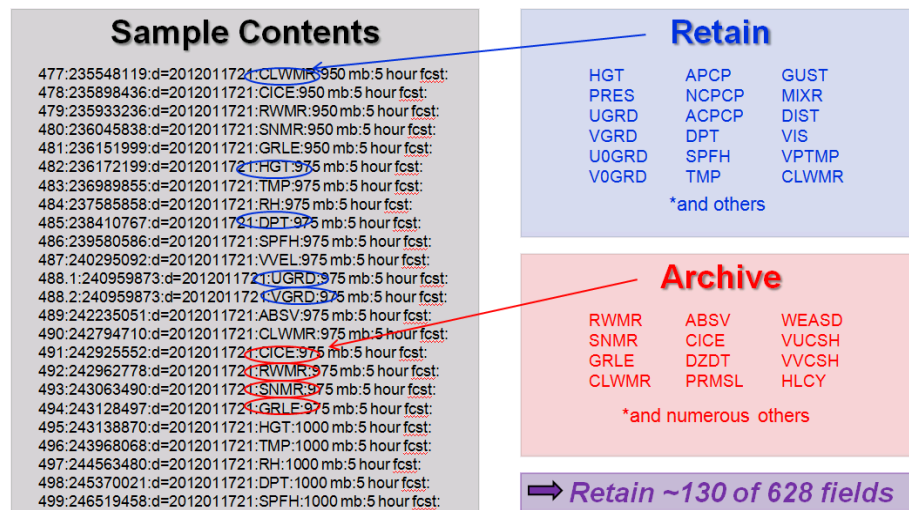


Figure 9: Each grid point in a single NOAA forecast contains dozens of different weather products, of which only a subset are of use to WFA.

These forecasts generated by NOAA provide a set of zero to +24 hour forecasts every hour (see Figure 10). The zero hour forecasts (also referred to as the “analysis” or “initialization” field), denoted as +0, are issued as much as 15 minutes after their validation time because of delays in acquiring input data and computing and disseminating the output data. From Figure 10, it is apparent that there are several forecast options to select from for a specific time of interest (e.g., 15 hours UTC in the figure), and it is up

to the consumer of this forecast data to select the forecast field(s) most suitable to their application. The WFA typically uses multiple forecast horizons at any given time, depending on the WTMx application.

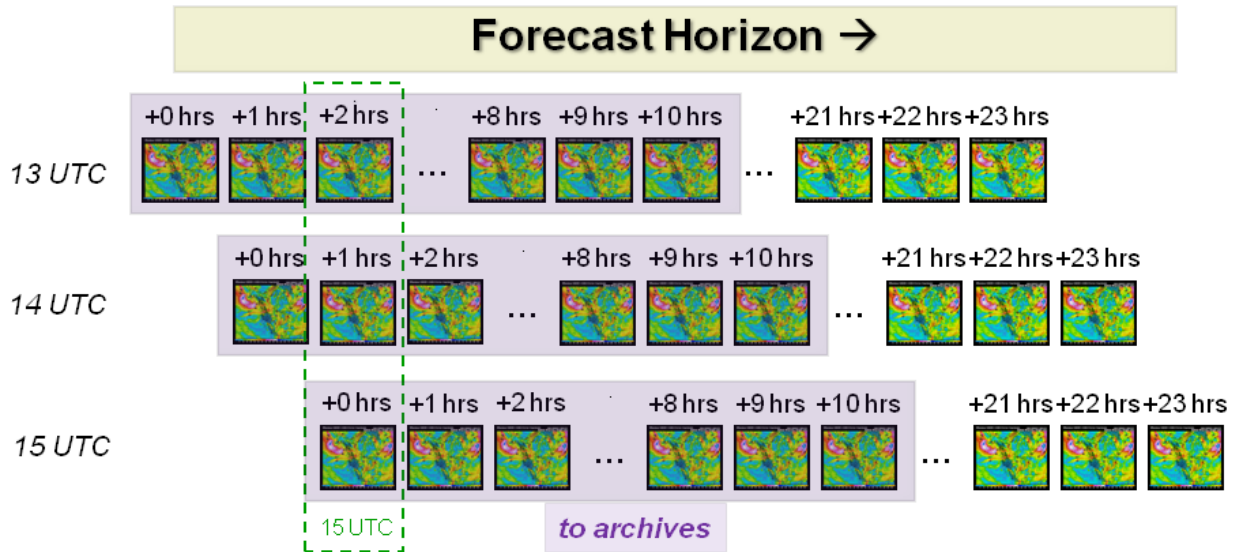


Figure 10: Weather forecasts generated by several NOAA products provide wind forecast data from zero up to 24 hours into the future, updated every hour.

Depending on the specific forecast model, the entire set of weather products may be computed at differing grid resolutions (requiring up to millions of grid points per forecast) and geographic coverage across the continental United States (CONUS) for about 50 altitude levels from the surface to high altitude (see Figures 11 and 12). The vertical levels can either be specified in terms of *pressure levels* (P) expressed in hectoPascals (hPa), or as *natural levels* (N) expressed in a normalized computation of sigma. Data requirements for these are determined by the specific WFA application. The standard atmospheric tables expressing conversion to or from pressure levels and natural levels are freely available in the literature and the web. It should be noted that the actual height of these two altitude measurement schemes will differ, and it is important that they be explicitly identified as part of the acquisition and archive process. Going forward, wind data source names will be appended with a -P or -N as appropriate to indicate which vertical level format applies to that data. For example, RAP-N would signify a RAP forecast data set using *natural* levels.

Latitude - Longitude

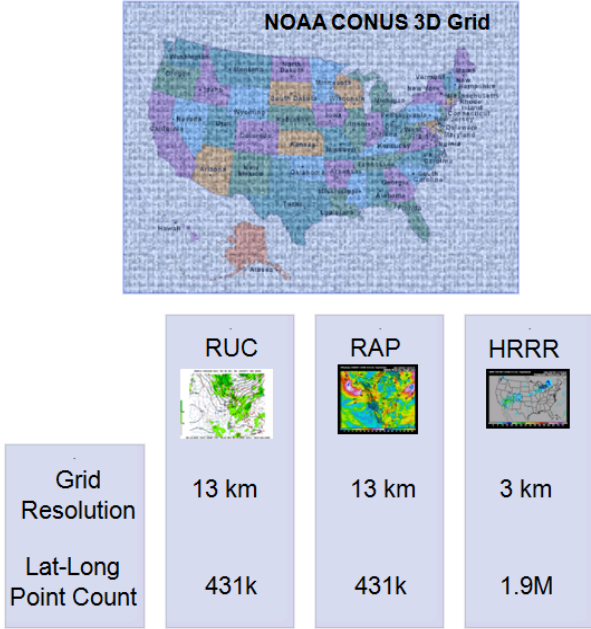


Figure 11: Weather forecast products by NOAA provide weather data at latitude and longitude resolution resulting in up to millions of data grid points per forecast.

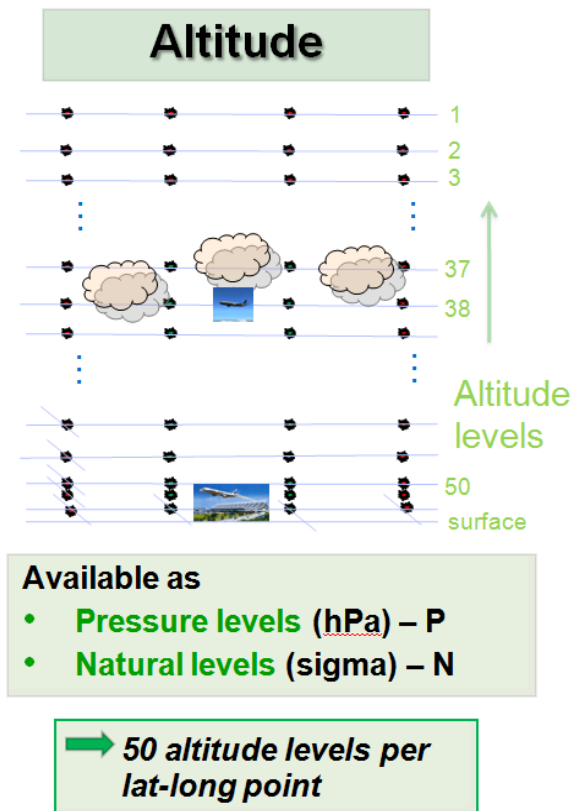


Figure 12: Weather forecast products by NOAA provide weather data at up to 50 levels of altitude resolution in either pressure level (P) or natural level (N) coordinates.

The next few subsections describe each of these forecast models in more detail, along with their suitability for various WTMx models.

3.2.1 Forecast Model: Rapid Update Cycle (RUC) Data

The RUC model was the earliest forecast model used in the course of WTMx development. As previewed in Figures 8, 11, and 12, the RUC model uses extensive physics coupled with radar and other meteorological sensor data to predict forecasts from +0 to +15 hours. Early analysis used data at 40-km resolution, and shortly thereafter a 13-kilometer grid resolution became the standard as applied to WFA development. The 13-km grid across the CONUS from the ground to 50 vertical levels height results in data sizes of about 300 megabytes (MB) per hour. For several years, this was the standard operational version generated by NOAA for FAA applications; it was made available via FTP (File Transfer Protocol) transfer to users in both pressure level and natural level formats. In May 2012, the RAP model replaced RUC as the standard operational model. At that time, NOAA officially dropped support for this

data feed, although it continued to be available in an unsupported fashion as of this writing. MIT LL continues to collect RUC model data as available to ensure WFA compatibility across the model transition. NOAA has informed the weather community that there is no assurance of continuing availability and support of the RUC model, and has advised users to transition their applications to the RAP model.

3.2.2 Forecast Model: Rapid Refresh (RAP) Data

The RAP forecast model supplants the RUC forecast model by offering more advanced physics and weather phenomenology for providing weather data across North America. It continues with the 13 km grid resolution, and offers 50 altitude levels in both pressure level and sigma level formats. The NOAA National Centers for Environmental Prediction (NCEP) issues operational RAP forecast data hourly up to a 24 hour forecast horizon. Another experimental RAP forecast model maintained by NOAA Earth System Research Laboratory (ESRL) also generates forecast data for an increased number of weather products, but this version is not guaranteed in its issuance as the NCEP model. Over time, the advances in the ESRL RAP forecast model are systematically transitioned into the official NCEP model as performance is tested and verified. For the purposes of WTMx development, acquisition of both sets of forecast model data are required, although only the NCEP RAP model data can be officially used in a real-time WTMx system. The ESRL-sourced RAP data provides additional weather products required to aid in Lincoln Laboratory analyses for specific airports.

3.2.3 Forecast Model: High Resolution Rapid Refresh (HRRR) Data

The HRRR model is under development as the future replacement for RAP once its performance has been validated. As such, MIT LL has committed to collecting and archiving model output in order to be prepared for an orderly transition with regard to their WFA application within WTMx, analogous to the transition that occurred from RUC to RAP. Additionally, the HRRR model holds promise for WFA forecast improvement that would require validation across concurrent HRRR and RAP data sets, thus requiring that data for the two models be collected in parallel for an extended period of time. The HRRR forecast model continues with RAP physics issued hourly, but offer a significantly higher grid resolution of 3 km with 50 altitude levels in either or both pressure altitude coordinates and sigma altitude coordinates, resulting in an increase of almost 20-fold in data size relative to RAP and RUC models. Furthermore, HRRR forecast models issue output with a forecast horizon of up to +24 hours. The increased resolution of HRRR has implications in wind data feed infrastructure requirements associated with handling this vastly larger data volume

3.2.4 Forecast Model Data System Requirements

From the past few subsections, it is apparent that the wind data system to be developed must be sufficiently extensible and scalable to acquire and process forecast model data streams from a wide variety of RUC, RAP, and HRRR data sources. For the highest resolution model, HRRR, this results in forecasts of several dozen weather products for each of 100 million grid points per forecast, with 24 new

forecasts issued every hour. The accessibility and latency limits of each data stream will vary depending on whether the WTMx consumer of the forecast data is conducting analysis, playback, or real-time mode of WTMx operations. This broad support for forecast model support among different variations of WFA/WTMx algorithms under development, coupled with strict demands for data accessibility, performance, and latency limits, imposes a significant challenge in the design of an extensible, scalable wind data system to meet these requirements.

3.3 AIRPORT SURFACE OBSERVATION SYSTEM DATA FEEDS

The National Weather Service (NWS) maintains a national network of several thousand Airport Surface Observation System stations that provide near ground-level (10 m height) weather observations to support their operational weather mission. The network was deployed as part of an interagency agreement that also included the FAA and the U.S. Department of Defense (DoD). ASOS data serves as the primary input of so-called “surface” wind forecast to support numerical weather predictions models (e.g., RUC, RAP), and is also the most-relied upon source of weather data for servicing public weather information needs. A large portion of the ASOS sensors are located at airports, where they serve the need of providing to air traffic managers key wind information critical to aircraft operations.

