Aviation Environmental Design Tool (AEDT)

System Architecture

Doc #AEDT-AD-01

1/29/2007

Prepared by:

Christopher Roof, Andrew Hansen, Gregg Fleming
USDOT Volpe Center

Ted Thrasher, Alex Nguyen, Cliff Hall
CSSI, Inc.

Eric Dinges, Raymond Bea
ATAC, Inc.

Fabio Grandi, Brian Kim, Scott Usdrowski
Wyle Laboratories, Inc.

Peter Hollingsworth
Georgia Tech.

Prepared for:

Federal Aviation Administration

Office of Environment and Energy

Table of Contents

List of Figures

List of Tables

1 Introduction

The Federal Aviation Administration's Office of Environment and Energy (FAA-AEE) is developing a comprehensive suite of software tools that will allow for thorough assessment of the environmental effects of aviation. The main goal of the effort is to develop a new capability to assess the interdependencies between aviation-related noise and emissions effects, and to provide comprehensive impact and cost and benefit analyses of aviation environmental policy options. The building block of this suite of software tools that integrates existing noise and emissions models is the Aviation Environmental Design Tool (AEDT). AEDT will provide a framework for consistent modelling and assessment of aviation environmental effects by Merging of existing tools and new modules into both a publicly available, regulatory/planning component (Local) and the policy component of AEDT (Global).

The central building blocks used for the AEDT system are four existing FAA noise and emissions modeling applications: (1) Integrated Noise Model (INM) – local noise; (2) Emissions and Dispersion Modeling System (EDMS) – local emissions; (3) Model for Assessing Global Exposure to the Noise of Transport Aircraft (MAGENTA) – global noise; and (4) System for assessing Aviation's Global Emissions (SAGE) – global emissions. This core of AEE developed applications contains the software implementations of best-practice environmental modeling and assessment techniques for aviation. Each application and its development history are summarized below.

1.1 Objective

While the four core software applications noted above implement the best-practice techniques for the respective local and global, noise and emissions models for aviation, they do so in a disjointed and, in some cases, inconsistent way; this is due to their unique historic timelines and factors that previously drove development. A prime objective in the AEDT architecture design is to construct a framework for the common components of these applications to provide coupled, and thereby consistent, analysis of the physical and logical processes being modeled. The result is a new capability for assessment and/or projection that ties noise and emissions effects together.

Additional objectives include improving user access and control, reducing application maintenance and distribution effort by the provider, and increasing the adoption and use of AEE models by aviation stakeholders worldwide.

Given this base of objectives for the system architecture, a requirements collection phase was undertaken to gather input from the wide array of AEDT stakeholders. The first step in identifying associated requirements was a series of stakeholder workshops hosted by the National Academies of Science (NAS) Transportation Research Board (TRB) [1]. Based on this input, further definition and refinement of AEDT requirements was undertaken by the development team (see Section 5.1 for a detailed description of the team and individual

[1] *http://www8.nationalacademies.org/cp/projectview.aspx?key=TAXX-P-03-05-A*

responsibilities). These requirements were consolidated into a single, living document [AEDT Software Requirements Document, Doc #AEDT-REQ-01, 1/25/2007] to provide the development team with a set of working specifications for the software implementation and tool integration. The requirements are further dealt with below (functional) and in more detail through the requirements document.

1.2 Background

The legacy environmental models upon which AEDT is built all have unique, historical development timelines, uses and motivations for existence.

1.2.1 Legacy Model History and Timelines

INM, the local or terminal area noise analysis model, has the longest history of the legacy models. First publicly released in 1978, the model has been utilized in support of Federal Aviation Regulations (FAR) Part 150 (Airport Noise Compatibility Planning), Part 161 (Notice and Approval of Airport Noise and Access Restrictions) and National Environmental Policy Act of 1969 (NEPA) analysis. Additionally, the model has been used extensively in national parks in support of the Grand Canyon Overflights Act and the Air Tour Management Act of 2000. A companion model of INM, the Heliport Noise Model (HNM), which was specifically designed to model rotorcraft operations, has been integrated into INM over the last several years. INM predicts noise levels for fixed and rotorcraft aviation sources.

EDMS, the local emissions and dispersion analysis model, was first publicly released in 1991. The model has been used in support of US Clean Air Act, National NEPA and State Implementation Plan (SIP) development analyses. EDMS has also been integral to global analyses of local air quality issues in support of the International Civil Aviation Administration’s (ICAO) Committee on Aviation Environmental Protection (CAEP). EDMS predicts emissions, as well as facilitates dispersion analyses, for fixed-wing aircraft as well as non-aviation emission sources around airports.

MAGENTA, the global noise model, has been used for analysis in support of ICAO CAEP as well as US domestic initiatives. First introduced in 1996 and the standard CAEP noise model since CAEP5 in 2000, MAGENTA predicts population exposed to a range of day-night average sound levels (DNL) for various policy scenarios for fixed wing aircraft.

SAGE, the global emissions model, has been used by FAA to develop global fuel burn and emissions inventories for the years 2004 through 2005, as well as specific data analyses that take advantage of a global gate-to-gate flight model. These inventories have been used to support ICAO CAEP as well as the United Nations Framework Convention on Climate Change (UNFCCC), and the Intergovernmental Panel on Climate Change (IPCC). Research efforts, such as the analysis of the impact of Reduced Vertical Separation Minimum (RVSM), have been undertaken by implementing the capability to change fleet-wide fuel burn values relative to specific criteria in SAGE. SAGE predicts global emissions for fixed-wing aircraft.

Figure 1 presents an overview of the individual legacy model development timelines. The figure illustrates the varying introduction dates (as early as 1978 and recent as 2001) of the models, as well as shows the staged merging of the tools into a single, integrated model.

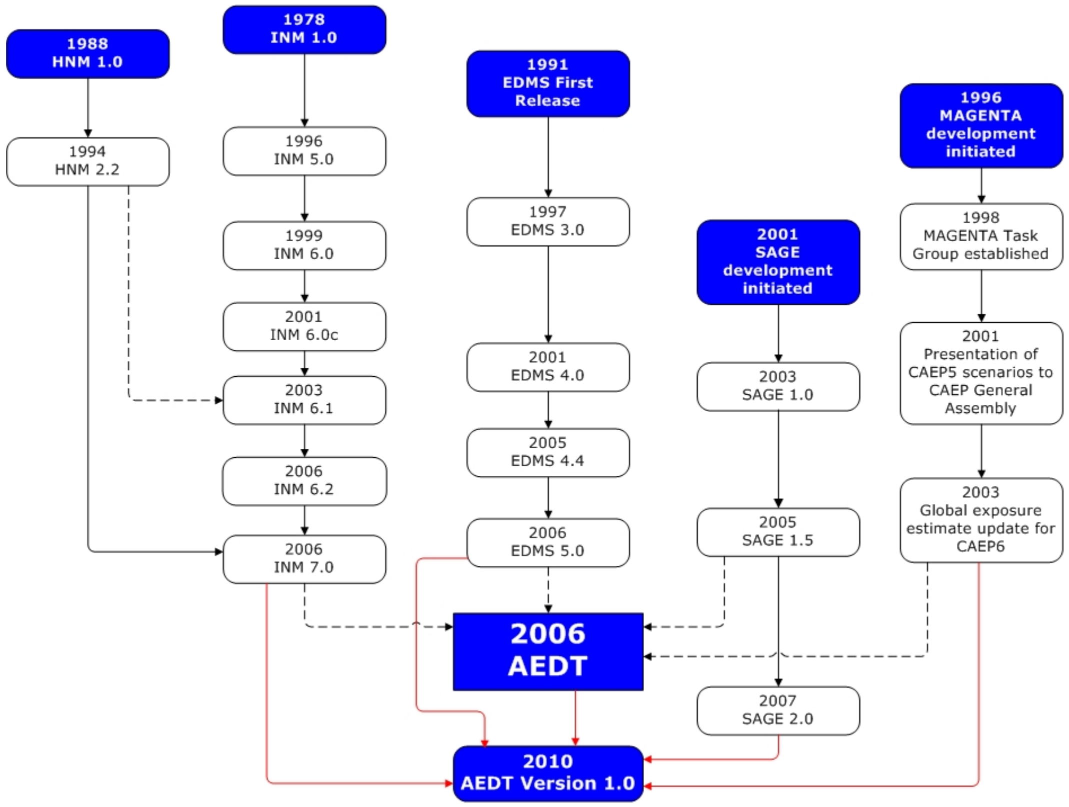


Figure 1. Development history of FAA AEE aviation environmental modeling and assessment applications

2 Development Environment Specifications

The requirement that AEDT be a single, integrated analysis tool leads directly to specifications for software development. The Microsoft integrated development environment, Visual Studio (MSVS) is used for AEDT development because it provides capabilities matching the scalability, flexibility, and efficiency required by AEDT. If necessary, other standards such as the ANSI C++ standard, can be maintained within the MSVS environment on a per module basis. The MSVS environment has direct benefit to three of the design goals noted above:

Scalability: The current MSVS product (MSVS .NET) enables developers to implement software that operates in a distributed fashion, either on a single computer or a network of computers. The main support provided for this approach to scalability is found in the tools for communication between software implementations. The .NET approach is also amenable to interfacing with applications on non-Windows platforms.

Flexibility: For both new and legacy implementations, MSVS offers the development team flexibility in choice of software language and level of technology adoption. For example, an existing application can be encapsulated within a communication layer and remain relatively untouched or completely broken down and re-implemented with state-of-the-art technology.

Efficiency: The MSVS integrated development environment includes explicit tools (version control, configuration management, software requirements and specifications, software life-cycle tracking, integration with project management tools, automated diagramming of legacy projects, extensive software support for database access, and a mature graphical user interface (GUI) library that can be shared across the development team but utilized though one interface. MSVS is also well suited to leverage prior development as the greater portion of existing software implementations already reside in the current or prior MSVS versions (i.e., C++).

2.1 Microsoft .NET Environment

Following is a brief description of the Microsoft.NET environment. It is presented with a specific focus on AEDT, namely the advantages .NET offers within the context of AEDT development. Specially, .NET 2005 has been selected as the development environment to be used by all AEDT developers beginning in January 2007.[2]

.NET is a framework for distributing application workload. It utilizes operating system independent messaging, called eXtensible Markup Language (XML) to exchange data between software modules. Among other benefits of XML, legacy applications can be integrated together relatively quickly by implementing a simple software layer that encodes, or wraps, the application's input and output data into XML format. In the case of applications that are being

[2] There are numerous resources on the web for providing additional background on .NET: http://www.microsoft.com/Net/Default.aspx, http://support.microsoft.com/ph/8291, http://support.microsoft.com/ph/8940. While much of the description in the above links centers on pure web applications, the concepts apply equally well to conventional software efforts, including AEDT .

developed from scratch or already reside on Microsoft Windows platforms, the .NET framework provides the XML software layer so no code development is needed for communication.

