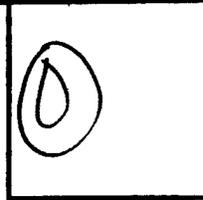


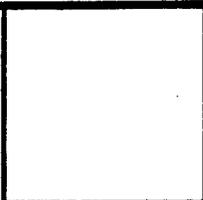
# LOAN DOCUMENT

PHOTOGRAPH THIS SHEET



INVENTORY

DTIC ACCESSION NUMBER



LEVEL

AFRL-ML-TV-TR-2000-4542

DOCUMENT IDENTIFICATION

Nov 99

**DISTRIBUTION STATEMENT A**  
Approved for Public Release  
Distribution Unlimited

DISTRIBUTION STATEMENT

ALTERNATE FOR	
NTIS	GRAM
DTIC	TRAC
UNANNOUNCED	
JUSTIFICATION	
BY	
DISTRIBUTION/	
AVAILABILITY CODES	
DISTRIBUTION	AVAILABILITY AND/OR SPECIAL
A-1	

DISTRIBUTION STAMP

DATE ACCESSIONED

DATE RETURNED

20000622 064

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

H  
A  
N  
D  
L  
E  
  
W  
I  
T  
H  
  
C  
A  
R  
E

This report was prepared under contract to the Department of Defense Strategic Environmental Research and Development Program (SERDP). The publication of this report does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official policy or position of the Department of Defense. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the Department of Defense.

**AFRL-ML-TY-TR-2000-4542**



# **THE EMISSION REDUCTION PLANNING MODEL**

**DANIEL M. MALONEY**

**ARGONNE NATIONAL LABORATORY  
ENVIRONMENTAL ASSESSMENT DIVISION  
9700 SOUTH CASS AVENUE  
ARGONNE IL 60439**

**Approved for Public Release; Distribution Unlimited**

**MATERIALS & MANUFACTURING DIRECTORATE  
AIR FORCE RESEARCH LABORATORY  
AIR EXPEDITIONARY FORCES TECHNOLOGIES DIVISION  
139 BARNES DRIVE, STE 2  
TYNDALL AFB FL 32403-5323**

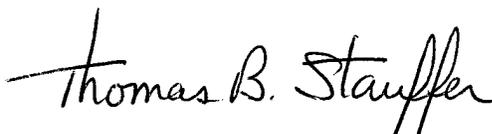
NOTICES

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

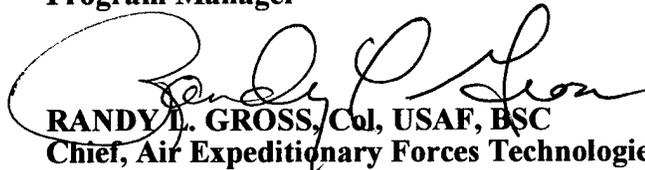
THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.



TIMOTHY G. WILEY, Lt Col, USAF, BSC  
Program Manager



THOMAS B. STAUFFER, PhD, DR-IV, DAF  
Chief, Weapons Systems Logistics Branch



RANDY L. GROSS, Col, USAF, BSC  
Chief, Air Expeditionary Forces Technologies Division

**REPORT DOCUMENTATION PAGE***Form Approved*  
*OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 11 May 1999	3. REPORT TYPE AND DATES COVERED Final Report: November 1992 – November 1999	
4. TITLE AND SUBTITLE The Emission Reduction Planning Model			5. FUNDING NUMBER MIPR – N28FY96-30 PE – 63723F PROJECT NO. – 2103H303	
6. AUTHORS Maloney, Daniel M.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Argonne National Laboratory Environmental Assessment Division 9700 South Cass Avenue Argonne IL 60439			8. PERFORMING ORGANIZATION REPORT NUMBER	
8. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/MLQL 139 Barnes Drive, Ste 2 Tyndall AFB FL 32403-5323			10. SPONSORING/MONITORING AGENCY REPORT NUMBER  AFRL-ML-TY-TR-2000-4542	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for Public Release; Distribution Unlimited			12b. DISTRIBUTION CODE  A	
13. ABSTRACT (Maximum 200 words) Day-to-day activities at both industrial and Department of Defense (DoD) facilities require the operation of a wide variety of air pollution sources, and achieving compliance with applicable air pollution regulations and permits governing the operation of such air pollution sources can be complicated. The Air Compliance Advisor (ACA) software was designed as a decision support tool to assist environmental coordinators in developing optimized compliance strategies.  This report presents a description Emission Reduction Planning Model (ERPM) and the ACA. The ACA is an extendable computer-based decision support system for developing air pollution compliance strategies. It was developed to assist environmental personnel to achieve and maintain compliance with applicable regulations and permit conditions. It accomplishes this by providing a framework for collecting site-specific data, providing reliable estimates of emissions in the absence of actual data, providing estimates of pollution control options' efficiencies and costs, and integrating the functionality of regulatory requirements for federal facilities to determine what regulations may apply. While the software was developed initially for use by the DoD, these capabilities are clearly useful to industry and regulators both in the U.S. and Canada.				
14. SUBJECT TERMS Compliance, air pollution control, clean air act, title V, air compliance advisor, control Device, cost estimation			15. NUMBER OF PAGES	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT  UL	

## ABSTRACT

Day-to-day activities at both industrial and Department of Defense (DOD) facilities require the operation of a wide variety of air pollution sources. Environmental Coordinators (ECs) at these facilities are tasked with maintaining compliance with applicable air pollution regulations and permits governing the operation of air pollution sources. A well planned, cost-effective compliance strategy requires ECs to be knowledgeable in many different technical areas. The Air Compliance Advisor (ACA) software was designed as a decision support tool to assist ECs in developing optimized compliance strategies.

This report presents a description Emission Reduction Planning Model (ERPM) and the ACA. The ACA is an extendible computer-based decision support system for developing air pollution compliance strategies. It was developed to assist environmental personnel to achieve and maintain compliance with applicable regulations and permit conditions. It accomplishes this by providing a framework for collecting site-specific data, providing reliable estimates of emissions in the absence of actual data, providing estimates of pollution control options' efficiencies and costs, and integrating the functionality of regulatory requirements for federal facilities to determine what regulations may apply. While the software was developed initially for use by the DoD, these capabilities are clearly useful to industry and regulators both in the U.S. and Canada.

## TABLE OF CONTENTS

Section	Page
1. INTRODUCTION.....	1
2. BACKGROUND.....	1
3. THE ACA SYSTEM.....	6
4. RESULTS.....	9
5. SUMMARY/CONCLUSION.....	10
APPENDIX A: EXGLUI Programmer's Manual	
APPENDIX B: The ACA User Guide	
APPENDIX C: ACA Training Material	

## **INTRODUCTION**

An extendible computer-based decision support system for developing air pollution compliance strategies entitled, the Air Compliance Advisor (ACA) has been developed as a result of the work done on the Emission Reduction Planning Model (ERPM) project. For base level Department of Defense (DoD) environmental personnel, the Microsoft Windows based ACA program has the ability to play a beneficial role in dealing with various issues including facility-wide emissions, control technologies, and regulations.

The work described in this report outlines the work done by D & E Technical and Argonne National Laboratory, sponsored by the US Army Corps of Engineers Construction Engineering Research Laboratory (USACERL), the US Air Force Airbase and Environmental Technology Division of the Air Force Research Laboratory (AFRL/MLQ), the Strategic Environmental Research and Development Program (SERDP), and the US Environmental Protection Agency (EPA).

## **BACKGROUND**

The day-to-day activities at DoD installations require the operation of a wide variety of air pollution sources in order to fulfill the installation's mission. Environmental Coordinators (ECs) at DoD facilities are tasked with maintaining compliance with applicable air pollution regulations and permits governing the operation of air pollution sources.

The ERPM research and development project was begun to create an automated system to assist experienced DoD air pollution managers tasked with ensuring facility compliance with applicable federal, state, and local air quality regulations. Environmental Coordinators (ECs) at DoD facilities are in the difficult position of maintaining compliance with all their facility's air pollution sources. To develop a successful compliance strategy for a facility the EC requires considerable knowledge in several areas: source characterization; emission reduction techniques (e.g., the application of control devices, or the modification of source operations); emission trading issues; federal, state, and local regulatory requirements; and permit conditions. Considering both the level of expertise required to develop a compliance strategy and the goal of creating an optimized compliance strategy, ECs face a daunting task. This type of complex problem, requiring detailed knowledge in many areas, is handled well by automated, computer-based data/analysis management system.

### **Regulatory Basis**

Passage of the 1990 Clean Air Act Amendments (CAA-90) supports the need for an automated system to assist experienced DoD ECs in the development of compliance strategies. This report will not attempt to detail the various Titles of the CAA-90, yet it is important to mention the three parts of the CAA-90 that will have the greatest impact on DoD facilities. The requirements of Titles I (Attainment and Non-attainment), Title III (Hazardous Air Pollutants), and Title V

(Permits) will likely have the greatest impact on stationary air pollution sources at DoD facilities. To illustrate the need for an automated system to assist experienced DoD air pollution managers in developing compliance strategies, a brief discussion of these three titles follows.

Title I of the CAA-90 considers the requirements for areas across the United States that have failed to achieve the National Ambient Air Quality Standards (NAAQS), and the requirements for maintaining the NAAQS for areas in attainment. Title I establishes new, lower limits for the classification of "major" sources in non-attainment areas. In some cases, the major category classification is a function of the degree of non-attainment. For example, in areas defined as "marginal" non-attainment for ozone, a source is classified as major if it has VOC or NO<sub>x</sub> emissions greater than 100 tons per year (TPY), whereas sources that generate more than 10 TPY of VOC or NO<sub>x</sub> would be classified as major in "extreme" ozone non-attainment areas. Similar requirements are imposed for CO, SO<sub>x</sub>, and PM<sub>10</sub> non-attainment areas. Reasonably Available Control Technology (RACT) applies to all existing facilities in non-attainment areas classified in the major source category.

Title III of the CAA-90 requires emission control for 189 chemical substances designated as Hazardous Air Pollutants (HAPs). Any new or existing major source in a particular HAP source category is subject to the Maximum Achievable Control Technology (MACT) requirements for that source category; major sources are generally defined as a source emitting greater than 10 TPY of a single HAP, or greater than 25 TPY of multiple HAPs. Sources can defer compliance with MACT requirements by participating in the early reduction program. This program requires that facilities meet specific emission inventory requirements and demonstrate an actual emission reduction between 90-95%, depending on the HAP.

Title V of the CAA-90 may have the greatest impact on DoD facilities. Title V requires (1) submission of Title V operating permit applications, (2) extensive reporting and monitoring of chemical emissions, and (3) annual permit fees assessed on a ton of pollutant per year basis. For major stationary sources, all of the CAA-90's regulatory requirements, and the regulations in effect before the 1990 amendments, will be implemented through the Title V operating permit program. All major facilities that currently have operating permits must reapply for Title V permits. Initially, Title V requirements will apply to sources that fall into the major source category (as defined by Titles I and III), and eventually may include non-major sources. Given the reality of compliance with Title V requirements, the best option for any facility is to write their permit applications carefully to allow the greatest degree of operational flexibility at a reasonable cost.

When discussing Title V permits, there are two main issues to consider. The first issue relates to the preparation and planning of Title V permit applications. The second issue relates to day-to-day compliance with the conditions of a Title V permit.

Preparation for a Title V permit is a considerable undertaking. When planning the Title V permit, the EC at a facility must understand the mission of the facility and anticipate any changes in mission based on mobilization requirements, base realignment, and base closure. The EC must have knowledge relating to the operation of all sources at the facility and collect data

pertaining to each source for an emissions inventory. The EC must also determine which federal, state, and local regulations apply to their sources and their facility as a whole.

The above information provides the "baseline" needed to plan for a Title V permit. When planning a Title V permit, the EC should consider the following strategies:

- Opting out of the major source category if possible, including becoming a synthetic minor,
- Process modification (where appropriate),
- Alternate operating scenarios, and
- Provision for a reasonable balance between operational flexibility and costs.

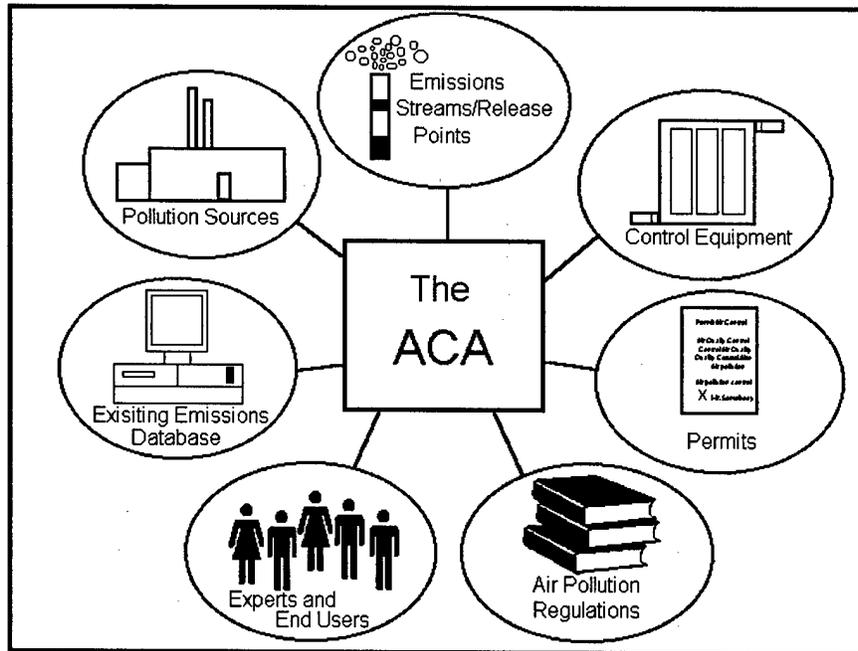
### **Issues Considered**

The diversity and magnitude of the task confronting ECs responsible for establishing and maintaining air pollution compliance at DoD facilities warrants the development of tools to assist ECs in performing this task.

Installations are often required to modify source operations, control pollutant emissions, monitor pollutant emissions, and/or significantly increase record keeping. A cost-effective compliance strategy cannot be achieved with a haphazard planning approach. A well-planned compliance strategy that considers all of the relevant issues needs to be developed.

A number of issues need to be considered in the development of an optimum air pollution compliance strategy, including: source characterization, emission reduction techniques (e.g., the application of control devices, and the modification of source operations), regulatory requirements of the federal, state, and local levels, and permit conditions. The optimal situation is for the EC to be somewhat knowledgeable in these areas and have the tools available to assist him/her in developing an optimized compliance strategy. Without a computerized tool to assist in decision making, ECs are clearly faced with a daunting task. Prior to the development of the ACA, there were no computerized tools available that simultaneously dealt with these key issues.

A combination of the requirements of CAA-90 and an investigation of existing resources (references, in-house expertise, and software packages) available to the base-level EC demonstrated the need for a comprehensive tool that would simultaneously deal with the key issues involved in developing air compliance strategies. A further need for providing expert-level guidance on developing compliance strategies and interpretation of air pollution data was recognized. Figure 1 presents a graphical representation of the issues that the envisioned software would consider.



**Figure 1. Issues considered in the ACA.**

### **Requirements Defined**

Three issues were identified to clarify the specifications of the required software tool: (1) an understanding of who the end-user would be, (2) the specific problems that would need to be solve and (3) the attributes of the problem.

Targeting the system to an end-user that lacked technical expertise could potentially sacrifice the power of the final tool, while targeting the most sophisticated end-user might limit the number of potential end-users. The end-user of the ACA was considered to be a DoD EC, with a moderate understanding in the area of air pollution sources/emissions and air pollution regulations. The end-user was also assumed to have at least a fundamental knowledge in the use of a personal computer (pc).

After an investigation into existing environmental software and base-level needs, the specific issues/problems that need to be simultaneously addressed were identified as:

- the representation of typical air pollution source data,
- solutions to "standard" air pollution related problems (e.g., calculate facility-wide actual and potential emissions, size control devices and estimate their costs),
- the need for software to be extendible by the end-user to represent atypical air pollution related data as well as solve non-standard problems that the end-user may encounter,
- the need to use minimal data for a quick analysis, and complete data for a more exact analysis,
- the need for the EC to play "what-if" games with the data to evaluate compliance options,
- the need to provide results in a way that the EC can easily understand the available options, and

- the need to bring the EC to the point that he/she could communicate effectively with regulators, engineering consultants, and vendors.

The attributes of air pollution related problems were classified in order to select the most appropriate framework, or development tool (e.g., a spreadsheet format, a procedural program, a database program), in which to develop the ACA system. Given the expected end user, an understanding of what the ACA needs to do, and the characteristics of air pollution data and analysis, the following problems were identified:

- a vast amount of data
- numerous resources for obtaining data
- incomplete data
- diverse set of analyses required
- the need to be extendible by end user

Each of the above problems will be considered below.

There is a vast amount of data relevant to air pollution problems (e.g., air pollution source characterization data, operating schedules, air pollution emissions data, chemical property data, material property data, permit conditions, regulations data). Both the diversity of these data types as well as the large number of instances of each data type (e.g., a facility could have hundreds of air pollution sources each operated in a multitude of ways on different operating schedules resulting in different emissions data) were considered in the selection of the development tool.

Given the diversity of the data types that are relevant to the air pollution compliance problem, there are numerous resources for obtaining this data. Since the data will likely come from a variety of resources, the format (e.g., units, level of data reduction) will also vary. The ability for the EC to enter disparate data types directly was a requirement that needed to be considered.

For the reasons outlined above much of the required data will be incomplete. In many cases a complete database will never be achieved, yet some level of analysis will still be required. The ability to provide suggestions based on a varying level of completeness in the data is an additional requirement that needed to be considered.

The potential types of analyses required in order to develop a compliance strategy are diverse and depend on the end-users objectives as well as completeness of the data. The ability to allow different types of analyses was a further requirement that needed to be considered.

It was decided that regardless of the number of analysis options available to the end-user, some end-users would find site-specific requirements. The ability to allow the end-user to build a database of "rules-of-thumb" and analysis capabilities that could be shared among DoD installations (as well as industry and EPA) was considered a requirement.

## THE ACA SYSTEM

The research phase of the ERPM project resulted in the determination that the ideal software tool for this specific application (e.g., air compliance management, control technology costing and applicability evaluation) would need to be:

- (1) Comprehensive — Capable of investigating the whole gamut of compliance strategies so that end-users would not have to learn and use many potentially incompatible tools that addressed only one industry or only one method of achieving compliance.
- (2) Integrated — Able to handle all the heterogeneous data required to solve these problems (control technology specific data; pollutant/chemical data; economic data; material data; and pollutant stream data [e.g., temperature, pressure, flow rate, pollutant concentration]).
- (3) Flexible — Allows the user to not only modify model defaults, analyses, data values and functions, but also to easily locate these changes and return to the original default values as needed.
- (4) Extendible — Allows the end-user to create new custom analyses and otherwise extend the functionality of the system.
- (5) User friendly — Presents the data in an obvious and intuitive manner and illustrates the relationship between the various data types and the functionality of each bit of data.

Unfortunately, neither procedural programming paradigms (e.g., FORTRAN) nor spreadsheets allow for all of these requirements to be met. To meet the needs identified in the ERPM research, the development team created a new custom computer language, EXLGUI, which became the heart of a new air compliance software tool, the Air Compliance Advisor. Appendix A of this report outlines the details of the EXLGUI computer language.

### System Description

The ACA software package contains the data structure and analysis tools designed to address many of the data management and analyses necessary for environmental managers to select a cost-effective control strategy for inclusion in its operating permit submission. The design of the ACA assumes that the typical end-user has some environmental/technical background with at least a moderate level of computer experience, yet the design allows the more advanced end-user virtually unlimited power and extendibility.

The ACA system is a Microsoft (MS) Windows-based tool. The minimum requirements to run the ACA effectively are:

- IBM PC or compatible computer
- 12 MB of RAM

- 12 MB of free hard drive space
- MS Windows 95, MS Windows 98, or MS Windows NT version 4.0 or later operating system

The general capabilities of the ACA can be divided into three categories: (1) database/program structure, (2) analytical functions, and (3) interface options and extendibility. Each of these areas is discussed below.

### **Database/Program Structure**

The ACA serves as a database for a significant portion of air pollution-related information. The data structure was designed to accommodate the diverse set of air pollution-related data required to perform typical compliance related studies, including emission unit data, control device data, chemical data, and source operational data. The ACA is designed with in an object-oriented, hierarchical structure data structure. This data structure allows the user to visualize the relationship between the various groups of data (i.e., objects) as well as effectively organize their data and any build custom analyses, as needed. Further, this type of data structure enables the developers of the ACA to easily create new analyses as well effectively manage the "warehouse" of existing analyses.

### **Analytical Capabilities**

A feature of the ACA is its ability to help develop compliance strategies. In order to develop these strategies, the ACA must be able to analyze diverse data. Some general analytical capabilities of the ACA include (1) automatic unit conversion, (2) minimal data entry requirements, and (3) user-defined analyses. Beyond the general analytical capabilities of the ACA, specific analytical capabilities of the ACA include (1) control device cost estimates and applicability grading, (2) calculations of potential and actual emissions and (3) checks for applicable federal regulations, including the National Emission Standards for Hazardous Air Pollutants (NESHAPS) and New Source Performance Standards (NSPS). Each of these capabilities is discussed in detail below.

Algorithms were developed and implemented for a variety of control technologies, including carbon adsorbers, condensers, electrostatic precipitators, baghouses, flares, thermal incinerators, wet scrubbers, and catalytic incinerators. For each control device, extensive research resulted in the development of algorithms for both the sizing and costing of the device. The analysis of an applicable control device considers both air pollution source-specific parameters (e.g., hours of operation, pollution stream temperature, pollution concentrations, pollution chemical characteristics, and particle size distributions) and control device specific parameters (e.g., achievable reduction efficiencies, cost estimation data, operation and maintenance requirements, and applicability constraints), as well as generic economic data (e.g., cost of labor, utilities, and cost indices).

The ACA also addresses the Title V Federal Operating Permit Program. A critical component of this program is the concept of potential to emit (PTE). PTE determines the applicability of the Title V Operating Permit Program and is a key factor in writing permit applications. Because the ACA data structure is designed to easily differentiate between actual and potential emissions through the use of operating schedules, calculation of the actual and potential emissions becomes

a simple bookkeeping exercise for an environmental manager once all relevant information regarding the operation and scheduling is entered into the ACA.

The *Regulations Data Engine* is another component of the ACA program. The ability to incorporate the functionality of individual federal, state, and local air quality regulations will assist environmental personnel with compliance issues. The Regulations Data Engine component of the ACA program contains an encoded version of both the applicability and the functionality of a regulation. Incorporation of regulations in this manner allows the computer to search available regulations to determine which regulations do not apply to a specific source. This eliminates a majority of regulations environmental personnel must manually investigate to determine applicability and compliance, resulting in a considerable saving of time and labor for environmental personnel. The ACA currently contains only US federal regulations, including NSPS, NESHAP, and MACT standards, but provides the data structure to add more regulations (e.g., state-specific regulations), as well as the built-in capability to search for potential applicability. While the other features of the ACA described above are mature, fully functional features, the Regulations Data Engine is still in the "prototype" phase of development.

### **Interface Options and Extensibility**

The user may select from three different skill levels when running the ACA. In novice mode, the ACA simplifies the displays by showing the minimum number of variables needed to perform a majority of the calculations. The other two modes (intermediate and expert) reveal progressively more detail. Beginning users can operate the ACA in novice mode, which is simple to learn but still quite powerful. Advanced users can make small revisions, replace entire equations, or extend the system with entirely new pollution sources and control devices.

The ACA was written in the EXLGUI custom software development tool. EXLGUI was designed specifically to support such scientific and engineering applications as the ACA. It consists of an object-oriented programming language and an environment that executes programs. EXLGUI takes the best features of several software development paradigms (procedural programming, spreadsheets, relational databases, artificial intelligence frameworks) and merges them into a single consistent framework that is both powerful and easy to use. It is easy to learn to program in EXLGUI and to make changes to source code written in EXLGUI. (see Appendix A) However, programming in EXLGUI is not required of the user, unless the user wants to extend the capabilities of the ACA.

The ACA, by virtue of being written in EXLGUI, is designed to be end-user extendible. This feature allows individual end-users to add unlimited capabilities beyond those currently included with the ACA. With the ACA, end-users are given all the tools they need to customize the ACA environment to meet their site-specific requirements. End-user modifications can range from simple (e.g., overwriting the equation to calculate one parameter) to more difficult (e.g., from adding a new control technology, to redefining an emission unit type, to adding a whole new class of data types, to adding new analytical features). End-user extensions would, by definition, be the intellectual property of the person/organization that made the extension. This leads to an exciting opportunity for third-party extensions that could potentially remain in-house, be sold, be submitted to the EPA for potential incorporation into the core ACA, or be freely distributed as

the owner(s) (of the third-party extension) desired. This free-market opportunity will help to insure that the ACA remains on the leading edge of technology in the future.

## **RESULTS**

The User Guide to the ACA (Appendix B of this report) provides a detailed description of the analytical features available in the ACA.

The first non-beta release of the ACA (Version 6.1) was in May 1998. The latest version of the software and user's manual for the ACA can be downloaded from the EPA's Clean Air Technology Center (CTC) World Wide Web site (<http://www.epa.gov/ttn/catc/>) and from the ACA Development Group's World Wide Web site (<http://quattro.me.uiuc.edu/~acad>).

An ACA workshop was held in May 1998 at the US EPA's Environmental Research Center in RTP, NC. Approximately 22 people attended the workshop, including representatives from the US EPA, the US Army, the US Air Force, Environment Canada, and US industry. The workshop provided the ACA development group with the first opportunity to fully demonstrate the ACA. Appendix C of this report provides the training materials that were developed for and used during the ACA workshop.