Because ASOS provides wind data at the high frequency rate of once per minute rate covering all airports of interest, including airports in locations such as Alaska for which forecast model data are not available, the acquisition and processing of the ASOS feeds is fundamental to conducting WFA evaluation analyses, especially in light of the stringent safety factor tolerances required for these analyses. In a real-time deployment, ASOS data bears a critical role in generating predictions since the WFA specifically delivers a “WTMx unavailable” status if even a few ASOS observations are missing within a short period of time, thereby setting a strict reliability requirement for the ASOS data feed in this wind data infrastructure. Provisions are currently being made in the wind data system design to ensure future compatibility for acquisition of real-time ASOS data from one or more ASOS feeds from actual airports as they become available.

ASOS data are collected, distributed, and archived on a number of different time scales. The highest resolution data are provided in near-real time at 1-minute resolution; this is the time fidelity most critical to aircraft operations. Data are archived in both 1-minute and 5-minute samples. Observations are also recorded and disseminated at 1-hour intervals, consistent with the historical convention for surface weather data reporting. Intra-hour reports are also automatically generated whenever a meteorological parameter crosses a key aviation-related threshold, typically associated with wind, visibility, or precipitation.

The wind data system being developed requires the acquisition and processing of several different ASOS data feeds with differing latency, perishability, resolution, and accuracy depending on the usage mode of the WTMx system. While the real-time mode of a WTMx system would require low latency, high reliability data acquisition and processing of ASOS observations every minute, the analysis mode would allow for the acquisition of ASOS data archives over greater time expanses, often extending over

years, in which latency is not an issue. The older ASOS data archives also offer the benefit of having been vetted and corrected for data errors and omissions, and employ a more rigorous data acquisition process to ensure data quality and omission detection and correction using techniques that are unavailable to a real-time system because of its strict latency requirements. The source and data format of each these ASOS data feeds also varies.

Although MIT LL does not yet have a direct real-time ASOS data feed to any of the 50+ selected airports of interest, the wind data system is currently acquiring continental-aggregated ASOS archive data from three different sources within NOAA. While each of the ASOS data feeds provides measurements (observations) of wind speed and direction along with gust speed and direction, these sources differ in their reporting and observation frequency and minimum resolution of speed (0.1 m/s vs. 1 knot) and direction (1 degree resolution vs. 10 degrees). Two of the long-term ASOS data archives from the National Climatic Data Center (NCDC) provide ASOS data at 1 minute and (averaged) 5 minute intervals, and are only made available for the previous month, leading to a latency of up to 30 days. To permit WFA analyses on more recent ASOS data, a third ASOS data feed also from NCDC is acquired that provides ASOS observations at one minute intervals, batched daily resulting in a latency of up to 24 hours, with the caveat that some of the ASOS observation data may be corrected over the following few weeks. Collectively, the three ASOS archive feeds provide data at a resolution and latency to support the WTMx analysis needs as part of the WFA algorithm development efforts, with the data redundancy offering a means for handling data quality evaluation and data gap mitigation.

3.4 AIRCRAFT WIND DATA

Wind data measurements are also available via the Aircraft Communications Addressing and Reporting System (ACARS) data. The ACARS system consists of an Aeronautical Radio Inc. (ARINC) datalink to transfer bidirectional data between an aircraft's avionics and a network of satellites and ground radio stations. While originally purposed with uplinking weather information from ground to aircraft and automated downlinking of amendments to aircraft flight phase, more recent modifications to ACARS provide automated communication of routine aircraft data such as sensor and maintenance information from the aircraft subsystems to ground stations. For some aircraft, this includes periodic reports of wind data, disseminated by ARINC, and archived and made available online by NOAA. The ACARS system uses weather instruments equipped in participating aircraft, which record weather data generally temporally-spaced at one-minute intervals.

The benefit of aircraft reports are that they provide wind data from the actual flight paths for departure and arrival from an airport rather than a fixed grid resolution in the vicinity of a runway. Actual data report rates vary, with recorded observations made available hourly. The data are relatively small in size due to the limited number of spatial points associated with this feed, and limited information contained in each individual report. The current source available to WTMx for these data has a latency of 54 hours and relatively short perishability, although other sources of ACARS data with near real-time availability are anticipated in the future. The primary challenge that ACARS wind data poses to a wind data system is the spatial variability of the wind data points selected per flight, as well as issues in data

quality provided by the ACARS sensor systems. There is also an ongoing effort to obtain aircraft wind data via Automatic Dependence Surveillance-Broadcast (ADS-B), which may impose additional data collection and storage requirements with regard to volume and latency.

3.5 LIDAR DATA

As part of the wake turbulence program's field data collection effort, the FAA has deployed and collected data from LIDAR sensors at a number of key airports. The cross-sectional beam of the LIDAR is oriented toward aircraft flight paths at runways of interest to characterize (radial) wind profiles, and measure wake vortex behavior through detailed wind measurements. LIDAR data is a key source of wind validation to support WFA development. Based on the geometry of the LIDAR orientation relative to the runway and aircraft flight path, a detailed cross-sectional plane (grid) of wind data can be measured roughly once per minute. Alternately, an additional technique involves Velocity Azimuth Display (VAD) scanning, which requires a conical scan of the LIDAR beam using a fixed angle of elevation while varying the azimuth angle, yielding a series scans of spatial tilts of wind data. These scans can collectively be used to derive a full vector steady-state wind over the region for each vertical level up to a height of several hundred meters. The scanning sequence allows for a complete vertical profile of winds approximately every 20 minutes.

The LIDAR data streams are currently being acquired in an ad-hoc fashion from just a handful of airports that have been temporarily equipped with these sensors. While the initial intent of the use of the LIDAR sensors has been to aid wind and wake analyses, there is also interest in possibly employing LIDAR sensors in a more deliberate, operational means as part of real-time WTMx deployments. If and when this occurs, the wind data system and WFA will be required to acquire and process the LIDAR wind data stream in real-time mode as well. These data sets tend to be smaller relative to the forecast model data for a single sensor, and any data requirement challenge posed by this source would only result if a great number of LIDAR sensors were installed.

3.6 DATA SOURCE SUMMARY

The past few subsections have explored the variety of RUC, RAP, AND HRRR forecast model data that must be acquired, processed, and archived for use by WTMx systems. The ASOS, aircraft, and LIDAR sensor wind data sources were also discussed in detail. With this survey of the variety of wind data streams to be supported now complete, we can now begin to focus on the design of a wind data system that is capable of meeting the WTMx requirements across this broad range of wind data input feeds.

This page intentionally left blank.

4. DATA SYSTEM DESIGN

The weather data feed infrastructure is responsible for wind data acquisition, data processing, and archiving of preprocessed and/or postprocessed wind data. Based on the application architecture requirements, the weather data feed infrastructure that enables wind data acquisition and processing and archiving must be:

- sufficiently flexible in data organization to the receipt of a broad diversity of wind data feeds with differing format and transmission methods and perishability
- highly scalable to ingest high-rate data input, possibly over durations of several years
- efficiently organized to allow the storage and select retrieval of varying types of wind data over large geographic expanse and calendar time
- fully reliable in terms of network connectivity and bandwidth, hard drive storage and archival storage, with additional processing power for CPU-intensive and memory-intensive tasks

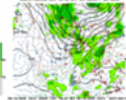
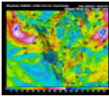



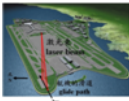
4.1 DATA ACQUISITION

The weather data feed infrastructure must acquire wind data from online Internet-based web and FTP servers, local and remote file systems, network-based protocol feeds, email protocol feeds such as Simple Mail Transfer Protocol (SMTP), and other transmission modes to be defined in the future. The data acquired can be in the form of Grib1 or Grib2 archive files, zip or tar archive files, ASCII text files, binary files, and other formats in the future. Although there may be some latency between the time of the wind data generation and its subsequent availability, the wind data feed infrastructure must acquire these data in a timely manner to support the wake mitigation algorithm computations.

The perishability of the weather data in the context of this architecture depends on the duration for which the data source maintains the availability of the weather data access. For some data feeds, the perishability is defined in years, whereas for other data sources, the duration can be as short as 6 hours. Real-time operational modes require that weather data are acquired as soon as they become available, often hourly for wind forecast data and every minute for ASOS wind data. Other data may be more temporary or lower temporal coverage, such as LIDAR and VAD sensor data. Table I catalogs the variety of data feeds being acquired in support of the WTMx application architecture, along with its frequency of acquisition. This acquisition frequency can put tremendous demands on the CPU and hard drive of the acquiring machine, along with the network it is connected to.

Table I

Summary of Data Acquisition and Archiving Requirements

		Source	Expense	Perishability	Frequency	Archive Size	Online Size
	RUC-N	ESRL	2010 – now	24 hours	hourly	7,000 MB	1,700 MB
	RUC-P	NCDC	2010 – now	years	hourly	5,472 MB	1,700 MB
	RAP-N	NCEP	2012 – now	24 hours	hourly	7,000 MB	1,700 MB
	RAP-P	ESRL	2012 – now	24 hours	hourly	10,560 MB	2,000 MB
		NCEP	none	months			
	HRRR-N	ESRL	sample	24 hours	sample	115,000 MB	24,000 MB
	HRRR-P	CoSPA	2010 – now	60 days	hourly	94,464 MB	20,000 MB
	ASOS	NCDC	2010 – now	years	hourly	17 MB	5 MB
	ASOS-RT	none					
	ACARS	MADIS	2009 – now	years	hourly	30 MB	30 MB
	ACARS-RT	MADIS	2009 – now	years	10 mins	30 MB	30 MB
	LIDAR	Volpe	pockets	years	varies	30 MB	30 MB
	LIDAR-RT	none					

Note: N=Normalized Levels, P=Pressure Levels, RT=Real-Time

4.2 DATA PROCESSING

An overview of the wind data filtering chain is shown in Figure 13. While the particulars of processing will be specific to the data type and its source, the majority of input wind data feeds will substantially undergo each of the filtering steps presented in Figure 13. Much of the discussion below is most applicable to model-derived data, which drive many of the system requirements because of their large volume, and requirement to acquire during a limited window of availability.

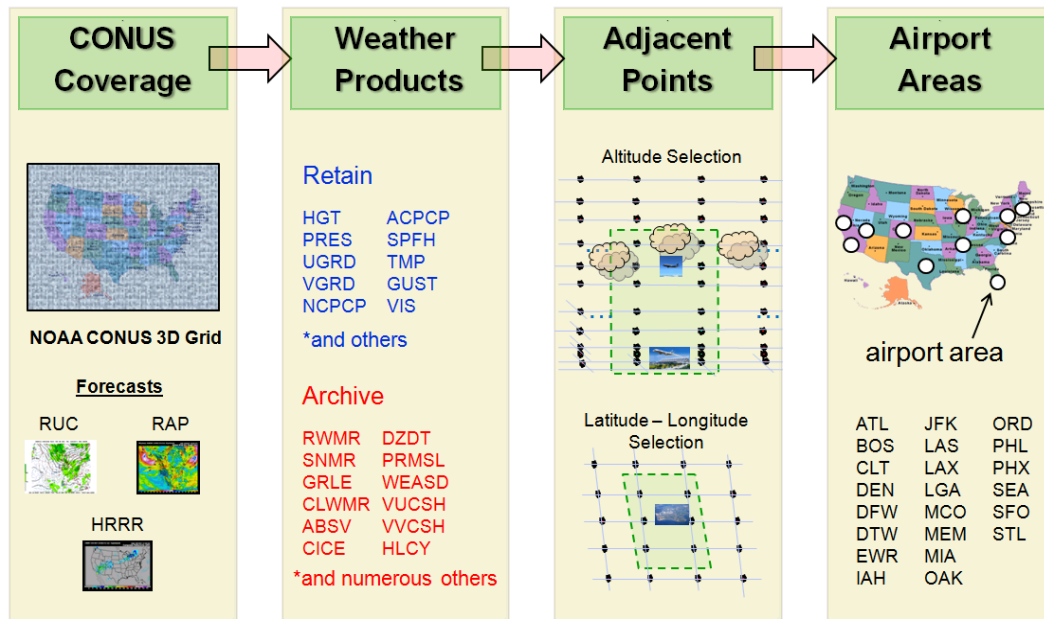


Figure 13: Weather data processing sequence starting from the acquired data and its filtering to specific airport wind data.

After the weather data are first acquired, a subset of the wind data products will be selected from the acquired dataset. While many of the datasets will provide weather data from the surface to as many as 50 altitude levels as high as 60,000 feet, the WTMx algorithms only require weather data at the surface up to about 3,000 feet. This permits as much as 60% reduction of weather data through altitude filtering by selecting only the surface and low-level aloft wind data. Very significant reductions often greater than 99% are achieved by surgically filtering the weather data by latitude and longitude based on geographical considerations of airport vicinity, generally ranging from 81–225 grid points per altitude level (either 9×9 grids of points around an airport for RAP or 15×15 grids for HRRR) surrounding each airport of interest.