Within the .NET environment, software applications benefit from:

- Scalable solutions through distributed computational workload;
- XML messaging protocol for platform independent data interface;
- independence of developer language preference;
- large library of foundation classes;
- industry standard network communications protocols; and
- efficient memory caching of stored data.

Microsoft Visual Studio (MSVS) .NET 2005 is the integrated development environment that supports .NET software development. MSVS .NET maintains backward compatibility with software developed either on earlier versions of MSVS or under ANSI C/C++ standards. From a developer's point of view, MSVS .NET provides access to improved technology for system implementation.

Other features of .NET 2005 important for AEDT include:

Efficient caching: Caching is the storage of information that will be reused in a memory location for faster access in the future. .NET allows the caching of data from a database so that Internet/Intranet access to large databases is more efficient. This may provide advantages within the context of SAGE and MAGENTA in particular. It may also allow FAA/AEE and other users to more easily access the large databases associated with AEDT-Global.

Memory leak and crash protection: .NET automatically recovers from memory leaks and errors through the use of what Microsoft terms Managed Code. Moreover, there are substantial security benefits associated with managed code, which may be beneficial to AEDT, should it be required to make the large databases of SAGE and MAGENTA (in particular) available over either the Internet or an Intranet.

Backward Compatibility: While not always seamless, MSVS .NET and MSVS 2005 automatically support all software development projects developed in earlier MSVS releases. Developers can take these prior implementations wholesale into the .NET and 2005 integrated development environments so that they are immediately useable, as is. This is especially important for AEDT, given the significant amount of legacy code throughout the existing applications. Over time, legacy code can be re-engineered as desirable with the latest technology.

Beyond MSVS .NET's access to new implementation technologies, it also has two key benefits as an AEDT development platform: (1) Cross-language support for developer convenience; and (2) Low integration/port cost to reuse computational components of legacy AEE applications (SAGE, INM, MAGENTA, EDMS).

There are other significant benefits in moving the AEDT team collectively to the MSVS .NET integrated development environment[3], particularly with as many as eight organizations involved in implementation. In one package MSVS provides capability for:

- Cross-language support (Basic, C, C++, C#, Java, J#, and other lesser languages);
- integrated source code version control and configuration management;
- software life-cycle tracking (test, alpha/beta release, bug fixes, etc.);
- Potential automated diagramming of C# modules;
- project management interface to MS Project (MSVS 2005 Team System);
- extensive software support for database access, e.g. Structured Query Language (SQL) querying;
- mature library for graphical user interface implementation.

The specific items called out in the capabilities list above are described in more detail below.

Multiple language support: With .NET, programmers can write code in more than 25 .NET languages. This allows programmers to develop in the language they know best. It also means improved maintainability for AEDT, as it will be easier to assimilate new programmers, as needed, given more flexibility with regard to language. .NET programming languages are translated into what is referred to as a Common Language Runtime (CLR) during compiling. This means that the same data types and calling conventions are used for every language - all languages are equal. It is expected that wherever possible AEDT code will be written in C# under the "managed code" infrastructure; it has been agreed that the Database Access and Taskmaster modules will be written in C#, whereas the Local GUI development team is currently considering the use of C# versus C++.

Version control and configuration management: Given the broad collection of developers and organizations contributing to building AEDT or using AEDT within other models such as APMT, the need for version control on both software implementation and description documents is clear. Subversion, which can be invoked directly from the MSVS .NET development environment, has been utilized in prior efforts by the AEDT development team. The Subversion system uses an open interface that can act independently or with client interfaces such as Tortoise. The Subversion system also handles data sets of binary or specially formatted files such as data, documents, or intermediate sources. In order to manage the anticipated continual AEDT development process, which will be undertaken somewhat independently by individual development groups for the separate modules, version control will also manifest itself in the individual module outputs.

Life-cycle tracking: The AEDT software life cycle before and after implementation must be considered, particularly including the large install base of INM and EDMS users to be supported by the AEDT application. Design and implementation are only a small part of the effort to build and maintain the system. Testing, validation, distribution and tech support/bug fixing are also important factors when considering the total effort. MSVS .NET has integrated tools to link

[3] A high level overview of the MSVS .NET integrated development environment can be found at: http://msdn.microsoft.com/vstudio/ and http://msdn2.microsoft.com/en-us/teamsystem/default.aspx

design, implementation, and validation of software, and to streamline the maintenance phase of software releases.

Interface to MS project management tools: If desired, the Team System version of MSVS 2005 can link directly with both SharePoint (the software used for the FAA's Knowledge Services Network - KSN) and MS Project applications to share data and more tightly bind the implementation and management of the software development cycle. Specifically, documents and source code implementation can be linked directly to project management plans through the SharePoint document interface so that the management and implementation of the project are more tightly coordinated.[4]

Software support for database access: The MSVS NET environment provides a library of highly optimized database access functions, which can be invoked directly by the developer for use in the software implementation. The .NET library covers a broad range of database technologies, from the latest SQL solutions to conventional (flat) database formats.

Graphical User Interface Library: The majority of AEDT Local users will be using a Microsoft Windows interface based on WinForms. The MSVS .NET environment provides the mature and streamlined WinForms library for quickly implementing GUIs, both simple and complex. Likewise, the Windows GUI has a wide array of third party, purpose built modules supporting everything from GIS and mapping to three-dimensional rendering to graphical database access/display. However, there is a programmatic/means restriction against integrating third party tools with end-user license fees as requisite AEDT functionality. If licensed software is incorporated it must either provide only optional functionality for the end-user or have free end-user redistribution licensing structure.

For AEDT Global development, another GUI consideration is the use of WebForms based interface. This approach employs a Service Oriented Architecture (SOA) for exposing functionality to the end user through a web browser interface, rather than a self-contained application.

Microsoft ASP.NET (Active Server Page) technology is one example of SOA infrastructure. The AEDT architecture design will utilize this technology both for its development efficiency as well as the compatibility with software modules (DLLs) implemented to support the WinForms application instance. The benefit here is a reduced development effort in initial implementation of AEDT functionality and avoidance of redundant maintenance effort when modifications and enhancements are implemented. The WinForms and WebForms application designs can utilize, for the most part, the same set of functional modules. The integrated .NET environment is well suited for this dual approach.

[4] A high-level description of the capability to link to project management tools is available at: http://msdn2.microsoft.com/en-us/library/aa302181.aspx

2.2 Architecture Modeling Tool

Conceptual schematics and diagrams are an important part of documenting and conveying system architecture information. There is a wide range of software modeling tools for designing and implementing software modules, e.g. Rational Rose/XDE, Rational Software Modeler, MS Visio, and MS Visual Studio Class Designer. A significant benefit beyond the basic software source code implementation is the extensive list of facilities for formally documenting the software. Specifically these facilities are used for preparing module Interface Control Documents and Algorithm Description Documents (ADD).

The MS Class Designer is an integrated tool with the Visual Studio integrated development environment. It provides the user with the ability to automatically generate C# and VB instances of source code. Further, this mechanism is bi-directional so that implementations of C# and VB can be automatically represented graphically in the software model.

Class Designer does have drawbacks when compared with other applications, for instance, it does not export Unified Modeling Language (UML) formatted documents (easy exchange with other development tools), it does not currently support C++ or Java, and its export format is Microsoft proprietary. The compromise in the decision to use Class Designer was based primarily on the deficiencies of the other applications when interfacing with Visual Studio .NET 2005. The other applications did not support any wider language base nor did they integrate as well with Visual Studio (separate applications). The end result is that Class Designer is used much like the integrated editor or compiler within Visual Studio.

3 System Description

3.1 Module Design Concepts

To focus AEDT system design, a set of high-level design paradigms for the project and its implementation have been identified as follows:

- **Modularity**—isolate functionality to minimize work when addressing requirement changes, h/w modernization, etc.
- **Scalability**—decompose units of work so that a plurality of h/w resources can be applied
- **Flexibility**—accommodate a variety of algorithms for accomplishing the same goal
- **Usability**—present streamlined interface for novice users and deep or intricate details for experts
- **Transparency**—expose pertinent methods, data, inputs, and assumptions for any given output
- **Performance**—meet computational demands of the current user base, anticipate bottlenecks and develop implementations for efficiency and optimization

3.2 Functional Requirements Overview

The AEDT Requirements Document outlines the AEDT requirements identified to date. Requirements are separated by global/local- and noise/emissions-derived inputs and outputs, as well as those common to two or more of the legacy model delineations. All requirements are notated as either critical or support in nature. Additionally, those existing requirements which may not be appropriate in the future are annotated for potential, future phase-out.

3.3 Unit of Work

Recalling the primary objective in the AEDT architectural design (a common framework for noise and emissions analysis), the starting concept for AEDT architectural design is the identification of a minimum operation that can be configured for an environmental analysis. The objective here is to find a common thread across both the noise-emissions dimension and the local-global dimension. This common thread provides the context for both distinguishing modularity and removing redundancy in the module breakdown of the system architecture. The former is the entry point for addressing the design goals listed in Section 1 and the latter reduces the effort expended to satisfy the combination of noise, emissions, local, and global requirements.

With regard to the implementation, this “minimum operation” implies the basic unit of work for the system. To conduct the simplest analysis, the system must process a minimum operation or one unit of work. The minimum operation is defined to be one gate-to-gate flight path. More complex or wider analysis is then iteration over multiple units of work. In cases where state information on only a portion of the entire path, say a local noise or air quality assessment, the context of a gate-to-gate trajectory is still pertinent as the aircraft conditions, e.g. take-off weight, and approach and departure procedures are dependent on the origin/destination (O/D) of the flight. However, in this case the calculation of the trajectory over the entire flight is foreshortened to the region of interest.

3.4 System Databases

The AEDT system will rely on four core input databases for source information. These are itemized below along with the output (results) database structure. These databases will, if not already, be implemented in a relational database management system (RDBMS). Each database has a database description document (DDD) which specifies the structure of the database, the units and fields stored within it, as well as the procedure for maintaining and/or updating the content in the database.

Table 1. AEDT Databases

Database (Lead)	Doc #	Purpose
Airports (AMalwitz)	AEDT-DDD-01-01	Collection of airport specific data including runway plan, navaids, local historical weather.
Fleet (CHall)	AEDT-DDD-02-01	Annual itemization of world-wide aircraft in service including performance, noise, and emissions parameters.
Movements/Operations (SUsdrowski)	AEDT-DDD-03-01	Itemization of flights including schedule or trajectory information for use in assessment or analysis.
Aircraft Retirement/Replacement (FGrandi)	AEDT-DDD-04-01	Update schedules for building future fleets in forward looking analyses.
Results Inventory (SBalasubramanian)	AEDT-DDD-05-01	Annual emissions and noise assessment inventories.