The ACA is currently being used as a teaching tool at the University of Illinois at Urbana-Champaign in the Civil and Environmental Engineering Department course entitled "Air Resources Engineering." Students in the course use the ACA with homework assignments and as part of a final class project.

### **US EPA Technology Transfer**

In May 1999, the US EPA's Office of Air Quality and Planning Standards (OAQPS), Innovative Strategies and Economics Group (ISEG) began to help support and maintain the ACA. The ISEG, which provides economic analysis for the Office of Air and Radiation (OAR) for the majority of its air regulations, considered the ACA to be an important tool for both the regulators and the regulated.

With the "adoption" of the ACA, ISEG takes on the tasks of distributing the software and documentation, providing the technical focus for future development, and funding future development. The ISEG plans to fund development in the following areas: (1) customizing the ACA to meet EPA requirements, (2) making the ACA more intuitive and user friendly, and (3) incorporating additional analytical modules and data. The EPA plans to host the ACA and support the distribution, as well as provide the technical focus for future development. Finally, the EPA may potentially expand the scope of the ACA to include other types of data and analytical capabilities, such as addressing multi-media issues.

While the EPA is currently making the ACA software and user guide available at the CTC Web site, soon it will be moving to the ISEG's Web site (which is currently being developed). The ACA will have its own Web page, where software and documentation will be available, as well

as additional support-type information, such as a "frequently asked questions" section. Finally, while there is nothing currently set, the US EPA is laying the groundwork for additional vehicles of technology transfer, including seminars and future workshops.

### **US DoD Technology Transfer**

To help maximize technology transition and use by military users, it was determined that the ACA should be able to interface with existing, related systems, and that APIMS had strong support and wide distribution, and would serve well as the first interface with the ACA. Therefore, a SERDP-sponsored study was done to determine the feasibility of establishing an interface between the ACA and APIMS. Such an interface would greatly enhance the technology transition effort for the ERPM project (i.e., ACA) as well as provide the APIMS model with a comprehensive set of analytical capabilities that are an important element of assessing air pollution compliance strategies. The feasibility study was completed and it was determined that such an interface would be possible. A simple, yet functional, prototype interface was developed as part of the feasibility study. This prototype interface allows the APIMS model to execute the ACA in batch mode and then import the ACA reports. Appendix D of the ACA User Guide (Appendix B of this report) provides a more comprehensive explanation of how the prototype interface works. Depending upon the feedback obtained from the APIMS developers and/or user community other ACA/APIMS interface options could easily be developed.

### **SUMMARY/CONCLUSION**

The ACA is an innovative, state-of-the-art tool that can help environmental managers deal with the complicated issues surrounding air pollution compliance. The ACA offers a unique array of data management and analytical capabilities as well as the ability for the end user to extend and modify program functionality. In its current state of development, it is anticipated that the ACA will be especially useful to industry in selecting cost-effective control technologies that will help them meet the regulatory requirements under which they must operate. The ACA's ability to estimate and/or manage all of the actual and potential emissions data that are frequently required to meet various permitting requirements (e.g., Title V Operating Permits) will also be of immediate benefit to industry. Since the ACA can be extended and/or modified to meet individual industrial users needs, the usefulness and functionality of the program is not limited to its current design.

Regulators will also find that the ACA can serve as a useful tool to meet their needs. For the most part, regulators need the same data management and analytical capabilities that the ACA offers to industrial sources. The ACA's key capabilities (e.g., analyze control technologies, analyze emissions) are needed by regulators to verify the data, assumptions, and calculations that sources have used in developing a permit application. Regulators can also use to ACA to assist smaller industrial sources in meeting standards and outlining permit conditions. Regulators may also find the ACA's end-user extensibility features useful to develop or automate a new analysis (e.g., evaluate a new or emerging control technology). When both the regulated and the regulator

are using the ACA, the process of developing a cost-effective permitting approach that is acceptable to both can be expedited.

With the US EPA's Office of Air Quality and Planning Standards (OAQPS), Innovative Strategies and Economics Group (ISEG) "adoption" of the ACA the continued development has been secured.

Technology transfer within the US DoD requires some additional effort. A more complete integration with either the APIMS model or another commonly used relational database will help to ensure the work done on this project profits the US DoD.

## **APPENDIX A: EXLGUI PROGRAMMERS' MANUAL**

## **APPENDIX B: THE ACA USER GUIDE**

## **APPENDIX C: ACA WORKSHOP/TRAINING MATERIALS**

## **APPENDIX A: EXLGUI PROGRAMMERS' MANUAL**

**EXLGUI Programmer's Manual**

**VER. 3.1**

**May, 1998**

**Fred Wasmer & Dan Maloney**

**Department of Mechanical and Industrial Engineering**

**University of Illinois at Urbana-Champaign**

**1206 West Green Street**

**Urbana, IL 61801**

**May, 1998**

# **EXLGUI Programmer's Manual**

**VER. 3.1**

**May, 1998**

For more information about EXLGUI contact Fred Wasmer or Dan Maloney at:

Fred Wasmer:

Department of Mechanical and Industrial Engineering  
University of Illinois at Urbana-Champaign  
1206 West Green Street  
Urbana, IL 61801

phone: (217) 352-9617  
fax: (217) 244-4416  
email: wasmer@wasmerconsulting.com  
www: <http://www.wasmerconsulting.com/>

Dan Maloney:

Department of Mechanical and Industrial Engineering  
University of Illinois at Urbana-Champaign  
1206 West Green Street  
Urbana, IL 61801

phone: (217) 244-6808  
fax: (217) 244-4416  
email: danm@detech.net  
www: <http://www.detech.net/>

# Contents

---

<b>CONTENTS</b> .....	<b>2</b>
<b>1) INTRODUCTION</b> .....	<b>5</b>
INTRODUCING EXLGUI .....	5
A WORD ABOUT THE AIR COMPLIANCE ADVISOR .....	5
SHOULD YOU USE EXLGUI? .....	6
EXLGUI SUMMARY FEATURES .....	6
ABOUT THIS MANUAL.....	7
<b>2) EXLGUI BASICS</b> .....	<b>8</b>
COMMENTS .....	8
GRAMMATICAL RULES .....	8
DECLARATIONS .....	8
CONSTANTS <i>CONST</i> .....	8
VARIABLES <i>VAR</i> .....	9
A WORD ABOUT UNITS .....	9
BASIC EXLGUI SYMBOLS .....	9
<b>3) KEY COMPONENTS OF EXLGUI</b> .....	<b>11</b>
LISTS .....	11
DATA TYPES .....	12
CLASSES AND INHERITANCE.....	12
<i>Classes</i> .....	12
<i>Inheritance</i> .....	13
MEMBER SLOTS .....	14
PROCEDURES .....	15
<i>global procedures</i> .....	15
<i>member procedures</i> .....	16
<i>Local procedures</i> .....	16
FUNCTIONS .....	17
<i>global functions</i> .....	17
<i>member functions</i> .....	17
<i>imbedded functions</i> .....	18
ACTIONS .....	19
CONSTRUCTORS .....	19
AUTOMATIC MEMORY MANAGEMENT .....	19
TYPE CASTING .....	19
EXCEPTIONS .....	19
PROGRAMMING WITH UNITS .....	20
<i>Why automatic units conversion?</i> .....	20
<i>How EXLGUI converts units</i> .....	21
<i>Unit conversion database</i> .....	21
<i>Prefixes</i> .....	24
<i>Compatible units</i> .....	25
THE UNITS OPERATOR "" .....	25
<i>Using units</i> .....	26
<i>Constants</i> .....	26
<i>Variables</i> .....	26
<i>Functions and slots</i> .....	27
<i>Math operations</i> .....	28

GRAPHICAL USER INTERFACE TOOLS .....	29
<i>Class Options</i> .....	29
<i>Member Options</i> .....	30
<b>APPENDIX A) GENERAL EXLGUI FUNCTIONS.....</b>	<b>33</b>
<i>Assert</i> .....	33
<i>BeginReport</i> .....	33
<i>CopyInstance</i> .....	33
<i>date.DayOfWeek</i> .....	34
<i>EndReport</i> .....	34
<i>GetChar</i> .....	35
<i>GetCharFromASCII</i> .....	35
<i>InGUIMode</i> .....	35
<i>IsDefined</i> .....	36
<i>LF</i> .....	36
<i>LinkInstance</i> .....	36
<i>list.Add</i> .....	37
<i>list.Delete</i> .....	37
<i>LoadStateFromFile</i> .....	37
<i>New</i> .....	38
<i>Now</i> .....	38
<i>Random</i> .....	38
<i>Randomize</i> .....	38
<i>RandomSeed</i> .....	39
<i>SaveStateToFile</i> .....	39
<i>SetGUIMainWindowTitle</i> .....	39
<i>SetReportTitle</i> .....	39
<i>SetSlot</i> .....	40
<i>string.Delete</i> .....	40
<i>string.Insert</i> .....	40
<i>string.Length</i> .....	40
<i>Time</i> .....	41
<i>Today</i> .....	41
<i>True</i> .....	41
<i>Typeof</i> .....	42
<i>Write, Writeln</i> .....	42
CONTROL FLOW .....	42
<i>case</i> .....	42
<i>foreach</i> .....	43
<i>If</i> .....	43
<i>return</i> .....	43
<i>while</i> .....	43
MATH FUNCTIONS .....	44
<i>Abs</i> .....	44
<i>Ceiling</i> .....	44
<i>Cos</i> .....	44
<i>ArcCos</i> .....	44
<i>CubeRoot</i> .....	45
<i>EvaluateIfPossible</i> .....	45
<i>Exp</i> .....	45
<i>False</i> .....	46
<i>Floor</i> .....	46
<i>Ln</i> .....	46
<i>Log10</i> .....	46
<i>Max</i> .....	47
<i>Min</i> .....	47

<i>Pi</i> .....	47
<i>Round</i> .....	47
<i>Sin</i> .....	47
<i>ArcSin</i> .....	48
<i>Sqr</i> .....	48
<i>Sqrt</i> .....	48
<i>ToFloat</i> .....	48
<i>Tan</i> .....	49
<i>ArcTan</i> .....	49
<i>Trunc</i> .....	49
SPECIAL.....	49
<i>OneOf( string list )</i> .....	49
<i>Uses</i> .....	50
<b>APPENDIX B) AVAILABLE ICONS IN EXLGUI</b> .....	<b>51</b>

# 1) Introduction

---

## INTRODUCING EXLGUI

EXLGUI is a custom designed, Microsoft Windows-based programming language that is a hybrid of other programming paradigms and styles. EXLGUI combines many features of other programming languages as well as incorporating novel concepts. EXLGUI has been instrumental in the success of the Air Compliance Advisor (ACA), a decision support system developed entirely in EXLGUI. EXLGUI is the ideal tool for developing software to solve engineering/scientific problems. EXLGUI is advantageous in that it can be used to: (1) organize the vast amount of heterogeneous data that needs to be considered in dealing many engineering problems; (2) organize the complex analyses required to be performed; (3) provide built-in quality assurance of accuracy in algorithm development (through the "programming with units" feature of EXLGUI); and (4) allow for end-user inspection and extendibility of the software.

EXLGUI goes one step beyond conventional programming languages that are currently used to develop engineering/scientific software in that source code can be written with associated units (e.g., meters, feet, hours, lbs, kg, etc.). This unique feature allows code to be developed that does NOT require either the programmer or the user to do any manual units conversion: all conversions are performed automatically. Maintaining consistent units is extremely difficult when developing engineering/scientific software that is more than a few hundred lines long. Regardless of the amount of documentation and attempts at uniformity in dealing with units, this is still a problem that plagues developers of scientific/engineering software. EXLGUI addresses and SOLVES this problem. Programs developed in EXLGUI also allow users to enter data in any combination of units (provided that they match the dimensional units of the required data).

EXLGUI is a by-product of an extensive research and development effort into the area of an optimum programming paradigm to address engineering/scientific problems. This research effort is part of the Air Compliance Advisor project. The development of both EXLGUI and the ACA has been sponsored by the Strategic Environmental Research and Development Program (SERDP), The United States Environmental Protection Agency (USEPA), and the United States Army Center for Public Works. The work on the ACA and EXLGUI was carried out by Wasmer Consulting, D & E Technical, Argonne National Laboratory, the University of Illinois, and the United States Army Construction Engineering Research Laboratory.

## A WORD ABOUT THE AIR COMPLIANCE ADVISOR

EXLGUI is the development tool used to write the Air Compliance Advisor (ACA). The ACA is a state-of-the-art computer-based decision support system designed to aid environmental personnel in developing installation-specific, cost-effective air pollution compliance strategies. The ACA was developed for the United States Department of Defense and United States Environmental Protection Agency. The ACA has the analytical capabilities to evaluate the cost and applicability of VOC, HAP and particulate matter (PM) control technologies as well as the capability to estimate emissions from emission units typically found at DoD installations. The ACA allows users to perform what-if scenario analyses as well as extend the capabilities of the system. The ACA is, by design, extendible (by both the developers and end-users) to address additional analyses and data types, as needed. The ACA is

currently available from the US EPA's Clean Air Technology Center bulletin board as well as the ACA web site <<http://quattro.me.uiuc.edu/~acad/>>. The uniqueness and success of the ACA software can be directly attributed to the fact that it was written in EXLGUI.

## SHOULD YOU USE EXLGUI?

Should you use EXLGUI to meet your engineering/scientific software development needs? This question is a relatively easy one to answer. Consider the following questions. If you answer yes to some of these questions, then there is a good chance that the EXLGUI is a good choice for you and we recommend that you phone or email us to discuss your specific needs.

1) Is the data, required for your proposed software model, more complex than what could be represented on a couple of homogenous tables? Spreadsheets are frequently used by scientist/engineers to developing programs that contain simple data. Unfortunately as these type of models evolve, the required data becomes more complex and the spreadsheet paradigm becomes cumbersome to use. EXLGUI is perfect for dealing with complex data.

2) Are "units" a problem that you encounter in the analyses that you will be required to perform? Small programs with variables that have one-dimensional units (e.g., feet, meters, pounds, dollar) can be modeled with most programming paradigms. Once the programs become larger in size (greater than ~500 lines) and/or variables have more than one-dimensional units (e.g., feet/second, meters<sup>2</sup>/lb-mole, Newton/meters-second<sup>2</sup>) units can become a nightmare. Further, if you want to allow the end-user to enter units into the program in a variety of conventions (e.g., feet or meters or miles or inches), then programs written in conventional programming paradigms quickly get convoluted. EXLGUI allows you to program with associated units, completely eliminating this insidious problem.

3) Would you like your engineers, scientists, and customers to be able to modify both input data and relationships? EXLGUI was designed with this capability in mind. Users can both view and modify all portions of an application written in EXLGUI. Furthermore, a user does not have to be an experienced EXLGUI programmer in order to make changes. Even complete novices can make some changes, such a tweaking individual functions. More experienced user can add entirely new analysis and data types.

## EXLGUI SUMMARY FEATURES

- hybrid programming language that takes the best features of other existing programming languages/styles.
- a new tool for programming engineering and scientific applications.
- an innovative environment for developing flexible software in easy-to-use Microsoft Windows environment.

- a by-product of an extensive R&D effort into the area of an optimum programming paradigm to address engineering/scientific problems.
- spreadsheet-like cells/slots
- procedural programming capabilities
- object-oriented data structures
- build-in Microsoft Windows based graphical user interface
- allows scientist/engineers be able to write, review and/or extend your application independent of a "Windows programmer"
- allows your end-users to be able to write small (or large) extensions to your software
- gives your end-users the capability of automatically playing what-if scenario analyses with your program
- the relationship between data element can be seen from the graphical user interface
- the resultant code does not have to be a "black box" to the engineers/scientist who wrote the specification/equations for the model. Rather, the engineer/scientist (as well as end users if desired) can browse both the data and the analyses that take place in the model to verify model accuracy.

**ABOUT THIS MANUAL** This manual is a tutorial and reference for the computer programming language EXLGUI. The reader is assumed to have a basic knowledge of programming concepts. However, advanced features of the language are explained.

## 2) EXLGUI BASICS

---

Before jumping into the Key Components of EXLGUI (Section 3), it is important to understand more basic components of the EXLGUI programming language. This section is devoted to outlining these basic components of the language.

**COMMENTS** There are two ways to incorporate comments into EXLGUI source code. Using the symbols // is the first way to define a comment line in EXLGUI. This will serve to comment all text on the same line that appears after the symbols //. The second way to comment text is with the use of the symbols (\* and \*). The symbols (\* signify the start of the comment and the symbols \*) signify the end of the comment. This second method of commenting can serve to comment out more than one line.

Below are some examples of commented code:

```
// This line is commented out
(* These two lines
   are also commented out *)
```

**GRAMMATICAL RULES** EXLGUI is case insensitive, which means that upper-case letters are indistinguishable from lower-case letters: "DATA", "Data", and "data" are all considered identical by EXLGUI. The only exceptions are in units (defined in Section 3: Programming with Units) and strings and texts (defined in Table 1).

All statements end with a semicolon, and statements may be arranged in any way: one statement per line, one statement spread over several lines, or several statements combined on the same line.

**DECLARATIONS** Declarations are the smallest components of an EXLGUI program that can be independently compiled.

They are separated from each other by *separators*, which are made by placing five dashes in a row on a line:

```
[declaration]
-----
[declaration]
```

Declarations are all of the various components that make up an EXLGUI source file. They define all aspects of the program, including the operations to be performed and the organization of the data which the operations will act on.

**CONSTANTS CONST** This is the simplest kind of EXLGUI declaration. Const is used to specify a constant value, and can not be changed. Here are some examples.

```
const CodeName = 'Air Compliance Advisor';
```

## VARIABLES *VAR*

Variables will likely be used throughout all programs written in EXLGUI. The operator *var* is used to define a new variable of any data *type*. Variables are similar to constants, except that a variables value can be reset/changed while constants cannot. Data types will be discussed in greater detail in the following sections. Standard, built-in EXLGUI data *types* include:

- integer;
- float;
- boolean;
- string;
- date;
- time;

Below are some examples of using the *var* operator:

```
var volume: float; // the variable "volume" has been initialized as type float
var count: integer;
var CostYear: date;
var check: boolean;
```

## A WORD ABOUT UNITS

One unique feature of EXLGUI is the write code with variables that have associated physical units. While this feature will be discussed in greater detail in later sections, it is important to make note of it here. Floating point numbers, variables (*var*) that are of type *float*, and constants (*const*) can have associated physical units. The associated physical units are specified immediately after the specification of *float* for data types and after the floating point value in equations. The associated physical units must be enclosed with `""`. An example of using associated units is give as:

```
var volume: float "feet^3";
volume := 10.0 "feet^2" * 2.9 "meters"; // units conversion is done automatically
```

## BASIC EXLGUI SYMBOLS

These are a number of symbols used by EXLGUI (some already discussed above) that are used to set variable, make comparisons, etc. These symbols also include math operators, statement punctuation, and other special characters. Table 1 below summarizes some of the basic symbols used in EXLGUI.

Table 1  
Basic EXLGUI symbols

Symbol	Meaning	Examples
<code>:=</code>	assignment	<code>x := 1;</code>
<code>;</code>	end of statement	<code>return x;</code>
<code>=, &lt;&gt;</code> <code>&lt;, &gt;</code> <code>&lt;=, &gt;=</code>	equal to, not equal to less than, greater than less (greater) than or equal to	<code>if (x &lt;&gt; 0) writeln('x is nonzero');</code> <code>1 &gt; 0</code> <code>a &lt;= b</code>
<code>:</code>	type declaration	<code>var x: float;</code>
<code>-----</code>	declaration separator	<code>var n: integer;</code> <code>-----</code> <code>var x: float;</code>
<code>// text</code>	single-line comment	<code>x := 1; // This line sets x to one.</code>
<code>(* text *)</code>	multiple-line comment	<code>(* one or more lines of comments, a column of asterisks is for appearance. Only the parentheses and the asterisks adjacent to them are required. )</code>
<code>'text'</code>	text delineator	<code>writeln('This is a line of output');</code>
<code>"units"</code>	unit delineator	<code>length := 1.0 "m";</code>
<code>- +</code> <code>* /</code> <code>^</code>	addition, subtraction multiplication, division, exponentiation	<code>x := -a/4 + 2*a^2;</code>

### 3) Key Components of EXLGUI

---

There are a number of building blocks of EXLGUI that give it its unique personality:

- Lists
- Data Types
- Classes and Inheritance
- Member Slots
- Procedures (global, member and imbedded procedures)
- Functions (global, member and imbedded functions)
- Actions
- Constructors
- Automatic memory management
- Type Casting
- Exceptions
- Programming with Units
- Graphical User Interface Tools

In this section, we will discuss these key features. These features are presented in a manner that allows each new feature to build on those features that have already been discussed. Some references will be made to the built-in EXLGUI functions that are discussed in detail in Appendix A, so it is advised that if you familiarize yourself with this information at this time.

**LISTS** EXLGUI offers strong support for lists. Using the *list of* operator, a variable can be defined to be list of any valid data type (see next subsection). List elements can be added and deleted from the list using the `list.add` and `list.delete` functions, as shown below,

```
var Numbers_To_Remember: list of integer;
Numbers_To_Remember.Add(1); // list contains: (1)
Numbers_To_Remember.Add(2); // list contains: (1,2)
Numbers_To_Remember.Add(3); // list contains: (1,2,3)
Numbers_To_Remember.Delete(1); // list contains: (2,3)
```

Manipulation of lists with EXLGUI can be an extremely powerful capability. The *foreach* block allows you to loop over a list, one element at a time:

```

foreach Member in TypeList do
    [put any commands here]
end foreach;

```

A specific example of the *foreach* block (with an imbedded *if* block) is given as:

```

foreach element in Numbers_To_Remember do
    if (element = 3)
        writeln('The number 3 is in the list');
    end if;
end foreach;

```

**DATA TYPES** The standard EXLGUI data types, as discussed in the previous section, include: boolean, integer, float, string, time, and date. New data types can also be created by first defining the type using either the *list of* operator or the *oneof* operator. The *list of* operator serves to specify a data type that is itself a list of a certain type, while the *oneof* operator serves to specify a data type that restricts the values to be only one of those specified.

```

type FuelType = oneof(
    'Aviation Fuels',
    'Crude Oil',
    'Diesel Fuel',
    'Distillate Oil No. 2');
-----
type Date_List list of Dates;

```

These custom *types* could then be used as follows:

```

var Dates_To_Remember: Date_List;
Dates_To_Remember.Add('June 10, 1997');
var Fuel_Used: FuelType;
Fuel_Used:= 'Crude Oil';

```

## CLASSES AND INHERITANCE

### Classes

Classes give EXLGUI its object-oriented nature. It is through classes that object-oriented features are implemented in EXLGUI. It is beyond the scope of this programmers manual to explain object oriented programming, but there are numerous

references that cover the theory and the basics of this programming technique. If you are not familiar with this programming style, it is recommended that you refer to external references if you plan to create your own EXLGUI application. If you plan to make modifications to an existing applications (such as the ACA), then this manual treatment of object oriented program may suffice.

Defining a class is a two-step process.

1. Create a class declaration, which lists all of the slots, member functions and member procedures contained by the class.
2. Define each of the slots, member functions, and member procedures as separate declarations.

This is illustrated with an example.

```
class SimpleObject
    procedure Constructor;
    slot Count: integer;
end class SimpleObject;

-----

procedure SimpleObject.Constructor;
    setslot (Count, 0);
end procedure SimpleObject.Constructor;

-----

slot SimpleObject.Count: integer;
    return ?;
end slot SimpleObject.Count;
```

Instances of these classes/objects can be created and manipulated with global function, global procedures or, more commonly, by the build-in features of the EXLGUI's Graphical User Interface.

### Inheritance

As with most object-oriented languages, EXLGUI supports class inheritance. Inheritance is a mechanism that allows a base class to be specialized. The inherited or derived class contains all the procedures, functions, and slots of the base class. The derived class can be further specialized by adding additional procedures, functions, or slots. In addition, the members of the base class can be overridden to fit the needs of the inherited class. The class example above will be continued to show how to create inherited classes.

```
class Inherited_Object (SimpleObject);
    procedure Constructor;
    slot DoubleCount: integer;
end class Inherited_Object;

-----

procedure Inherited_Object.Constructor;
    // In Inherited_Object start the count from 1
    setslot (Count, 1);
```

```
end procedure Inherited_Object.Constructor;
```

```
-----
```

```
slot Inherited_Object.DoubleCount: integer;  
    return 2 * Count;  
end slot Inherited_Object.DoubleCount;
```

## MEMBER SLOTS

All slots are by definition Member Slot. In other words, they must belong to a class. Slots do not take parameters, but do return a value of a specific type. In this manner, they are equivalent to member functions (defined below) with zero input parameters. Slots cannot be accessed directly from anywhere in the program, rather they can only be accessed from in an instance of the class to which they belong. From an instance of the class that they belong, they can either be accessed directly or from another slot, a function, or procedure that belongs to that same class. An example is as follows:

```
class glob;  
    slot Mass: float "kg";  
    slot Volume: float "m^3";  
    slot Density: float "kg/m^3";  
end class;  
-----  
slot glob.Mass: float "kg";  
    return ?;  
end slot glob.Mass;  
-----  
slot glob.Volume: float "m^3";  
    return ?;  
end slot glob.Volume;  
-----  
slot glob.Density: float "kg/m^3";  
    return Mass/Volume;  
end slot glob.Density;
```

EXLGUI is designed to provide spreadsheet-like functionality. When an EXLGUI program is run in the GUI environment, data can be displayed and entered in the form of slots, as is done in a spreadsheet. Slots are typically displayed as boxes which contain room for a numerical or text value. The slot may have a predefined equation whose value is generally displayed, or the slots equation may be overridden by the user. As the slots are one of the main features of the EXLGUI that are displayed to the end user, there are special attributes (member option) which can affect how the slots are displayed (see GRAPHICAL USER INTERFACE TOOLS, Member Options for more detail).