To support current and future anticipated WTMx algorithm development efforts, the weather data are extracted for the 54 airport candidates for WTMx study. These airports are in either the top 35 busiest rankings by commercial passenger traffic (2011 traffic figures), the top ten rankings by cargo volume traffic, or specifically requested by the sponsor and/or collaborators in the wake turbulence program. While only a subset of these 50+ of these airports are anticipated to participate in a future WTMx deployment, they were selected for the following reasons:

- already part of, or in future consideration for operational demonstrations of WTMx prototypes

- presence of complementary data measurement assets or compatible FAA programs
- strong support of local airport officials and/or commercial and cargo airlines for introducing WTMx to that airport
- significant commercial traffic for which runway slots may be at a premium during peak operational hours
- geographically situated such that local weather conditions favor the benefits from the WTMx work
- proximity to major adjacent airports being studied and sharing a local airspace and requiring coordinated operations

Through these stages of weather data filtering, weather data acquired from the source are filtered by weather product, altitude, and airport vicinity, resulting in a data reduction from as many as 100 million data points (in the case of HRRR) to about 2500 data grid points, with only about 60% of the weather data product fields retained for each point. This 40,000-fold reduction in data necessitates significant resources in terms of hard drive storage, server memory, and CPU power to allow these filtering operations to occur in a reasonable timeframe, or even real-time for future operational demonstrations.

Once the weather data has been filtered to a more usable size, the weather data for the remaining grid points is further processed in preparation for use by the WFA algorithm. These processing steps will vary for each weather data feed, and can include any of:

- rotation and shift of lattice of grid points from one orientation and displacement to that expected by the WFA algorithms
- conversion of the units of the weather product values to the standardized units expected by WFA and WTMx
- computation of additional intermediate weather products from the data given
- filtering outliers from the datasets based on specific conditions for each weather product
- adding missing values in the data set, usually zero values for missing wind measurements that may occur due to hardware or software issues during generation of the source data

The processing times and frequency for each weather data feed are specified in Table I. Once the filtering and final processing steps have been completed for each data set for this acquired weather data feed, these data are collated into an airport-specific text file containing all the data over daily, monthly, and yearly time ranges to facilitate access to this data by the WTMx algorithms.

4.3 DATA ARCHIVING

A significant component of the wind data infrastructure requirements are directed towards archiving of acquired source data, filtered data, and processed airport data files. The archival procedures will be specific to each weather data feed due to the inherent differences in data structure, data size, accessibility, perishability, and processing times.

The acquired source data is archived provided it is not excessively large. While the sensor-based weather data feeds such as ASOS and ACARS and ultimately LIDAR are sufficiently small enough to be archived in their raw source form, the weather forecast data obtained from NOAA such as RUC, RAP, and HRRR are many orders of magnitude larger in size. This necessitates the initial subsetting based on forecast times, in that only the first 0 to +15 hour forecasts are acquired to obtain a reasonable balance between having sufficient forecasts into the future while limiting total archive file size (and processing time and memory). Further filtering based on weather products results in a 60% reduction in file size, provided the weather data is archived in Grib2 format to offline storage. If additional server hard drive capacity is obtained, this filtered data may be kept online, space permitting.

The disk space required for each weather data feed is shown in Table I. Upon comparing daily download size for archived and online storage of each of the weather data feeds, it is apparent that the NOAA forecast model data is several orders of magnitude larger than the sensor-associated data feeds such as ASOS and ACARS. Furthermore, it is apparent that the HRRR forecast model data is five times larger than the RUC and RAP forecast model data, primarily due to its 3 km grid point resolution relative to the 13 km grid point size used in the RUC and RAP forecast models (see Figure 11).

The airport-specific wind data files, already substantially reduced in file size, are also archived, both offline and online. This data must always be readily accessible by the WTMx algorithms, and can be archived for many years using relatively small amounts of hard drive memory.

4.4 WEATHER DATA FEED INFRASTRUCTURE

The weather data feed infrastructure is composed of a variety of scripts and Java code involved in the various acquisition, processing, and archiving stages of the wind data.

The data acquisition part of the weather data feed infrastructure is primarily composed of Unix/Linux-based Bourne-Again Shell (bash) shell scripts running in automated fashion via automated Linux *cron jobs*. Bash scripting is a universal, freely available shell scripting language with broad tool support and has been in use for decades in mission-critical applications. Linux *cron jobs* are often used by information technology (IT) staff for initiating automated batch-processing functionality for routine IT systems maintenance for well over 30 years. The acquisition scripts are activated hourly for the NOAA forecast models and ACARS aircraft data, whereas the ASOS scripts run on a daily basis as the airport sensor data becomes available. The script logs for each script activation are stored for about a week to allow investigations of any recent data acquisition issues, and then subsequently purged.

These acquisition bash scripts use either FTP or *wget* commands to request the data as it becomes available at the source server. For the weather data feeds for which *wget* can be used, the primary benefits are the ability to request source files using sophisticated filename wildcarding and the ability to offer self-healing acquisition scripts, in which each hourly activation of the script will download the current hour's weather data files while also identifying and downloading any other weather data files missing from the previous hours provided they are still available on the source server, which depends on the perishability of that data at the source. This feature permits the weather data infrastructure system to self-heal its acquired weather data should the Linux machine running these scripts suffer any form of relatively short-lived outage, usually measured in hours and up to a day, due to any failure mode such as:

- system lockup due to Linux kernel failure requiring a forced reboot
- loss of available free hard drive memory to store the data
- network hiccups or complete loss of network connectivity
- some forms of file corruption if it results in the complete loss of a file
- loss of power due to announced scheduled power interruptions or inadvertent power losses originating within the building or the electricity supplier to the facility
- human error such as inadvertent data deletions or system shutdowns

It is noted that over the past year, the wind data infrastructure has endured each of the failure modes described above and was able to self-recover without any human intervention in each of them. To improve the reliability of the weather data infrastructure and offer load-balancing, the self-healing nature of the scripts is exploited further by running a duplicate set of these bash data acquisition scripts on another server on another subnet within the facility network. The use of a duplicate machine on a separate network subnet minimizes the probability of a failure to acquire weather data due to the uncorrelated nature of failures of the two systems and networks. It is still noted that several correlated failure modes could still occur which would impair the self-healing, redundant nature of these scripts such as:

- failure of the data storage server due to a failed hard drive, loss of power or network connectivity, or physical damage resulting from fire or flood
- loss of electrical power to the entire building housing both data acquisition servers running these bash scripts
- loss of Internet connectivity of the entire facility
- loss of access to the weather data source due to failures at the source

To help identify the system specifications required for the large capacity storage server, the requirements defined in Table I served as the primary sizing criteria. For all intents, the HRRR data size is driving the requirements for this data storage server, since the majority of other wind data feeds can be handled using existing hardware. As stated in Section 3.2, the HRRR data size estimate was based on HRRR data archiving needs over a three year rolling window assuming vertical levels extending from the surface to about 2,500 feet. It is also noted that this storage hardware being acquired is *not* a compute server; it can only provide disk storage.

To meet these requirements, a computing environment was configured at Lincoln Laboratory consisting of two primary physical servers with 5 terabytes (TB) and 40 TB+ of storage capacity and two virtual compute servers for data processing. The two servers each have 12 CPU cores @2.66 Gigahertz (GHz) and 24 gigabytes (GB) of memory. The first server has 5 TB of fast storage for data processing and the second has 40 TB+ for data archival. Each system is accessible by the two virtual compute servers with 4 vCPUs and 8 GB of Random Access Memory (RAM) each for data analysis and processing.

After data acquisition, the next step involves the data filtering and data processing scripts. These also are in the form of bash scripts running on a Linux machine, although these bash scripts routinely can call other helper scripts in Perl or MATLAB, often to take advantage of pre-existing, manually-activated scripts. The script logs for each script activation are stored for about a week to allow investigations of any recent data acquisition issues, and then purged. For simplicity, these processing scripts are activated in automated fashion as *cron jobs* only once a day per data feed, generally during overnight hours when system load is expected to be lower outside of regular work hours. While a bash script activated more frequently, such as hourly, might be preferable in terms of data processing latency, the sophisticated requirements of handling issues such as time delays associated with self-healing and download time variations make this unfeasible at this time.

For a real-time deployment of this wind data infrastructure, failure to acquire the weather data in a timely fashion will result in the conservative default WFA prediction of unavailable until the weather data become available again. This conservative feature permits a more simplified approach to handling data interruptions with regard to real-time wind data processing that cannot be applied to archiving these same data feeds long-term for research purposes.

The final step of processing involves the creation of yearly airport-specific wind data files. That is, for each airport, the wind data for an entire year is concatenated into a single text file and stored online. The production of these files is also automated, and activated by the bash processing scripts defined above. The contents of these files tend to vary depending on the source of the data based on the types and spatial grid and temporal frequency of the weather data recorded by these individual weather data feeds.

After processing the wind data, either or both of the raw source wind data and post-processed wind data are archived to offline storage. Because of the extremely large data sizes associated with these data feeds, often handled in batches of hundreds or thousands of gigabytes of memory, these data transfers from online to offline storage are done by hand to allow human verifications of adequate storage

space availability, to allow for human verification that the wind data was correctly processed and without any data corruption or omissions, and to conduct these taxing disk copy operations during less demanding times such as overnight or weekends.

This design approach used for the weather data infrastructure offers numerous benefits including but not limited to:

- permits the use of very custom acquisition and processing and archiving scripts that are specific to each weather data feed and the source server's file hierarchy and network protocol preferences
- full automation of virtually all of the data acquisition and processing scripts with built-in self-healing while improving reliability by employing two servers to allow for load-balancing and fail-over
- compatibility and portability to other Linux systems without modification through the use of standard bash scripts with exception to the MATLAB processing functionality, which is currently being converted to a nonproprietary scripting solution
- extensibility and flexibility to incorporate new weather data feeds that are similar or in a completely different form from those already being acquired and processed, conducive to a future common interface to an online wind data repository that would be of benefit for use as part of long-term research studies

With the design and initial implementation for the weather data infrastructure largely complete, the next step is to incorporate this component into the overall WTMx application design. Before embarking on that design, it is useful to review the requirements for the WTMx application design first.

5. APPLICATION ARCHITECTURE

The past few sections have introduced the WTMx portfolio of technologies, surveyed in detail the wide variety of wind data sources required as input to these algorithms, and specified the requirements of the various clients of the WTMx application(s) today and in the future. After a brief review of these requirements, the focus can shift to proposing a design and architecture for a system capable of meeting all of these requirements and the path to implementing this system. This all-WTMx encompassing architecture is hereafter referred to as the Wake Turbulence Application Architecture (WTAA).

5.1 REVIEW OF REQUIREMENTS

An introduction to the current system and processes supporting WTMx technologies development and deployment was discussed in Section 2.1, along with its inherent deficiencies relating to performance, fragility, and issues associated with scaling to the proliferation of new wind data sources being introduced.

In the course of defining the application architecture requirements in Sections 2.2 and 2.3 for the three application modes of operation, the analysis mode involves very large time spans of data as long as three rolling years across the 50+ airports of interest, leading to very large demands for data storage while being more tolerant of computation time delays. Conversely, the real-time mode operates over relatively small amounts of wind data but with intense requirements for computational speed to satisfy the real-time requirements of the data processing and prediction presentation. The playback mode has more intermediate needs for data storage and computational capacity because of its weather data spanning over days to months and minimal real-time needs.

Section 3 defined a broad yet detailed survey of all of the current and anticipated weather data feeds that must be acquired, processed, and archived with regard to data size, timeliness, latency, and performance. The need to conduct these operations in an efficient, automated, reliable manner with minimal hardship due to hardware and network issues was also discussed. Efforts to future-proof this design via the ability to accommodate future weather data feeds in an extensible fashion with particular attention to increasing data size were also discussed.

With the core requirements of the WTMx applications specified, we now turn to presenting an application architecture that meets these requirements.

5.2 WAKE TURBULENCE APPLICATION ARCHITECTURE (WTAA)

A detailed characterization of the business process rules, superimposed upon the business process flow diagram for each of the WTMx application usage modes, was presented in Section 2.3. Based on the variations in how these three application modes have differing demands for data storage and computational capability, a natural WTMx application architecture begins to emerge in the form of three

application components, as shown in Figure 14. In this architecture, the wind data are acquired and processed from all available weather data feeds by an application component hereafter referred to as the *Wind-ETL*, where ETL refers to the common software classification of Extract Transform and Load software commonly used to categorize applications that involve data acquisition, data conversion, and data archiving procedures used in many enterprise-level applications. An obvious second application component, the WFA algorithm component, ingests the wind data and generates availability outlook predictions. The third application component that emerges is the *WTMx Display* component, which is comprised of the functionality required to process WFA predictions into a standardized format for archiving and for display to the tower controller.