3.5 System Modules

The modules currently identified for AEDT development, including the individual module leads, are as follows:

Table 2. AEDT Modules

Module (Lead)	Doc #	Purpose
Aircraft Performance (EDinges)	AEDT-ICD-01-01	Calculate gate-to-gate aircraft performance.
Aircraft Emissions (AMalwitz)	AEDT-ICD-02-01	Calculate aircraft emissions.
Aircraft Acoustics (EBoeker)	AEDT-ICD-03-01	Calculate aircraft noise.
Local GUI (RBea, ANguyen)	AEDT-ICD-04-01	Graphical user interface for running local analyses.

Module (Lead)	Doc #	Purpose
Database Access (SBalasubramanian)	AEDT-ICD-05-01	Input/output all data for all AEDT modules.
Fleet Operations (FGrandi)	AEDT-ICD-06-01	Calculate future year fleet and operations utilizing FESG and other baseline forecasts.
Taskmaster (RBea)	AEDT-ICD-07-01	Coordinate interface/processes for all communications between AEDT modules.
GIS (TLedoux)	AEDT-ICD-08-01	Performs all graphical illustration and manipulation of data.
Fuel-Burn (AMalwitz)	AEDT-ICD-09-01	Calculates aircraft fuel burn.
Interpolation (JRuggerio)	AEDT-ICD-10-01	Undertakes all required interpolation and extrapolation.
Terrain (PGerbi)	AEDT-ICD-11-01	Undertakes all terrain-related calculations for noise and other modules.
Global GUI (AMalwitz)	AEDT-ICD-12-01	Graphical user interface for running and querying global analyses.
Data Validation (AHansen)	AEDT-ICD-13-01	Validates input/output data from AEDT and 3rd party modules.
Ground Track Dispersion (SBalasubramanian)	AEDT-ICD-14-01	Disperses enroute OAG flight tracks per statistical analysis of historic radar data.
Weather Module (CHall)	AEDT-ICD-15-01	Processes weather for aircraft performance and local dispersion modeling.
Radar Flight Profile Module (EDinges)	AEDT-ICD-16-01	Calculates AEDT-compatible trajectories (including thrust) from aircraft position data, currently relies on radar data.

3.6 Overall System Architecture

The overall system architecture is depicted in Figure 2 below. In addition to the system databases (System Data and Study Data) and modules highlighted above, AEDT will also make use of third-party components. Some third-party software will be used as mandated by EPA and FAA guidance and regulations; other software will be used as deemed most appropriate from a development standpoint.

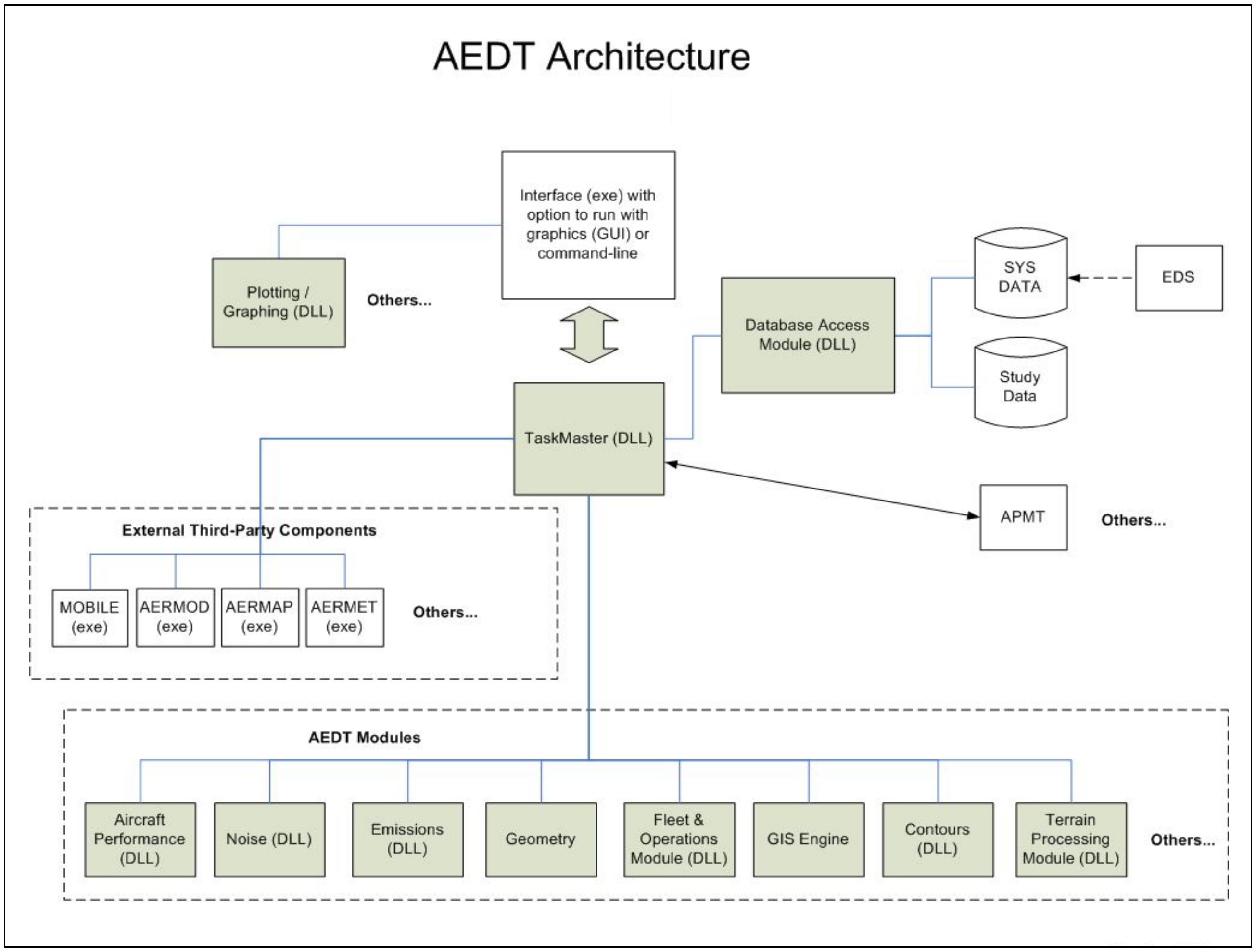

AEDT Architecture
Interface (exe) with option to run with graphics (GUI) or command-line
Plotting / Graphing (DLL)
Others...
Database Access Module (DLL)
SYS DATA
EDS
Study Data
TaskMaster (DLL)
APMT
Others...
External Third-Party Components
MOBILE (exe)
AERMOD (exe)
AERMAP (exe)
AERMET (exe)
Others...
AEDT Modules
Aircraft Performance (DLL)
Noise (DLL)
Emissions (DLL)
Geometry
Fleet & Operations Module (DLL)
GIS Engine
Contours (DLL)
Terrain Processing Module (DLL)
Others...

Figure 2. Overall AEDT Architecture

As can be seen in Figure 2, the Database Access (DAM) and Taskmaster modules are integral to all AEDT functionality. The DAM will be used to access and manipulate all system and user data required for use in AEDT. The Taskmaster will be the single control module which will manage all AEDT processes. Together the DAM and Taskmaster will form the core functionality of the tool. They also ultimately will provide significant support to the Aircraft environmental Portfolio Management Tool (APMT), providing consistency between AEDT and APMT through the use of harmonized processes. The overall system architecture will continuously be revisited as the DAN and Taskmaster development efforts progress further.

3.7 System Graphical User Interface Development

The overall system architecture is common to the entire tool set, including both local and global applications. Due, however, to the significantly different functionality and user bases of the local and global applications, the decision was made to maintain separate GUIs for the two portions of the tool.

3.7.1 Local Graphical User Interface Development

The Local GUI team is tasked with the harmonization and merging of the local tools. This consists of the merging of existing capabilities contained within INM and EDMS, as well as the underlying workflows and associated data structures. Additionally, due to the significant user bases for the legacy tools, consideration will be made for maintaining consistency with the existing features in the legacy tools, where possible. A single local interface will eliminate redundancy inherent in the use of two separate GUIs and tools, the obvious example being the need to only generate a single set of aircraft performance characteristics for both noise and emissions modeling.

Figure 3 below shows examples of the INM and EDMS GUIs, illustrating their merging into a single, AEDT Local GUI. Significant development progress will be realized on the Local GUI after the releases of INM Version 7.0 and EDMS Version 5.0. The formation of a single, combined noise and emissions Design Review Group (DRG) will also aide the development effort. A single, integrated Local GUI is scheduled to be released in 2010.

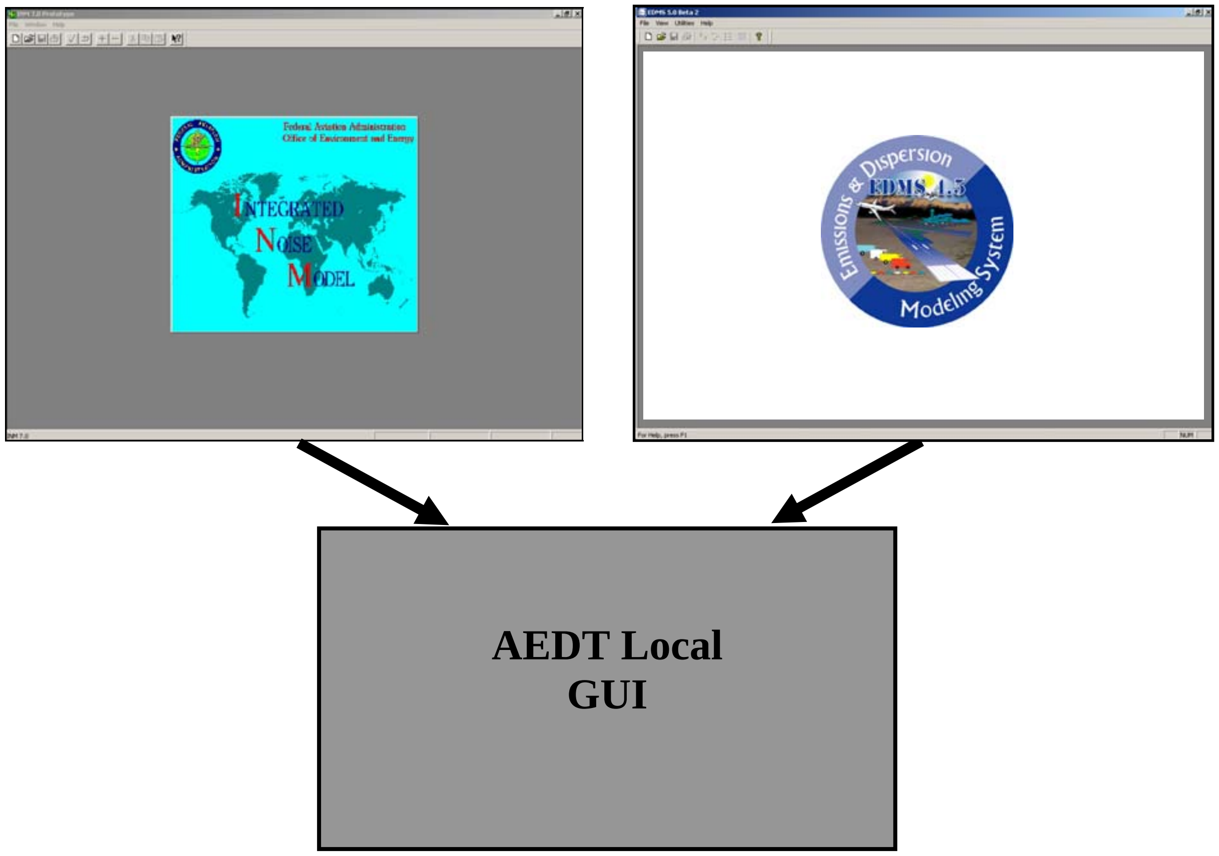

Federal Aviation Administration
Office of Environment and Energy
Integrated Noise Model
Emissions & Dispersion Modeling System
EDMS 4.5
AEDT Local GUI