There is one important difference between slots and functions. Functions are evaluated when they are called, no matter what. Slots, on the other hand, store their

value the first time they are called, and simply retrieve the stored value after that. Transparent to the user, each slot maintains a list of other slots that depend on it. If a slot is changed, it automatically erases the stored values for any other slots that depend on it.

For example, suppose that we called glob's MassDensity slot the first time. The expression in the return statement would be evaluated, the value stored, and the value also returned. If we called MassDensity a second time, the stored value would be returned. Now suppose that the value of the Volume slot changed. This would result in MassDensity's stored value being disposed of, since MassDensity depends on Volume. It would need to be recomputed the next time MassDensity was called.

The relationship between slots is similar to the relationship between cells in a spreadsheet. Object-oriented programming with slots, as opposed to the more traditional member fields and methods, allows the programmer to do object-oriented programming using the spreadsheet as a mental model. Using slots speeds up the program because retrieving a stored value is much quicker than computing it every time it is needed.

**PROCEDURES** A procedure performs some kind of operation. It can take any number of parameters (including zero parameters), but cannot return a value. Procedures can be global, they can belong to a class, or they can be imbedded into an action, a slot, a function or another procedure. Examples of each of these types of procedures will be presented below.

### **global procedures**

Global procedures are procedures that can be called from any location in the program. They do not belong to a specific class. Here are a couple of examples of a global procedure, first a simple global procedure:

```
procedure WriteMe (S: string);  
    writeln ('The string is ', S);  
end procedure WriteMe;
```

Next, a more complex global procedure (from the ACA):

```
procedure ReportEmissionData(chem: ChemicalObject;  
    emissions: float "lb/hr"; Ref: string; Rating: string;  
    factor: string);  
    writeln(' -----');  
    writeln(' For Chemical/Pollutant: ',  
chem.GetInstanceName);  
    if( factor <> 'None')  
        writeln(' Emission Factor is ', factor);  
    end if;  
    writeln(' Emission Rate is: ',  
CheckNA_MassPerTime(emissions));
```

```

        writeln('      Emission Factor Rating is: ', Rating);
        writeln('      Emission Factor Reference is: ', Ref);
    end procedure ReportEmissionData;

```

### member procedures

Member procedures, by definition, belong to a class and are similar to global procedures, except that they cannot be accessed from anywhere in the program. These procedures can only be accessed from in an instance of the class to which they belong. From an instance of the class that they belong, they can either be called directly or from another procedure, slot or function that belongs to that same class. Constructors (described below) are a prime example of a member procedure. An example is as follows:

```

class dollar_type
    slot year : integer;
    procedure setYear(y : integer);
end class;
-----
procedure dollar_type.PrintYear(y : integer);
    writeln('The Year for cost estimate: ', y);
end procedure;
-----
// this procedure can only be called from an instance
// of this class. Suppose "adollar" is an instance of
// the "dollar_type" class, then
adollar.setYear(1995); // is a valid procedure call

```

### Local procedures

Procedures can be local or imbedded in another procedure, function, action or slot. These imbedded procedures are similar to the other types of procedures, but they can only be called from within the procedure, function, action or slot in which they are imbedded. Note that the imbedded procedure will have direct access parameters that are passed to procedure or functions in which they are imbedded. - An example of this type of procedure is the procedure selective\_write which is imbedded in the global procedure WriteMe:

```

procedure WriteMe (S: string);
    procedure selective_write
        writeln(S);
    end procedure;

```

```

    if (S = 'Hello')
        selective_write
    elseif (S = 'Hi')
        selective_write
    end if;

end procedure WriteMe;

```

**FUNCTIONS** A function returns the result of some kind of operation. It can take any number of parameters (including zero parameters) and returns a specific data type. Function, like procedures, can be global, they can belong to a class, or they can be imbedded into a slot, an action, another function or a procedure. Examples of each of these types of functions will is presented below.

### global functions

Global functions are functions that can be called from any location in the program. They do not belong to a specific class. Here are some examples of a global function:

```

function AddStrings (S1: string; S2: string): string;
    return S1 + S2;
end function;

```

-----

```

function IsBiggerThanSeven (Value: integer): boolean;
    return (DivideBySeven (Value) > 7.0);
end function IsBiggerThanSeven;

```

-----

```

function DivideBySeven (X: integer): float;
    return (X/7.0);
end function DivideBySeven;

```

### member functions

Member functions, by definition, belong to a class and are similar to global functions, except that they cannot be accessed from anywhere in the program. These functions can only be accessed from in an instance of the class to which they belong. From an instance of the class that they belong, they can either be called directly or from another function, a slot or procedure that belongs to that same class. An example is as follows:

```

class dollar_type
    slot year : integer;
    function Next_Year(y : integer): integer;
end class;

```

-----

```

function Next_Year(y : integer): integer;

```

```

    return y + 1;
end function;
-----
// this function can only be called from an instance //
// of this class. Suppose "adollar" is an instance of //
// the "dollar_type" class, then
var A_Year_From_Now: integer;
A_Year_From_Now := adollar.Next_Year(1995); // will be
                                           // set to 1996

```

### imbedded functions

Functions can be local or imbedded in a procedure, another function, an action or a slot. These imbedded functions are similar to the other types of functions, but they can only be called from within the procedure, function, action or slot in which they are imbedded. Note that the imbedded functions will have direct access parameters that are passed to procedure or functions in which they are imbedded. Further note that all imbedded functions (and procedures) must be declared at the beginning of the action, procedure, function or slot in which they are imbedded. An example of this type of function is the function testit which is imbedded in the global procedure WriteMe:

```

procedure WriteMe (S: string);
    function testit : boolean;
        if (S = 'Hello')
            return true;
        elseif (S = 'Hi')
            return true;
        else
            return false;
        end if;
    end function;

    if (testit)
        writeln(S)
    end if;

end procedure WriteMe;

```

**ACTIONS** Actions are similar to global procedures, except that they do not take any input parameters. Action are generally used to control major operations in the system. All of the analyses that are done from the GUI's pull-down Analyses menu bar will call actions.

**CONSTRUCTORS** Constructors are essentially member procedures that are automatically called when an instance of a class is initialized with the *new* operator.

**AUTOMATIC MEMORY MANAGEMENT** EXLGUI automatically takes care of all memory management issues.

**TYPE CASTING** Sometimes a derived object's type is only specified as its base class, but the situation requires the information in the inherited class. While it is usually best to avoid these situations, the EXLGUI language provides a way around them. To understand the EXLGUI casting operator, consider the following classes.

```
class Animal
  slot Mass: float "kg";
end class Animal;
-----
class Bear(Animal);
  slot color: colortype;
end class Bear;
```

Now consider the case of a global procedure that takes a list of animals and prints the color of each bear in the list. Each animal in the list may or may not be a bear. We can use the Typeof operator to determine this. Even though an animal may be identified as a bear, the program still only identifies each item in the list as an animal. This problem can be resolved by using the casting operator in EXLGUI.

```
procedure Bear_Color(Animal_List: list of Animal);
// This procedure writes the color of each bear in
// the animal list
foreach beast in Animal_List do
  if(beast typeof Bear)
    var Yogi: Bear;
    Yogi:= cast(bear, beast);
    write ('There is a ');
    write (Yogi.color);
    writeln(' bear in the list');
  end if;
end foreach;
end procedure Bear_Color;
```

Notice that only animals in the list that are truly bears can be cast in this manner. A general animal object or any other animal cannot be cast into a variable of type Bear.

**EXCEPTIONS** EXLGUI has an exception-handling facility based on the exception handling of C++.

This is the syntax:

```
try
  statements
```

```

    catch(type of exception);
        error-handling commands
    end try;

```

The try command signals the beginning of exception handling, and the catch is the end. If an exception occurs between these two statements, the block of statements between the catch and the end are executed. Generally these statements inform the user of the error or redirect control of the program.

## PROGRAMMING WITH UNITS

EXLGUI has the unique ability to manipulate physical units along with numerical values. This is extremely valuable in scientific and engineering applications, where many physical quantities have units. A quantity with units is represented by a numerical value followed by a combination of names and/or abbreviations of the units. The units are always enclosed in double quotation marks.

The EXLGUI programming language supports automatic physical units conversions. Constants, variables, member slots and function return values can be declared with an associated set of physical units (for example, "meters/second" or "kg/hr"). The compiler will then automatically make conversions as needed. Therefore, the programmer doesn't have to worry about whether all the variables used in an equation have matching units. Of course, the equations will still need to be dimensionally correct (e.g., when setting a variable that is float in units of "m/s", then the resultant units of the equation must be in distance/time). If the equation is not dimensionally correct, then the EXLGUI will report an error at compile time.

### Why automatic units conversion?

When writing a large program that models some aspect of the physical world, it is often difficult to reconcile the physical units used in various parts of the program. For example, suppose that we have two functions. The first, DeltaTime, returns an elapsed time in seconds. The second, DeltaDistance, return a distance traveled in feet. We wish to print the value implied by this time and distance. In most programming languages, we would write something like this.

```

    writeln ('Speed = ',
            ConvFtSecToMPH * (DeltaDistance/DeltaTime),
            'miles per hour');

```

The units conversion constant ConvFtSecToMPH must be explicitly defined by the programmer. Computing such constants has been shown to be quite error-prone. There is also a program maintenance problem. If we later decide that DeltaDistance should return its value in meters instead of feet, we would have to go throughout the entire program, inspecting each use of DeltaDistance and possibly modifying conversion constants.

One alternative is to use a standard set of units throughout the program (for example, MKS, or CGS), but this also has its share of problems. In a program such as the ACA, it is expected that equations will be taken from a variety of technical references. If the programmer has decided to use all MKS units, then whenever an equation in English units is entered into the system, the programmer will need to convert every term in the equation to MKS.

With EXLGUI, conversion of units is done automatically. Wherever a float data type is used in the language, an optional set of units may be specified, enclosed in double

quotes ". In the above example, the function DeltaDistance would be declared like this:

```
function DeltaDistance: float "feet";  
// ...rest of declaration  
  
function DeltaTime: float "second";  
// ...rest of declaration
```

The EXLGUI compiler would then know that the result of the division DeltaDistance / DeltaTime has units of feet per second. To print the results in miles per hour, the following line would be used.

```
writeln ('Speed = ',  
        (DeltaDistance/DeltaTime) "mph" );
```

The programmer does not need to manually insert and maintain the conversion factor. EXLGUI also prevents unit mismatch errors by checking all units when the program is compiled. For example, suppose that an attempt was made to compile the following line.

```
writeln ('Speed = ', (DeltaDistance/DeltaTime)  
        "kg/hr" );
```

The compiler would issue this error message:

```
Units mismatch, cannot convert "distance second^-1" to  
"mass time^-1"
```

### How EXLGUI converts units

When an EXLGUI program is compiled, the compiler accesses a database that describes which sets of units are to be used, as well as how to convert between compatible units. Any two units which do not have conversion factors are incompatible, so quantities measured in one set of units cannot be converted into a value in the other.

### Unit conversion database

To determine what unit conversions are allowed, the EXLGUI compiler accesses an external database of unit conversion information. - an understanding of this database is not strictly necessary for using automatic units conversion, however, it will provide a good basis for comprehending the remainder of this chapter.

The units database consists of the three sections.

**Base units** Every consistent set of units must be reducible to an set of base units, each of which defines a fundamental physical quantity. The set of base units used by EXLGUI is presented in Table 2.

Table 2  
Base physical units

Quantity	Primary Unit	Other Acceptable Units
<b>Length</b>	meter	meters, m, Meters
	foot	feet, ft, Feet
	inch	inches, in, Inches
	yard	yards, yd, Yards
	mile	miles, mi, Miles
	micron	microns
<b>Mass</b>	angstrom	angstroms, Angstrom, Angstroms, A
	gram	grams, g, Grams
	pound	pounds, lb, Pounds
	grain	grains, gr
	ton	tons, t
	<b>Time</b>	second
minute		minutes, min, mins, Minutes
hour		hours, hr, hrs, Hours
day		days, Days
week		weeks, wk, Weeks
year		years, yr, Years
<b>Temperature</b>	kelvin	Kelvin, K, kelvins, Kelvins
	rankine	Rankine, R, rankines, Rankines
	C*	
	F*	
<b>Plane Angle</b>	radian	radians, Radians
	degree	degrees, deg, Degrees
	revolution	revolutions, rev, Revolutions, Revs
<b>Area</b>	acre	acres, Acres
	Hectares	

Quantity	Primary Unit	Other Acceptable Units
<b>Volume</b>	liter	liters, l, Liters
	gallon	gallons, gal, Gallons
	pint	pints, Pints
	quart	quarts, Quarts
	cf	ft <sup>3</sup>
<b>Velocity</b>	knot	knots, Knots
	mph	
<b>Frequency</b>	hertz	Hertz, hz, Hz
<b>Force</b>	newton	newtons, Newton, Newtons, Neutons, N
	dyne	dynes, dyn, Dynes
	lbf	
<b>Energy</b>	joule	joules, Joule, Joules, J
	WH	Watt-hours
	erg	ergs, Ergs
	btu	btus, Btu, BTUS, BTUs, btu
	calorie	calories, cal
<b>Power</b>	watt	watts, Watt, Watts, W, w
	horsepower	hp, BHP, Horsepower, horsepower
<b>Pressure</b>	pascal	pascals, Pascal, Pascals, Pa
	bar	bars, Bars
	atmosphere	atmospheres, Atmosphere, Atmospheres, atm
	mmHg	mmMercury
	torr	torrs, Torr, Torrs, torr
	inchwater	incheswater, inches_water, inwater, in_water, inchwater
	footwater	feetwater, footwater, ftwater, ft_water, feet_water
<b>Value</b>	dollar	dollars, DOLLAR, DOLLARS
<b>Volume/Time</b>	cfm	cubic (foot)/min, CFM

Quantity	Primary Unit	Other Acceptable Units
<b>Current</b>	ampere	Ampere, Amperes, Amp, amp, amps, Amps
<b>Voltage</b>	volt	V, Volt, Volts, volts
<b>Amount of Substance</b>	mole	gmole, gmmole
	lbmole	
<b>Luminosity Candela</b>		
<b>Concentration</b>	ppm	
<b>Relative Amount of Substance</b>	percent	

\* Not allowed in programming, only data entry.

### Prefixes

It is customary, especially in the metric system of units, to modify a unit name with a prefix, such as "*kilo*" or "*milli*". The unit conversion database contains a section where prefixes are specified. The prefixes in the default EXLGUI units conversion database are listed in Table 3.

Table 3  
Unit prefixes

Name	Abbreviation	Value
atto	a	10 <sup>-18</sup>
femto	f	10 <sup>-15</sup>
pico	p	10 <sup>-12</sup>
nano	n	10 <sup>-9</sup>
micro	u	10 <sup>-6</sup>
milli	m	10 <sup>-3</sup>
centi	c	10 <sup>-2</sup>
deci	d	10 <sup>-1</sup>
deca	---	10 <sup>1</sup>
hecto	h	10 <sup>2</sup>
kilo	k	10 <sup>3</sup>
mega	M	10 <sup>6</sup>
giga	G	10 <sup>9</sup>
tera	T	10 <sup>12</sup>
peta	P	10 <sup>15</sup>
exa	E	10 <sup>18</sup>

### Compatible units

Every set of units has an associated dimension. A dimension is simply the integer power to which each of the base unit quantities in the unit is raised.

*Non-integer dimensions (such as length<sup>0.5</sup>) are not allowed.*

For example, the dimensions of both the units “meter” and “foot” would be *length<sup>1</sup>*. The dimension of a measure acceleration, such as “meters/second<sup>2</sup>”, would be *length<sup>1</sup> time<sup>-2</sup>*.

Two sets of units are said to be convertible if a value expressed in one can be converted into a value expressed in the other by multiplying by a scaling factor. For two sets of units to be equivalent, their dimension must match exactly.

For example, “meters/second” is convertible to “miles/hour”, since they share the dimensions of *length<sup>1</sup> time<sup>-1</sup>*. However, “meters/second” is not convertible to “kilograms/hour”, since the second set of units dimension, *mass<sup>1</sup> time<sup>-1</sup>*, does not match the first set of unit’s dimension.

If an attempt is made to convert between incompatible units in an EXLGUI program, an error is generated when the program is compiled.

### THE UNITS OPERATOR ""

The units conversion operator is a set of double quotes “” enclosing text that describe the units being declared. This text consists of one or more unit names (as defined in the unit conversion database), separated by unit operands that define how the unit names related to each other.

A blank space between two adjacent unit names implies multiplication.

For example, in the set of units "kg m/s<sup>2</sup>", the unit names are kg, m, and s. The unit operators are /, ^2, and the space between kg and m. Table 4 gives a complete list of the unit operators.

Table 4  
Unit operators

Operator	Associativity	Meaning
( )	---	Grouping
sq, square, cu, cubic	prefix right	Exponentiation
^	infix right	Integer Exponentiation
Space * -	infix left	Multiplication
/	infix left	Division

Note that in units, multiplication has a higher precedence than division. Therefore, "a b / c d" is the same as "(a b) / (c d)". This allows us to write units in a more natural form. Also, any embedded periods '.' are ignored, so "m./sec." is equivalent to "m/sec".

Think of indicating that a value is to be squared by placing a small 2 superscript after the item to be squared, like this: X<sup>2</sup>. This is an example of a postfix operator.

The unit operator " " operates on the value preceding it. Therefore, it must appear after the value that it applies to. For example, the units of a floating point literal are indicated like this.

1.234 "m/s"

In EXLGUI, every floating point value is either a pure number, or has some set of associated units. Applying the units operator to a pure number indicates that the number is to be interpreted in that set of units.

### Using units

It is not difficult to use automatic units conversion in your EXLGUI program. To the extent possible, the syntax of using this feature has been designed to mimic the conventional equation notations used in engineering textbooks.

### Constants

Whenever a floating point constant is declared, it can be given units by placing the units conversion operator after the value. The following example should make this clear.

```
const StandardPressure = 1 "atm";
const C1 = 1.234 "dollar/Mw";
```

### Variables

A floating point variable can be given an associated set of units by following the float keyword with the units conversion operator. Consider this example.

```
var Distance1: float "kilometer";
var Distance2: float "meter";
```

```

Distance1 := 1 "km";
// Since Distance2 has units of "meters", the value in
// Distance1 is converted before assignment
Distance2 := Distance1;

writeln (Distance1, ' equals ', Distance2);

```

This would result in the following line being written.

```
1 "km" equals 1000 "m"
```

---

## Functions and slots

Any units associated with the return values for functions and object slots are declared in the same way as for variables.

```

function ComputeDistance: float "meter";
// ...rest of declaration

class MyClass
    slot Power: float "watt";
end class MyClass;

slot MyClass.Power: float "watt";
// ...rest of declaration

```

Parameter types for functions, procedures, and constructors are also declared in an analogous manner. Conversion between formal and actual parameter values for pass-by-value parameters follow the same rules as variable assignment.

```

procedure PrintTime (T: float "seconds")
    writeln( 'T = ', T );
end procedure PrintTime;

procedure PrintTest
    PrintTime (1 "minute");
end procedure PrintTest;

```

A call to the PrintTest procedure would print the following line.

```
T = 60 "seconds"
```

If a parameter is declared as pass-by-reference, then the units of the formal and actual arguments must match exactly.

```

procedure P (var T: float "seconds");
// ...
end procedure P;

var V1: float "sec";
var V2: float "min";
// "seconds" matches "sec" since the conversion factor
// between them is exactly 1.0
P (V1); // Fine; units match exactly
P (V2); // Causes a compile-time error

```

## Math operations

Mathematical operations, such as addition and squaring, are possible on floating point values with units. A series of rules govern what operations can be performed. In general, these rules are what one would expect.

**Addition and subtraction** Two quantities with units can be added or subtracted as long as their units are compatible. If the two units are compatible but not equal, then the result is expressed in the units of the first term.

```
var Dist: float "m";
Dist := 5000 "mm" - 200 "cm";
writeln (Dist); // Prints 3 "m"
Dist := 1 "m" + 1 "kg"; // Compile-time error:
                        // units are not compatible
```

**Multiplication and division** Two values with units can always be multiplied or divided. The units of the result will just be the value of the two units with the integer exponents of each base unit added or subtracted, respectively.

```
var Area : float "square inches";
var Speed : float "miles per hour";
var Length : float "feet";

Area := 10 "meters" * 5 "feet";
Speed := 20 "meters" / 5 "seconds";
Length := Area / 2 "miles";
```

**Exponentiation** A value with units can be raised to any integral power. Also, the square and cube roots of a variable may be taken with the **sqrt** and **cuberoot** functions, as long as it does not result in the final value's units having a base unit raised to a non-integral power. For example, the following is legal.

```
var Area : float "ft^2";
var Volume : float "ft^3";
var Distance: float "meter";

Area := Distance^2;
Volume := Distance^3;
Distance := Sqrt (Area)
Distance := CubeRoot (Volume);
```

However, the following is illegal, since it would result in a value having the units of  $length^{0.5}$ .

```
writeln (Sqrt (Distance)); // Error:
                        // unacceptable units
```

Any attempt to raise a value with units to a non-integral power is will result in an error at compile time.

**Trigonometric functions** The trigonometric functions **sin**, **cos**, and **tan** are all declared to take a floating point value with units of "*radians*". Likewise, the inverse function **arcsin**, **arccos**, and **arctan** all take a pure number as a parameter, and return a result with the units of "*radians*".

```

var Angle: float "degrees";

Angle := 90 "deg";

writeln ('sin (', Angle, ') is ', sin (Angle));
writeln ('arctan (1) is ', arctan (1) "degrees" );

```

This would result in the following lines being printed.

```

sin (90 "degrees") is 1
arctan (1) is 45 "degrees"

```

## GRAPHICAL USER INTERFACE TOOLS

Programs written in EXLGUI are automatically displayed to the end-user in the form of a graphical user interface (GUI). You, as the developer, have some control over how the program will appear as well as (to some extent) how the end-user will be allowed to add/replace data. This control is given to the programmer in the form of *Class Options* and *Member Options*. These two features are discussed in detail below.

If a value or option is not specified by users, then a default value/option will be used. The various defaults for each option are given below.

### Class Options

Class options provide control over how an object or a class will appear in the GUI. The class options that are available include *Label*, *Icon*, *Root*, *UserLevel*, and *Order*. These five class options are described below.

#### Label

**Remarks** Specifies the name of a class as displayed by in the EXLGUI graphical user interface.

**Default** None

**Example** `ClassOption <Label, 'Chemical'>;`

**Associated Parameter Type** Text strings

#### Icon

**Remarks** Specifies the icon that will be used to represent a class or object in the EXLGUI graphical user interface. See Appendix B for list of images.

**Default** A gray square with a black X in it

**Example** `ClassOption <Icon, 'ChemicalObjectIcon'>;`

**Associated Parameter Type** Fixed EXL name, see Appendix B

#### Root

**Remarks** Specifies if an object of this class is considered root-level. Root-level objects are the roots of the object containment hierarchy, as displayed in the left pane of the EXLGUI standard view.

**Default** False

**Example** `ClassOption <Root, True>;`

**Associated Parameter Type** Boolean

**UserLevel**

**Remarks** Specifies at which user level a class will appear in the "Types of Objects" view.

**Default** Novice

**Example** `ClassOption <UserLevel, Intermediate>;`

**Associated Parameter Type** Choices of three: Novice, Intermediate and Expert

**Order**

**Remarks** Specifies the order relative in which classes are displayed in the "Types of Objects" view.

**Default** 0

**Example** `ClassOption <Order, 4>;`

**Associated Parameter Type** Integer

### Member Options

Member options provide control over how a slot will appear in the GUI. Some of the member options also provide some control over the type of data that can be added/replaced in the EXLGUI. The member options that are available include *Label*, *Help*, *Order*, *UserLevel*, *Source*, *Hidden*, *Disabled*, *MinValue*, *MaxValue*, *AddFrom*, *AddFrom\_AllowNew*, *ObjectRef*, and *StartSlotGroup*. These five member options are described below.

#### Label

**Remarks** Specifies the title of the slot when displayed in the EXLGUI graphical user interface.

**Default** the slot name

**Example** `MemberOption <Label, 'Title'>;`

**Associated Parameter Type** Text strings

#### Help

**Remarks** Specifies help information about a slot when the Information (I) button is pressed in the EXLGUI graphical user interface.

**Default** None

**Example** `MemberOption <Help, 'I am the help string'>;`

**Associated Parameter Type** Text strings

### Order

**Remarks** Specifies the relative order in which slots are displayed in the EXLGUI graphical user interface.

**Default** 9999

**Example** `MemberOption <Order, 1>;`

**Associated Parameter Type** Integers

### UserLevel

**Remarks** Specifies at level at which a slot will appear at in the EXLGUI graphical user interface.

**Default** Novice

**Example** `MemberOption <UserLevel, Intermediate>;`

**Associated Parameter Type** Three choices: Novice, Intermediate, Expert

### Source

**Remarks** Specifies if the value for the slot is intended be entered by the user, is calculated or is externally defined. Note that setting this slot does not effect the definition of the slot, just the way it is presented to in the GUI. For example, a slot with a source of Calculation can still be overridden by a user-supplied value.

**Default** User

**Example** `MemberOption <Source, User>;`

**Associated Parameter Type** Slot: User, Calculation, External

### Hidden

**Remarks** Specifies if slot is hidden in the Standard view. When this slot is set to True it will not appear in the Standard view, but it will still appear in the advanced view under Types of Objects. Hidden slots are always evaluated when updating the EXLGUI

**Default** Hidden, false

**Example** `MemberOption <Hidden, true>;`

**Associated Parameter Type** Boolean

### Disabled

**Remarks** Specifies if slot is disabled or not in the Standard view. When this slot is set to True it will appear in the Standard view, but it will not be editable.