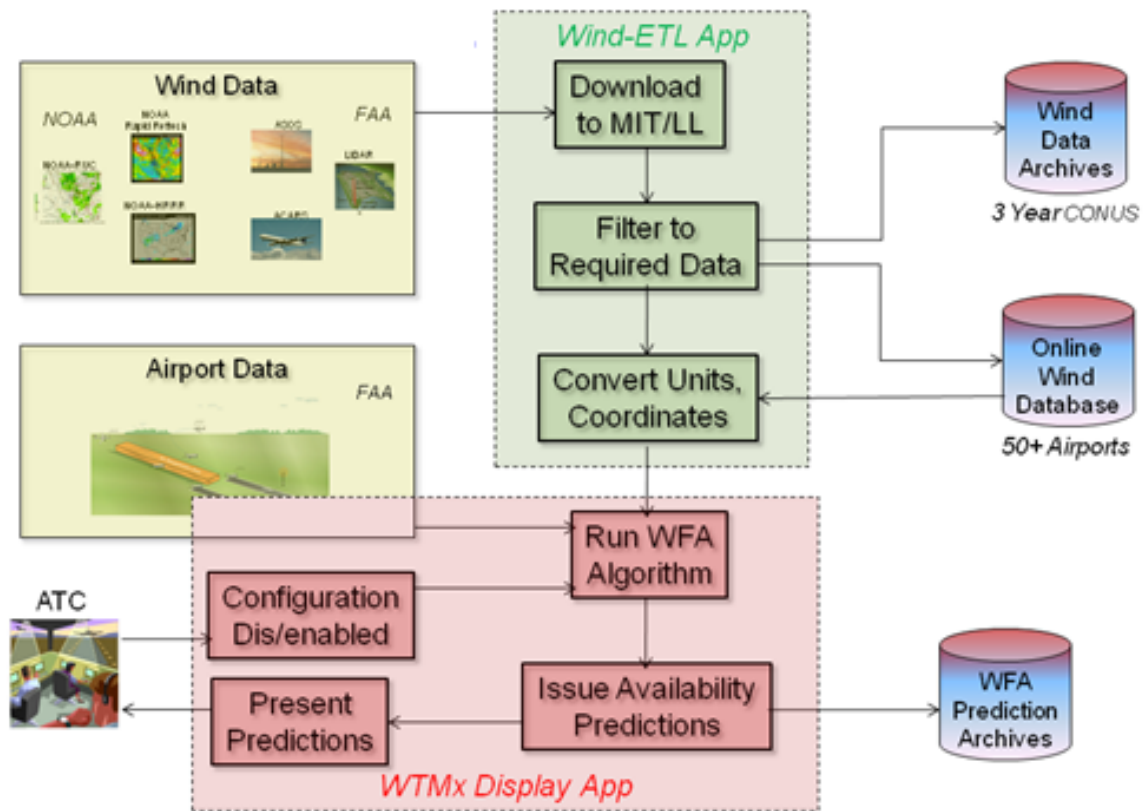


Figure 14: The WTMx application architecture emerges as composed of the Wind-ETL and WTMx Display application components, in addition to the WFA algorithm itself and databases.

In the WTMx application architecture, the Wind-ETL application component will acquire the weather data and archive this data into offline storage. The weather data will be further processed to allow

for lattice grid point rotations and units conversions as well as adjustments for outlier and missing data and data arbitration to form airport-specific files which will be archived to a database. This data is then forwarded on to the WFA algorithm.

The WTMx algorithm component encapsulates all of the algorithm-specific scientific theory and computations of the wake turbulence mitigation application, separate from the rest of the operational WTMx application architecture. By isolating the WFA algorithm logic from the other WTMx components, this permits the easy drop-in substitution of a particular WFA component version with either a MATLAB- or Java-based implementation, or for side-by-side regression testing of a new revision of WFA with a prior version, for any of the WTMx systems under development while the remainder of the WTMx data architecture remains unchanged. In the future, this will also provide scaling to run concurrent operations of multiple WTMx systems within the same data architecture instance, such as simultaneous WTMD and WTMA on a single computer system. This WFA encapsulation is already a requirement for current MIT LL needs, in which initial prototyping of the algorithm development is performed using MATLAB scripts for investigatory efficiency and flexibility, whereas subsequent performance versions of WFA will be coded in Java for computational efficiency to support long duration algorithm analyses and real-time application modes.

This algorithm component requests wind data from the Wind-ETL application component, and also requests airport specification information such as runway orientation and configuration and airport operating procedures from custom airport input files and uses these to compute predictions of WTMx availability and related data products to disseminate to the display component.

In terms of display, the developmental WTMA developmental/mock display was used as a template in defining system data requirements, as it is more sophisticated than the baseline WTMD. However, the design is driven to meet all anticipated WTMx variations. The WTMA concept includes a very short term forecast analogous to that for WTMD, as well as a longer term “outlook” of multiple hours. The WTMA Display application component receives the availability outlook predictions, along with various other intermediate data for display and/or logging, and converts this into a standardized prediction data format. This standardized prediction is archived into a database, and also simultaneously directed to the WTMA application display for viewing.. In a real time operational environment, this display component would also likely be responsible for interfacing with the air traffic managers to receive updates on airport runway configuration in use as well as whether the WTMA procedure is currently enabled.

5.2.1 Application Architecture In Detail

To further develop this WTMx application architecture, it is useful to define all the data flows between key application components and the constituents that interact with them. This is illustrated at a high level in Figure 15, in which all of the weather data feeds, shown on the left, send their data into the WTAA, which generates availability outlook predictions that are presented to the end users via the display. In an operational setting, the air traffic manager would use the prediction decision support tool to advise pilots on their routes and runway slotting.

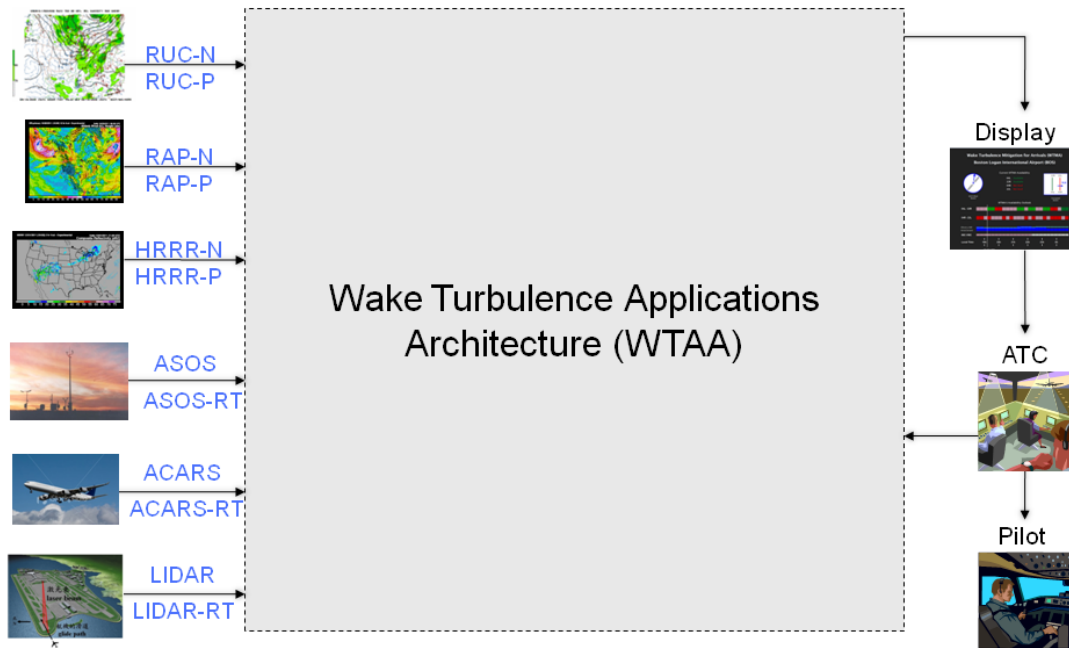


Figure 15: High-level wake turbulence applications architecture illustrating data flows between key role players. This schematic uses WTMA as the example mitigation procedure.

It should be noted that in many situations, multiple instances of weather data feeds may be used for the same class of data. For example, for NOAA-based forecast data, the pressure level (P) and natural level (N) editions of the forecast data are provided via this architecture, in support of research needs. For sensor measurement data flows, archived and real-time (RT) data feeds are shown to support both the analysis and real-time modes of this application.

This application architecture is illustrated in further detail in Figure 16, which incorporates all of the application components described in the previous sub-section. In this figure, the wind data flows to the Wind-ETL component, which is archived to the offline data archive. This data is also filtered and processed by the Wind-ETL component and the resulting airport-specific wind data archived into an online database. This wind data flows on to the WFA algorithm component, whether in MATLAB or Java form, and the ensuing availability outlook predictions are transmitted to the WTMx display (and archived) for viewing by the tower controller.

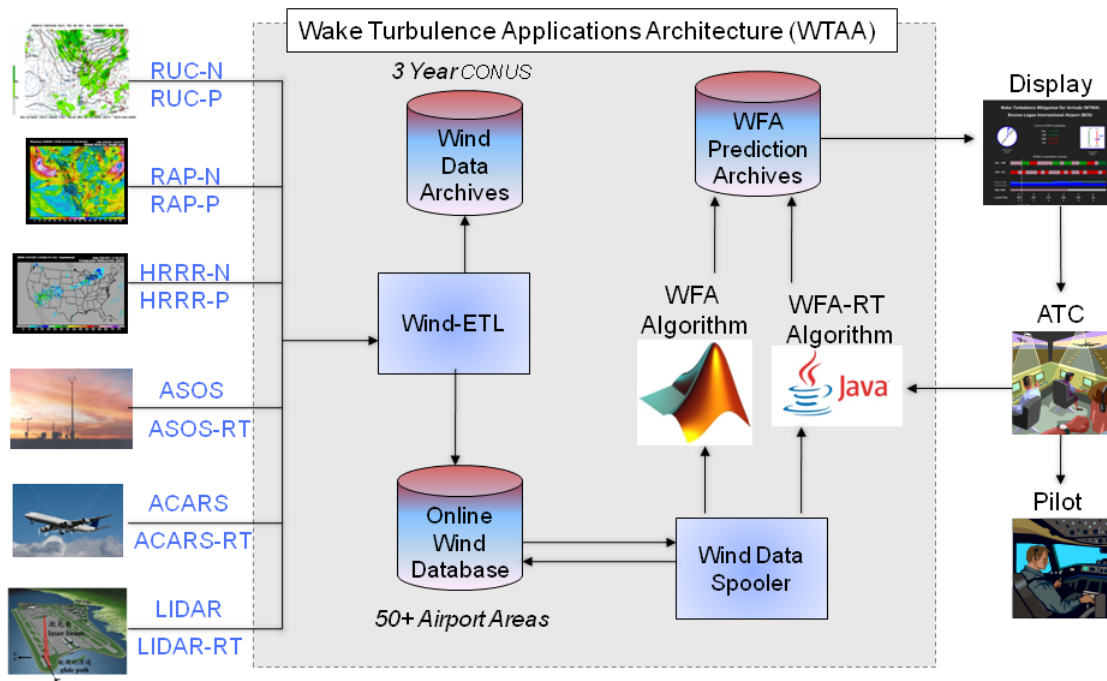


Figure 16: Detailed WTMx application architecture illustrating the data flows amongst the WTMx application components.

5.2.2 Wind-ETL Application Component

To further zoom into the left side of Figure 16, the next diagram focuses on the data flows from the variety of weather data feeds into the Wind-ETL application component (see Figure 17). This architecture diagram introduces the concept of a data message bus, which acts as a sort of mailing list and message collator for notifying interested listeners as new data becomes available on a designated message topic to which they have subscribed. This data message bus architecture component, commonly used in enterprise integration for integrating different enterprise systems and components into a single enterprise architecture, facilitates the communicability and distribution of new weather data to multiple listener components without the limitations of a direct link, which tend to be proprietary and vulnerable to single point of failure reliability issues. The use of the data bus also allows for simple replacement of any weather data source, or the addition of new weather data sources for extensibility, without any modification to the components listening to that specific *wind data topic*. In fact, the listeners would not even be aware that a change occurred unless it was noted in the specific data transmitted. Finally, the use of the data bus also supports the use of multiple servers used for acquiring or processing wind data due to the inherent multisystem, multinetwork fundamental features of data buses as part of enterprise architectures.