Figure 3. AEDT Local GUI

Parallel to development of a single, local GUI, the legacy INM and EDMS modules and databases are being harmonized and integrated in a step-wise manner. This (1) provides for a manageable coding integration process, and (2) in parallel to the combined DRG effort highlighted above, minimizes the potential to disturb the large user base of the legacy models. Figures 4 through 7 highlight the step-wise integration of the local modules and databases.

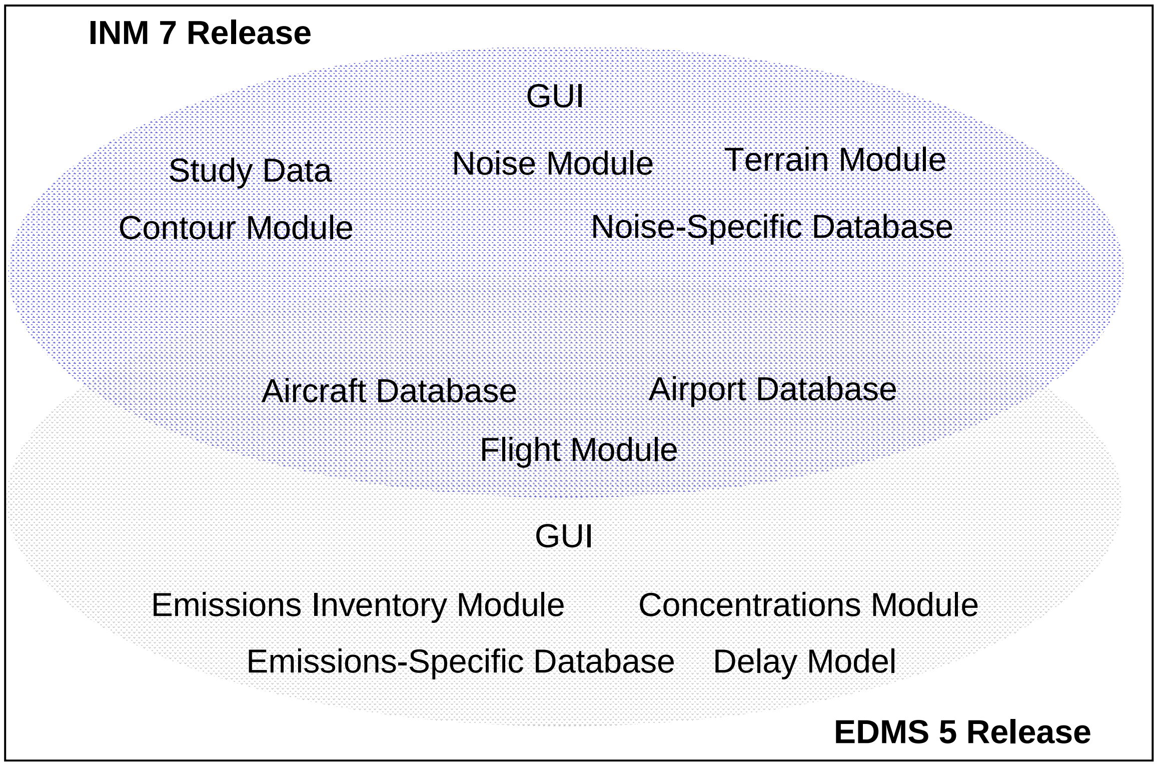


Figure 4. AEDT Version 0.0

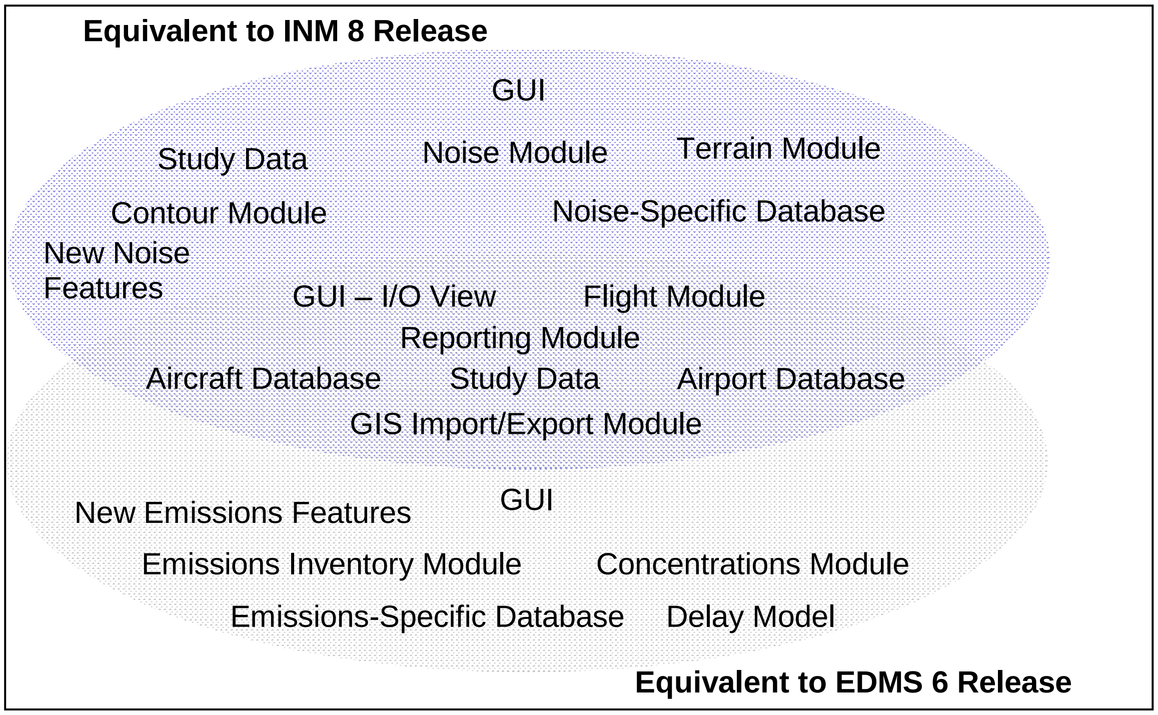


Figure 5. AEDT Version 1.0

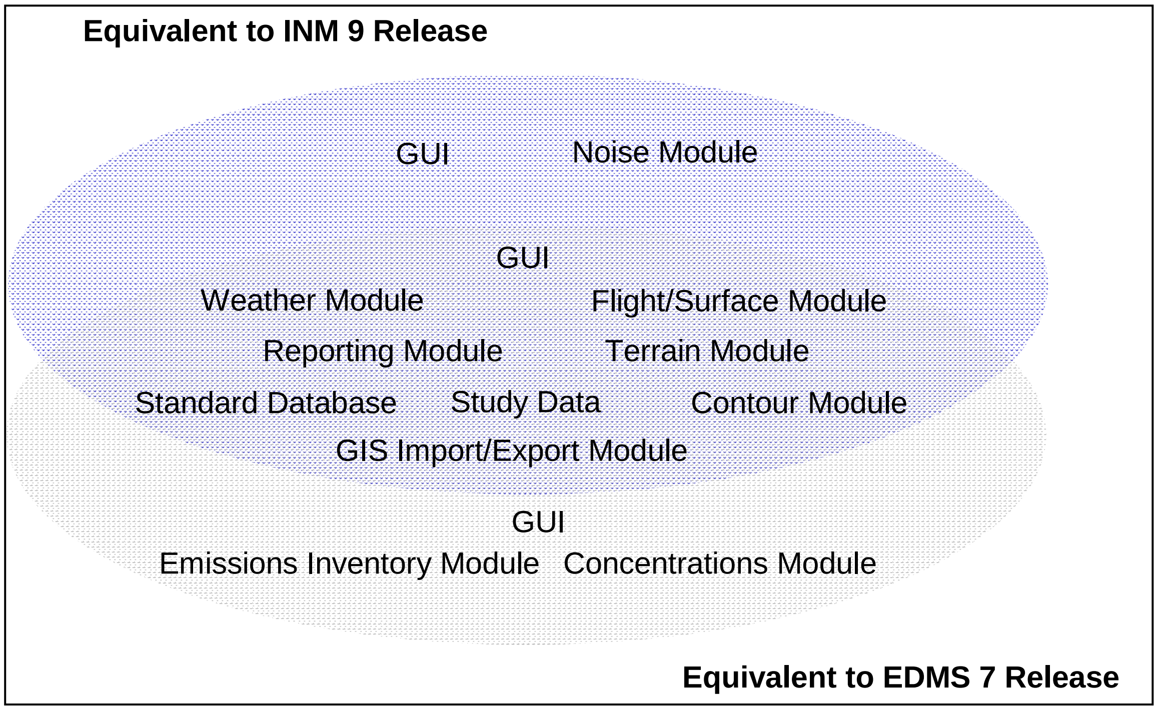


Figure 6. AEDT Version 1.2

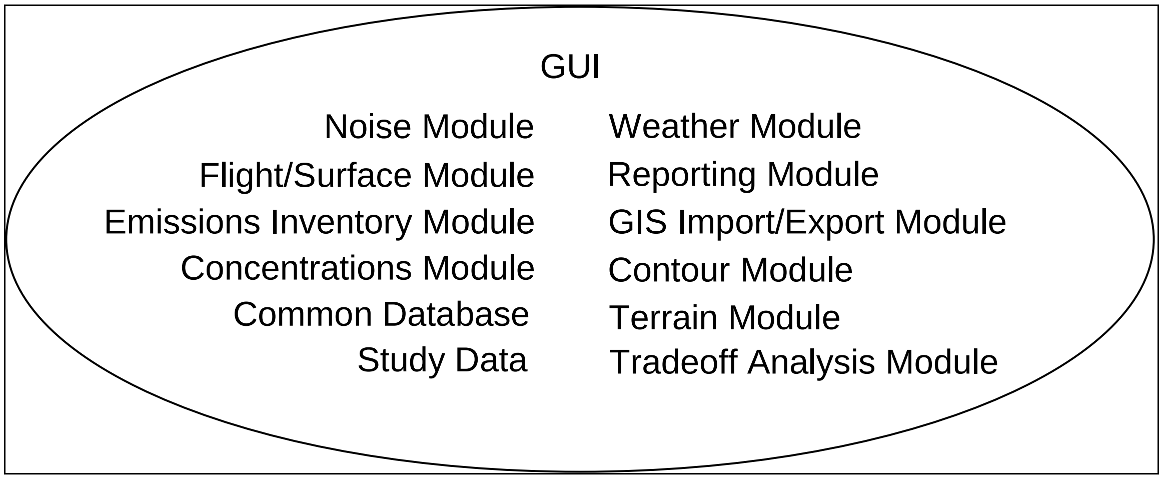


Figure 7. AEDT Version 2.0

3.7.2 Global Graphical User Interface Development

The MAGENTA-SAGE (MASAGE) integration team is tasked with the harmonization and merging of the global tools into a single, global tool. Unlike the local tool development, while the MAGENTA and SAGE systems are relatively mature, they have not historically been fielded beyond the development team. While this means that more actual development may need to be undertaken, it also allows for a development process from scratch to support the global needs.

As outlined in Section 2.1, integral to the .NET environment is the ASP.NET web service. Use of this technology allows for the running of actual AEDT modules (DLLs) remotely via web applications. Figure 8 below highlights a potential use of ASP.NET.

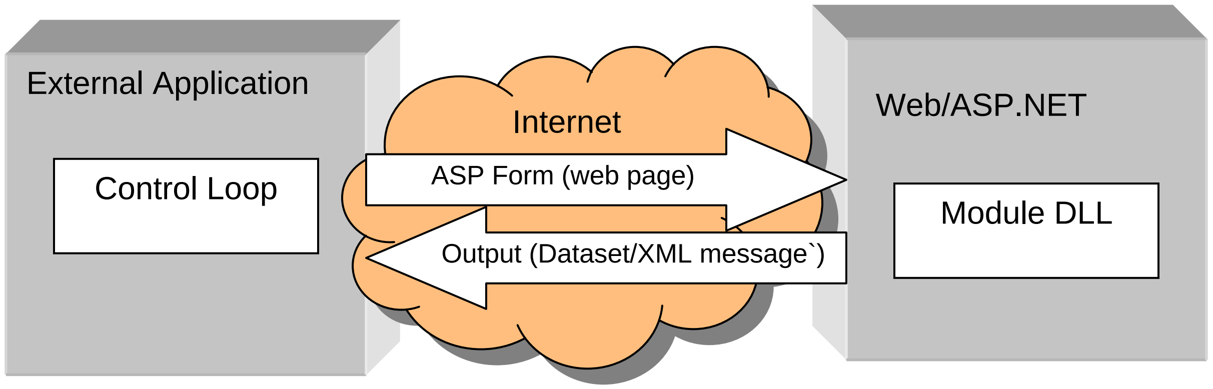