**Default** Disabled, false

**Example** `MemberOption <Disabled, true>;`

**Associated Parameter Type** Boolean

### **AddFrom**

**Remarks** Valid only for slots that return either objects or lists of objects. It is an association with another slot that is a list of objects of the same type. The associated slot must be in the same class/object and it can be an empty list. When the user clicks on the Add (+) button in the EXLGUI graphical user interface, a list of available objects is displayed in a pop-up window. This list will consists of the objects returned when the AddFrom slot is evaluated.

**Default** None

**Example** `MemberOption <AddFrom, AddFromStreamList>;`

**Associated Parameter Type** a slot name (that is a "list of" of the same type as the slot with this option).

### **AddFrom\_AllowNew**

**Remarks** Specifies whether or not an Add New button will appear on the popup window that allows the user to select from a list of object to add. See member option AddFrom above.

**Default** False

**Example** `MemberOption <AddFrom_AllowNew, true>;`

**Associated Parameter Type** Boolean

### **ObjectRef**

**Remarks** Specifies if an object or a list of objects slot is a "Copy" or a "Link". Objects that are specified to be copies will contain a unique instance of the object while objects that are specified as links will be a "short cut" to a unique instance of the object that resides elsewhere.

**Default** "Link"

**Example** `MemberOption < ObjectRef, Copy>;`

**Associated Parameter Type** "Copy" or "Link"

### **StartSlotGroup**

**Remarks** Specifies that a divider line with associated text should appear in the EXLGUI graphical user interface right above the slot with which this member option is associated.

**Default** None

**Example** `MemberOption <StartSlotGroup, 'Put the divider label text here'>;`

**Associated Parameter Type** Text strings

## Appendix A) General EXLGUI functions

---

EXLGUI has built-in functions for doing various things. Functions and variables are declared by putting the return type after the name, with a colon in between.

### Assert

**Purpose** Verify a condition: throw an exception if it is false.

**Declaration** `procedure Assert (X: boolean);`

**Remarks** This function is intended to be used as a debugging aid in developing EXLGUI programs. It evaluates its single boolean argument: if it is false, assert throws an exception. Assert is intended to be used to verify conditions that should always be true in a correctly functioning program. Consider the example below. The procedure has a single float argument, X. As part of the procedure's body, the square root of X is calculated. Therefore, PerformCalculations should never be called with a negative value of X. This is verified by the assert statement.

**Example** `procedure PerformCalculations (X: float);  
 assert (X >= 0);  
 DoSomething (Sqrt (X));`

### BeginReport

**Purpose** Start collecting information for a report.

**Declaration** `procedure BeginReport;`

**Remarks** Called to begin the creation of an EXLGUI report. Once BeginReport is called, all text output written using Write and Writeln is directed to the report. The report is completed by calling EndReport.

**See also** EndReport, SetReportTitle

**Example** See example associated with EndReport.

### CopyInstance

**Purpose** Returns a copy of an EXLGUI object.

**Declaration** `function CopyInstance (X: <Any Object>): <Any Object>;`

**Remarks** This function returns a copy of an object defined with the `class` command. The copy is a new instance of the same class as the argument. Each of the copy's slots has the same value as the corresponding slot in the argument. Note that while the object itself is copied, objects referenced by the copied object are not copied.

**Example** `var Animal1: AnimalClass;  
 Animal1 := new AnimalClass;  
 var Animal2: AnimalClass;  
 Animal2 := CopyInstance (Animal1);`

## Date

**Purpose** Converts the string argument into a value of type date.

**Declaration** `function Date (DateString: string): date;`

**Remarks** This function takes a date in string form and converts it into a value of type date. The format for DateString should be as in "January 5, 1992". Three-letter abbreviations of the month name are acceptable, as are alternative capitalized forms, e.g. "JAN. 5, 1992" or "Jan. 5, 1992".

Note that dates of the form ##/##/## are not supported, due to the potential confusion as to which century the 2-digit year refers to.

If the actual argument is a string constant, then the conversion to a date type is done when the program compiles, and any errors in DateString's format would be caught. Otherwise, the conversion is done at run time.

**See also** Time, Today

**Example**

```
var D: date;
D := Date ('Dec. 30, 1943');
if (D > Date ('Dec 31, 1950'))
    writeln (D, ' is after 1950.');
```

## date.DayOfWeek

**Purpose** Returns the day of the week that a date falls on.

**Declaration** `function date.DayOfWeek: integer;`

**Remarks** Returns an integer code for the day of the week, where Monday = 1, Tuesday = 2, ..., Sunday = 7.

**Example**

```
var D: date;
D := Date ('30 May 1995');
writeln (D.DayOfWeek);           // Writes 2, the code
                                // for Tuesday.
```

## EndReport

**Purpose** Completes and displays a report.

**Declaration** `procedure EndReport;`

**Remarks** The procedures BeginReport and EndReport are used to collect data for an report. Calling BeginReport begins data collection: , - all text written using Write and Writeln would be collected for the report, instead of appearing on standard output. Other report option procedures, such as SetReportTitle, may also be called. When EndReport is called, all data collected since the last BeginReport is assembled into a report, which is then displayed to the user.

**Restrictions** Must not be called before BeginReport.

**See also** BeginReport, SetReportTitle

**Example** **BeginReport;**  
**Writeln** ('This is the first line of the report.');

```
Writeln ('This is the second line of the report.');
```

**SetReportTitle** ('A Short Report');

```
EndReport;
```

### GetChar

**Purpose** Reads a single character from the keyboard.

**Declaration** **function** GetChar : **string**;

**Restrictions** GetChar is only useful in non-GUI programs.

**Example** **var** TempChar: **string**;

```
// A single keystroke is read.
```

TempChar := GetChar;

```
Writeln('You pressed the ',TempChar,' Key');
```

### GetCharFromASCII

**Purpose** Converts an integer into a string that contains the ASCII character represented by the integer.

**Declaration** **function** GetCharFromASCII (ASCIICode: **integer**): **string**;

**Restrictions** ASCIICode must be between 0 and 255.

**See also** LF

**Example** **var** S: **string**;

```
S := 'abc' + GetCharFromASCII (10) + 'def';
```

```
// 10 is the ASCII code for a newline character
```

**Writeln** (S);

```
// Prints 'abc' and 'def' on separate lines
```

### InGUIMode

**Purpose** Returns true if the currently executing EXLGUI program is being accessed through a graphical user interface (GUI).

**Declaration** **function** InGUIMode: **boolean**;

**Remarks** Returns false if it is being accessed any way other than through a graphical user interface (usually through a text-based user interface).

An EXLGUI program may use this function to control its behavior depending on the way the user is interacting with it. For example, if a text-based interface is being used, a text-based menuing system may be implemented using EXLGUI.

**Example** **if** not InGUIMode

```
ExecuteTextBasedMenuSystem;
```

**end if**;

## IsDefined

- Purpose** Determines whether or not an expression returns a defined value.
- Declaration** `function IsDefined (X: <Any Type>): boolean;`
- Remarks** IsDefined attempts to evaluate its argument *expression* and returns True if *expression* has a defined value. If *expression* is undefined, IsDefined returns False.
- This function takes a single parameter - an expression of any type - and returns true if the expression evaluates to a defined value. False is returned if the expression evaluates to an undefined value (i.e., if evaluating the expression causes an UndefinedValueException to be thrown).
- The actual value that the expression evaluates to is discarded. If this value is of interest, use a try..catch block to wrap up the storing of the value and the undefined test into a single construct.

**See also** EvaluateIfPossible

**Example**

```
function ReturnSeven: integer;
    return 7;
end function;
function ReturnUndefd: integer;
    return ?;
end function;
Writeln (IsDefined (ReturnSeven)); // Prints true
Writeln (IsDefined (ReturnUndefd)); // Prints false
```

## LF

- Purpose** Equal to a string containing the newline character.
- Declaration** `constant LF;`
- See also** GetCharFromASCII
- Example**
- ```
var S: string;
S := 'abc' + LF + 'def';
Writeln (S);
    // Prints 'abc' and 'def' on separate lines
Write ('Hello' + LF); // These two lines
Writeln ('Hello');   // are equivalent
```

## LinkInstance

- Purpose** Returns a link of an EXLGUI object.
- Declaration** `function LinkInstance (X: <Any Object>): <Any Object>;`
- Remarks** This function returns a link to an object defined with the `class` command. The link refers to the original object: thus, a link functions much like a pointer in other programming languages.

**Example** `var Animal1: AnimalClass;  
Animal1 := new AnimalClass;  
var Animal2: AnimalClass;  
Animal2 := LinkInstance (Animal1);`

### **list.Add**

**Purpose** Adds an element to a list.

**Declaration** `procedure list.Add (Element: <ElementType>);`

**Remarks** Adds Element to the end of a list. Duplicate elements are allowed.

**See also** list.Delete

**Example** `var L: list of integer;  
L.add (5);  
L.add (7); // L now contains the elements 5 and 7`

### **list.Delete**

**Purpose** Removes an element from a list.

**Declaration** `procedure list.Delete (Element: <ElementType>);`

**Remarks** The list is searched, beginning to end, for an element equal to Element, as determined by the equality operator =. If there is more than one such element, then only the first one is removed. If Element does not exist in the list, a run-time error occurs.

**See also** list.Add

**Example** `var L: list of integer;  
L.add (1);  
L.add (2);  
L.add (3);  
L.add (2); // L now contains 1, 2, 3, 2  
L.delete (2); // L now contains 1, 3, 2  
L.delete (9); // This causes a run-time error.`

### **LoadStateFromFile**

**Purpose** Reads an EXLGUI program's global data and object instances from a file.

**Declaration** `procedure LoadStateFromFile (FileName: string);`

**Remarks** This procedure can be used to load the state of an EXLGUI program previously saved by a call to SaveStateToFile.

**Restrictions** The specified file must exist, and the contents must be in EXLGUI Command format; otherwise, a run-time error occurs.

Should only be called from inside an action. Since this procedure can result in existing object instances being deleted, calling this procedure from a class member function will produce undefined results.

**See also** SaveStateToFile

**Example** `LoadStateFromFile ('testfile.xml');`

### **New**

**Purpose** New allocates memory for an object, creating a new instance of a class.

**Declaration** `function New: <any type>;`

**Example** `chem := new ChemicalObject;  
class SourceClass  
    procedure constructor;  
end;  
var Source: SourceClass;  
Source := new SourceClass;`

### **Now**

**Purpose** Returns the current time according to the system clock.

**Declaration** `function Now: time;`

**See also** Time, Today

**Example** `if (Now >= Time ('7:00 PM'))  
    writeln ('It is evening.');`  
`end if;`

### **Random**

**Purpose** Returns the random number.

**Declaration** `function Random (Max: integer): integer;`

**Remarks** Returns a random integer drawn uniformly from the range of 0 to Max-1.

**See also** Randomize, RandomSeed

**Example** `var i: integer;  
i := Random (8);       // Sets i to a random number  
                          // between 0 and 7`

### **Randomize**

**Purpose** Sets the random number generator seed to a random value.

**Declaration** `procedure Randomize;`

**Remarks** Sets the random number generator seed to an integer based upon the current system clock time. Calling Randomize will insure that the random number generator will begin in a random state.

**See also** Random, RandomSeed

**Example** `Randomize;`

### RandomSeed

**Purpose** Sets the random number generator seed to the value of the argument.

**Declaration** `procedure RandomSeed (Seed: integer);`

**Remarks** Sets the random number generator seed to a known value. By resetting the random number seed to a previous value, a sequence of random numbers generated by the Random function can be repeated.

**See also** Random, Randomize

**Example** `RandomSeed (57); // Sets random number seed to 57`

### SaveStateToFile

**Purpose** Writes an EXLGUI program's global data and object instances to a file.

**Declaration** `procedure SaveStateToFile (FileName: string);`

**Remarks** This procedure causes all object instances, and the values of all global variables, to be written to the a file named FileName in EXLGUI Command format. The state can later be restored by a call to LoadStateFromFile.

**See also** LoadStateFromFile

**Example** `SaveStateToFile ('testfile.oxl');`

### SetGUIMainWindowTitle

**Purpose** Sets the title of an EXLGUI application's main window.

**Declaration** `procedure SetGUIMainWindowTitle (Title: string);`

**Remarks** Sets the text that appears at the top of the main window when the currently executing EXLGUI program is being accessed through a graphical user interface.

If a graphical user interface is not being used (for example, if the text-based interface is being used), then this function does nothing.

**Example** `SetGUIMainWindowTitle ('EXLGUI Application version 1.0');`

### SetReportTitle

**Purpose** Sets the name of the report currently being created.

**Declaration** `procedure SetReportTitle (Title: string);`

**Remarks** Sets the name of the current report to Title. When the current report is completed with EndReport, the report is displayed.

If SetReportTitle is not called, then the report is given the default title 'Untitled'.

**Restrictions** Must be called after a call to BeginReport, but before the matching call to EndReport.

**See also** BeginReport, EndReport

**Example** See example associated with EndReport.

### SetSlot

**Purpose** Sets a slot to a specified value.

**Declaration** `action SetSlot(slot, expression);`

**Remarks** SetSlot overrides the definition or current value of a slot, and sets the value of the slot to the second argument of SetSlot.

**See also** Reset (?)

**Example** `SetSlot(a_slot, expression);`

### string.Delete

**Purpose** Removes a substring from a string.

**Declaration** `procedure string.Delete (I, N: integer);`

**Remarks** Removes N characters, beginning with the I'th character, from the target string.

**See also** string.Insert

**Example**

```
var S: string;
S := 'ABCDEFGH';
S.delete (3, 2);
writeln (S); // S now equals 'ABEFG'
```

### string.Insert

**Purpose** Inserts a string into another string at a given location.

**Declaration** `procedure string.Insert (I: integer; S: string);`

**Remarks** Inserts string S into the target string. S is inserted just before the I'th character in the target string.

**See also** string.Delete

**Example**

```
var S: string;
S := 'ABCDEFGH';
S.insert (4, 'xyz');
writeln (S); // S now equals 'ABCxyzDEFG'
```

### string.Length

**Purpose** Returns the number of characters in the target string.

**Declaration** `function string.Length: integer;`

**Example** `var S: string;  
S := 'ABCDEFGH';  
writeln (S.Length); // Writes 7  
S := ''; // Sets S to the empty string  
writeln (S.Length); // Writes 0`

## Time

**Purpose** Converts the string argument into a value of type time.

**Declaration** `function Time (TimeString: string): time;`

**Remarks** This function takes a time in string form and converts it into a value of type time. The format for TimeString should be as in "10:52 PM". The PM may have periods, and be in lower case, as in "10:52 p.m.". If AM or PM is not present, the time is assumed to be in military (24-hour) format. In this case, the hour should be between 0 and 23.

Seconds may be optionally specified, as in "10:52:45".

If the actual argument is a string constant, then the conversion to a time type is done at compile time, and any errors in TimeString's format are caught then. Otherwise, the conversion is done at run time.

**See also** Date, Now

**Example** `var T: time;  
T := Time ('8:45 am');  
if (T < Time ('12:00'))  
 writeln (D, ' is in the morning.');`  
`end if;`

## Today

**Purpose** Returns the current date according to the system clock.

**Declaration** `function Today: date;`

**See also** Date, Now

**Example** `if (Today >= Date ('Jan. 1, 2000')) and  
(Today <= Date ('Dec. 31, 2099'))  
 writeln ('We are in the 21st century.');`  
`end if;`

## True

**Purpose** Equal to the boolean value true.

**Declaration** `constant True: boolean;`

**See also** False

**Example** `var Flag: boolean;  
Flag := True; // Sets Flag to true`

## Typeof

**Purpose** Used to determine whether an object's type is the specified type.

**Example**

```
var beast: bear;
if (beast TypeOf Bear)
    writeln('This is a bear');
else
    writeln('This is not a bear');
end if;
```

## Write, Writeln

**Purpose** Prints text to output.

**Declaration** `procedure Write(<list of strings>);`

**Remarks** Write prints the text in its argument to the current output. The text may be in the form of a comma-separated list of items, including both strings and other variable types. Writeln prints a newline character at the end of the output, whereas Write does not. EXLGUI prints its output to files named "report*n*.rpt", where *n* is an integer which is set to 1 when the program starts, and is incremented to the next higher integer each time a new report is requested.

**See also** GetChar

**Example** `write( 'Result = ', result );`

**Control flow** Control-flow commands allow the programmer to specify different operations to be executed depending on the circumstances, or to execute an operation several times.

## case

**Purpose** Executes one of several alternative operations.

**Remarks** Rather than test each possibility separately, as the if statement does, case handles all options in the same block, using the value of a test expression to distinguish between them. Each case may be specified by a single value of the test expression, or by a range of values using the ".." operator.

**See also** If

**Example**

```
var X: integer;
case X is
in ..(-1):
    writeln('X is negative. ');
in 0:
    writeln('X is zero. ');
in 1 .. :
    writeln('X is positive. ');
end case;
```

## **foreach**

**Purpose** Executes a set of operations on all elements in a list.

**Remarks** Foreach searches XList for elements, and operates on each element found by placing its value in X and executing the commands between the foreach statement and the corresponding end statement. Xlist must be the name of a declared list object.

**See also** while

**Example** **foreach** X **in** XList **do**  
    Total := Total + X;  
**end foreach;**

## **if**

**Purpose** Executes operations conditionally.

**Remarks** if evaluates the condition following it, and if the condition is true, executes all commands up to the first else, else if, or corresponding end statement. The else if and else statements are optional. else if may also be written as elseif. In the event that the if condition is false, else if is called and behaves exactly like if. Else executes the commands following it, providing the previous if and else if conditions are false. All if blocks must be terminated by an end statement.

**See also** Case

**Example** **if** (x>0)  
    **writeln**('x is positive');  
  
    **elseif** (x<0)  
        **writeln**('x is negative');  
    **else**  
        **writeln**('x is zero');  
**end if;**

## **return**

**Purpose** Exits the current routine and returns to the point where it was called.

**Example** **function** AdditiveInverse(x: **float**): **float**;  
    Return -x;  
**end function;**

## **while**

**Purpose** Performs a set of operations as long as a condition is true.

**Remarks** while evaluates the condition following it, and executes the commands following the while up to the corresponding end statement, as long as the condition is true. Once the condition becomes false, the program continues at the point following the while block.

**See also** Foreach

**Example**

```
var n: integer;
n := 1;
// print out the powers of 2 up to 100.
while n < 100
    write( n, '...' );
    n := n * 2;
end while;
```

## Math functions

### Abs

**Purpose** Returns the absolute value of the argument.

**Declaration** function Abs (X);

**Remarks** X can be any numeric type. The return type is the same as the type of X.

**Example**

```
var i: integer;
var f: float "m";
i := Abs (-10);           // Sets i to 10
f := Abs (-1.5 "m");     // Sets f to 1.5
```

### Ceiling

**Purpose** Returns the value of the argument rounded to the smallest integer greater than or equal to the argument.

**Declaration** function Ceiling (X: float): integer;

**See also** Floor, Round, Trunc

**Example**

```
var i: integer;
i := Ceiling (1.3);     // Sets i to 2
```

### Cos

**Purpose** Computes the cosine of the argument.

**Declaration** function Cos (Angle: float "radians"): float;

**See also** ArcCos, Sin, Tan

**Example**

```
var f: float;
f := Cos (180 "deg");   // Sets f to -1.0
```

### ArcCos

**Purpose** Returns the inverse cosine of the argument.

**Declaration** function ArcCos (X: float): float "radians";

**Restrictions** X must satisfy  $-1.0 \leq X \leq 1.0$

**See also** ArcSin, ArcTan, Cos

**Example**

```
var Angle: float "radians";
Angle := ArcCos (0.0);      // Sets Angle to  $\pi/2$ 
```

### CubeRoot

**Purpose** Returns the cube root of the argument.

**Declaration** `function CubeRoot (X: float): float;`

**Remarks** If the argument has associated units, then the return type has the cube root of those units. The resulting units must have all integer exponents.

**Restrictions** X must not be negative.

**See also** Sqrt

**Example**

```
var Length: float "m";
Length := CubeRoot (8 "m^3");    // Sets to 2 "m"
```

### EvaluateIfPossible

**Purpose** Attempts to evaluate a boolean expression.

**Declaration** `function EvaluateIfPossible (X: boolean): boolean;`

**Remarks** EvaluateIfPossible attempts to evaluate its argument, which is a boolean expression composed of two or more terms joined with the logical operators **and**, **or**, and **not**. The function returns a true or false value even if some of the expression's terms are undefined.

Normally, if any quantity used by a function evaluates to undefined, then the function returns undefined. However, under some conditions, the value of a logical expression may be independent of one or more of its components. Unlike most functions, EvaluateIfPossible returns the value of its argument even if one or more terms in the expression evaluate to undefined, as long as the value of the expression itself can be determined.

As an example, consider three boolean variables T, F, and U, where T evaluates to true, F evaluates to false, and U evaluates to undefined. If the argument of EvaluateIfPossible is the expression (U or T), the function will return true, since (U or T) evaluates to true regardless of whether U is true or false. On the other hand, EvaluateIfPossible returns undefined if its argument is (U or F), since that expression's value is true if U is true, but false if U is false.

**See also** IsDefined

**Example**

```
B := EvaluateIfPossible (F and U);
// Sets B to false
B := EvaluateIfPossible (T and U);
// Throws an undefined value exception
```

### Exp

**Purpose** Returns the exponential of the argument.

**Declaration** `function Exp (X: float): float;`

**Remarks** Exp calculates the exponent of X, that is,  $e^X$ .  
**Restrictions** x must be a float with no units.  
**See also** Ln, Log10  
**Example**

```
var X: float;  
X := Exp (1); // Sets X to 2.7183
```

### False

**Purpose** Equal to the boolean value false.  
**Declaration**

```
constant False;
```

  
**See also** True  
**Example**

```
var XisPositive: boolean;  
If X <= 0  
    XisPositive := False; // Sets flag to false
```

### Floor

**Purpose** Floor returns the value of the argument rounded down to the largest integer less than or equal to the argument.  
**Declaration**

```
function Floor (X: float): integer;
```

  
**See also** Ceiling, Round, Trunc  
**Example**

```
var i: integer;  
i := Floor (1.7); // Sets i to 1  
i := Floor (-1.7); // Sets i to -2
```

### Ln

**Purpose** Returns the natural logarithm of the argument.  
**Declaration**

```
function Ln (X: float): float;
```

  
**Restrictions** X must be greater than zero.  
**See also** Exp, Log10  
**Example**

```
var X: float;  
X := Ln (1); // Sets X to 0.0
```

### Log10

**Purpose** Returns the base 10 logarithm of the argument.  
**Declaration**

```
function Log10 (X: float): float;
```

  
**Restrictions** X must be greater than zero.  
**See also** Exp, Ln  
**Example**

```
var X: float;  
X := Log10 (100); // Sets X to 2.0
```

## Max

**Purpose** Returns the maximum of the arguments.

**Declaration** `function Max (X1, X2...);`

**Remarks** This function can have any number of arguments. The return type is the type of the first argument, unless a combination of integer and float arguments is given, in which case the return type is float.

**See also** Min

**Example** `var i: integer;  
i := Max (3, 4, 1, 2); // Sets i to 4`

## Min

**Purpose** Returns the minimum of the arguments.

**Declaration** `function Min (X1, X2...);`

**Remarks** This function can have any number of arguments. The return type is the type of the first argument, unless a combination of integer and float arguments is given, in which case the return type is float.

**See also** Max

**Example** `var i: integer;  
i := Min (3, 4, 1, 2); // Sets i to 1`

## Pi

**Purpose** Returns the value of pi ( $\pi$ ).

**Declaration** `function Pi: float;`

**Example** `var Angle: float "degrees";  
Angle := Pi "radians"; // Sets Angle to 180 "deg"`

## Round

**Purpose** Returns the value of the argument rounded to the closest integer.

**Declaration** `function Round (X: float): integer;`

**See also** Ceiling, Floor, Trunc

**Example** `var i: integer;  
i := Round (1.7); // Sets i to 2  
i := Round (-1.7); // Sets i to -2`

## Sin

**Purpose** Returns the sine of the argument.

**Declaration** `function Sin (Angle: float "radians"): float;`

**See also** ArcSin, Cos, Tan

**Example** `var f: float;  
f := Sin (90 "deg"); // Sets f to 1.0`

## ArcSin

**Purpose** Returns the inverse sine of the argument.

**Declaration** `function ArcSin (X: float): float "radians";`

**Remarks** X must satisfy  $-1.0 \leq X \leq 1.0$

**See also** ArcCos, ArcTan, Sin

**Example** `var Angle: float "radians";  
Angle := ArcSin (1.0); // Sets Angle to  $\pi/2$`

## Sqr

**Purpose** Returns the square of the argument.

**Declaration** `function Sqr (X: float): float;`

**Remarks** If the argument has associated units, then the return type has those units squared.

**See also** Sqrt

**Example** `var Area: float "meters^2";  
Area := Sqr (2 "meters");  
// Sets Area to 4 m^2`

## Sqrt

**Purpose** Returns the square root of the argument.

**Declaration** `function Sqrt (X: float): float;`

**Remarks** If the argument has associated units, then the return type has the square root of those units. The resulting units must have all integer exponents, which means all components of the argument's units must be raised to even powers.

Sqrt takes the square root of *arg*, and converts the units if they can be expressed as the square of another set of units. For instance, if *arg* is measured in square meters (m<sup>2</sup>), then Sqrt will return an answer in meters (m). If the units are not reducible, Sqrt returns undefined.

**Restrictions** X must not be negative.

**See also** CubeRoot, Sqr

**Example** `var Length: float "m";  
Length := Sqrt (4 "sq m"); // Sets Length to 2 "m"  
Example: Radius := Sqrt (Area )/[PI or 3.14];`

## ToFloat

**Purpose** Removes any units from a variable of type `float`.

**Declaration** `function ToFloat (expression: <any type>(?)): float;`

**Remarks** ToFloat returns the numerical value *expression* would have if measured in *units*.

*units* must be consistent with the units of *expression*.