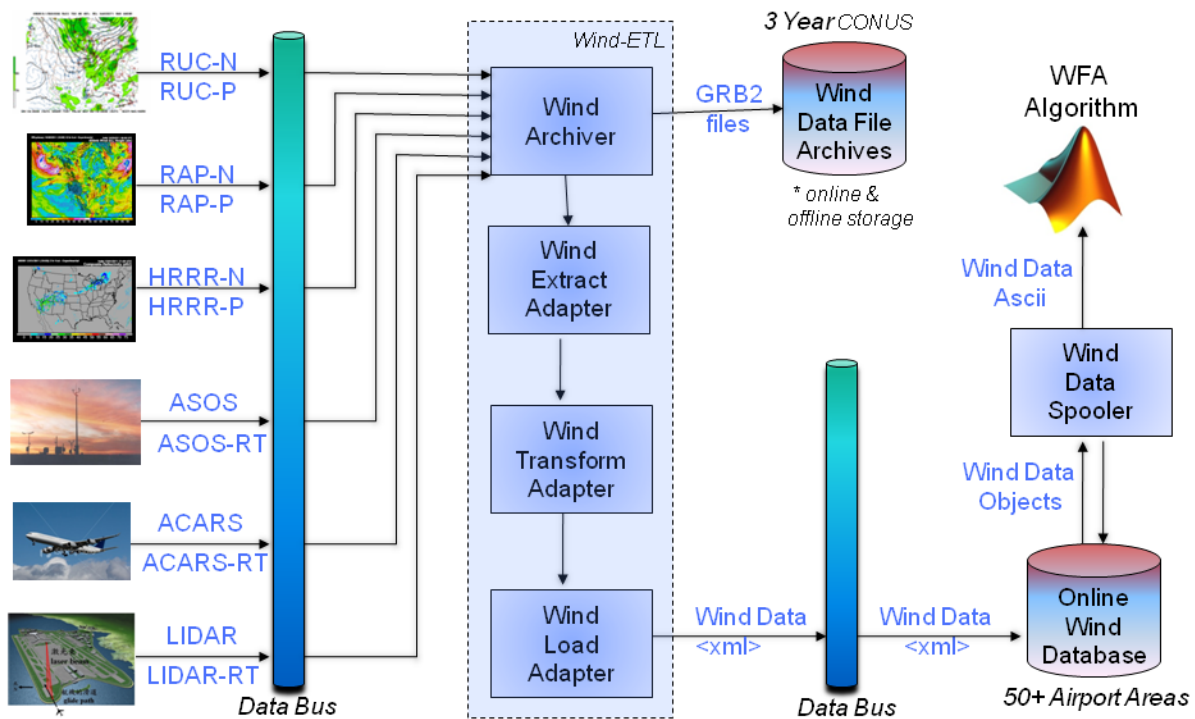


Figure 17: Data flows amongst the Wind-ETL related application components in the WTMx application architecture

The Wind-ETL application component listens to the data bus for each of the data feeds through its data acquisition subcomponent. Because of the large data size associated with the source data, the messages sent to the data bus serves to notify the Wind-ETL component to the receipt and cache location of new weather data rather than to actually transport the complete data payload. The newly arriving weather data is archived offline, as well as passed on to the remaining Extract, Transform, and Load sub-components (see Figure 17). The acquisition, extract, and transform components are composed of the bash scripts (and supporting MATLAB and Perl scripts) described earlier as part of the wind data infrastructure. The load component, which writes the airport-specific wind data into a database for broad accessibility and fast query times, is composed of Java code coupled with automated data bean and database table creation and update tools. The database component is configurable, and the specific database technology required will depend on the usage mode of the WTMx system as it relates to amount of wind data to be stored. The design considerations for selecting the specific database technology will be discussed in Section 5.2.5.

Considerable progress has been made on the Wind-ETL application component as part of early research studies on WTMA and earlier efforts via hand-activated bash weather data acquisition and processing in support of prior and ongoing WTMx analyses. This legacy work has been expanded and

integrated to form a more coherent Wind-ETL application component, although some parts remain to be finished. Figure 18 uses DVD-player-style buttons over each subcomponent or data flow to indicate its current status. For instance, all of the RUC and RAP data feeds acquisition and archiving subcomponents are complete, whereas HRRR acquisition and processing scripts are prepared and optimized to its large data size; however, activation of these scripts will begin shortly now that the new data storage system is available for data processing. Acquisition of research-source ASOS and ACARS data feeds, which suffer from a 24–54 hour reporting delay and are thereby suitable for research analyses but not real-time applications, are complete. No real-time source of ASOS and ACARS aircraft data are currently available at MIT LL, although some efforts are being extended outside of this project to gain access to real-time feeds of these data. Finally, the extract and load adapters are largely complete, although the final step of loading airport-specific wind data files into the database has not been performed due to disk storage limitations at the time it was developed.

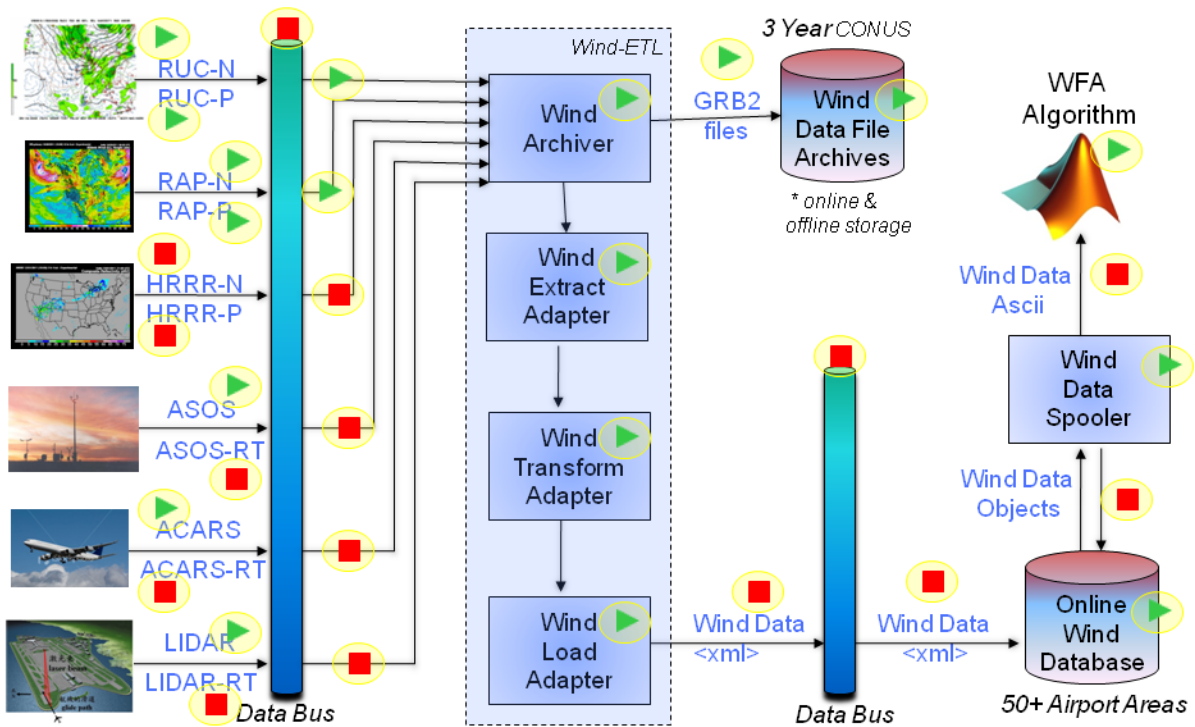


Figure 18: Progress on the Wind-ETL related components, in which green arrowheads indicate it is working and red stop buttons indicate not yet functional.

5.2.3 WTMx Algorithm Component

The WFA for each of the WTMx wake solutions, which represents the core of the wind-dependent wake turbulence solutions, is being developed independently of any specific WTMx applications architecture. This algorithm uses advanced scientific and mathematical principles to make predictions of the availability outlook of operational throughput based on very conservative safety factors derived from detailed historical performance analysis.

In the course of developing the WTMx algorithm component, it will be necessary to develop a modernized WFA component as well. A decision on whether the new WFA should be completely independent from, or a part of, its associated WTMx technology has not yet been made, and will depend on the amount of customized WFA behavior required for each WTMx algorithm among the portfolio of WTMx technology systems being developed. Any airport customization required for WFA can be written into the algorithm code in a generalized form, and the input files specific to that airport can contain properties to activate these specialized features. Where possible, a similar approach would be used to activate WFA features specific to a particular WTMx system to maximize the commonality of the core WTMx algorithm component among all the proposed WTMx systems.

Because the algorithm is currently being optimized and improved based on continuing weather data spanning years for operational availability, safety, and benefits, the algorithm is currently coded in MATLAB to support ongoing rapid prototyping and testing. Simultaneously, a Java-based real-time performance version of the algorithm is being developed to support WFA analyses spanning multiple years among the full array of airports, and to serve as the real-time developmental implementation of WFA. The availability outlook prediction data produced by the Java-based WTMA algorithm will be output in Extensible Markup Language (XML) form to promote openness and simplicity in data sharing, such as to the WTMx display component, which will be discussed in more detail in Section 5.2.6. This Java version of the WFA is also designed to be used to support ongoing and future WTMD analyses as needed.

5.2.4 Display Application Component

The ability to display WTMx/WFA information is generally intended to serve two distinct purposes. The first is to allow information visualization during the development and testing phases. The second is to accommodate the potential for future display requirements within the operational FAA environment. For this discussion, we turn to the display concept under development for WTMA to serve as a general model, as its design establishes a reasonable target for establishing WTMx display requirements. The developmental WTMA display application, which arises from the right side of Figure 16, accepts the runway-specific WFA availability outlook predictions updated every minute for archiving and display.

The WTMA display design as currently conceived would allow it to serve as the primary interface between the WTMA application and the human air traffic managers. A screenshot of a prototype WTMA

display is shown in Figure 19. It is recognized that this specific iteration is subject to change as operational requirements become more clearly defined and the specific FAA environment in which it will reside is more firmly established.

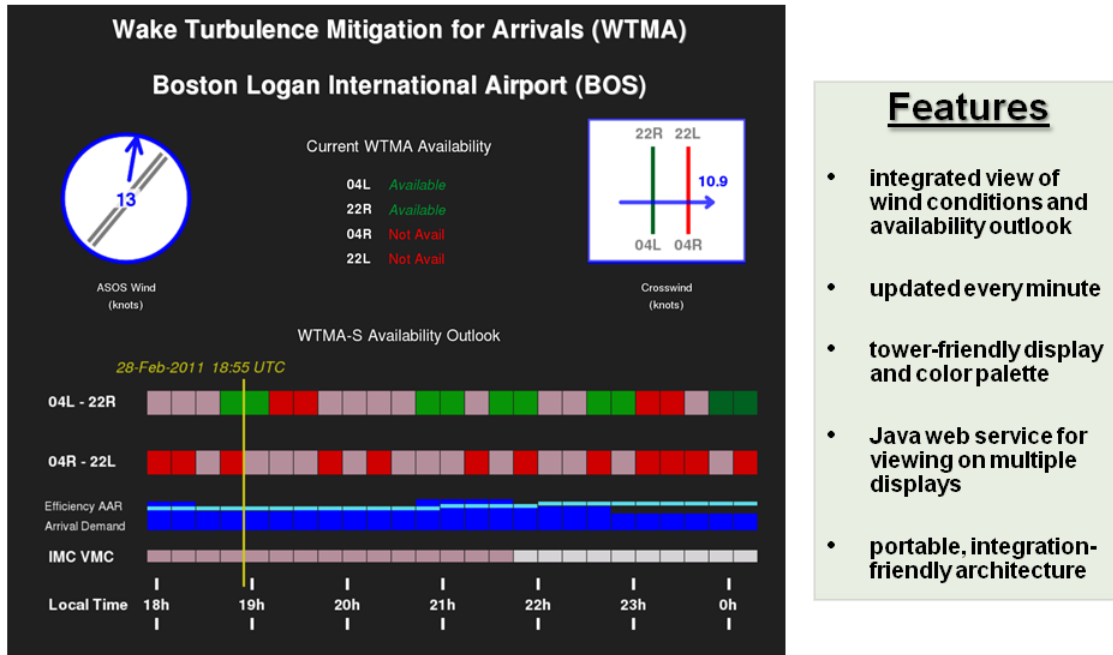


Figure 19: Display concept of the WTMA Display user interface component.

The WTMA display as designed would allow the capability to:

- notify the controller if the WFA currently permits the use of reduced aircraft spacing based on observed local wind conditions and forecasts
- track whether WTMA operations are currently enabled or disabled
- provide situational awareness of current winds and runway crosswind conditions, including direction and velocity for the parallel runway set
- display the longer-term availability outlook prediction for each runway in 15 minute increments for the next 4–6 hours into the future from the current time
- overlay the airport arrival rate and airport efficiency on the same time scale for the same multihour outlook

- display whether Instrument Meteorological Conditions (IMC) or Visual Meteorological Conditions (VMC) operations are anticipated to be in effect over the same future outlook

The WTMA display GUI is designed to be tower-friendly in its appearance with regards to color palette and information layout. It provides all of the current state information relating to current WTMA operations, and aggregates the wake turbulence outlook information that an airport supervisor would need to determine whether to enable or disable WTMA procedures. This tool would support decision-making regarding flight operations planning, and specifically, whether to assign runway arrival slots to aircraft prior to pushing back at their origin airport.

The WTMA display interface is designed as a part of larger WTMx application architecture illustrated in Figure 20, which is in effect a more detailed representation of the right side of Figure 16. In this architecture, the WFA algorithm, either in the MATLAB prototype or Java real-time form, is designed to receive updates on current airport runway configuration and WTMx enabled/disabled status from the airport tower supervisor and computes the availability outlook based on the most recent wind/forecast data. The design of this information flow has not yet been determined, pending further guidance regarding future WTMx operational procedures during airport runway configuration changes. Once computed, it transmits its availability outlook predictions and supporting wind data products to a predictions data transformations adapter subcomponent, which wraps the prediction data and posts it to the *predictions topic* on the data bus for dissemination.