Figure 8. ASP.NET Example

A prototype AEDT Global Inventory Analyzer tool has been developed using the ASP.NET technology. This tool allows for the querying of historic AEDT global emissions inventories. Various means for querying the archival data exist, including: (1) high level, aggregate inventory analysis (i.e., summaries making use of a partial or entire year's inventory); (2) analysis of single or multiple airport operations; (3) custom queries, which may target specific airframes and/or engines, or may involve specific subsets of airports, or specific dates for analysis; and (4) manual SQL queries, which allow expert relational database users to undertake highly specialized queries.

Figure 9 below presents the initial screen of the beta AEDT Global Inventory Analyzer application.

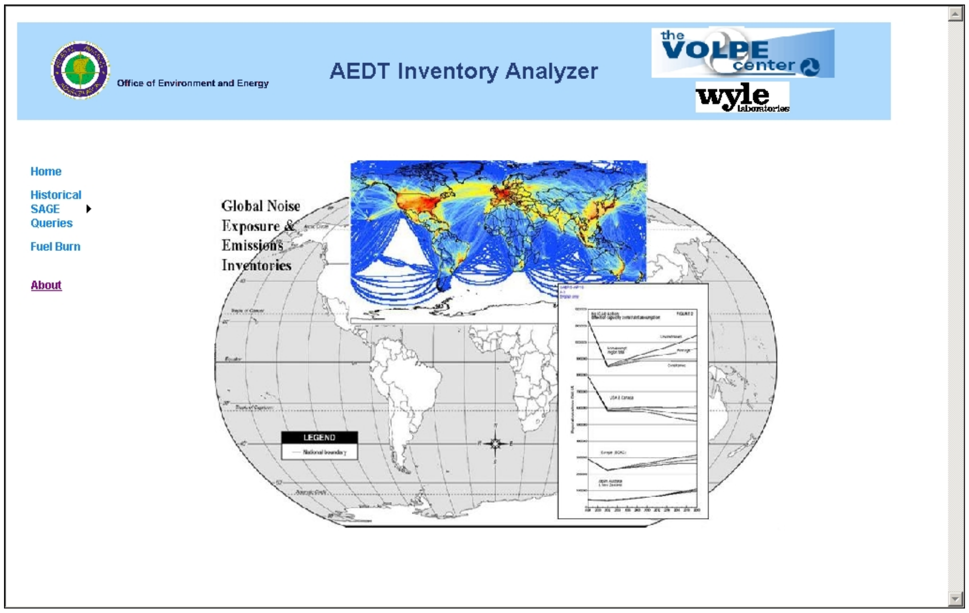
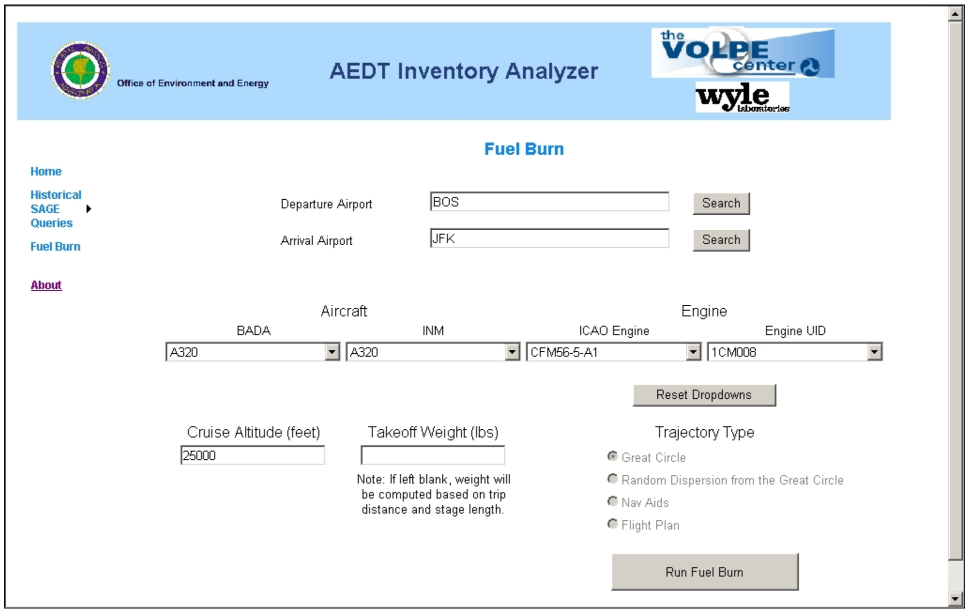


Figure 9. Exemplar AEDT Global Inventory Analyzer Interface

Fuel burn computations may be initiated in the application to calculate airframe/engine-specific fuel burn, given a specific O/D pair, as well as cruise altitude and aircraft takeoff weight. Figures 10 and 11, respectively, present the fuel burn input windows and example output results.



AEDT Inventory Analyzer

Fuel Burn

Departure Airport: BOS

Arrival Airport: JFK

Aircraft		Engine	
BADA	INM	ICAO Engine	Engine UID
A320	A320	CFM56-5-A1	1CM008

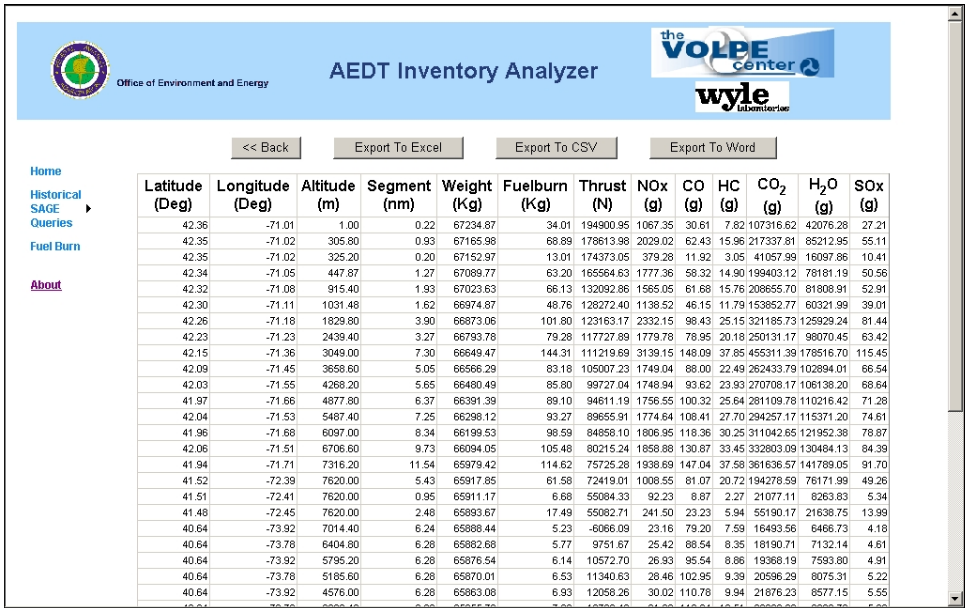
Cruise Altitude (feet): 25000

Takeoff Weight (lbs):

Note: If left blank, weight will be computed based on trip distance and stage length.