**Restrictions** X must not be negative.

**Example**

```

var distance: float "km";
var distMile:float;
distance:= 10 "km";
writeln('Distance= ', distance); // will write out
                                // Distance= 10 "km"
distMile:=ToFloat(distance, "mile");
writeln('Distance (miles): ', distMile);// will write out
                                //Distance (miles): 6.2

```

### Tan

**Purpose** Returns the tangent of the argument.

**Declaration** `function Tan (Angle: float "radians"): float;`

**Restrictions** Angle must satisfy  $-90 \text{ "deg"} < \text{Angle} < 90 \text{ "deg"}$

**See also** ArcTan, Cos, Sin

**Example**

```

var f: float;
f := Tan (45 "deg"); // Sets f to 1.0

```

### ArcTan

**Purpose** Returns the inverse tangent of the argument.

**Declaration** `function ArcTan (X: float): float "radians";`

**See also** ArcCos, ArcSin, Tan

**Example**

```

var Angle: float "radians";
Angle := ArcTan (1.0); // Sets Angle to  $\pi/4$ 

```

### Trunc

**Purpose** Removes the fractional part from a float value and returns the result as an integer.

**Declaration** `function Trunc (X: float): integer;`

**Remarks** If the float is too big for the integer data type, an EXLGUI exception is thrown.

**See also** Ceiling, Floor, Round

**Examples**

```

Capacity := Trunc( SizeOfContainer / SizeOfObjects );
var i: integer;
i := Trunc (1.7); // Sets i to 1
i := Trunc (-1.7); // Sets i to -1

```

### Special

#### OneOf( string list )

**Purpose** Defines the allowed values of an enumerated string variable.

**Remarks** OneOf is used only when declaring a variable as an enumerated string. The allowed values are the strings in the enumerated list. Unlike regular string variables, if a

variable which is a string enumeration is set to a value which is not in the list of allowed strings, the error will be caught by the compiler, and will have to be fixed before running the program.

**Example** `type color = oneof ('red', 'green', blue');`

### **Uses**

**Purpose** Directs the EXLGUI compiler to read a source file.

**Declaration** `function(?) uses ... ;`

**Remarks** `Uses` is found only in the definition file to indicate which source files are part of the program.

**Example** `Uses 'Main.exl';`

## Appendix B) Available Icons in EXLGUI

---

The following is a list of available images that are available in EXLGUI. These images can be associated with classes/objects by using the class option "Icon" in the form:

```
Class ChemicalObject;  
    ClassOption <Icon, 'ChemicalObjectIcon'>;  
    // ...  
    // the rest of ChemicalObject class declaration  
    // ...  
end class;
```

For this example, the an image that represents a chemical would be associated with any instance of a "ChemicalObject" in the GUI.

Note that there is currently not a way for EXLGUI programmers to incorporate their own images (i.e., bitmaps) into an EXLGUI program, but this would be a very simple extension for the developers to make if it is desired. The images listed below are a subset of all of the icons available in creating a custom EXLGUI application. The developers will provide a complete list of all 295 icons upon request.

|                                    |                                                                                       |
|------------------------------------|---------------------------------------------------------------------------------------|
| Chemical_Pollutants_bmp            |   |
| Materials_Library_bmp              |   |
| BioactivityDataIcon                |   |
| Wet_Tubular_ESPs_bmp               |   |
| Venturi_Scrubbers_bmp              |   |
| Miscellaneous_Chemical_Process_bmp |  |
| WasteWaterPlantSourceObjectIcon    |  |

## **APPENDIX B: THE ACA USER GUIDE**

# Air Compliance Advisor User Guide - Version 7.0



US EPA  
Office of Air Quality Planning and Standards (OAQPS)  
Innovative Strategies and Economics Group (ISEG)  
OAQPS/ISEG  
MD-15  
RTP NC 27711

Copyright 2000

## **Disclaimer of Warranty**

THIS SOFTWARE AND THE ACCOMPANYING FILES ARE PROVIDED "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OR MERCHANTABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED OR IMPLIED.

The user assumes the entire risk of using this program.

This software program was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor the University of Chicago, nor the University of Illinois, nor Wasmer Consulting, nor D & E Technical, nor any of their employees or officials, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The view and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

# 1. Introduction

## 1.1 Background

The Air Compliance Advisor (ACA) is a software tool designed to assist on-site air pollution managers in developing strategies for addressing air pollution compliance issues. The ACA utilizes source characterization, emission reduction techniques, permit requirements, and existing air pollution regulations. Results from the multiple analyses that the ACA can perform are presented in a concise, non-biased manner. These results can be used by air pollution managers to address the requirements of the 1990 Clean Air Act Amendments, including:

- Title I (Attainment and Non-attainment)
- Title III (Hazardous Air Pollutants)
- Title V (Permits)
- and other environmental programs and policies.

The ACA is a joint project between Argonne National Laboratory, the University of Illinois, and the US Army (Corps of Engineers) Construction Engineering Research Laboratory (USACERL). The work is sponsored by USACERL, the US Air Force Environics Directorate of Armstrong Laboratory (AL/EQS), the US Army Center for Public Works (CPW), the Strategic Environmental Research and Development Program (SERDP), and the US Environmental Protection Agency (USEPA).

## 1.2 New Features in Version 7.0 of the ACA

This release, Version 7.0 of the ACA, is the second full public release of this program. Users of any of the Version 6.0 releases will find a number of changes, including:

- Wizards
- Formatted Reports

The information contained in **Appendix B — Tips on Obtaining and Estimating Chemical Property Data** and **Appendix D: Generating an ACA Control Device Report from the Command Line** is new in this version of the ACA.

### *Wizards to Streamline the Data Entry Process*

Wizards are now available to streamline the data entry process for three of the most common applications of the ACA: (1) adding volatile organic compounds (VOCs) to the Chemical Database, (2) adding particulate matter (PM) pollutants to the Chemical Database, and (3) entering the data required to evaluate air pollution control technologies. The Wizards greatly simplify data entry by providing easy-to-follow data forms with associated help. Example values are available for some of the data entry cells in order to further assist the user. All three Wizards are available from the main menu options titled Wizards. See Section 5 for more details.

### *Formatted reports*

The output reports of the ACA are now formatted and can be exported to HTML (for web page) or HTML (for word processor input) formats by selecting **File | Export | Report**.

## **1.3 Features Coming Soon in Version 7.1 of the ACA**

Version 7.1 of the ACA, which will be the third full public release of this program, will be available May 2000. Some of the changes that will appear in Version 7.1 are currently being beta tested and appear on a limited functionality basis in this version of the ACA. The changes in Version 7.1 will include:

- Ability to Copy and Paste Objects
- Ability to Export Data in Pieces
- Ability to Import Data Objects
- Sort/Reorganize Slots Based Upon User Preferences
- Capability to Merge Multiple Pollutant Streams

### *Ability to Copy and Paste Objects*

In Version 7.1, users will be able to copy any object in the ACA using either the **Edit|Copy** menu or using the copy toolbar shortcut . Once an object has been copied, it will reside on the clipboard of the user's computer. The copied object can be pasted to an appropriate location in an ACA data structure — either in the ACA file it was cut from or by opening a different ACA file and pasting there. Pasting an object is accomplished by first selecting an appropriate location to add the object contained on the clipboard (note that an object can only be placed in a location that allows for that type of object. If the user is not sure what the object is, they can always save the object in the **Library Data | Miscellaneous Objects Library**, which can accept any object type, and then they can move the object from there as needed). Next, the user will need to select the **Edit|Paste** menu option or use the paste toolbar shortcut .

### *Ability to Export Data in Pieces*

In Version 7.1, the user will be able to save objects (e.g., the ACA Library, a chemical object, a control device) to an external file. The external file will exist in standard ASCII formatted OXL format. Previous versions of the ACA only allowed the user to save the entire data set (i.e., all of the library data, all of the installation data and all of the What-if Scenarios data), which also included any user-supplied data (e.g., emission units at the installation). Saving a portion of the data (i.e., a data object) can be done by selecting the object to export in the **EXLGUI Standard View**, then selecting **File|Export|Object**. The user will then be directed to select a filename for the data. Note that export works like clipboard copy. That is, first you select the object to export or copy, then choose export or copy from the menu.

### ***Ability to Import Data Objects***

In Version 7.1, the user will be able to import an object from an external file (e.g., an object saved via the Export feature). In order to import an object, the user will first have to select either a valid object slot (to replace) or a valid list of objects from the ACA's **Standard View**. Next, the user will have to select **File|Import|Object**. The user will then be presented with an Open File view that will allow them to select the filename that contains the data of interest. An object can only be placed in a location that allows for that type of object. Note, if the user is not sure what the object is, they can always save the object in the **Library Data | Miscellaneous Objects Library**, which can accept any object type, and then they can move the object from there as needed. Note that import works like clipboard paste. That is, first you select the location where you want to import or paste an object, then choose import or paste from the menu.

### ***Sort/Reorganize Slots Based Upon User Preferences***

In Version 7.1, the user will be able to sort/reorganize slots based upon their own preferences with a new menu selection: **Options | Slot Display Order**. The user can then select from the following choices:

- Default Order
- Name
- User Level
- Source

The "Default Order" choice will sort the slots based upon the criteria that has been pre-selected by the developers. The "Name" choice will sort the slots in alphabetical order based upon the slot name/title. The "User Level" choice will sort the slots based upon the user level required to view the slots (i.e., Novice, Intermediate, Expert). The "Source" choice will sort the slots based upon the type of slots it is (i.e., user-defined, functionally defined, or externally defined).

### ***Sort/Reorganize Objects in a List Based Upon User Preference***

The user will soon be able to sort the instances of object in a list (e.g., chemicals in the chemical database) based upon their own criteria. To sort an object list, first select the list in the **EXLGUI Standard View**, then choose **Utilities | Sort Objects** from the menu. The user will then be presented with a dialog box listing the slots of the objects in the list. The user will be prompted to select a slot to sort by. The objects in the list will then be sorted based upon that slot's value in each object.

### ***Capability to Merge Multiple Pollutant Streams***

Users will soon be able to create equivalent stream, based upon the combination of multiple streams, when evaluating control technologies.

## **1.4 User Guide**

This User Guide explains the features of the ACA and gives examples of its use. Note that the different fonts used in this document are selected to mirror the appearance of on-screen text. This User Guide can be viewed directly from the ACA's graphical user interface by selecting **Help | View User Guide**.

## **1.5 Contact Information**

Questions and comments regarding the installation and use of the ACA program should be directed to Dan Maloney at the following address:

Daniel M. Maloney  
D & E Technical  
1008 W. William Street  
Champaign, Illinois 61821  
(217)244-6808 (phone)  
(217)244-4416 (fax)  
dan@detech.net

The ACA software and this User Guide are available from the ACA World Wide Web site: [www.detech.net/ACA](http://www.detech.net/ACA). This site also contains notices on the development of the ACA, papers and reports related to the ACA, and answers to frequently asked questions (FAQ).

## 2. Installing the ACA Program

This chapter identifies the minimum system requirements and describes the procedure for installing the ACA software and loading sample data files.

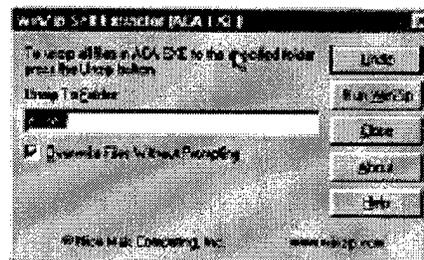
### 2.1 Minimum System Requirements

The ACA software requires the following computer components to run properly:

- IBM PC or compatible computer
- 486 (or later) type processor
- VGA monitor with minimum screen area of 800 by 600 pixels
- Mouse (or other pointing device) supported by MS Windows 95®, MS Windows 98® or MS Windows NT®
- Microsoft Windows 95, Windows 98 or Windows NT
- 9 MB of RAM (minimum)
- 8 MB free hard drive space (minimum)
- Screen area: 800x600 pixels (or greater)

### 2.2 MS Windows 95, Windows 98 and NT Installation

The ACA is distributed as one compressed executable file `aca.exe`. To install the ACA to your computer's hard disk, run `aca.exe` from a temporary directory on the hard disk. The installation program asks for a destination directory as shown in the figure below. In most cases, you will want to select the default choice, `c:\aca`, however, you may select any location or drive. Click the Unzip button and all the necessary files will be expanded into the selected directory.

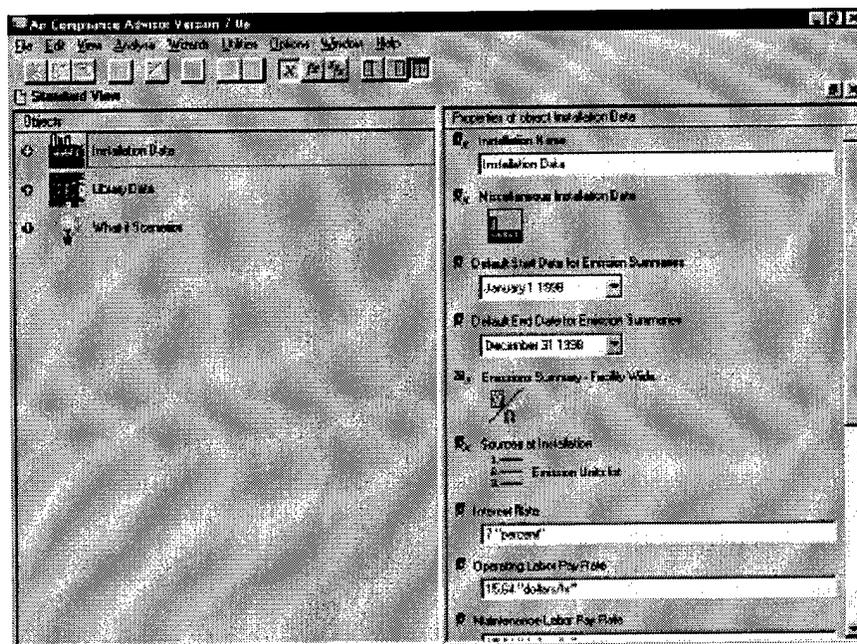


Selecting a destination directory for the ACA

### 2.3 Starting the ACA

Execute the file `c:\aca\exlgui.exe` to start the ACA program. Once the ACA finishes compiling, the ACA's graphical user interface (GUI) will be displayed, as shown in the figure below. The

ACA's GUI contains pull-down menus and toolbar buttons for performing actions. The Standard View features a split screen, which is used for viewing the data and the overall data structure (the object/left pane of the split screen), as well as for entering data (the Properties of Objects/right pane of the split screen). The toolbar mirrors some of the key functions available from the pull-down menus. Toolbar and menu selections will be "grayed out" and unavailable to the user when they are inappropriate for the item highlighted in the active view (e.g., Standard View, report view). Users can find an explanation of each toolbar button by holding the mouse pointer over the button, causing a small "balloon help" window to appear. Note that a program and icon group can be made using the procedure as described in the MS Windows 95®, Windows 98® or NT® User's Manual.



ACA GUI initial screen

## 2.4 Exiting the ACA

To close the ACA, select File | Exit from the main menu.

## 2.5 Bits and Pieces

- The ACA contains a large amount of built-in information known as library data. The chemical, material, and control device information is contained in the core library data file (library.xml). The Regulation, Pollution Prevention, and Suggestions databases are not contained in the core library data file, but can be loaded as needed via File | Load. Any changes a user makes during an ACA session, including changes to the library data, will be stored if the user saves an ACA file using the File | Save or File | Save as functions. However, the original library data will be unaffected and will be available whenever a new ACA session is started.

- Values may be specified in any set of units recognized by the ACA that are dimensionally correct for a particular property. See Appendix A for a list of units recognized by the ACA. The

ACA contains a utility for converting units, available under **Utilities | Units Conversion Utility** in the main menu.

- If insufficient information has been entered to perform an analysis, the analysis ceases and the ACA lists the missing variables.

## 3. Fundamentals of Using the ACA

This section discusses several basic concepts related to using the ACA. This section will outline the user interface, the representation of data, data entry methods, user extensibility, and the standard libraries.

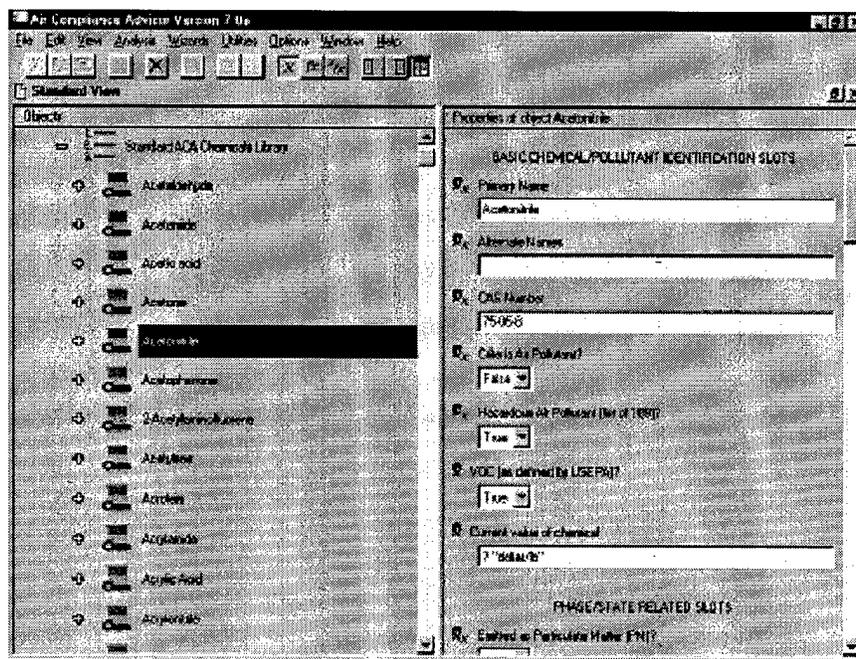
Examples illustrating the use of the system have been included in Section 6 of this manual. The reader should refer to these examples in order to gain additional insight into the information presented in this section. The examples do not present a complete description of all the functions contained in the ACA. This information is best learned through experimentation with the system.

### 3.1 Standard View

The **Standard View** offers a simple interface for viewing and editing information. The figure shown in the previous section, *Starting the ACA*, shows the three core objects of the **Standard View**. Users view and edit installation and air pollution source information with the **Installation Data** object, library data with the **Library Data** object, and air pollution stream data with the **What-if Scenarios** object. Users can then run analyses based on this information and view the results of these analyses in the **Standard View**. More than one **Standard View** can be open at one time. Each time users select **View | Standard View** a new **Standard View** will be opened. Each opened **Standard View** will always contain the same information as every other opened **Standard View**. Opening multiple **Standard View** screens is sometimes helpful for dragging and dropping information between the opened views.

Except for the three core objects discussed above, all other data and data structures are contained in a "slot." A slot can be an object, a list of objects, or a data entry slot. Objects are data structures that typically represent something in the real world (e.g., pollutants, pollution sources, and pollution control devices). Objects can contain data entry slots, lists of objects, and sub-objects. The ACA is made up of a data hierarchy that begins with the three core objects. All other information in the ACA is related to one of the three core objects. The ACA represents these relationships graphically in the **Standard View**.

The **Standard View** is similar to MS Windows Explorer®. There are two sides to the **Standard View**. The left side shows objects or lists of objects that are analogous to the folders found on the left side of Windows Explorer. These objects or object lists can be expanded or contracted to reveal more or less information about an object or list in the same way that folders can be expanded or contracted in Windows Explorer. A plus sign (+) to the left of an object or list indicates that this object or list can be expanded; a minus sign (-) indicates that this object or list has already been expanded; and a small circular bullet indicates that this object or list does not have other objects or lists associated with it at the current user level. The **Standard View** depicts relationships between objects or lists in the left pane by displaying and indenting "child" objects directly underneath the associated "parent" objects. The figure below shows the chemical **Acetonitrile**, which is an object contained in the **Standard ACA Chemicals Library** object list. Notice how all the information is ultimately related to the core object type **Library Data**.



### Standard View

The right side of the Standard View, the Properties of Object pane, shows "slots" that are analogous to folders or files found on the right side of Windows Explorer®. The ACA displays slots associated with objects or object lists selected in the left pane of the Standard View and the title of this pane will reflect the chosen object or object list. These slots contain information that describes the selected object. Slots may be objects, object lists, or data entry slots. Users can view and edit information contained in data entry slots. The figure above shows some of the slots associated with the chemical object Acetonitrile.

The toolbar icons , , and  are used to control and indicate which panes are visible in the Standard View. Users select the leftmost icon to display the left pane only, the middle icon to display the right pane only, or the rightmost icon to display both panes. The selected icon will appear depressed in comparison to the other two icons. In the figure above, both panes (the rightmost icon) of the Standard View are selected.

The user level setting in the ACA affects the appearance of the Standard View. Users may select from three different levels in the Options | User Level menu. These levels vary in degree of detail presented. The ACA starts in the "novice" mode (Options | User Level | Novice). This mode is the easiest to work with because it only presents the data that is frequently required. The novice mode reveals the minimum number of slots required for performing the majority of the calculations. The "intermediate" mode reveals a wider array of slots needed for detailed analyses. The final mode, "expert," allows the user access to all slots.

## 3.2 Slot Types

There are three types of slots in the ACA:

1. User-defined slots
2. Calculation slots
3. External slots

User-defined slots are parameters that typically are either entered by the user or would be information the user might need to change occasionally. For example, when describing a pollutant stream in the ACA the temperature of the stream is a value that the user would normally supply and therefore is a user-defined slot. User-defined slots may contain a default value or a default function that calculates a value. The value may or may not affect other slots in the ACA. In any case, these slots are designated as slots that can be changed by the user.

Calculation slots are parameters that, by default, are calculated by a function. These functions may be dependent on other slots. Users must be very careful about modifying calculation slots since they are not slots that users would typically need to change. Calculation slots often either contain information that is a prime result of an analysis or is a crucial parameter used in the calculation of other slots. The slot "stream gas density," also associated with a pollutant stream, is an example of a parameter that is a calculation slot. The ACA has a built-in algorithm that will calculate the stream density based upon a number of other slots, such as temperature, pressure, pollutant concentrations, etc. If the user overrides this slot with a value, changes to stream parameters used in calculating stream density would no longer affect the value displayed in the stream density slot. Note that once a slot is overridden, with either data or a function, the user can revert back to the default value by using the reset toolbar button (i.e., ) — this will reset a slot to the default function or value.

Finally, external slots are parameters that are typically set by an outside action (e.g., executing one of the "analysis" options from the menu bar) or provided as built-in information (e.g., the Standard ACA chemical library object list). External slots are very similar to user-defined slots in that they can, and in some cases should, be modified. External slots are distinguished from user-defined slots because of the slot's information source rather than the types of actions that can be performed on the slot. There are actually very few data entry slots that are external slots.

Users have the ability of modifying or replacing information found in almost all data entry slots. Those slots that can not be modified are visible, but are inaccessible to the user. Information can be modified or replaced whether slots hold information that is calculated or is entered directly. For example, equations used to size control devices can be changed, as can the equations used to determine the cost of control devices, or the actual emissions of emission units. Users can override any slot type with either a user-supplied value or a user-supplied function. After a slot is overridden, with either data or a function, the user can revert back to the default value by using the reset toolbar button (i.e., ) — this will reset a slot to the default function or value.

With all the possible combinations, the type of data found in a slot can become confusing. Therefore, small icons are used to indicate both the originally defined type of slot, and the type of information that is actually in that slot.

These icons are located to the left of each slot in the Properties of Objects window, and are duplicated below with a brief discussion. Browsing the Properties of Objects window for each of the objects contained in the standard library will conveniently illustrate many of these notations. Holding the mouse over these icons in the Properties of Objects window causes a small "balloon help" window to display a description of the icon.

### *User-Defined Slots*

-  This icon indicates a user-defined slot, set to a default value. A User-defined slot, set to a default value typically starts out undefined since users enter information into these slots. An undefined slot will contain a "?" in place of a value. An example of a slot that would have this icon associated with it is the temperature of a pollutant stream before the user supplies a value for the temperature. Once users enter a value or function, the slot icon changes to one of the following two icons.
-  This icon indicates a user-defined slot, set to user-defined value. The "X" notation that is associated with this icon indicates that the user has entered a particular value for the parameter, and that the parameter is no longer set to the default value (which is typically set as a "?"). Using the example of the temperature slot, this icon would appear when the user specifies a temperature, such as 293 "K". Much of the built-in library information in the ACA is stored in this slot type. Note that you can revert back to the default state of this type of slot by using the reset toolbar option (i.e., ) — in many cases the default value for this type of slot will be undefined or "?".
-  This icon indicates a user-defined slot, set to a user-supplied function. The "f(x)" notation associated with this icon indicates that the value of this slot is calculated by a function supplied by the user. For example, in the materials library users can enter an equation to describe the density of the material (a user-defined parameter) as a function of the properties of the individual components of the mixture. Note that you can revert back to the default state of this type of slot by using the reset toolbar option (i.e., ) — in many cases the default value for this type of slot will be undefined or "?".

### *Calculation Slots*

-  This icon indicates a calculation slot, value computed by default function. As the image of the user is absent from this icon, the function used to define the particular parameter is a default equation contained within the ACA. The user can modify this type of slot by directly entering data or by supplying a function. For example, the mass-based heat of combustion (e.g., "BTU/lb") of a chemical is determined from the volumetric-based heat of combustion (e.g., "BTU/ft<sup>3</sup>") and the vapor density. If the user enters a particular value for the mass-based heat of combustion of a chemical, the icon associated with the slot will change to that for a calculation slot, overridden with a user-supplied value icon. If the user modifies or replaces the equation with a different (user-supplied) equation, then the icon associated with the slot would change to

that for a calculation slot, overridden with a user-supplied function icon. The explanation of the icons associated with each of these cases is illustrated below.