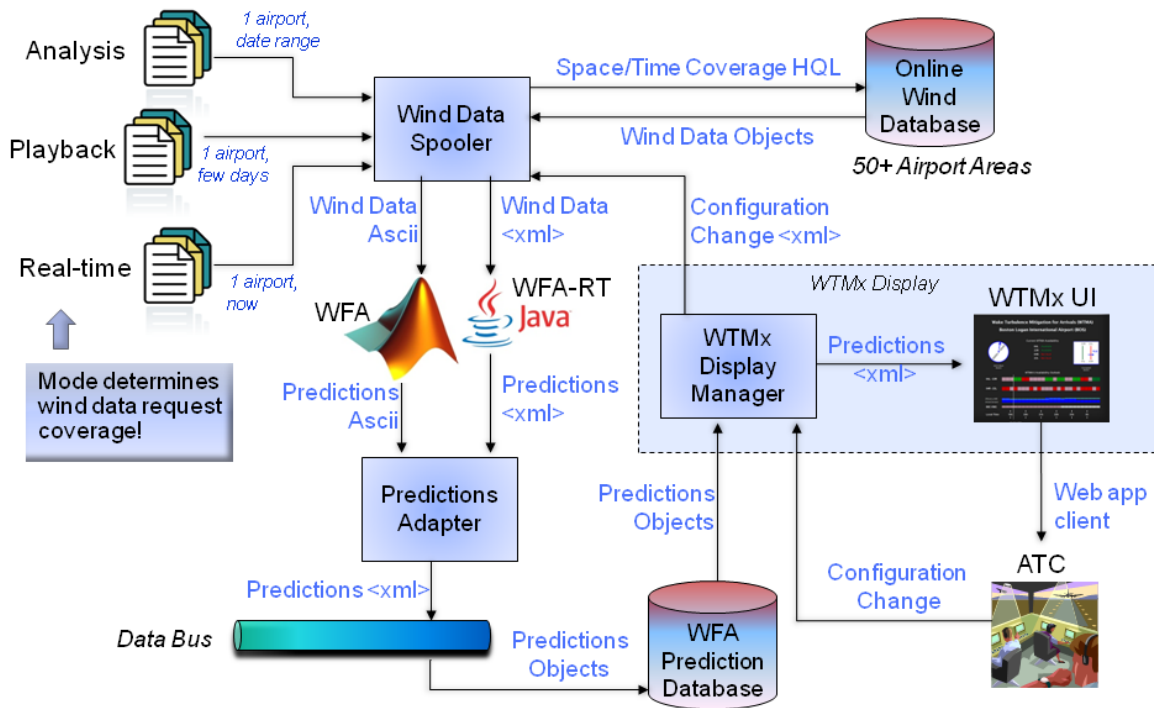


Figure 20: Data flows among the WTMx Display-related application components for each of the three application modes.

Among the listeners to this *predictions* topic is the predictions database, which archives the predictions sequentially every minute to support research studies and future playback demonstrations. For analysis mode and playback mode deployments of the WTMx display application, the WTMx display controller will access this prediction archive to present to the user via the WTMx display GUI sub-component, with the privileged ability to step forward or backward in time through the prediction archive.

In WTMx display application environments running in real-time mode, the WTMx display controller will also listen to the *predictions* topic on the data bus, and process the prediction information into a form that can be parsed by the WTMx display user interface for real-time viewing of the WTMx outlook predictions for the current minute. In real-time mode, the display GUI must automatically update itself to be valid for the most recent minute, or warn the controller if it is unable to acquire or display valid current prediction availability outlook. Real-time predictions also are likely to be archived to support future analyses, such as benefits studies.

To support the analysis, playback, and real-time modes of the WTMx display application, this architecture was designed with the intent of being flexible to allowing multiple means of distributing prediction information to numerous consumers of this data while maintaining the high performance

requirements for real-time dissemination of availability outlook predictions. All of the data transformation adapters and display controllers are coded in Java for performance, and to promote platform independence of the WTMx display application. The WTMx display GUI uses a SOA built on Java servlet technology, making use of HTML5 (HyperText Markup Language) and CSS3 (Cascading Style Sheet) markup language for the display to exploit the multi-platform and performance benefits of current conforming web browser technology, including familiar open source browsers such as Firefox. All of the data transmissions through the data bus are XML-based to support the incorporation of the WTMx display application into existing and future tower operations equipment, and to enable a capability to publish availability outlook prediction data to upcoming FAA data sharing technology for use by other NextGen applications.

Progress has been made into the development of the WTMA display application, albeit at a smaller rate than the Wind-ETL application discussed in the previous section (see Figure 21). The WTMA display GUI prototype is complete for the real-time mode version, whereas the analysis mode and playback mode versions of the GUI are still being designed. The WTMx display controller driving the GUI is complete, although its data processing component for accepting predictions data from either the data bus or the predictions database remain to be completed. A prototype of the predictions database is awaiting testing once the WTMA display controller is ready to connect to it. The tower supervisor feedback loop delivering the current airport runway configuration and WTMA enabled/disabled status has not been designed, and will be postponed until the means of accessing this tower supervisor input is determined. Based on current project plans, the WTMx display application development efforts will be directed towards completing the analysis and playback modes of the application, with the real-time mode version delivered at a future date.

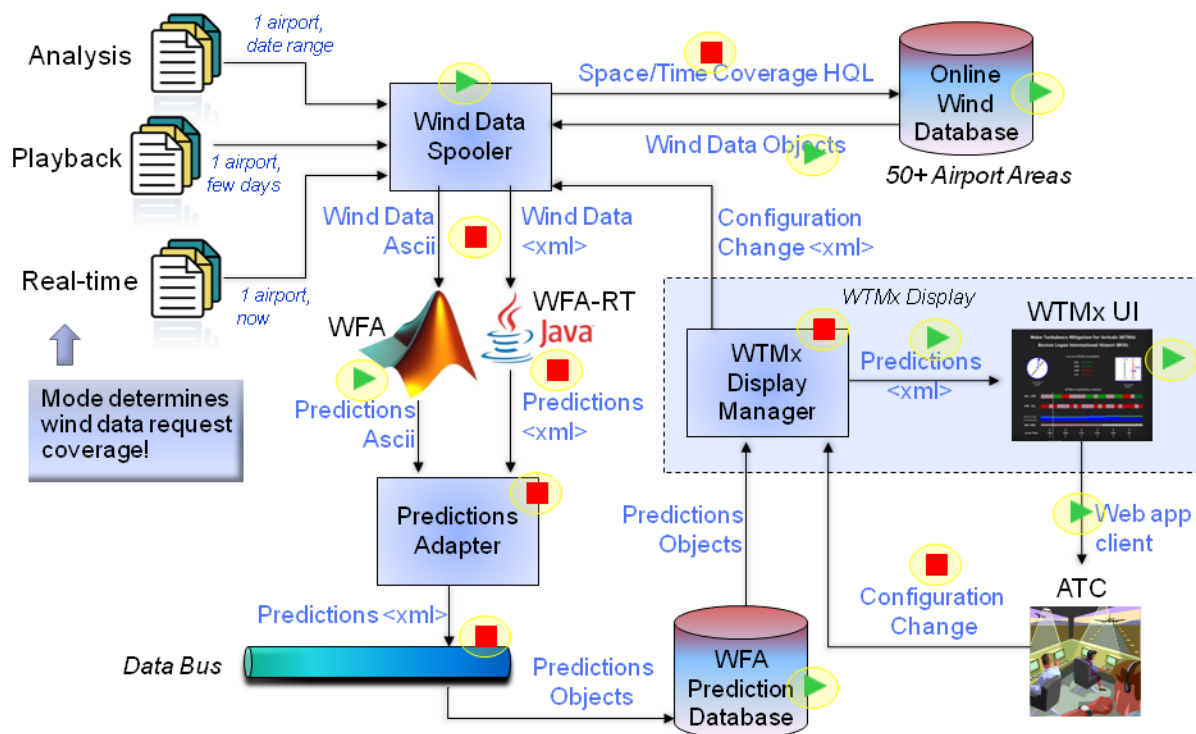


Figure 21: Current progress on the WTMx Display-related application components, in which green arrowheads indicate it is working and red stop buttons indicate not yet functional.

With a detailed review of the design and interactions of the key WTMx application components including Wind-ETL, WFA component, and WTMx Display now complete, we shift our focus to the design of the underlying data storage systems for WTMx, specifically the databases.

5.2.5 WTMx Database Design

In the WTMx data architecture overview presented earlier in Figure 16, three distinct database storage mediums are introduced with the emphasis on defining their primary function and what data flows into and out of each of them. The three storage mediums are:

- (1) Offline Wind Data Archives
- (2) Online Wind Database
- (3) WFA Prediction Archives

To refine the database storage design, a characterization and sizing of the data flows and storage sizes is necessary to help determine what database technology selections would be appropriate for each database medium, as illustrated in Figure 22.

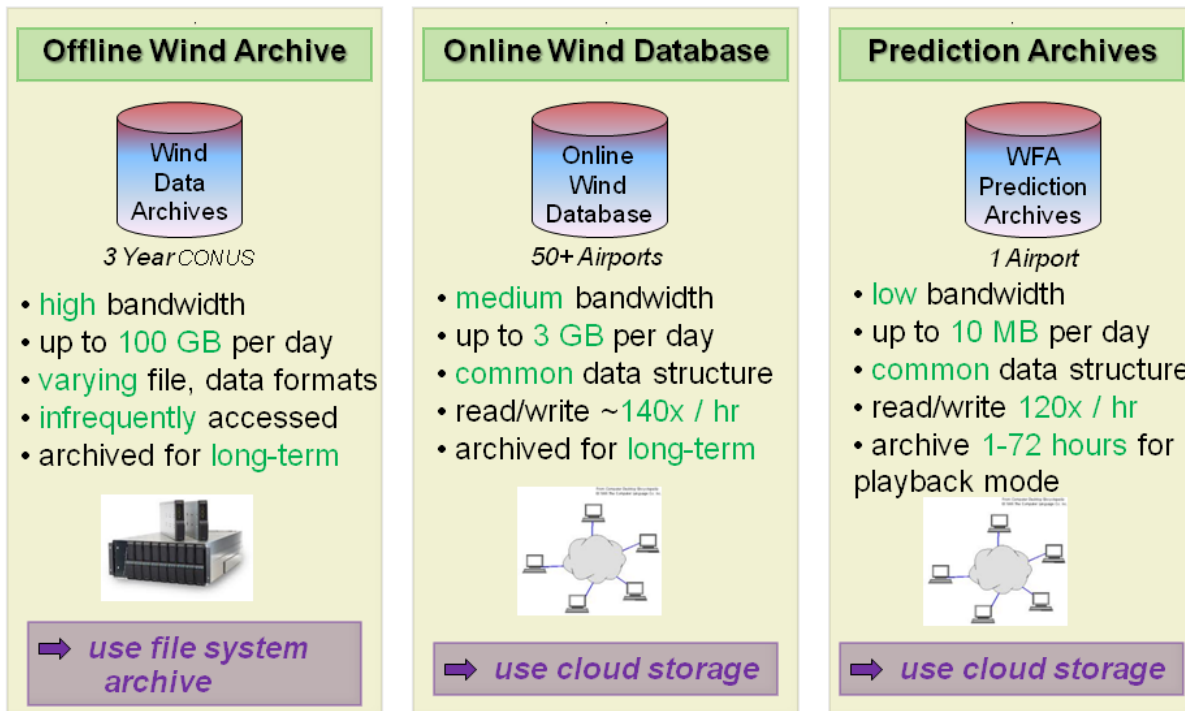


Figure 22: Database technology selection is driven by the data flow rates and archive sizes of each of the three databases introduced as part of the WTMx data architecture.

The long-term wind data archives, shown in the left panel of Figure 22, contains all the wind data from a broad array of wind data feeds over a rolling three year timeframe. This data originates as minimally filtered data from the Wind-ETL component on an hourly basis for forecast model data and a per-minute basis for ASOS and aircraft data, at a data rate of up to 100 per day. Furthermore, the wind data is composed of a wide variety of weather data products contained in differing file formats and data formats. To retain the inherent data compression advantages of their native file formats and to minimize processing time on these extremely large files, these long-term wind data will be archived directly to the file system using their native file formats if the data comes in *GRIB2*, *zip*, or *tar* formats; *text* formats will either be stored as-is, or further compressed into a *zip* file. This approach also preserves the ability for data sharding according to wind data source, a “Big Data” technique useful for splitting large data storage volumes into multiple storage locations across numerous (often distributed) systems.

The online wind database, shown in the center panel of Figure 22, receives the key wind data parameters such as u , v , pressure, and height for each latitude-longitude-altitude coordinate in the airport areas of interest, about 3 GB per day. This database also will span many years, and retrieves wind data to deliver to the MATLAB and Java-based versions of WFA for the domain of wind data points of spatial and temporal interest. Although any single wind data request will span coordinates within a single airport terminal area regardless of usage mode of WTMx, MIT LL analyses can be expected to request wind data over time from any of the 50+ airports that are to be studied.

To support this segmentation of the data, a distributed database design is under consideration for the online wind database using a series of independent cloud nodes for database storage, in which a natural data sharding approach will separate data into each storage node by airport, ensuring that all related wind data for a single airport remain on a single storage node for network bandwidth efficiency. The specific cloud database technology has not been decided at this time, although there are numerous open source cloud databases available that satisfy the desired features and have demonstrated a proven track record in large deployments of well-known social networking and search websites. Many of these distributed databases are also compatible with the more sophisticated data query languages required for retrieving data from distributed databases, such as the Hive Query Language (HQL) operating on an open source Hadoop (MapReduce-based) distributed file system. With cloud database technologies advancing at a rapid pace, the specific technology selections for database and query language will be made at the time of implementation.