Trajectory Type
- ● Great Circle
- ○ Random Dispersion from the Great Circle
- ○ Nav Aids
- ○ Flight Plan

Figure 10. AEDT Inventory Analyzer, Fuel Burn Capability



AEDT Inventory Analyzer

Latitude (Deg)	Longitude (Deg)	Altitude (m)	Segment (nm)	Weight (Kg)	Fuelburn (Kg)	Thrust (N)	NOx (g)	CO (g)	HC (g)	CO_2 (g)	H_2O (g)	SOx (g)
42.36	-71.01	1.00	0.22	67234.87	34.01	194900.95	1067.35	30.61	7.82	107316.62	42076.28	27.21
42.35	-71.02	305.80	0.93	67165.98	68.89	178613.98	2029.02	62.43	15.96	217337.81	85212.95	55.11
42.35	-71.02	325.20	0.20	67152.97	13.01	174373.05	379.28	11.92	3.05	41057.99	16097.86	10.41
42.34	-71.05	447.87	1.27	67089.77	63.20	165564.63	1777.36	58.32	14.90	199403.12	78181.19	50.56
42.32	-71.08	915.40	1.93	67023.63	66.13	132092.86	1565.05	61.68	15.76	208655.70	81808.91	52.91
42.30	-71.11	1031.48	1.62	66974.87	48.76	128272.40	1138.52	46.15	11.79	153852.77	60321.99	39.01
42.26	-71.18	1829.80	3.90	66873.06	101.80	123163.17	2332.15	98.43	25.15	321185.73	125929.24	81.44
42.23	-71.23	2439.40	3.27	66793.78	79.28	117727.89	1779.78	78.95	20.18	250131.17	98070.45	63.42
42.15	-71.36	3049.00	7.30	66649.47	144.31	111219.69	3139.15	148.09	37.85	455311.39	178516.70	115.45
42.09	-71.45	3658.60	5.05	66566.29	83.18	105007.23	1749.04	88.00	22.49	262433.79	102894.01	66.54
42.03	-71.55	4268.20	5.65	66480.49	85.80	99727.04	1748.94	93.62	23.93	270708.17	106138.20	68.64
41.97	-71.66	4877.80	6.37	66391.39	89.10	94611.19	1756.55	100.32	25.64	281109.78	110216.42	71.28
42.04	-71.53	5487.40	7.25	66298.12	93.27	89655.91	1774.64	108.41	27.70	294257.17	115371.20	74.61
41.96	-71.68	6097.00	8.34	66199.53	98.59	84858.10	1806.95	118.36	30.25	311042.65	121952.38	78.87
42.06	-71.51	6706.60	9.73	66094.05	105.48	80215.24	1858.88	130.87	33.45	332803.09	130484.13	84.39
41.94	-71.71	7316.20	11.54	65979.42	114.62	75725.28	1938.69	147.04	37.58	361636.57	141789.05	91.70
41.52	-72.39	7620.00	5.43	65917.85	61.58	72419.01	1008.55	81.07	20.72	194278.59	76171.99	49.26
41.51	-72.41	7620.00	0.95	65911.17	6.68	55084.33	92.23	8.87	2.27	21077.11	8263.83	5.34
41.48	-72.45	7620.00	2.48	65893.67	17.49	55082.71	241.50	23.23	5.94	55190.17	21638.75	13.99
40.64	-73.92	7014.40	6.24	65888.44	5.23	-6066.09	23.16	79.20	7.59	16493.56	6466.73	4.18
40.64	-73.78	6404.80	6.28	65882.68	5.77	9751.67	25.42	88.54	8.35	18190.71	7132.14	4.61
40.64	-73.92	5795.20	6.28	65876.54	6.14	10572.70	26.93	95.54	8.86	19368.19	7593.80	4.91
40.64	-73.78	5185.60	6.28	65870.01	6.53	11340.63	28.46	102.95	9.39	20596.29	8075.31	5.22
40.64	-73.92	4576.00	6.28	65863.08	6.93	12058.26	30.02	110.78	9.94	21876.23	8577.15	5.55

Figure 11. AEDT Inventory Analyzer, Fuel Burn Capability Example Results

The tool allows for the easy export of all query results to Microsoft Excel and Word for documentation and further analysis purposes, as well as to comma-delimited CSV format.

The next phase of development will include the addition of global noise inventory results to the querying capability. Ultimately, the tool is envisioned to be capable of generating both noise and emissions global inventories and/or policy scenario computations. With the ability to remotely query noise and emissions inventories, the tool will be useful for policymakers at both the domestic and global level, without the need for the development of analysis-specific software or the direct involvement of software developers.

4 AEDT Development

4.1 Timeline

Figure 12 presents the anticipated AEDT development timeline though the year 2010. In addition to AEDT-specific milestones, since AEDT is envisioned to directly support the ICAO CAEP decision process, specific important CAEP meetings/milestones are identified throughout the development process.

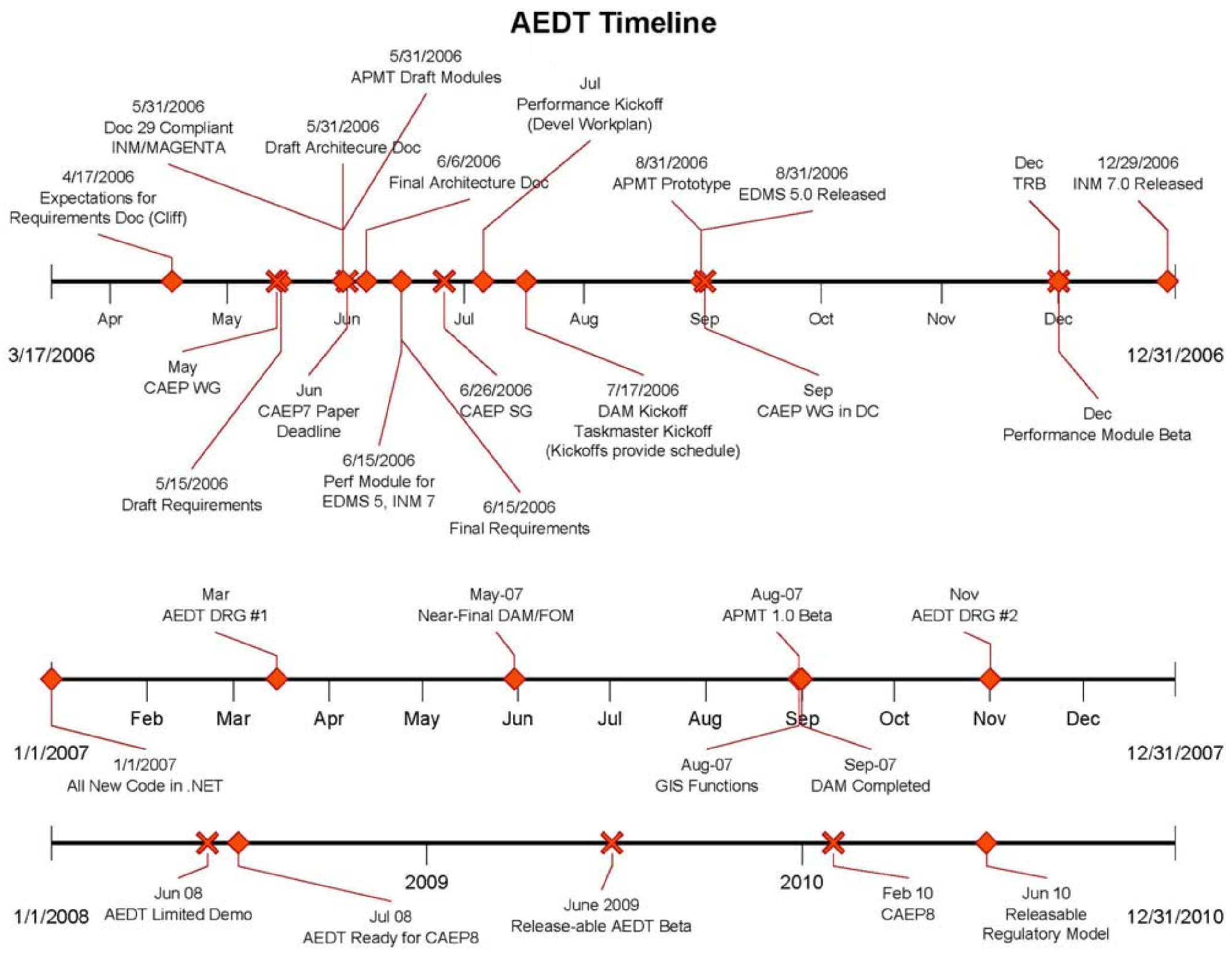


Figure 12. AEDT system development follows a four year implementation timeline with phased implementation of stakeholder driven functionality requirements

4.2 Connectivity with Other Tools

AEDT will be a standalone system capable of modeling impacts due to aviation sources. In addition, however, AEDT will need to directly interface with other tools. Sections 4.2.1 through 4.2.3 outline that connectivity.

4.2.1 Environmental Design Space

The Environmental Design Space (EDS) will be the tool used to estimate source noise, exhaust emissions, performance, and economic parameters for potential future aircraft designs under different technological scenarios. Incorporating detailed future aircraft designs in environmental analyses will allow for more accurate prediction of future trends of aviation impacts.

AEDT will interface with EDS directly through the Fleet Database. In particular, EDS output data will directly populate additional records in the database. This will allow for direct use of EDS-defined aircraft within AEDT.

4.2.2 Aviation environmental Portfolio Management Tool

The Aviation Environmental Portfolio Management Tool (APMT) will be the tool used to effectively assess and communicate environmental effects, interrelationships, and economic consequences based on integrated analyses. It will directly link the environmental predictive capabilities of AEDT with comprehensive economic analysis capabilities, which have previously been done independently.

APMT will link to AEDT by using the AEDT databases and modules outline in sections 3.4 and 3.5 of this document. This direct linkage is made possible by the detailed code development and documentation standards outlined in sections 3 and 4.

4.2.3 Other Tools

Connectivity with all tools is preferred to be via direct use of modules (specifically dynamic link libraries – DLLs). In rare occasions, module source code may be exchanged. Specific guidance is available for these exceptions.

4.3 System Maintenance and Coding Standards

For the purposes of AEDT development, implementation and coding standards will be applied at the module level. Application of these standards down to the unit level is encouraged, but in most cases opaque to the overall system. The standard consists of four components: 1) Interface Control Document (ICD), 2) ADD, 3) exemplar I/O data set (to be included within the appropriate ICD) for verification, and 4) a set of formatting rules for the module interface.