- This icon indicates a calculation slot, overridden with a user-supplied value. As previously discussed, when a slot whose default value is computed by a function, such as a chemical's mass-based heat of combustion, is replaced by the user with a particular value, the definition of the slot changes to this category. This allows the user to determine at a glance which parameters he/she has changed, the default designation of the slots, and their new designation. The user can then determine the effect of any changes he/she may have made on the results of the analyses. However, the red "X" is a reminder that this type of slot is not typically something a user would change and that great care should be exercised when modifying a calculation slot, value computed by default function. Note that you can revert back to the default function/state of this type of slot by using the reset toolbar option (i.e., ).

- This icon indicates a calculation slot, overridden with a user-supplied function. As previously discussed, the value of a calculation slots is, by default, computed by a function (for example, a chemical's mass-based heat of combustion). If this function is modified or replaced by the user, the definition of the slot changes to this category. This allows the user to determine at a glance which parameters he/she has changed, the default designation of the slots, and their new designation. The user can then determine the effect of any changes he/she may have made on the results of the analyses. However, the red "X" is a reminder that this type of slot is not typically something a user would change and that great care should be exercised when modifying a calculation slot, value computed by default function. Note that you can revert back to the default function/state of this type of slot by using the reset toolbar option (i.e., ).

### **External Slots**

- This icon indicates an external slot, set to a default value. External slots are similar to user-defined slots. Like user-defined slots, these slots typically start out undefined because they are typically intended to be set by an external action or calculation and not by a pre-defined equation. Once the value of this type of slot is set, its icon changes to one of the following two icons discussed below. Note that you can revert back to the default state of this type of slot by using the reset toolbar option (i.e., ) — in many cases the default value for this type of slot will be undefined (i.e., "?") or a constant value.

- This icon indicates an external slot, set to a value supplied by either an external calculation or the user. The "X" notation associated with this icon indicates that an external action, or the user, has entered a particular value for the parameter and that the parameter is no longer set to the default value (which is typically set as "?"). Note that the ACA does not distinguish between a slot that is set by an external action and one that is set by the end user. You can revert back to the default state of this type of slot by using the reset toolbar option (i.e., ) — in many cases the default value for this type of slot will be undefined (i.e., "?") or a constant value.

- This icon indicates an external slot, overridden with a user-supplied function. The "f(x)" notation associated with this icon indicates that the user has entered a user-supplied function for an external slot. Note that you can revert back to the default state of this type of slot by using the

reset toolbar option (i.e., ) — in many cases the default value for this type of slot will be undefined [i.e., "?"] or a constant value.

It is important to note that whenever the value of a slot is changed, whether to a particular value or to a function that is dependent on other parameters, all of the parameters that depend on the changed slot will be updated automatically. By way of comparison, one can compare this aspect of user extendibility to the operation of a spreadsheet. When a particular cell is updated, all of the cell values that depend upon the value of that cell will be updated automatically as well.

The table below provides a quick reference to the icons discussed in this section. A more detailed discussion of how users can extend the ACA through the modification of slots is presented in Appendix C.

| Icon                                                                                | Meaning                                                                              |
|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| <b>User-defined slots</b>                                                           |                                                                                      |
|    | User-defined slot, set to a default value                                            |
|    | User-defined slot, set to user-defined value                                         |
|   | User-defined slot, set to a user-supplied function                                   |
| <b>Calculation slots</b>                                                            |                                                                                      |
|  | Calculation slot, value computed by default function                                 |
|  | Calculation slot, overridden with a user-supplied value                              |
|  | Calculation slot, overridden with a user-supplied function                           |
| <b>External slots</b>                                                               |                                                                                      |
|  | External slot, set to a default value                                                |
|  | External slot, set to a value supplied by either an external calculation or the user |
|  | External slot, overridden with a user-supplied function                              |

#### Summary of icon slot definitions

### 3.3 Data Entry

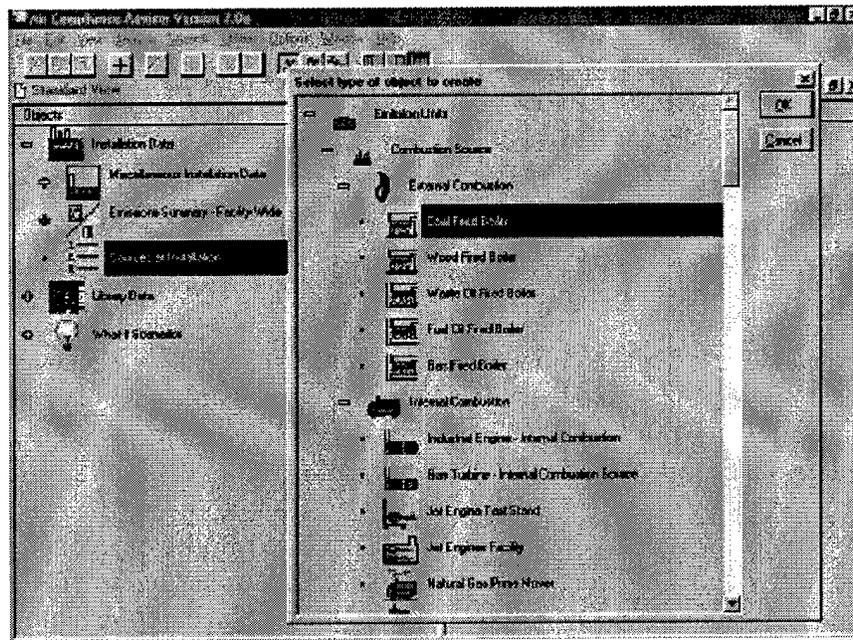
Data is entered into the ACA to provide input for the ACA's analyses or as a way of storing this information in a database. Data entry consists of modifying either objects or data entry slots. Objects can be added to object lists, replaced by other objects, and deleted. The types of data entry slots and the actions that can be performed on data entry slots were described in the

previous sections. The analytic features of the ACA are described in greater detail in Section 4. These descriptions in Section 4 include the information requirements of each analysis option.

## Objects

Users manipulate objects through the add button , dragging and dropping, and the delete button . The usage of each of these object manipulation methods is discussed in the paragraphs below.

The primary purpose of the add button  is to add objects to object lists. To add an object to a list, users first click on an object list in the left pane of the **Standard View**, then click the add button to add another instance of that object (e.g., an emission source). After clicking the add button , a screen with a context relevant pick-list appears. Users select the desired object by highlighting it and then clicking . The figure below shows an example where coal fired boiler is about to be added to the list of sources at an installation.

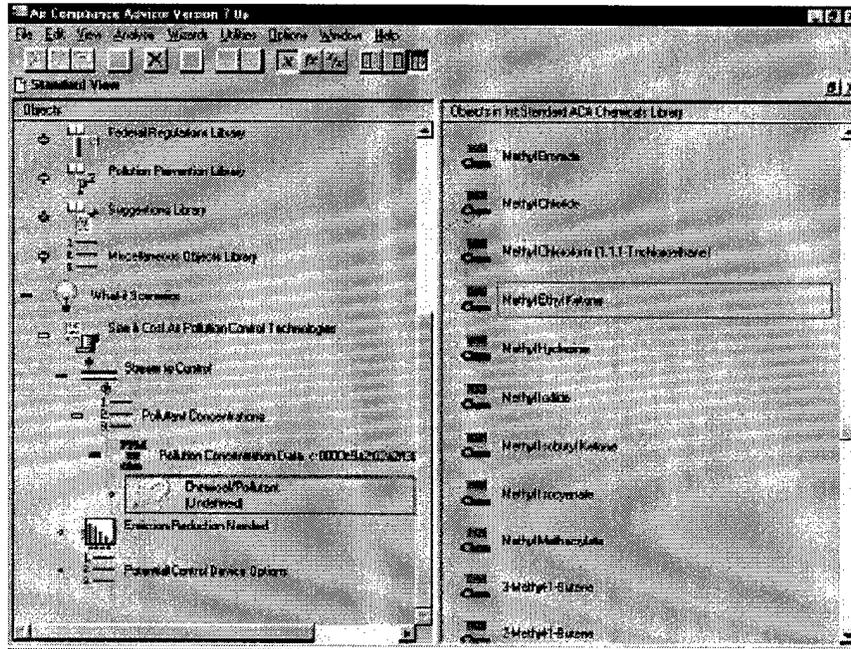


Adding an object to an object list

The add button  can also be used to replace certain objects. If the add button  is available while an object is selected, then that object can be replaced by clicking on the add button and selecting from the available objects. When objects should not be added or replaced, either the add button will be unavailable or no objects will be displayed in the pick list when the add button is clicked.

Objects can also be manipulated by "dragging and dropping" an object from one part of the **Standard View** to another. A drag and drop is accomplished by clicking and holding the left mouse button on an object, dragging it to the appropriate object list or object, and dropping the object by releasing the mouse button. A rectangular box will appear around an object list or object when a "dragged" object can be dropped there. When an object is dropped on a list, the

object will be added to the list. When an object is dropped on an object of the same type, the object will be replaced. The figure below shows methyl ethyl ketone being dragged from the standard chemical data library to an undefined chemical/pollutant object. After this operation is complete, methyl ethyl ketone would become a chemical included in the pollutant stream.



**Dragging and Dropping a chemical**

When objects are added or replaced, either by the user or by the system, the new object is either a copy of an object or of a link. A copied object is an independent object and changes made to this object will not affect the object used to create the copied object. For example, the Coal Fired Boiler added to the Sources at Installation object list from the figure entitled "Adding an object to an object list" above is a copy of the Coal Fired Boiler object. Changes to the copied Coal Fired Boiler object will not effect the original Coal Fired Boiler object built in to the ACA. A linked object, on the other hand, is not independent. Any changes made to a linked object will also be made to the object used to create the linked object. For example in the figure above, the dragging and dropping of the Methyl Ethyl Ketone object will create a link to the Methyl Ethyl Ketone object found in the chemical library. Any changes made to the linked Methyl Ethyl Ketone object will be duplicated in the Methyl Ethyl Ketone object found in the chemical library. A linked object is designated graphically by a small arrow (↖) on the bottom left-hand side of the linked object's icon.

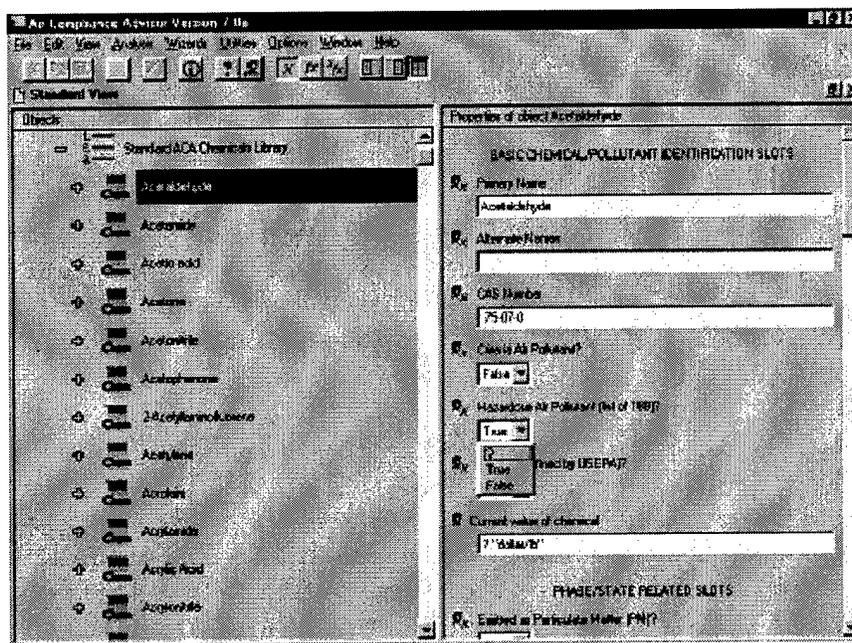
The delete toolbar button  is used to delete objects from the left-hand side of the Standard View. Users can delete all objects except for some higher-level object types. The deletion button will be unavailable for those objects that should not be deleted.

### *Data Entry Slots*

Data entry slots contain text, numbers, times, dates, and logical data (e.g., true, false). Numeric data entry slots often also have units associated with the numeric data. The units are enclosed in quotation marks (e.g., "dollar/lb"). Information is added to data entry slots either by typing it in

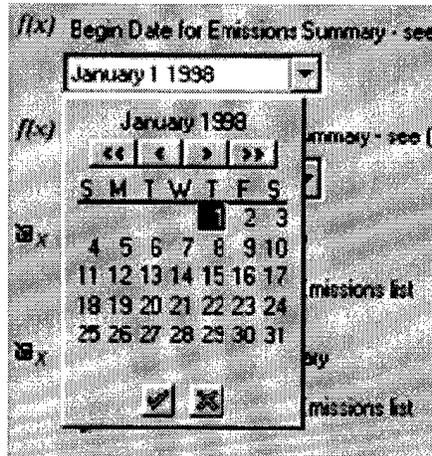
or by selecting it from a list. If a data entry slot contains numeric data with units, both the value and the units must be supplied. Users are free, however, to enter a wide array of units as long as the units are dimensionally correct for that slot. Appendix B contains a list of units recognized in the ACA.

Information is entered into data entry slots either by typing information into the slot or by selecting from a pick list. A slot with an associated pick list will have a small down arrow key on the right side of the slot. Data entry slots containing dates or logical data always have a pick list. Data entry slots without pick lists will scroll when information is added beyond the length of the slot, and pressing the Enter key can expand these slots. Users can perform standard operations, such as copy, edit, and paste, on information contained in data entry slots. The figure entitled "Entering data in the Properties of Objects pane" shows some of the data entry slots for the chemical object acetaldehyde. This figure shows examples of data entry slots with text, logical data, numeric data with units, and a pick list.



Entering data in the Properties of Objects pane

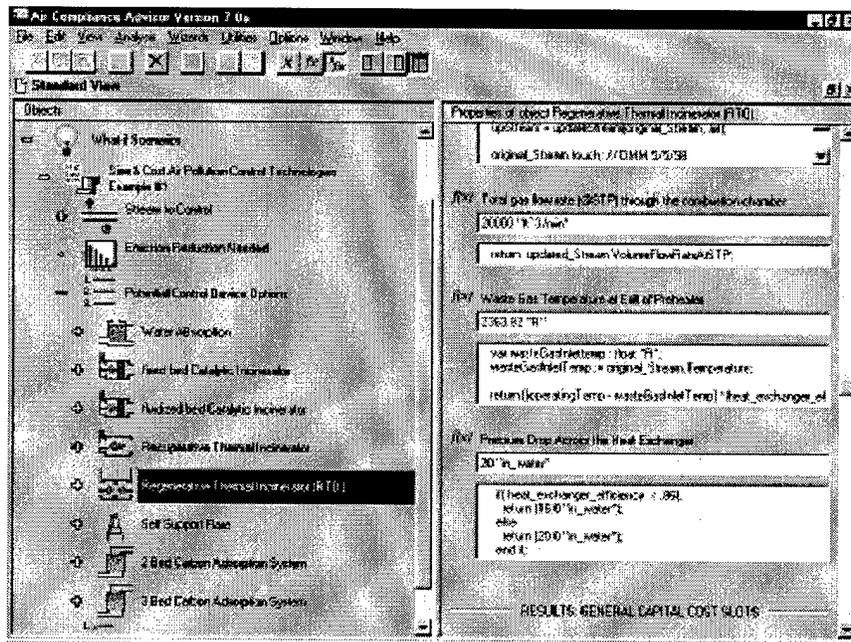
The ACA has a built-in special utility to assist in entering dates into slots. Click on the down arrow key on the right side of the date slot and a calendar appears. Click on << or >> to change the year. Click on < or > to change the month. Click on the desired date in the body of the calendar to select a specific day. See the figure entitled "Pop-up calendar for entering date."



Pop-up calendar for entering date

### 3.4 Basic User Extendibility

The information found in a data entry slot might actually be the result of a function. Users can view the underlying functions of data entry slots by selecting the  $f(x)$  button in the toolbar to view both the functions and the values or the  $f_x$  button to view the functions alone. The  $X$  button will toggle the view back to just values. Note that in the figure entitled "Data entry slot values and functions" the  $f(x)$  button has been selected.



Data entry slot values and functions

Users can obtain detailed information about a data entry slot by clicking on the toolbar button . The information includes the slot name of the data entry slot. The slot name is required to create user-supplied functions that depend on that slot.

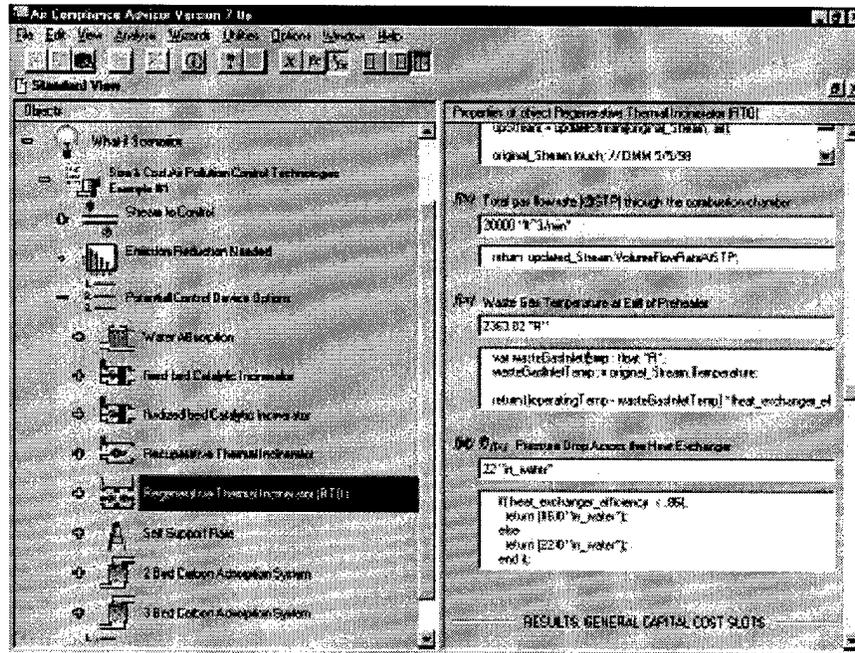
| <b>Icon</b>                                                                       | <b>Meaning</b>                           |
|-----------------------------------------------------------------------------------|------------------------------------------|
|  | Reveal detailed information about a slot |
|  | show slot values                         |

#### **Toolbar buttons**

Users can edit existing functions or create new functions in the function view. The figure entitled "Data entry slot values and functions" shows an example of the right pane of the **Standard View** displaying both values and functions. The list of potential control device options is generated after running the **Apply Control Technologies** analysis for a **What-if Scenario**. Details of this analysis will be discussed in the "Apply Control Technologies" Section. The data entry slot for **Pressure drop** will be used for a simple example of how users can modify functions.

The function view for the **Pressure drop** data entry slot shows a small section of computer code. This code is the function for pressure drop. In this case the function is a simple if-then type statement and the value of the slot is dependent on the slot name `heat_exchanger efficiency`. If a user had updated information from a vendor indicating that the pressure drop should really be 22 inches of water for heat exchanger efficiencies greater than or equal to 0.86, the user could modify the function to reflect this new information. The figure entitled "Results from editing pressure drop function" shows an edited function and the result this change had on the value of the **Pressure drop** data entry slot. The icon showing the slot type also changed to indicate that a user-supplied function had been entered.

This brief example and discussion only covers a very small portion of the user extendibility options within the ACA. Appendix C contains much more detailed information on this subject.



Results from editing pressure drop function

Recall that the toolbar buttons  and  are used to set slots to unknown and to reset slots to the default value, respectively. For data entry slots, setting a slot to unknown sets the value to unknown and also removes any underlying function. For data entry slots with a function, resetting the slot to the default value will regenerate the function and the value calculated by the function.

 **Warning:**

If user-defined slots that begin with either a value or a function (i.e., slots with the  or  icons) are overridden by the user with either a new value or a new function, then the reset button will NOT reset to the initial value or function. Rather, it will reset the user-defined slot to the default value, which is typically undefined (i.e., "?").

### 3.5 Standard ACA Libraries

This section considers the five standard libraries contained in the ACA:

- Chemicals Library
- Materials Library
- Control Devices Library
- Federal Regulations Library
- Pollution Prevention Library

- Suggestions Library
- Miscellaneous Object Library

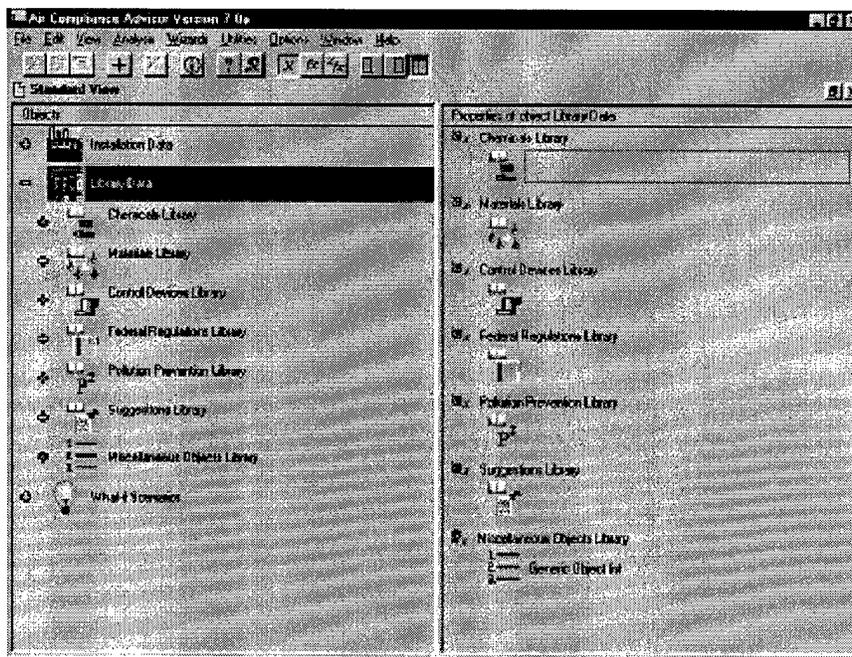
These library objects are shown in the figure entitled "Library data types found in the ACA."

Note that only the Chemicals, Materials, and Control Devices Libraries are loaded automatically at program startup. The other three libraries can be loaded manually as needed through **File | Load**.

Library data can be edited at will and changes a user makes during an ACA session will be stored if the user saves an ACA file using the **File | Save** or **File | Save as** functions. However, the original library data will be unaffected and will be available whenever a new ACA session is started. Each library object contains at least one standard and user-defined object list. A standard list contains the built-in library data and a user-defined list is designed to store user additions to the standard library items.

Users can add new entries in the user-defined libraries either by creating a new library item or dragging and dropping an existing library item into the appropriate user-defined library list. To create a new entry into any of the libraries, the user first clicks on the appropriate user-defined library list in the left pane of the **Standard View**, then clicks the add button  to add another instance of that object (e.g., a new chemical object). After clicking the add button, a screen with a context relevant pick-list appears (i.e., "select type of object to create"). Users select the object shown by highlighting it and then clicking .

The drag-and-drop method will create a copy of an existing object that the user can independently edit. To create a new entry into any of the libraries using the drag-and-drop method, the user first clicks and holds the mouse on the appropriate object they wish to copy. Then simply drag that object to the appropriate user-defined library list in the left pane of the **Standard View** and release the mouse to drop it onto that list. This new object can now be edited by the user.



**Library data types found in the ACA**

### ***Chemicals Library***

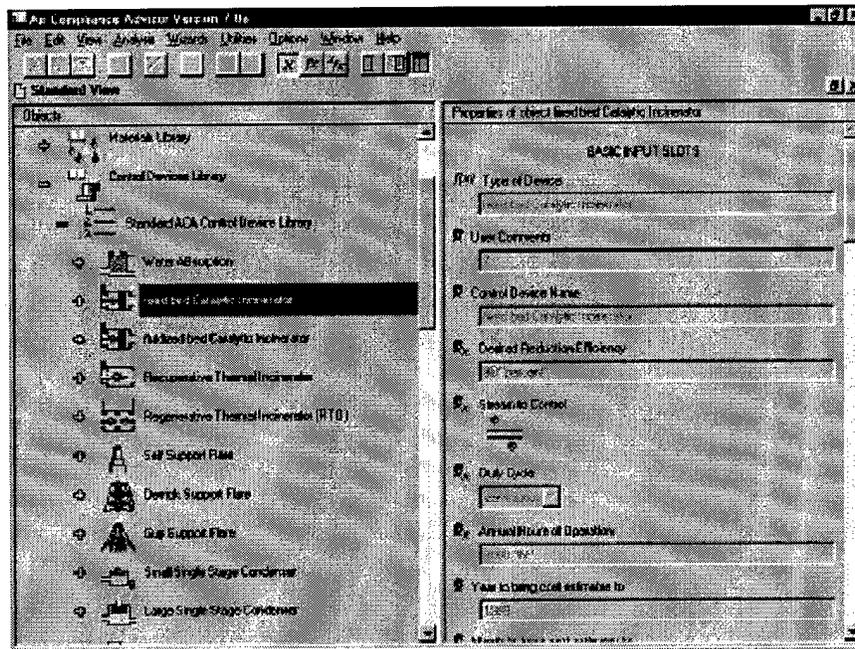
The Chemicals Library contains chemical properties for all of the hazardous air pollutants (HAPs) listed in Title III of the Clean Air Act Amendments of 1990, as well as the chemical properties of the criteria pollutants. The Chemicals Library is used to define the components of air pollution streams; determine chemical emission inventories on both a source specific and a facility-wide basis; assess applicable control technologies; and check the applicability of regulations, pollution prevention opportunities, and suggestions.