The WFA predictions database, described in the right panel of Figure 22, receives the WFA predictions and stores them, both to serve the WTMx display controller on a per-minute basis for real-time applications and for aiding in the forward/reverse playback usage mode of WTMx. It has also been suggested that in a real-time deployment demonstration, it would be desirable to archive the WFA predictions for the duration of the demonstration for later benefits analysis. For this reason, while data flow rates may be expected to be about 10 MB per day, the total size of the archive depends on whether WTMx is to be used in playback mode or real-time mode. While even freely available open source databases such as Java Derby and MariaDB can support the storage and data flow rate demands of playback mode, the longer term database size for real-time deployments requires a more robust database technology, and for this reason a distributed database design is proposed for the real-time instance of the predictions database. It is important to note that the component-based design of the WTMx data architecture allows for an easy drop-in substitution of either database technology according to the WTMx usage mode.

5.2.6 Considerations for WTMx Integration

In the last few subsections, the detailed application architecture for WTMx has been presented, with a significant focus on the key components required, the interactions and data flows between these components, and discussion regarding how this application architecture meets many of the requirements laid out in Sections 2.2 and 2.3. This subsection details the selection of specific software technologies to support this architecture, and how the design of this data architecture and its technologies meet current

development and testing needs at MIT LL and facilitate future deployment and integration into one or more test sites as part of a possible future operational demonstration.

A key design element of the WTMx data architecture is its ability to share data, specifically wind data and WFA predictions, within itself and among other applications and tools on-site and in remote locations. This is done to accommodate the data sharing concept that is central to the FAA NextGen framework that is currently under development. To share data, an application requires a means of identifying the location of this data, details about its structure and format, and a way of accessing this data. This is the subject of enterprise integration patterns (EIP), and is detailed in the following paragraphs.

To specify the location of data to be shared, EIP makes use of a common enterprise integration message bus (or service bus) that promotes synchronous and asynchronous data sharing among disparate applications regardless of platform or programming language. The technology that provides this common message bus is commonly referred to as a message broker, which acts similarly to a post office to pass application data messages within and between applications in which the message formats and data formats and data exchange protocols are predefined during development and common to all components. Applications communicate with and access data on the message data bus through the use of enterprise service bus (ESB) adapters, which transform (adapt) foreign data messages received into the preferred format of the application; these ESB adapters can also encode application data sent by the application to the message bus employing the predefined data format. Message brokers also provide the ability to categorize and sort messages through the use of unique *message topics*, and employ a *publisher-subscriber* pattern for allowing any of one-to-one or one-to-many or many-to-many relationships between message senders and receivers for a specific message topic. These relationships can also be modified dynamically, offering tremendous flexibility in defining how an array of different applications can be integrated, and how their integration relationships may evolve through time.

Although ESB adapters allow various parts of an application to send and receive messages to and from the message broker, proper application data flow requires a means of coordinating the identification of message endpoints (origination or destination), usually as an ESB, within an application, along with the routing information to connect these endpoint(s) with the appropriate message topic on the message broker. Apache Camel, an open source software framework, is selected as the technology to enable WTMx applications to specify which message topics to send and receive from, and to which ESB adapter endpoints within an application that messages will originate from or be delivered to. Apache Camel has been routinely demonstrated in both the corporate world and government systems to operate across multiple sites simultaneously in real-time under heavy loads.

Part of the WTMx data architecture requires a need to explicitly specify the data format to all providers and consumers of data on the message bus so that it can be shared. XML provides a technique for allowing data to describe itself, and the use of XML Schema Definition (XSD) allows a programmer to specify the format of the XML file and its restrictions to enable automated data checking and validation to confirm that a specific data object conforms to this XSD format without errors due to: misalignment of

data fields, bit errors, data dropouts, invalid data, data out of range, etc. The early work of defining these schemas has been completed for the WTMA application, although the ESB adapters to connect the WTMx components to the message bus have not yet been written. As other WTMx applications are implemented, new XML/XSD specifications relevant to those WTMx applications will be required as the data structure and data formats for each application are defined.

The use of message brokers is consistent with the System Wide Information Management (SWIM), network-centric, service-based concept advocated by parts of the FAA in other projects. The Apache ActiveMQ message broker, which is the dominant open source message broker in use today, is the broker selected for WTMx data architecture, Commercial versions of ActiveMQ are supported by FuseSource (recently acquired by Red Hat Inc.) and are already in use in other parts of the FAA, as well as other large government and private industry enterprise applications. FuseSource also includes a supported version of Apache Camel and an ESB adapter specification, ensuring that the integration approach used for the WTMx data architecture is consistent with the network-centric approach favored in other programs within the FAA.

With this foundation for the data sharing architecture, these concepts can now be applied to the design of the WTMx data architecture. In Figure 23, an illustration of the WTMx enterprise integration plots the basic flow of data from the data *publisher(s)* on the left side, through a custom ESB adapter to the message broker for that data topic, which is then received by the *subscriber(s)* on the right after a transform by another custom ESB adapter. For instance, the left side of the prediction data sharing panel in Figure 23 reveals how a MATLAB-WFA and Java-based WFA can co-exist within the same WTMx data architecture simply by selecting only one algorithm implementation to publish WFA prediction messages (in XML) to the *predictions data topic* in the message broker. Similarly, the right side of this same panel demonstrates how wind data and prediction data can be shared with a variety of subscribers such as traditional and cloud databases and outside applications through the use of custom ESB adapters that transform the common XML-based data into the preferred format of the subscriber. In each of these situations, the only change required to “hot-swap” one data flow to another is an update of the routing rules defined for the Apache Camel engine.

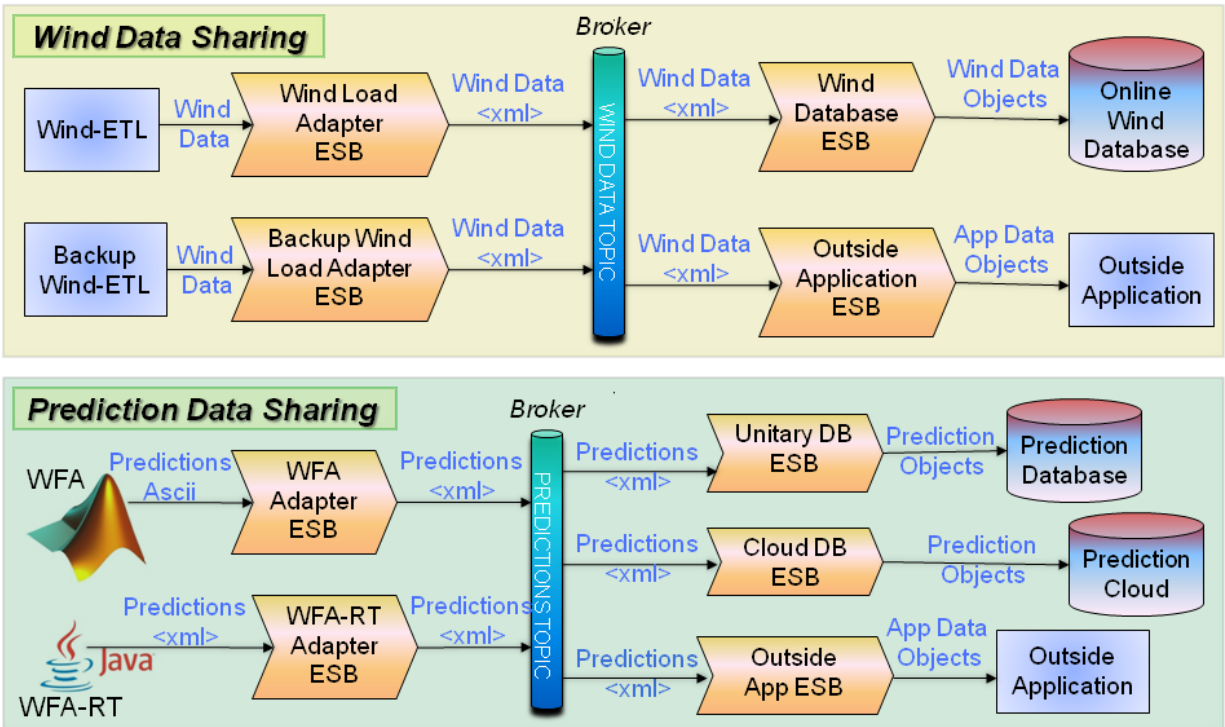


Figure 23: Enterprise integration in the WTMx data architecture, in which wind data and predictions data in each message broker topic can independently originate with one or more publishers (on the left) and be shared with a variety of subscribers (on the right). The use of Enterprise Service Bus (ESB) adapters permit applications to connect to the message broker.

The architectural flexibility gained by this enterprise integration pattern offers the WTMx data system the ability to support a variety of substitutions and alterations in its component and data flows. This flexibility is useful to support varying development and test evaluation needs for data sharing in the analysis, playback, and real-time usage modes, and to incorporate future anticipated and unanticipated data flows, data sources, and data destinations.

Besides uncertainties in the future capabilities of the system, there are other more longer-term design decisions to be examined. One area of ambiguity relates to the manner of deployment topology of WTMx during the evaluation prototype demonstration. While no specific deployment topology as been defined as of yet, it is useful to evaluate the data architecture proposed here in the context of what the selected deployment topology might be when the system is ultimately fielded. In Figure 24, three possible candidate deployment topologies are introduced in the form of integration schematic overviews, demonstrating how the two primary data sharing hooks (corresponding to the *wind data* and *prediction data* message broker topics in Figure 23) proposed in the WTMx data architecture support each topology.

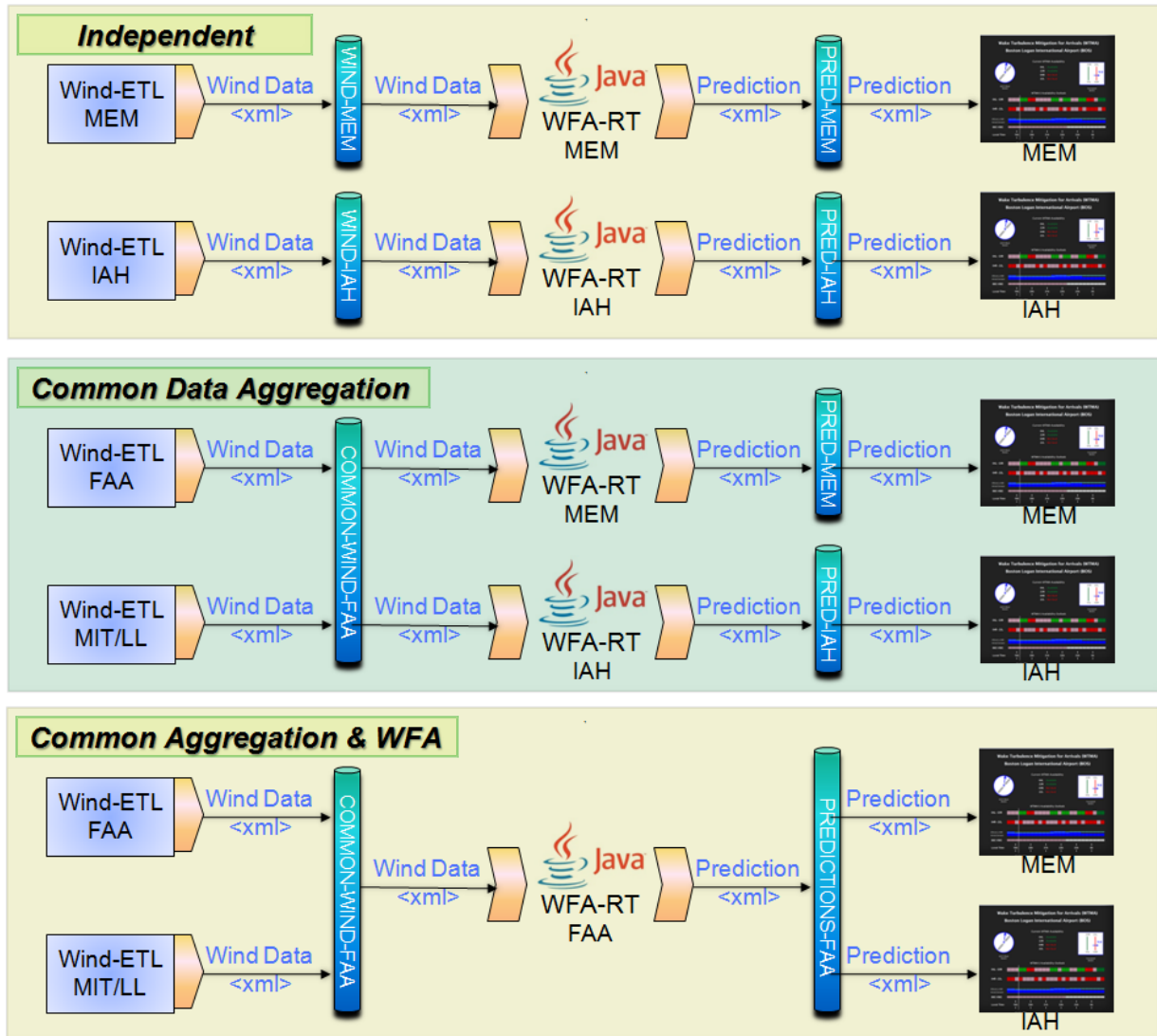


Figure 24: WTMx data architecture supports three deployment topologies including independent airport deployments, deployments using a shared wind data aggregator, or deployments with a centralized aggregator and WFA, which is piped to each airport site-based display (specific airport site references are strictly notional).