Each module will have a single point of authority and responsibility for version release. For a software module release to be accepted, each of these components must be available for the given version to be released. Example ICD, ADD, and header file templates can be found in the appendix. The release of a module is accomplished through the publication of draft ICD, ADD, example I/O, header file, and module library information on the FAA AEE KSN and notification of Andrew Hansen, the AEDT Technology Expert at Volpe. A rigorous review process is then undertaken involving Mr. Hansen and module point of contact, to ensure (1) internal consistency and completeness for the module and documentation, and (2) to verify consistency across all AEDT modules, in terms of both coding implementation and supporting documentation. Module release is accomplished at the end of the joint review process, after which the module is entered into the AEDT software module and database matrix located on the KSN[5]. Upon release, the module authority is the contact through which users feed back information about compatibility, usability, and any bug reports.

The ICD sets the input and output specification for the module release. It identifies name, type, units, and assumptions on each variable in the module interface and falls under the same version control as the software implementation.

The ADD conveys the mathematical and logical concepts that encode the software module. Much like a technical manual for the module, it includes underlying equations and schematics for the algorithms   The ADD references all constants used in the module implementation and assumptions about the physical or logical system being modeled by the encoded algorithm. As with the ICD, the ADD falls under the same version control as the software module

The header file formatting rules serve to make the reading and interpreting of the module interface more efficient. They establish naming conventions that map with dictionary and ICD entries so that the header file is a summary of the ICD. The resulting header file is, of course, tied to the software module and falls under the same version control.

The example I/O data set provides users instruction on preparing data for the associated module as well as a test mechanism to ensure that the implementation was invoked properly. The data set must express all mandatory and optional input parameters. The test data set also establishes a formal QA/QC evaluation mechanism.

In addition to the software implementation standards above, the AEDT standard for supporting databases and their release is the Database Design Document (DDD). The DDD establishes the sources, format, and parameters to be contained in the database, the process for updating or adding information to the given database, and the dated release version of the database itself. As with software modules, a database release is accomplished through the publishing of the Database and its DDD on the FAA KSN by the respective database lead, and successfully completing the review process for consistency and completeness.

Manipulation of the input databases (fleet, movements, airports) will be a primary way in which external users will interact with the AEDT system. The details in the respective DDDs provide the information that the interface designers and end users will need to effectively present and manipulate configurable data or pre-process new data for inclusion into the system.

[5] https://ksn.faa.gov/km/aee/aedt/Documentation/In%20Development/Architecture/AEDT_doc_checklist.doc

These coding and database standards improve the transparency of the AEDT application under both external review/auditing and future development, if extending the implementation to accommodate new requirements. In the immediate process, the ICD/ADD/DDD documents are the foundation of the modularity in the AEDT design. As a case in point, the ICD and DDD alone provide enough information to build a graphical interface for a given module or database.

The guideline further specifies the platform to be used for design and implementation. As noted above, MSVS .NET 2005 is the integrated development environment being utilized on the AEDT project. This choice has implications on software module implementation and the integration of those modules into target applications. However, .NET introduces additional options for software implementation that are not constrained by the module requirements noted above. Additional guidelines are provided here for the software engineer executing the software implementation from a given module design.

- Microsoft .NET Framework 2.0
    - Implementation of module software should be based on Framework 2.0
    - CLR compilation
- Module Namespace
    - Scope for a module's data and functions must be defined in its namespace
    - Each module's ICD must document the module namespace
    - Namespace descriptions in the ICD are the .NET analog of header files
- Compilation Settings
    - Computation module build targets must be Dynamic Link Libraries
    - New module implementations must be thread-safe

5 Project Management

A long-term objective for AEDT development is a close integration of project management with the software development and implementation. As such, there are several benefits to the choice of the MSVS integrated development environments.

- Integrated project management between project plans and managed objects
    - Project plans (timelines, receivables, deliverables, funding/costing, etc.)
        - Direct interface with MS Project
        - Project planning, estimation, and tracking
        - Completion of milestone triggers (i.e., uploading final database to server maps to completion of milestone)
    - Interface with Microsoft SharePoint (FAA-KSN)
    - Potential for integrated version-control and configuration management
- Three categories of managed objects (explicit versioning and revision control)
    - Algorithm description, interface control, and DDDs
    - Software implementations (source code, libraries, programs)
    - Test & validation data (scenarios, source databases, results)
    - Each linked back to milestones/deliverables in project plans
- Accessibility to plans and managed objects
    - Location/people
        - FAA
        - Volpe Center
        - National Aeronautics and Space Administration (NASA)
        - Private contractors (3)
        - Academia (2)
    - Addressable user permissions (read-only, author, administrator, etc.)
    - Centralized administration but collaborative development
    - Web-based interface
- Rollout Schedule: Phased approach across managed objects
    - Project plans--need for immediate application (next couple of weeks)
    - Algorithm and interface document control--next few months
    - Software implementation version control—6 to 8 months
    - Test and validation data--over the next year

5.1 Organizational Relationships

The AEDT Development Team is made up of the FAA-AEE, the John A. Volpe National Transportation Systems Center, Environmental Measurement and Modeling Division (Volpe Center), ATAC Corporation, CSSI Inc. and Wyle Laboratories, Inc. Due to the tight linkage between AEDT and the EDS and the Aircraft Portfolio Management Tool (APMT), the Massachusetts Institute of Technology (MIT) and Georgia Tech, the developers of EDS and APMT, are also tightly coupled with all AEDT-related architecture.

Specific roles of each of the AEDT Development Team members are outlined below. Where appropriate, specific individuals at each organization are identified. Each team member also has at least one representative who participates in regularly-scheduled, bi-weekly conference calls, which are a mechanism to (1) update the group on recent development activities; and (2) define the path forward on specific architecture issues.

Volpe Center

<u>Project Management –Roof (Electronics Engineer)/Reherman (General Engineer)</u>

Develop detailed monthly progress reports for development team activities; Develop various project plans; Ensure adherence to individual project plans; lead bi-weekly AEDT Leads calls, Maintain and track annual AEDT budget and task spreadsheet, Assist in the preparation of contractor-related statements of work

<u>CAEP Support - Fleming</u>

Chair ICAO CAEP Working Group 2, Task Group 2; Provide technical support to TG3 and TG4, as appropriate, Provide technical support to Steering Group and full CAEP meetings; liaison and attend Forecasting and Economic Analysis Support Group (FESG) meetings, and meetings of WG1 and WG3 and their subgroups, as appropriate (e.g., in support of Campbell Hill, Best Practices, In Production databases); coordinate all AEDT Development Team activities in support of WG1-3, SG, CAEP and FESG

<u>APMT Support –Roof (Electronics Engineer)/Hansen (Electronics Engineer)</u>

Represent AEDT by participating in all APMT conference calls, including Leads, Integration, PEB, Assessment, and Analysis and Display teams; provide AEDT module assessment support to APMT team; provide AEDT modules and databases for use by APMT; support APMT team in the integration of AEDT modules and databases

<u>Other FAA Development Initiatives Liaison - Fleming</u>

Coordinate development activities and provide module technical support to other FAA environmental modeling activities such as NIRS, ATO/ATC, etc.; lead Joint Planning and Development Office (JPDO) Environmental Integrated Product Team (EIPT) Tools Panel; liaison between AEDT and Evaluation Analysis Division (EAD); coordinate AEDT development with Partnership for Air Transportation Noise and Emissions Research (PARTNER) Center of Excellence (COE)

<u>Architecture –Roof (Electronics Engineer)/Hansen (Electronics Engineer)</u>

Develop AEDT Architecture Document to summarize, standardize and harmonize all development activities via QA/QC of all module and database architecture documentation; lead bi-weekly architecture calls; advise development team on architecture-related initiatives; lead effort to integrate AEDT modules and databases into a single system that can be accessed through a common interface

<u>Verification and Validation - Fleming</u>

Develop, maintain and execute an AEDT Verification and Validation (V&V) plan which will guide all system, module and database V&V efforts.

<u>Database Access Module - Balasubramanian (Senior Engineer)</u>

Lead development efforts of the AEDT database access module; this includes direct coordination with APMT and associated requirements to ensure module meets common needs

<u>Emissions Module - Malwitz (Electronics Engineer)</u>

Develop and maintain the algorithms for the AEDT emissions module; perform measurement activities in support of algorithm development and validation and verification

<u>Terrain Module – Gerbi (Middle Engineer)</u>

Develop and maintain the algorithms and interfaces for AEDT access to terrain models and databases.

<u>Noise Module - Boeker (Physical Scientist)</u>

Develop and maintain the algorithms for the AEDT aircraft noise module; perform measurement activities in support of algorithm development and validation and verification for the module; liaison with international groups (i.e., Society of Automotive Engineers (SAE) A-21 and European Civil Aviation Conference (ECAC)/AERMOD (steady-state plume dispersion regulatory model) to ensure module capabilities are consistent with international guidance

<u>Taskmaster Module – Hansen (Electronics Engineer)</u>

Work with ATAC, CSSI, and Wyle on development of algorithms for the AEDT taskmaster module; this includes direct coordination with APMT and associated requirements to ensure module meets common needs.

<u>INM - Boeker (Physical Scientist)</u>

Continue to support public releases of INM, including noise module enhancements and bug fixes, V&V of Noise-Power-Distance (NPD) updates, maintenance of research versions, including acoustically hard ground and multiconfiguration capabilities; provide user support related to noise computation issues,

<u>AEDT/SAGE - Malwitz (Electronics Engineer)</u>

Continue to develop annual global aviation emissions inventories; update AEDT/SAGE per state of current research (i.e., algorithm enhancements such as Specific Fuel Consumption (SFC) corrections, updated flight track dispersion methodologies, etc.)

<u>MAGENTA-SAGE Integration (co-lead) - Hansen (Electronics Engineer)</u>

Co-lead development efforts to harmonize and combine existing and planned SAGE and MAGENTA capabilities, including development of as standalone, integrated inventory querying capability and GUI

<u>Global Inventory Query GUI - Malwitz (Electronics Engineer)</u>

Develop and maintain a global noise and emissions inventory querying capability, including the ability to run OD pair-based fuel burn.