### ***Materials Library***

This library contains a few sample materials. The user can add materials to the library as needed as previously described. The material property data is used to define the material usage of air pollution sources; determine chemical emission inventories on both a source specific and a facility-wide basis; assess applicable control technologies; and check the applicability of regulations, pollution prevention opportunities, and suggestions.

### ***Control Devices Library***

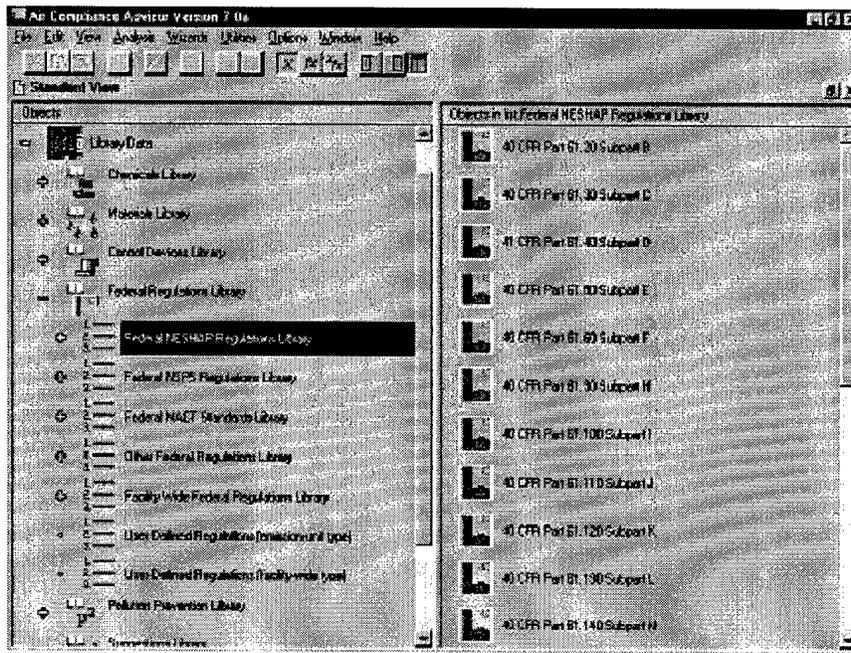
The Control Devices Library contains all the air pollution control devices found in the US EPA's *OAQPS Control Cost Manual* and a few other control devices specified elsewhere. Each object in the Standard ACA Control Devices Library is a separate control device. The figure entitled "Control Devices Library" shows the beginning of the Control Devices Library and a few of the slots associated with the fixed bed Catalytic Incinerator control device. The Control Devices Library is primarily used during the Apply Control Technology analysis described in detail in the "Apply Control Technologies" Section. The ACA uses the methods and information found in the *OAQPS Control Cost Manual* when determining the costs of applicable control devices.



### Control Devices Library

#### *Federal Regulations Library*

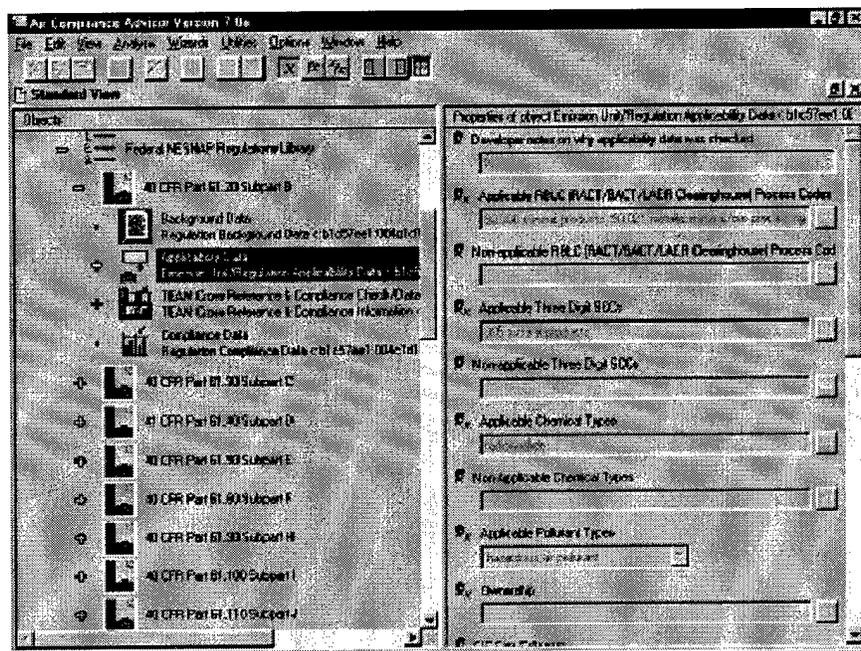
The Federal Regulations Library contains five lists of built-in regulations: NESHAP, NSPS, MACT Standards, Other Federal Regulations, and Facility-Wide Federal Regulations. The Federal Regulations Library also contains two user-defined regulation lists for emission-unit type regulations and facility-wide type regulations. Remember that this library data will need to be manually loaded in order to have access to it. Select File | Load | ACA Federal Regulations Database in order to load this library.



#### Regulation lists contained in the Standard ACA Federal Regulations Library

Portions of the US Army Construction Engineering Research Laboratories' environmental compliance assessment checklist manual have been incorporated into the Federal Regulations Library. The manual, also known as *The Environmental Assessment and Management (TEAM) Guide*, was developed for use by all DoD components to help determine compliance with current environmental regulations. Information from the *TEAM Guide* found in the ACA includes: the TEAM checklist ID, the summary of the requirements for the individual regulations, and the reviewer checklist that accompanies each regulation. Incorporation of the *TEAM Guide* in the ACA provides further context for assessing the applicability of particular regulations. This information appears as a data object associated with the corresponding federal regulation object.

Individual regulations are defined by specific properties that appear in the Properties of Objects window, and by information contained in the Background Data, Applicability Data, TEAM Cross Reference & Compliance Check/Data, and Compliance Data objects as shown in the figure entitled "Data structure of a regulation." Each of these data objects has unique properties, just as individual chemicals have unique chemical properties. It should be noted that not every regulation will contain entries for the TEAM Cross Reference & Compliance Check/Data object, as the *TEAM Guide* considers only those regulations that have direct applicability to DoD facilities.



**Data structure of a regulation**

The Federal Regulations Library within the ACA is used in determining which regulations may apply to a specific source or sources. This is accomplished by identifying those regulations that are not applicable to the source.

The ACA checks the applicability of regulations based on source type classifications (e.g., BLIS Process Codes and SCC values), chemicals used in a process, chemicals released during a process, facility ownership, and dates of construction, modification, reconstruction, and start up. As discussed in the "Prototype Analysis" Section of this manual, specification of these properties for a source facilitates the check for applicable regulations.

### ***Pollution Prevention Library***

Pollution prevention (P2) strategies and alternatives form the basis for this library. The user can explore details of various P2 strategies, including the cost, advantages and disadvantages of a particular strategy, chemical emissions, and obtain reference citations for further investigation of the strategies presented. The Pollution Prevention Library within the ACA is used in determining which pollution prevention opportunities may apply to a specific source or sources. Remember that this library data will need to be manually loaded in order to have access to it. Select **File | Load | ACA Pollution Prevention Database** in order to load this library.

### ***Suggestions Library***

The Suggestions Library provides the user with information on the control of nitrogen oxides (NOx) and emerging VOC control strategies. The information presented with each suggestion includes a general description of how the technology works, a discussion of the applicability of the technology to the waste stream, a discussion of necessary stream and operating conditions, and other information relevant to the technology. Remember that this library data will need to be

manually loaded in order to have access to it. Select **File | Load | ACA Suggestions Database** in order to load this library.

### ***Miscellaneous Objects Library***

The **Miscellaneous Objects Library** provides the user with a place to store any ACA data object for future reference. The ACA is distributed with two **Operating Schedule** objects (i.e., "always on" and "On Monday – Friday 8hrs/day") which can be copied, as needed, by the user to define how equipment is operated. The **Miscellaneous Objects Library** is a good location for the user to store installation-specific objects that they may want to reuse and are a different type of object than those in the other libraries.

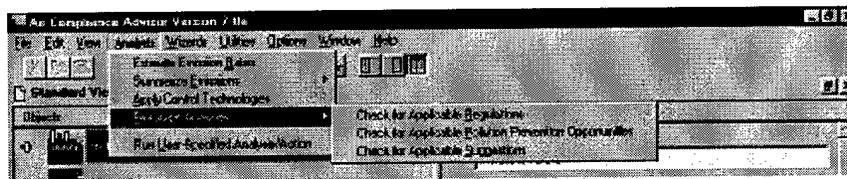
## 4. Analysis Options of the ACA

While numerous calculations in the ACA are made "behind the scenes" (much like a spreadsheet automatically re-calculates in the background), there are several other specific analyses currently available within the ACA program. These analyses are accessed via the **Analysis** menu within the main menu of the program as shown in the figure below entitled "Standard Analysis options menu available in the ACA" Each of the analyses generates a report. To control the amount of detail in the reports, the menu selection **Options | Report Level** can be selected to be "Low," "Medium" or "High," with low being minimal information, medium being an intermediate amount of information and high being very detailed.

The standard analyses are listed below:

- Estimate Emission Rates
- Summarize Emissions
  - Directly out of Emission Units
  - From Stack (to Atmosphere)
- Apply Control Technologies
- Prototype Analyses
  - Check for Applicable Regulations
  - Check for Applicable Pollution Prevention Opportunities
  - Check for Applicable Suggestions

This section will discuss the basics of these analysis options. In order to give the user practice with these analysis options and the mechanics of entering data, example problems are provided in Section 6 .



Standard Analysis options menu available in the ACA

## 4.1 Estimating Emission Rates

The Estimate Emission Rates analysis option estimates emission rates for many sources of air pollution. The ACA can estimate emission rates for the following emission units:

- Aboveground Tanks
- Abrasive Blasting
- Asphalt Cutbacks
- Carpentry Operations #
- Clean up Materials
- Coal Fired Boilers
- Degreaser Operations
- Electroplating #
- External Floating Roof Tanks
- Fuel Dispensing Facility
- Gas Fired Boilers
- Gas Turbine (IC)
- Graphic Arts / Printing
- Industrial Engine
- Internal Floating Roof Tanks
- Jet Engine Test Stands
- Loading Racks
- Medical Waste Incinerators #
- Multi-Chamber Incinerators
- Natural Gas Prime Mover (IC)
- Oil Fired Boilers
- Petroleum Spill
- Single-Chamber Incinerators

- Solvent Spill
- Spray Cleaner
- Surface Coating
- Tactical Haulers
- Tank and Drum Cleaning
- Thermal Metal Sprayers
- Underground Tanks
- Waste Oil Fired Boilers
- Waste Solvent Recovery Operation
- Waste water Treatment Units
- Water Solvent Recovery Operation
- Welding #
- Wood Fired Boilers

**\*Not fully tested**

The ACA calculates emissions by first estimating an emission rate (e.g., lbs/hr) and then using operating schedules to estimate emission summaries for any time period specified by the user. This emission estimating method allows realistic calculations of daily, weekly, monthly, or yearly emissions for all emission units. Users can input both actual and potential operational data sets for all emission units and the ACA will use this information to estimate both actual and potential emissions.

ACA emission estimates are based on data that is typically available or can be measured. In general, source, operational, and material-specific data are required to make emissions estimates in the ACA. The ACA uses internal algorithms based on standard emission estimation techniques to calculate emissions. Most emission estimates are calculated using the US EPA's AP-42 emission estimation guidance. If a user does not enter information into a data entry slot required for emission estimation, the ACA will identify the name of the required data entry slot and its location.

 **Note:**

Caution must be taken when interpreting the hourly emission rates for emission units that utilize AP-42 emission factors. These emission factors are designed to estimate yearly averages and not hourly values. The ACA makes these hourly estimates of emission rates to enable multiple operating scenarios, but when looking at emission summaries for less than one year, the limitations of the results should be considered.

As the data requirements for estimating emission rates are dependent on the type of emission unit considered (e.g., gas-fired boiler vs. waste water treatment plant), the exact procedure for entering data into the ACA will be a function of the emission unit.

However, the following are general steps that are true for most types of emission units:

1. In **Standard View**, select **Installation Data**. Then, add an emission unit to the **Installation Data | Sources at Installation** object list and expand the **Sources at Installation** object list and the emission unit object by selecting the  button in the toolbar. The figure entitled "Addition of emission unit for estimating emissions" shows an example of this step where a surface-coating source was added to the **Sources at Installation** object. Notice that the **RBLC** process code has already been set; the **Actual Operational Data** and **Potential Operational Data** slots are attached but are empty, and the **Attempt to Estimate Emissions?** data entry slot is set to **False**. See the screens that follow this section for an example.

2. Set the emission unit data entry slot **Attempt to Estimate Emissions?** to **True**.

3. Review the other slots at the emission unit level for data that may be required for estimating emissions.

4. Add as many operational data objects as needed to describe all of the modes in which the emission unit operates. Make sure to add operational data objects to both the **Actual Operational Data** object list and to the **Potential Operational Data** object list. Fill in the data entry slots for each operational data object as needed. The figure entitled "Data entry slots for operational data" shows data entry slots for the **Surface Coating Operational Data** object. Do this by clicking on the **Actual Operational Data** slot in the **Standard View** and clicking on the add button  in the toolbar. See the screens that follow this section for an example.

5. Describe the operating schedule. This step is optional since it is not required to estimate emission rates. However, this is a good time to enter this data, and an operating schedule is required to perform the **Summarize Emissions** analyses. To describe an operating schedule, select the **Operating Schedule** object and fill in the following data entry slots:

- Schedule Name (optional but good for bookkeeping purposes)
- Schedule Start Date
- Schedule End Date
- Average Operational Time per Day
- Days Operating per Week

Note that by going to the "Intermediate" User Level (**Options | User Level**) you are given the opportunity to enter daily operating schedules. In this case, the data entry slots **Average Operational Time Per Day** and **Days Operating Per Week** will be calculated by the ACA. The figure entitled "Operating Schedule data entry slots" shows the data entry slots associated with the **Operating Schedule** object in **Novice** mode. See the screens that follow this section for an example.

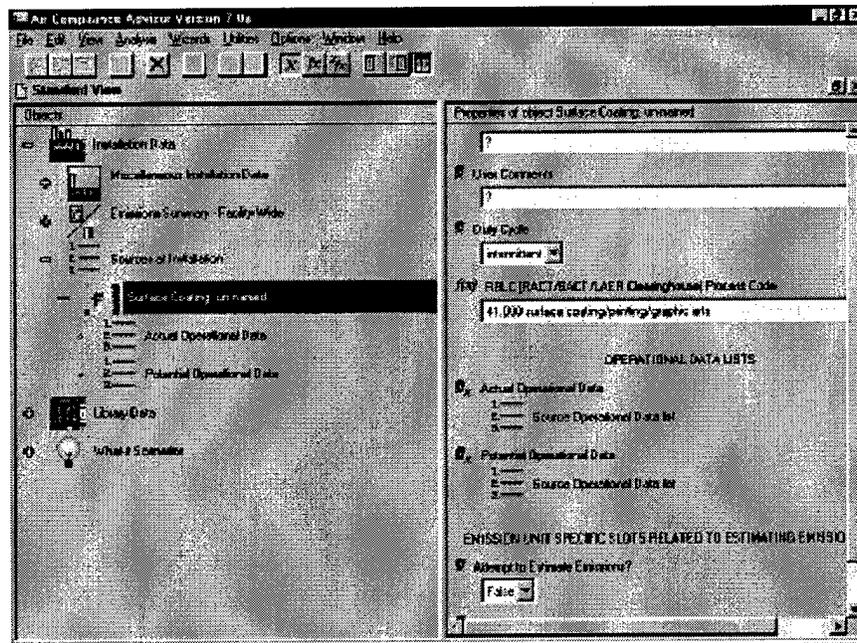
6. Describe the material used by the emission unit and enter its usage rate (for some emission units [e.g., coal-fired boilers], users can just enter in the summary material information contained in the Materials Inputs object at the Novice user level and then optionally enter in the actual material used at the Intermediate user level). To do this, first select and expand the Materials Inputs object by clicking on it in the Standard View. Next enter either the Mass Usage Rate or the Volumetric Usage Rate into the appropriate data entry slot. Then add a specific material

to the undefined ( ? ) Material object by selecting from the menu provided when you click on the add button (+) in the toolbar. Note that not all emission units (e.g., a gas-fired boiler) will require the addition of a specific material. Finally, review the other data entry slots of the Materials Inputs object for additional data that may be required for estimating emissions. The figure entitled "Materials Inputs object data entry slots " shows an example of data entry slots associated with the Materials Inputs object. See the screens that follow this section for an example.

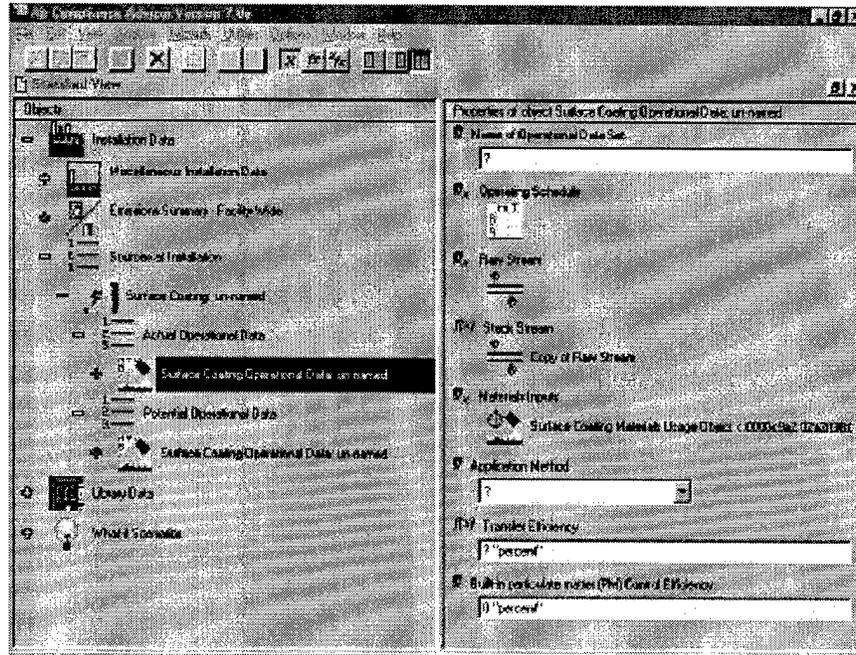
7. Select Analysis | Estimate Emission Rates to perform the emission estimate analysis.

 **Note:**

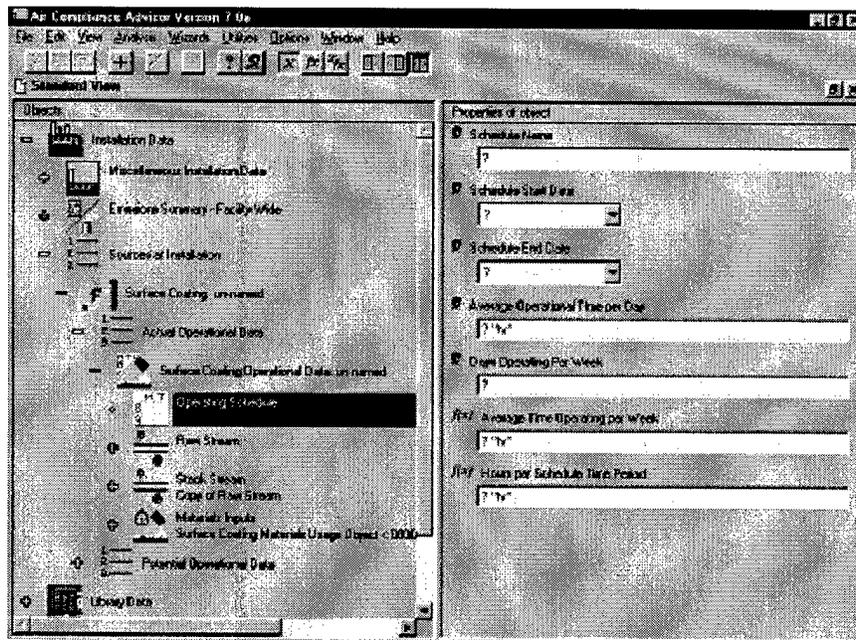
Steps 5 & 6 are to be repeated for each operational data set that you added in Step 4.



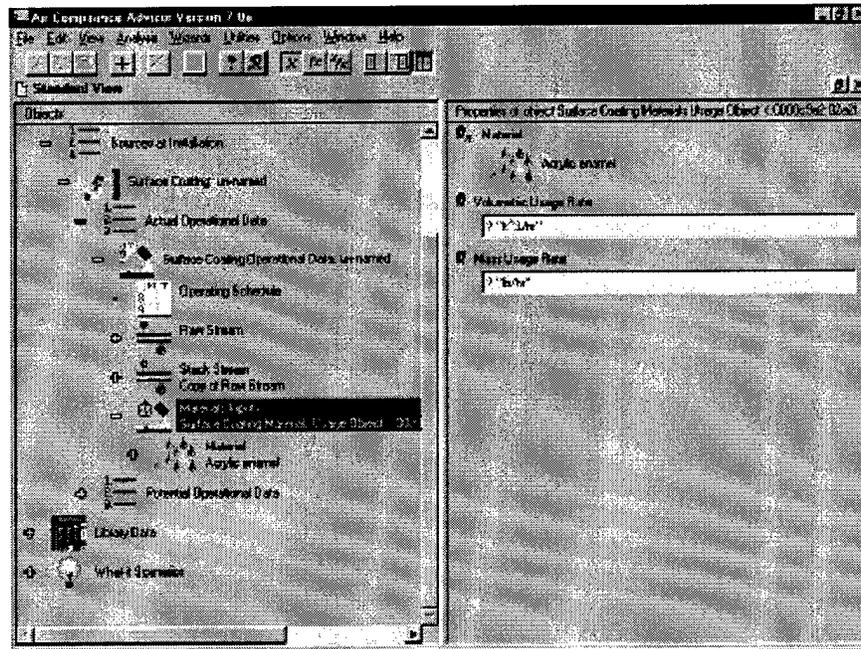
**Addition of emission unit for estimating emissions**



Data entry slots for operational data



Operating Schedule data entry slots



### Materials Inputs object data entry slots

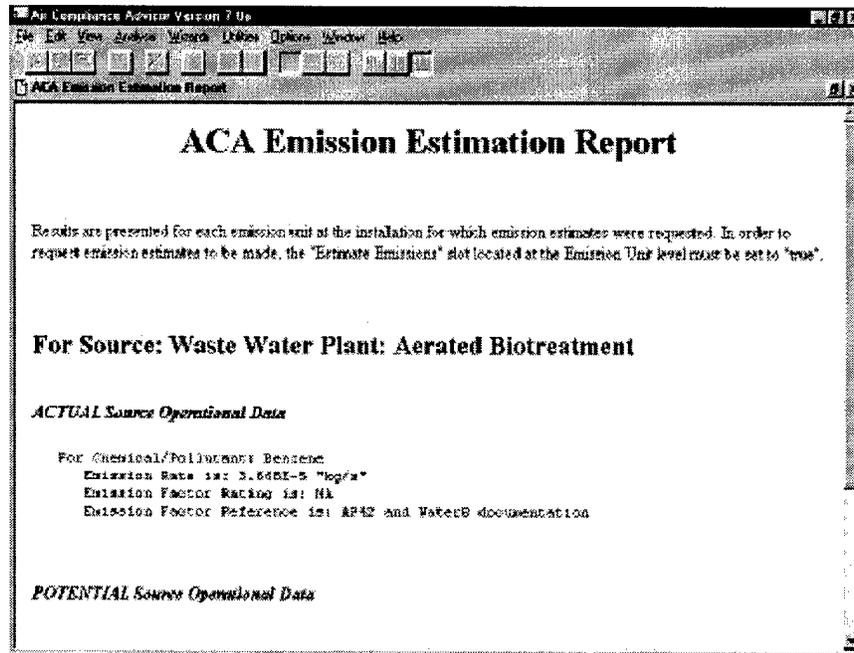
When the emission estimate analysis is complete, three things will happen:

1. A report will be generated for each emission unit.
2. Estimated emissions are added to the Raw Stream and Stack Stream objects of each operation data set in both the Actual Operational Data and Potential Operational Data object lists.
3. Any errors, notes or warnings related to the analysis will be stored in a slot for each operational data set.

### ✓ **Important:**

Be sure to read these reports as they contain important information.

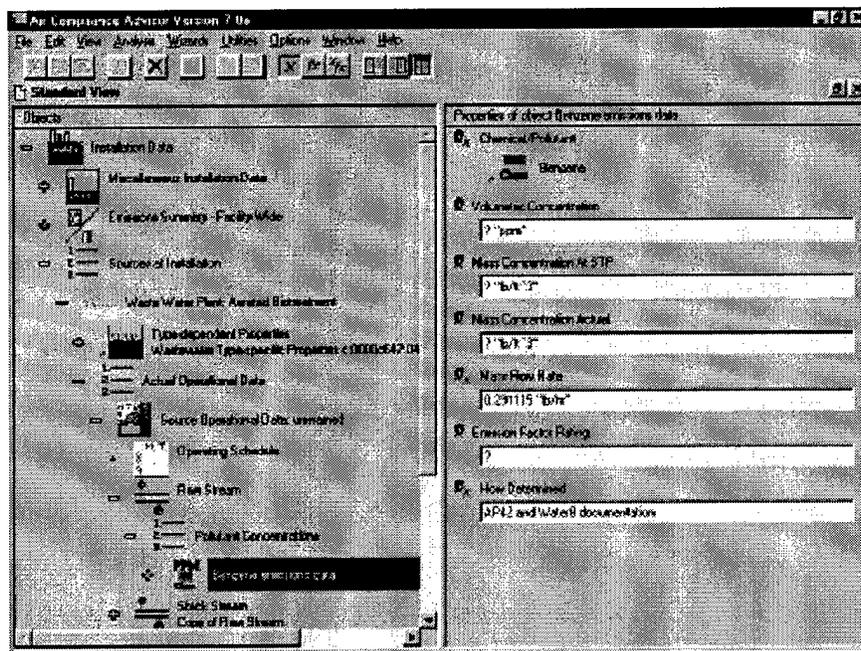
The ACA Emission Estimation Report appears in a new window. If no required data is missing, this report will contain estimates for most air pollutants the source emits. If a necessary piece of information was left out, the report will contain an error message directing the user where to provide additional information. The figure entitled "Emission Estimation Report" shows an Emission Estimation Report generated for an example benzene emission estimate from waste water treatment and an error message.