Although a WTMD system has been deployed to serve multiple individual airports (upper panel in Figure 24), the high bandwidth and computational requirements for processing large HRRR data feeds (expected to support future WTMx iterations) exceeding 100 gigabytes per day pose significant computing and network challenges, especially in a wider rollout to multiple airports, and could influence future deployment strategies. One possible solution to restrain the impact of the HRRR data feed size is to

aggregate all the wind data feeds at a single site (or two if a backup site is employed), and distribute the smaller highly filtered, airport-vicinity wind data individually to each airport site participating in the prototype evaluation, whereupon receipt, the airport site would complete WFA processing locally and display the WTMx status to the end user (middle panel in Figure 24). Another alternative is to conduct both the wind data feed aggregation *and* generation of WFA predictions at a single centralized site on behalf of *all* participating airports, and distribute only the airport-specific prediction data to each airport control tower for display to the controller (lower panel in Figure 24). While the specific FAA deployment topology has yet to be decided, it is important to note that the WTMx data architecture proposed in this document is compatible with any or all of these topologies. Furthermore, this flexibility to deployment topology also offers advantages during the development and testing phase of the WTMx data architecture, in which one topology can be used by the analysts and developers at its inception, quickly altered to another topology to suit testing at MIT LL and/or the sponsor site(s) and collaborator site(s), and amended once again to a third topology for deploying the evaluation prototype at one or more airport tower(s).

An additional deployment consideration concerns system scalability and application portability with regard to the range of hardware and software platform(s) to which it can be deployed. Historically, enterprise-scale IT systems have been built at high cost on a high performance CPU and memory equipped server/mainframe machine (or a small server cluster) optimized for running large applications. Recently, as the costs of now commodity CPU and memory hardware have fallen and the network bandwidth capability increased, enterprise applications are being developed in a more net-centric manner to take advantage of these inexpensive, massively-multi-core compute farms composed of numerous server nodes for increased application scalability and enhanced reliability while gaining redundancy of both software and hardware. To expand the capabilities and resources of the server farm, one simply provisions additional server nodes that seamlessly assimilate into the server farm. A key benefit of this redundancy is that if a hardware or software component or server node fails, regardless of the cause, another unit will automatically assume the responsibilities of the failed component, provided that the application by design does not store any *irreplaceable* state information on the failed component by employing the widespread practice of data replication.

To operate across hundreds or even thousands of these stateless servers in a cohesive manner, the traditional direct client-server model of application development has been replaced by a net-centric, service-based software architecture employing virtualized server platforms with access to virtualized data storage (since servers can no longer share or persist state information locally) using open web-based protocols (specifically Web Services Description Language or WSDL) and standardized data formats to facilitate data sharing. These advances ultimately led to the birth of cloud technology, and in turn, cloud-based applications. By conceptualizing an application as a collection of loosely coupled, modular distributed services composed of a set of multiple cloned software instances running in parallel across the compute farm, the benefits of parallel processing drive scalability and application component redundancy, all with no direct application dependence to any specific computer hardware or operating system or programming language.

Although the FAA has not formally established any specific deployment plans to cloud platforms at this time, it seems prudent to examine the suitability of the WTMx data architecture to being deployed on a public or private (or hybrid) cloud in the future. To facilitate this analysis, the WTMx data architecture can be divided into four principal deployment constituents:

- (1) data acquisition and Wind-ETL,
- (2) WFA,
- (3) WTMx display, and
- (4) data storage databases.

The ability to operate parts or all of a WTMx application in the cloud will depend on the feasibility of running each of these four deployment constituents, a set of loosely coupled distributed services, on a cloud server farm.

The first deployment constituent, data acquisition and ETL processing, is built on variety of platform-specific bash, Perl, and MATLAB scripts and Java components, and owing to the large data sizes involved and network bandwidth requirements along with the inherent state dependence required to monitor the progress of data acquisition, this constituent is not well suited to running in the cloud. In addition, the high costs associated with attaining the hardware redundancy inherent to cloud computing as it relates to CPU power and data storage memory are beyond current budget considerations, thereby restricting the data acquisition and ETL deployment constituent to a less portable yet highly efficient single platform, non-cloud solution. Continuing downstream within this constituent, however, the reduced-size, filtered wind data (extracted during ETL processing) is sent to the *wind data topic* on the message bus, and this data message transmission subset is well handled via a cloud. So while this constituent as a whole is ill-suited to running within a cloud, its filtered wind data output can be connected to the cloud.

For the WFA deployment constituent in (2), although the MATLAB version of WFA could only run in a limited capacity in a cloud server because of the inherent inefficiencies of MATLAB with regard to its use of volatile memory and reduced CPU performance, the Java-based version of WFA is fully capable of running in a cloud environment, and being based in Java, is portable to any hardware platform on which a Java Virtual Machine (JVM) is available. Moreover, the WTMx display constituent in (3) is inherently built from its inception as a service-based web application, and can already run in a cloud without *any* modification.

For the data storage databases deployment constituents in (4), the cloud suitability for the two downstream databases containing the filtered, airport-vicinity wind data and WFA predictions data is apparent since the WTMx data architecture proposes to use cloud databases for each of them. However, for the file database archiving the newly acquired CONUS-wide wind data from the source, transitioning

to a cloud database system would be difficult because of the large data sizes and costs involved, especially in light of the high data flow rates and data storage requirements and additional costs incurred because of the built-in data and server redundancy intrinsic to cloud systems. In particular, the specifications for the new data storage server being acquired are based on the requirements for storing data from all of the wind data feeds (dominated primarily by HRRR) for a three year rolling window, and any attempt at data redundancy motivated by the use of a cloud database make this prerequisite unattainable.

Returning to the feasibility of deploying the four WTMx constituents (in part or as whole) to the cloud, although the data acquisition and ETL processing and file-based, long-term database archiving require the use of a stateful server, thereby discounting operations in the cloud, all of the WFA processing and WTMx display and downstream databases for wind data and predictions data storage are compatible with cloud software design, and could be transitioned to a cloud infrastructure if such a need should arise in the future advancements of the portfolio of WTMx programs.

In summary, this subsection has presented the data sharing and system integration aspects of the WTMx data architecture through a detailed discussion of its foundation on enterprise-level messaging and a distributed loosely coupled, service bus-based component structure, including a synopsis of the data sharing hooks and many of the software technologies selected to support WTMx integration. Of several plausible WTMx deployment topologies envisioned, the data architecture being developed is compatible with any or all of these topologies. Concepts in SOA and cloud computing were reviewed with regards to their suitability to hosting WTMx applications, and the data architecture proposed for WTMx as it is currently being developed appears to be largely compatible with cloud computing with the exception of the data acquisition and ETL processing constituent which is not optimal for cloud deployment due to its data/bandwidth size and hardware costs.

6. SUMMARY AND FUTURE WORK

A generalized, extensible WTMx application architecture has been designed and partially implemented for intended use as a candidate common core architecture for each of the future WTMx systems under consideration. This architecture aggregates hourly wind forecast information from NOAA such as RUC, RAP, and HRRR with weather sensor observations in the vicinity of the airport including ASOS, ACARS, and LIDAR for archiving to long-term offline storage. This weather data is further processed to filter the weather data to the preferred wind products in the limited geographic and altitude levels of interest (surface and low level aloft winds) for the set of 50+ airports likely to be investigated for WFA and WTMx development and possible WTMx operational evaluation and deployments.

The magnitude of the size and diversity of data acquired, archived, and processed presents unique challenges in data storage and CPU processing power; the former are addressed through the use of performance-optimized code and the latter through the acquisition of an additional data storage server.

The WTMx application is designed to operate in analysis, playback, and real-time mode. While the analysis and playback modes operate over large expanses of weather data spanning durations of years for up to 50+ airports conforming to more relaxed performance requirements, the real-time mode operates over narrow weather data expanses at a single airport with high performance requirements to ensure timeliness in propagating availability outlook predictions. To simultaneously support all three modes of operation, the WTMx application architecture is divided into three major components including the Wind-ETL component, WFA algorithm, and WTMx display component, in which the data flow requirements associated for each mode can be identified and accommodated for in the design of each of these components.

The Wind-ETL application component aggregates the weather data and is responsible for archiving, extract, transform, and loading the data into small airport-specific weather data files for archiving into a wind database. The wind data is processed by the WFA algorithm in MATLAB or Java form, and availability outlook predictions are generated. These predictions are directed to the WTMx display application component, which archives the availability outlook predictions and directs them to the WTMx display user interface in the airport tower. The feedback loop notifying the WTMx system of changes to airport runway configuration and the enabled/disabled status of WTMx operations will be designed at a later date as this operational workflow becomes better defined.

Progress on the weather data aggregation software is nearing completion, although limitations in available disk storage and CPU processing power could pose long term challenges if additional data storage requirements are imposed. The Wind-ETL data filtering code development is nearly complete, whereas the weather data processing is a work in progress. The WFA algorithm, which exists in MATLAB form, continues to gain improvements in benefits and safety. The development of the Java-based performance version of the WFA algorithm is in progress. The WTMA display component, which

represents the general display capability requirements of future solutions beyond WTMD, is partially complete, in which the display interface prototype and supporting display controller are complete with additional work remaining to complete the ingestion of availability outlook prediction information. Although the data sharing formats have been specified, the corresponding service bus adapters to provide enterprise integration have not been written as of yet.

When complete, this WTMx application architecture offers the opportunity to provide a platform for developing and testing the WFA algorithm in all three application usage modes as part of WTMx program development in support of ongoing analysis and operational prototype efforts. This architecture is flexible to accepting new weather data feeds as they become available, advances in WFA algorithm specification, and ongoing changes to the content of the availability outlook prediction data set as data needs evolve.

This new wake turbulence application architecture directly promotes extensibility, portability, and performance with open source at its roots and is designed to be used stand-alone or integrated into larger infrastructures without regard to network environment. This includes deployment topologies such as independent data flows for each airport, common data aggregation at a centralized FAA facility, and / or common data aggregation and WFA at a centralized FAA facility, and supports simple integration of a backup data acquisition and processing site. Because of its service oriented architecture and scalable component design, the majority of constituents of the WTMx application architecture are compatible with future deployment to a highly redundant cloud system, with the exception of the Wind-ETL constituent which is hampered by the large data flow sizes.

GLOSSARY

ACARS	Aircraft Communications Addressing and Reporting System
ADS-B	Automatic Dependence Surveillance-Broadcast
AGL	above ground level
API	application programming interface
ARINC	Aeronautical Radio Inc.
ASOS	Automated Surface Observing System
bash	Bourne-Again Shell
BPM	Business Process Modeling
BPR	Business Process Rule
CONOPS	concept of operations
CONUS	continental United States
CPU	central processing unit
CSPR	closely spaced parallel runways
CSS	cascading style sheet
DoD	Department of Defense
DST	Decision Support Tool
EIP	Enterprise Integration Patterns
ER	en route operations
ESB	Enterprise Service Bus
ESRL	Earth System Research Laboratory
ETL	extract-transform-load
FAA	Federal Aviation Administration
FTP	File Transfer Protocol
GB	Gigabytes
GHz	Gigahertz
GUI	graphical user interface
hPa	HectoPascals
HQL	Hive Query Language
HRRR	High Resolution Rapid Refresh
HTML	HyperText Markup Language
IMC	Instrument Meteorological Conditions
IT	Information Technology
JVM	Java Virtual Machine
LIDAR	light detection and ranging
MB	Megabytes
MIT LL	MIT Lincoln Laboratory
N	natural levels

NCDC	National Climatic Data Center
NCEP	National Centers for Environmental Prediction
NextGen	Next Generation Air Transportation System
NOAA	National Ocean and Atmospheric Administration
NWS	National Weather Service
P	pressure levels
PA	Paired Approaches
RAM	random access memory
RAP	Rapid Refresh Model
RUC	Rapid Update Cycle
SMTP	Simple Mail Transfer Protocol
SOA	service-oriented architecture
SWIM	System Wide Information Management
TB	Terabytes
TPD	Time Paired Departures
VAD	Velocity Azimuth Display
VMC	Visual Meteorological Conditions
WFA	Wind Forecast Algorithm
WSDL	Web Services Description Language
WTAA	Wake Turbulence Application Architecture
WTMA	Wake Turbulence Mitigation for Arrivals
WTMD	Wake Turbulence Mitigation for Departures
WTM-SR	Wake Turbulence Mitigation on Single Runways
WTMx	Wake Turbulence Mitigation, generic system reference
XML	Extensible Markup Language
XSD	XML Schema Definition

REFERENCES

Tittsworth, Jeffery A., Steven R. Lang, Edward J. Johnson, and Stephen Barnes, 2012: “Federal Aviation Administration Wake Turbulence Program – Recent Highlights,” 57th Air Traffic Control Association (ATCA) Annual Conference & Exposition, October 1–3, Fort Washington, MD.

This page intentionally left blank.