<u>Airports Database – Malwitz (Electronics Engineer)</u>

Develop and maintain the AEDT airports database, including harmonizing with existing legacy tools and newer sources as they become available, coordinate this activity with ICAO/CAEP, as appropriate

<u>Global Movements Database – Balasubramanian (Senior Engineer)</u>

Develop and maintain the AEDT global movements database, including generating annual databases (from International Official Airline Guide (IOAG), Enhanced Traffic Management System (ETMS), Enhanced Traffic Flow Management System (ETFMS) and other sources as they become available); this includes liaison activities with groups such as Eurocontrol in order to further international buy-in to the database

<u>Joint EDMS-INM DRG –Boeker (Physical Scientist)</u>

Co-lead joint group made up of EDMS and INM DRG members

ATAC

Aircraft Performance Module – Dinges (Program Manager)

Develop and maintain the algorithms for the AEDT aircraft noise module; perform measurement activities in support of algorithm development and validation and verification for the module; liaison with international groups (i.e., SAE A-21 and ECAC/AERMOD to ensure module capabilities are consistent with international guidance

Taskmaster Module - Bea (Engineer)

Lead development efforts for the AEDT taskmaster module used to coordinate interface/processes for all communications between AEDT modules; this includes direct coordination with APMT and associated requirements to ensure module meets common needs

Radar Flight Profile Module – Dinges (Program Manager)

Develop and maintain the algorithms for the AEDT radar flight profile module used to create AEDT-compatible flight paths, including thrust, from aircraft position data. Support development of an SAE guidance document covering the methodology and update module to stay consistent with SAE guidance as appropriate.

INM - Dinges (Program Manager)

Continue to support limited public releases of INM, which will end at least a year prior to public release of AEDT, including GUI and flight module enhancements and bug fixes, V&V of NPD updates, maintenance of research versions, including acoustically hard ground and multiconfiguration capabilities; provide user support related to GUI and aircraft performance issues; continue harmonization of methods and databases with EDMS

Local GUI (co-lead) – Bea (Engineer)

Co-lead development of an integrated AEDT Local GUI for noise and emissions analysis

Terminal Area Movements Data - Dinges (Program Manager)

Maintain and update the Performance Data Analysis and Reporting System (PDARS) database for use in various AEDT assessments

Continuous Descent Approach (CDA) Demonstration - Dinges

Develop algorithms and software required to support coordinated noise and emission analyses of the effects of CDAs including making use of PDARS data, perform CDA Demonstration performance and noise analyses.

Fleet Database - Dinges (Program Manager)

Support AEDT Fleet Database development related to aircraft flight performance data including ensuring consistency between the AEDT database and current best practice flight profile and performance data definitions.

Joint EDMS-INM DRG – Dinges (Program Manager)

Co-lead joint group made up of EDMS and INM DRG members

CSSI

Aircraft Queuing and Delay Module – Nguyen (Program Manager)

Develop and maintain the algorithms for the AEDT module to model aircraft delay, queuing, and sequencing of aircraft operations.

EDMS - Nguyen (Program Manager)

Continue to develop and support limited public releases of EDMS, which will end at least a year prior to public release of AEDT, including GUI and computation enhancements and bug fixes, provide user support related to EDMS, and continue harmonization of methods and databases with INM.

Weather – Nguyen (Program Manager)/Yirenkyi (Software Developer)

Develop and maintain the algorithms for a module for reading, processing, and storing atmospherics / weather data for AEDT.

Taskmaster Module – Hall (Senior Software Engineer)/Nguyen (Program Manager)

Work with ATAC, Volpe, and Wyle on development of algorithms for the AEDT taskmaster module; this includes direct coordination with APMT and associated requirements to ensure module meets common needs.

Flight Operations Module - Nguyen (Program Manager)

Work with Wyle on enhancements to the FOM to include consideration of airport capacity and spreading of aircraft schedules when demand exceeds capacity.

Local Dispersion Modules - Hall (Senior Software Engineer)

Maintain AEDT compatibility with and implement the latest EPA atmospheric dispersion models.

Local GUI (co-lead) - Hall (Senior Software Engineer)

Co-lead development of an integrated AEDT Local GUI for noise and emissions analysis.

Requirements - Nguyen (Program Manager)

Develop AEDT Requirements Document to summarize all AEDT global and local noise and emissions requirements.

Fleet Database - Hall (Senior Software Engineer)

Develop and maintain the AEDT fleet database, including harmonization of aircraft types between EDMS and INM, and groupings between the US and Europe.

Joint EDMS-INM DRG –Thrasher (Executive Director)

Co-lead joint group made up of EDMS and INM DRG members

Wyle

Flight Operations Module – Grandi (Senior Engineer)

Develop and maintain the AEDT flight operations module; this includes the generation of scenario-specific flight operations and movements for AEDT and APMT sample problem scenario analysis

Forecasting Databases - Grandi (Senior Engineer)

Maintain and update databases (e.g., FESG projections, replacement aircraft, etc.) related to forecasting and scenario assessments

MAGENTA – Grandi (Senior Engineer)/Usdrowski (Engineer)

Continue to develop domestic and international MAGENTA runs; update AEDT/MAGENTA per state of current research (i.e., updates to core INM noise and flight trajectory computations)

GIS – Ledoux (Senior GIS Specialist)

Lead AEDT and APMT GIS development activities in order to leverage a single set of consistent tools across the entire development effort

MAGENTA-SAGE Integration (co-lead) - Grandi (Senior Engineer)

Co-lead development efforts to harmonize and combine existing and planned SAGE and MAGENTA capabilities, including development of as standalone inventory querying capability

All Organizations

International Technical Conference Support

Promote AEDT development and gain stakeholder buy-in through participation in and presentations at international technical conferences such as INTERNOISE, AWM&A and ICAS

6 AEDT Dictionary

6.1 Glossary of Terms

Algorithm Design Document: Written or schematic description of the math, logic, and definitions implemented by a structural element (module or sub-module).

Architecture: The description of the structural elements in a system, their externally visible properties, and the relationships between elements. *Alt*: A framework for the disciplined introduction of change to a design.

Bandwidth Constraint: An information processing constraint where the output depends on the product of two finite, often competing, resources. Classic example is signal processing—frequency and time.

Behavioral View: Architectural document describing the instantiation of the structural modules in the architecture, e.g. database, software library, stand-alone application, procedure, script, etc., as well as their host platforms and, potentially, user interfaces.

Computational Pipeline: A processing model whereby data is manipulated in a preconfigured way at different stages along the pipeline often in a stream based approach. Effective parallelism can be achieved by replicating pipelines.

Data Flow Diagram: Architectural document containing a schematic that identifies the information exchanged between structural elements of the system.

Linear Dynamic System: Description of a physical phenomenon that can be represented as a linear state transition accounting for time varying forces via a state transition matrix.

Functional View: Architectural document containing a schematic that identifies the structural elements of the system and describes the scope of the each element's function.

Interface Control Document: Specification of the format, communication protocol, units, valid ranges, and assumptions for input and output data exchanged by a structural element (module or sub-module)..

Kinematic System: Description of a mechanical system representing the equations of motion for a body with constant forces.

Software Layer: The level at which a software developer implements code: Source, Application Programmer Interface (API), Dynamic Library, and Application. Generally the software developer encodes (or compiles) from one given level to the next.

Structural Element: A uniquely identifiable module that can be isolated within a complete system. Structural elements may be hierarchical, e.g. a software module, which is a structural element, may be further broken down into units, which are themselves structural elements of the software module.

6.2 Acronyms

A-21	SAE Committee on Aviation Noise
ADD	Algorithm Description Document
AEDT	Aviation Environmental Design Tool
AEE	Office of Environment and Energy
AERMOD	Steady-state plume dispersion regulatory model
API	Application Programmer Interface
APMT	Aviation Environmental Portfolio Management Tool
CAEP	Committee on Aviation Environmental Protection
CDA	Continuous Descent Approach
CLR	Common Language Runtime
COE	Center of Excellence
DDD	Database Design Document
DRG	Design Review Group
ECAC	European Civil Aviation Conference
EAD	Evaluation Analysis Division
EDMS	Emissions and Dispersion Modeling System
EDS	Environmental Design Space
EIPT	Environmental Integrated Product Team
ETMS	Enhanced Traffic Management System
ETFMS	Enhanced Traffic Flow Management System
Eurocontrol	European Organization for the Safety of Air Navigation
FAA	Federal Aviation Administration
FAR	Federal Aviation Regulations
FBE	Fuel Burn and Emissions
FESG	Forecasting and Economic Analysis Support Group
GUI	Graphical User Interface
HNM	Heliport Noise Model
ICAO	International Civil Aviation Organization
ICD	Interface Control Document
INM	Integrated Noise Model
IOAG	International Official Airline Guide
IPCC	Intergovernmental Panel on Climate Change
JPDO	Joint Planning and Development Office
KSN	Knowledge Services Network
MAGENTA	Model for Assessing Global Exposure to the Noise of Transport Aircraft
MASAGE	MAGENTA–SAGE
MIT	Massachusetts Institute of Technology
MSDN	Microsoft Developers Network
MSVS	Microsoft Visual Studio
NAS	National Academies of Science
NASA	National Aeronautics and Space Administration

NOAA	National Oceanic and Atmospheric Administration
NPD	Noise-Power-Distance
O/D	Origin/Destination
PARTNER	Partnership for Air Transportation Noise and Emissions Research
PDARS	Performance Data Analysis and Reporting System
QA	Quality Assurance
QC	Quality Check
RDBMS	Relation Database Management System
SAE	Society of Automotive Engineers
SAGE	System for assessing Aviation’s Global Emissions
SFC	Specific Fuel Consumption
SQL	Structured Query Language
TRB	Transportation Research Board
UML	Unified Modeling Language
UN	United Nations
V&V	Validation and Verification

7 References

Fleming, G., et. al., “Aviation Environmental Design Tool Work Plan”, FAA Technical Report, August, 2004.

AEDT Software Requirements Document, Doc #AEDT-REQ-01, 8/15/2006].

Letter Report, Workshop #1, FAA Aviation Environmental Design Tool (AEDT), March 31 – April 2, 2004, Washington, DC, Transportation Research Board of the National Academies. (http://onlinepubs.trb.org/onlinepubs/reports/aedt_nov_2004.pdf)

Letter Report, Workshop #2, FAA Aviation Environmental Design Tool (AEDT) and Aviation environmental Portfolio Tool (APMT), August 24 – August 26, 2004, Washington, DC, Transportation Research Board of the National Academies. (http://onlinepubs.trb.org/onlinepubs/reports/aedt_april_2005.pdf)

Letter Report, Workshop #3, FAA Aviation Environmental Design Tool (AEDT) and Aviation environmental Portfolio Tool (APMT), January 31 – February 2, 2005, Washington, DC, Transportation Research Board of the National Academies. (http://onlinepubs.trb.org/onlinepubs/reports/aedt_may_2005.pdf)

http://www.microsoft.com/Net/Default.aspx - “What is Microsoft .NET Framework?”

http://support.microsoft.com/ph/8291 - “Microsoft .NET Framework 2.0”

http://support.microsoft.com/ph/8940 - “Microsoft ASP.NET 2.0”

http://msdn.microsoft.com/vstudio/ - “Microsoft Visual Studio 2005”

http://msdn2.microsoft.com/en-us/teamsystem/default.aspx - “Redefining Database Development with SQL Server and Visual Studio Team System”