### Emission Estimation Report

The estimated emissions are also shown in the Pollution Concentrations object list contained in the Raw Stream and Stack Stream objects. The Raw Stream object represents the emissions that come directly out of an emission unit. A "raw stream" may then pass through add-on control technologies before being emitted to the atmosphere through a "stack." The Stack Stream object represents the actual "stack stream" emissions to the atmosphere. A Pollution Concentrations object list has a Pollution Concentration Data object for each pollutant reported. Each Pollution Concentration Data object contains the emission rate of the pollutant and how this rate was determined. The figure entitled "Example pollutant concentration list" shows benzene emission data from a waste water treatment operation. The icon associated with the Chemical/Pollutant object indicates that it is a linked object. Any changes made to this object would also occur in the benzene object contained in the ACA Standard Chemicals Library.

The Stack Stream object is actually a carbon copy of the Raw Stream object unless users override this relationship. Since the Stack Stream object is initially a function of the Raw Stream object, all data entry slots associated with the Stack Stream object are made unavailable to users to preserve this relationship. Users can override the relationship between Raw Stream objects and Stack Stream objects by creating a new "empty" Stack Stream object through the toolbar's add button  or by dragging another stream object onto the Stack Stream object. If another stream object is dragged onto the Stack Stream object, the Stack Stream object will exhibit all the properties of the dragged stream. For example, users could first estimate emission with the Estimate Emission Rates analysis, use the calculated stream parameters to perform an Apply Control Technologies analysis (see the "Apply Control Technologies" Section), and then drag the resultant stream object onto the Stack Stream object. Users can also enter actual emissions information directly into the appropriate data entry slot. In either case, the Stack Stream object would then represent the stream being emitted to the atmosphere.



**Example pollutant concentration list**

The Estimate Emission Rates analysis fills out the data entry slot **Errors, Warnings, and Notes Report Generated When Estimate Emission Rates Was Last Run**. This data entry slot is contained in each operational data element and is available at the Intermediate user level. As the name implies, this slot contains any errors, warnings, or notes associated with the most recent execution of the Estimate Emission Rates analysis.

### ***Additional Things to Try/Consider When Estimating Emission Rates***

If, instead of estimating emission rates, you have source emissions data, you can simply manually enter this data into the "Raw Stream" of each operational data set in both the "Actual Operational Data" list and to the "Potential Operational Data" list. If you want to enter the emissions by hand, then make sure that the emission unit slot "Attempt to Estimate Emission?" is set to FALSE.

If you would like to view the algorithms for estimating emissions, then simply switch to the show slot function mode  and set the User Level to "Intermediate." Locate the slot "Emission Rate Calculations Slot" at the emission unit level. You are also able to modify these algorithms by going into the slot definition window and reprogramming as desired. If you wish to modify the emissions, it is advised that you become familiar with the "End User Extensibility in the ACA" — refer to Appendix C of this document.

### ***Emission Unit Types That the ACA Can Estimate Emission Rates***

| <b>Emission Unit Type</b>    | <b>Fully Tested</b> | <b>Can Be Edited Via the "Easily Viewable" Slot Format</b> |
|------------------------------|---------------------|------------------------------------------------------------|
| Aboveground Tanks            | Yes                 | Yes                                                        |
| Abrasive Blasting            | No                  | Yes                                                        |
| Asphalt Cutbacks             | Yes                 | No                                                         |
| Carpentry Operations         | No                  | Yes                                                        |
| Clean Up Materials           | Yes                 | Yes                                                        |
| Coal Fired Boilers           | Yes                 | Yes                                                        |
| Degreaser Operations         | Yes                 | No                                                         |
| Electroplating               | No                  | Yes                                                        |
| External Floating Roof Tanks | Yes                 | Yes                                                        |
| Fuel Dispensing Facility     | Yes                 | Yes                                                        |
| Gas Turbine                  | Yes                 | Yes                                                        |
| Gas Fired Boilers            | Yes                 | Yes                                                        |
| Graphic Arts / Printing      | Yes                 | Yes                                                        |
| Industrial Engine            | Yes                 | Yes                                                        |
| Internal Floating Roof Tanks | Yes                 | Yes                                                        |
| Jet Engine Test Stands       | Yes                 | Yes                                                        |
| Loading Racks                | Yes                 | Yes                                                        |
| Medical Waste Incinerators   | No                  | Yes                                                        |
| Multi-Chamber Incinerators   | Yes                 | No                                                         |
| Natural Gas Prime Mover      | Yes                 | Yes                                                        |
| Oil Fired Boilers            | Yes                 | Yes                                                        |
| Petroleum Spill              | Yes                 | Yes                                                        |
| Single-Chamber Incinerators  | Yes                 | No                                                         |
| Solvent Spill                | Yes                 | Yes                                                        |

|                                  |     |     |
|----------------------------------|-----|-----|
| Spray Cleaner                    | Yes | Yes |
| Surface Coating                  | Yes | Yes |
| Tactical Haulers                 | Yes | Yes |
| Tank and Drum Cleaning           | Yes | Yes |
| Thermal Metal Sprayers           | Yes | Yes |
| Underground Tanks                | Yes | Yes |
| Waste Solvent Recovery Operation | Yes | Yes |
| Waste Oil Fired Boilers          | Yes | Yes |
| Waste Water Treatment Units      | Yes | Yes |
| Water Solvent Recovery Operation | Yes | Yes |
| Welding operations               | No  | Yes |
| Wood-Fired Boilers               | Yes | Yes |

## 4.2 Summarizing Emissions

The Summarize Emissions analysis option is used to summarize all emissions (actual and potential) for all of the sources/emission units that are stored in the Installation Data | Sources at Installation.

When executing this analysis, you can select between the following options:

- Directly Out of Emission Units
- From Stack (to Atmosphere)

These two sub-analyses are similar in that they both present results in the form of three reports that summarize:

1. The facility-wide actual emissions
2. The facility-wide potential emissions
3. The actual and potential emissions on an emission unit-by-emission unit basis

There are a couple of differences between these two sub-analyses. First, the "Directly Out of Emission Units" analysis summarized the "Raw Stream" emissions. These emissions are in each Operational Data set for each emission unit in the Installation Data | Sources at Installation

list. Meanwhile, the "From Stack (to Atmosphere)" analysis summarizes the "Stack Stream" emissions that are also in each Operational Data set for each emission unit in the Installation Data | Sources at Installation list. Secondly, only the "From Stack (to Atmosphere)" analysis adds emissions data to the Installation Data | Emission Summary - Facility Wide object. The "From Stack (to Atmosphere)" analysis also adds emissions data to the "Emission Summary for Source" object that is contained within each emission unit (note that you will have to set the Options | User Level to "Expert" to see this data). When summarizing the emissions, the ACA also considers the operating schedule for the Operational Data set.

The following general steps are required for the Summarize Emissions analyses:

1. Expand the Emission Summary - Facility Wide object contained in the Installation Data core object by double clicking on it.

2. Enter data into the following data entry slots contained in the Emission Summary - Facility Wide object:

1. Title of Emissions Summary (optional, but good for book keeping purposes)

2. Begin Date for Emissions Summary\*

3. End Date for Emissions Summary\*

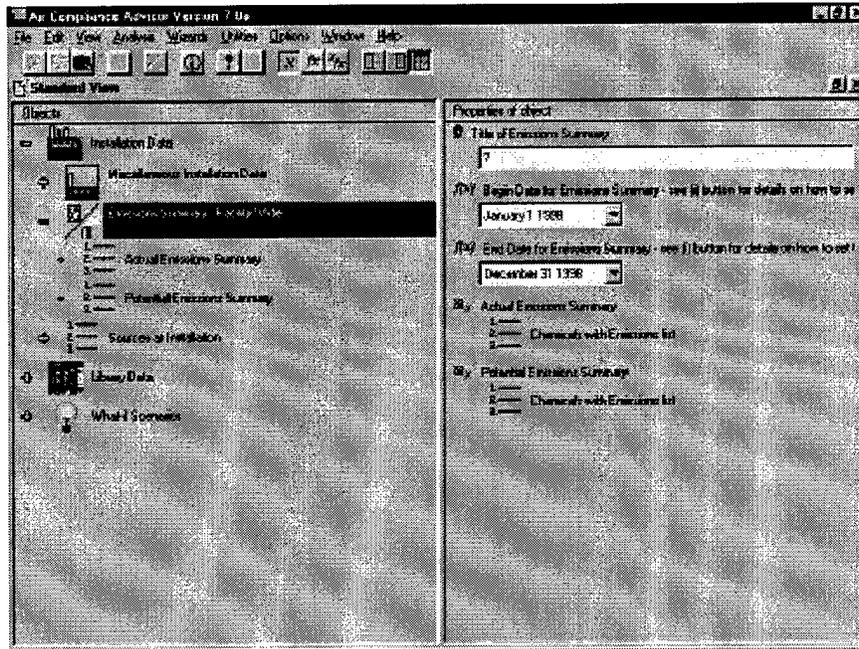
\*Note that you are not limited to any set time period. The default for these two slots are taken from the Installation Data object data entry slots Default Start Date for Emission Summaries and Default End Date for Emission Summaries. Users can either edit the Installation Data object defaults or the two slots contained in the Emission Summary-Facility Wide object. In order to keep the emission unit-specific summaries consistent with the facility-wide summaries, it is advisable to edit the data entry slots contained in Installation Data object.

The figure entitled "Slots contained in the Emission Summary-Facility Wide object" shows the slots contained in the Emission Summary- Facility Wide object.

3. Make sure that the Raw Stream and Stack Stream objects have been defined as described in the previous section, "Estimate Emission Rates."

4. Make sure the Operating Schedules object for all of the Actual Operational Data and the Potential Operational Data objects have been defined as described in the previous section, "Estimate Emission Rates."

5. Select either Analysis | Summarize Emissions | Directly Out of Emission Units or Analysis | Summarize Emissions | From Stack (to Atmosphere) from the Menu.



#### Slots contained in the Emission Summary-Facility Wide object

When an emission summary analysis is complete, three emission summary reports will be generated and displayed in separate windows. The reports summarize (a) actual and potential emissions on an emission unit-by-emission unit basis, (b) facility-wide actual emissions, and (c) facility-wide potential emissions. Each report shows the time period covered for each emission summary, and if a necessary piece of information was left out, the report will contain an error message directing the user to where additional information is needed. The following three figures show examples of these three reports. Note that the information contained in the reports for these figures is from example problem #5 in Section 6 of this User Guide.

Ap Compliance Advisor Version 7.0a

File Edit View Analysis Methods Utilities Options Database Window Help

Emissions by Source Stack Stream(s)

### Emissions by Source Stack Stream(s)

Emission Unit Name: Surface Coating: un-named

**Actual Emissions**

Emission Unit Name: Surface Coating: un-named  
 Emission Summary for Dates: January 1 1998 - December 31 1998  
 Emission Unit Actual Emissions

| Chemical Name                  | Mass Quantity |
|--------------------------------|---------------|
| Cyclohexanone                  | 4,487 "kg"    |
| Methyl Ethyl Ketone            | 7,686 "kg"    |
| Methyl-n-Propyl-Ketone         | 3,482 "kg"    |
| Methylene bisphenyl isocyanate | 3,824 "kg"    |

Report of actual and potential emissions on an emission unit-by-emission unit basis

Ap Compliance Advisor Version 7.0a

File Edit View Analysis Methods Utilities Options Database Window Help

Facility-Wide Potential Emissions Stack Stream(s)

### Facility-Wide Potential Emissions Stack Stream(s)

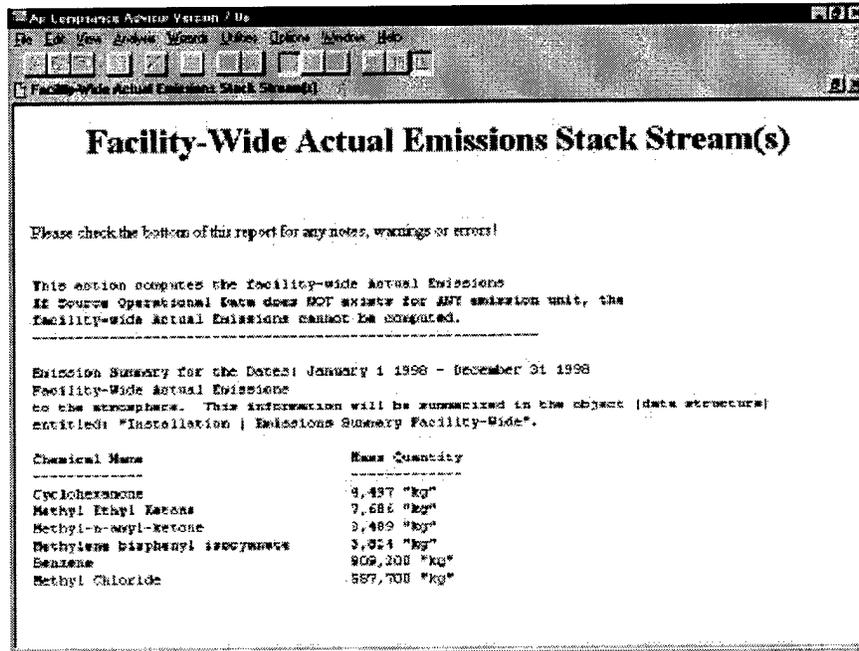
Please check the bottom of this report for any notes, warnings or errors!

This section computes the Facility-Wide Potential Emissions. If Source Operational Data does NOT exist for ANY emission unit, the Facility-wide Potential Emissions cannot be computed.

Emission Summary for the Dates: January 1 1998 - December 31 1998  
 Facility-Wide Potential Emissions  
 to the atmosphere. This information will be summarized in the object [data structure] entitled: "Installation | Emissions Summary Facility-Wide".

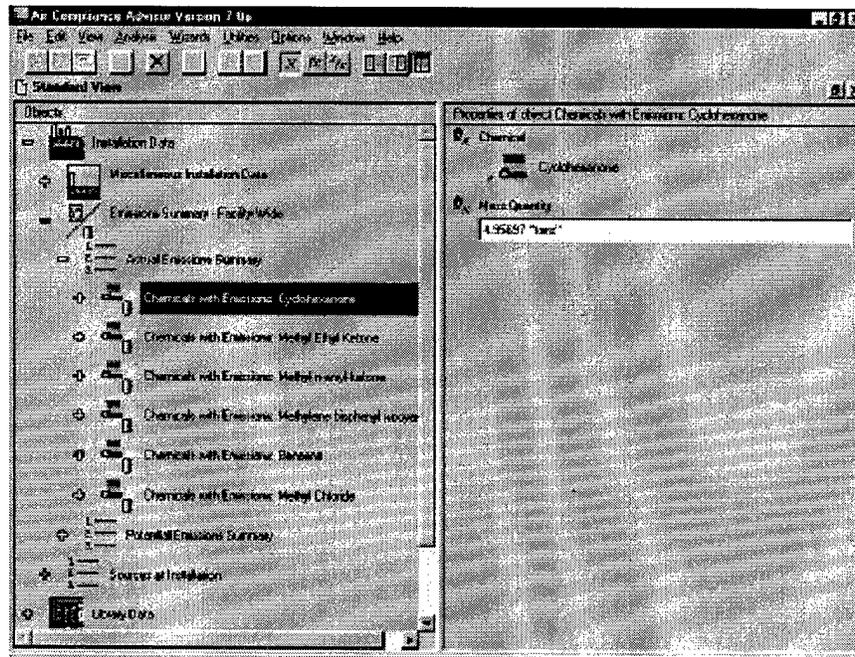
| Chemical Name                  | Mass Quantity |
|--------------------------------|---------------|
| Cyclohexanone                  | 137,400 "kg"  |
| Methyl Ethyl Ketone            | 234,800 "kg"  |
| Methyl-n-Propyl-Ketone         | 106,800 "kg"  |
| Methylene bisphenyl isocyanate | 116,800 "kg"  |
| Benzene                        | 909,200 "kg"  |
| Methyl Chloride                | 587,700 "kg"  |

Facility-wide potential emissions report



### Facility-wide actual emissions report

The From Stack (to Atmosphere) analysis adds emissions data to the Emission Summary-Facility Wide object. Both the Actual Emissions Summary and Potential Emissions Summary object lists will be filled in with a Chemicals with Emissions object for each pollutant emitted from the installation. Each of the Chemicals with Emissions objects will display the installation-wide total emissions for a specific pollutant. The figure entitled "Pollutant summary information in Emissions Summary - Facility Wide object" shows the data entry slot associated with the object Chemicals with Emissions: Cyclohexanone. The information contained in example5.oxl was used to generate this screen. The From Stack (to Atmosphere) analysis also adds emissions data to the Emission Summary for Source object that is contained within each air pollution source object. However, the user-level must be set to Expert to see this data.



Pollutant summary information in Emissions Summary — Facility Wide object

### 4.3 Apply Control Technologies

The Apply Control Technologies analysis option will provide information on the control technologies that can be used to control a user-specified air pollution emission. Users describe an air pollution emission in the Size & Cost Air Pollution Control Technologies object contained in the What-If Scenario core object. The analysis requires data related to:

- the air pollution stream to control
- the annual hours of operation
- the desired reduction efficiency

The ACA will determine which control devices are applicable to a specific scenario and estimate the size and cost (both capital and operating) of the applicable control devices. The ACA can provide information on VOC control devices (i.e., carbon adsorption, regenerative thermal incineration, recuperative thermal incineration, catalytic thermal incineration, refrigerated condensation, and flares) and particulate matter (PM) control devices (i.e., electrostatic precipitators, baghouses, cyclones, mist eliminators, and wet scrubbers).

The ACA contains a large library of control technologies located in the Control Devices Library object contained in the Library Data core object. All of the control technologies in this library will be considered for each study. Note that the Control Devices Library object contains both a list of standard ACA control technologies and a list of any user-defined control technologies. The control technologies in the standard ACA control technologies list are models based primarily upon guidance provided in the US EPA's *OAQPS Control Cost Manual*, William M. Vatauvuk's *Estimating Costs of Air Pollution Control*, Louis Theodore and Anthony J.

Buonicore's *Air Pollution Control Equipment — Volumes I & II*, and the Air & Waste Management's *Air Pollution Engineering Manual*. Other technical references were used (to a much lesser extent) in these models and are typically referenced within the source code as needed. Many of the source code equations are referenced, and since the source code can be viewed from the ACA's graphical user interface, it is relatively straight-forward to follow the "flow" of the various models.

The following general steps are required to size and cost air pollution control technologies:

1. Expand the **Size & Cost Air Pollution Control Technologies** object contained in the **What-If Scenarios** core object.

2. Enter data into the following data entry slots contained in the **Size & Cost Air Pollution Control Technologies** object:

- Study Title (optional, but good for bookkeeping purposes)
- Yearly Hours of Operation
- Duty Cycle (this has a default that will appear after you enter data into the **Yearly Hours of Operation** slot; however, it can be overridden)

The figure entitled "Data entry slots for **Size & Cost Air Pollution Control Technologies** object" shows these data entry slots (note that the information contained in this figure comes from data provided in example problem #1 of this user guide).

3. Double click on the **Stream to Control** object contained in the **Size & Cost Air Pollution Control Technologies** object to expand it. Then, enter information into the following data entry slots:

- Temperature
- Pressure
- Either: Volumetric Flow Rate Actual **OR** Volumetric Flow Rate At STP (not both)
- Moisture Content

It is also a good idea to enter data for all of the other data entry slots at this level for which you have data available. Depending on the control technology under consideration, some or all of these other slots may be required. Users are notified if additional "undefined" data entry slots are required when you attempt to run the analysis. The figure entitled "Data entry slots for **Stream to Control** object" shows some of the data entry slots for the information contained in **example1.oxl**.

4. Add a **Pollution Concentration Data** object to the **Stream to Control** object for each chemical/pollutant that is contained in the stream. This is done by using the toolbar's add button  when the **Pollutant Concentrations** object list is selected.

5. Describe the **Pollution Concentration Data** object by performing the steps below:

Associate it with a chemical/pollutant. Select the undefined object Chemical/Pollutant, then select the add button  on the toolbar and pick one of the available chemicals. (Note that all of the chemicals that are stored in User-Defined chemical database and the Standard ACA chemical database are available from the pick-list.)

Enter one (and only one) measure of the pollutant's concentration\*, namely one of the following data entry slots contained in the Pollution Concentration Data object:

- Volumetric Concentration
- Mass Concentration At STP
- Mass Concentration, Actual OR
- Mass Flow Rate

 **Note:**

\*If users enter more than one measure of pollutant concentration they risk the possibility of these "co-dependent" variables becoming inconsistent. The ACA will, by default, calculate the other three measures of concentration from the one that was entered (provided that the pollutant stream's temperature, pressure and volumetric flow rate are known, as well as the molecular weight of the chemical/pollutant). To see the ACA calculated versions of all of the measures of concentration, go to the "Intermediate" User Level and view the disabled (grayed out) versions of these slots. The Volumetric Concentration data entry slot will not be calculated for pollutants that are particulate matter (PM), as this measure of concentration does not apply to PM pollutants.

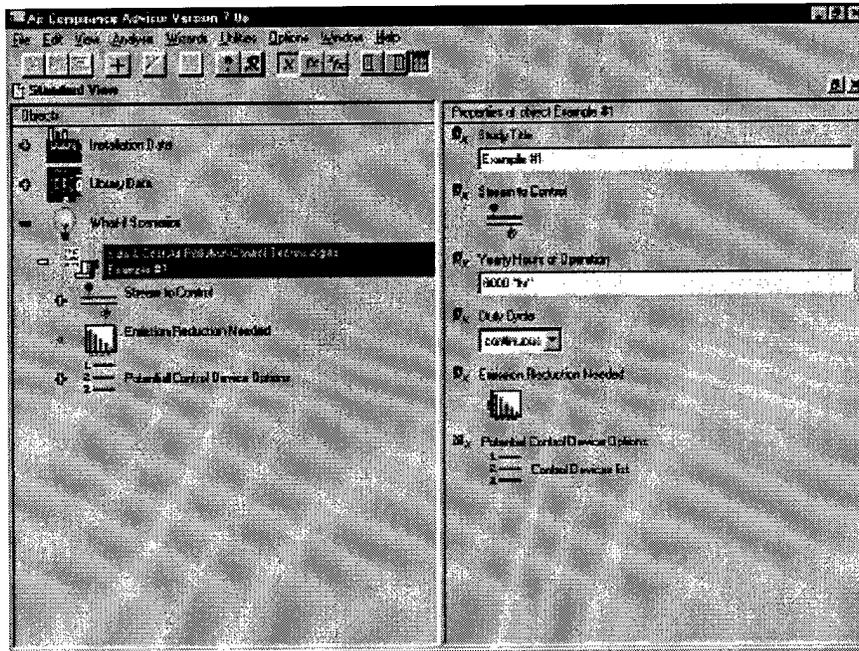
 **Note:**

Repeat this step for each Pollution Concentration Data object added in the previous step.

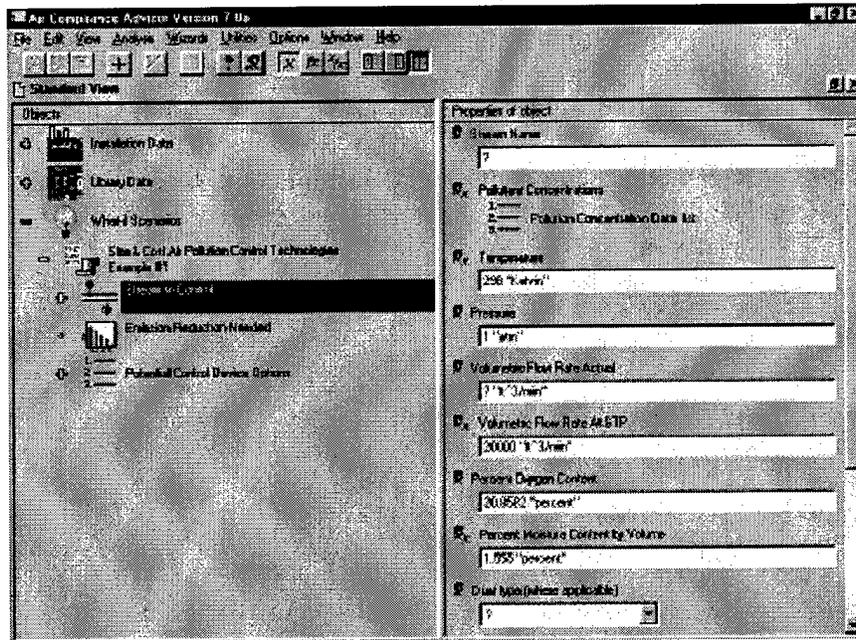
The figure entitled "Pollution Concentration Data objects added to Pollutant Concentrations object list" shows benzene and methyl chloride Pollutant Concentrations objects and the Volumetric Concentration data entry slot filled in.

6. Select the Emission Reduction Needed object contained in the Size & Cost Air Pollution Control Technologies object and enter the percentage of VOC or Particulate Matter reduction needed into the appropriate data entry slot.

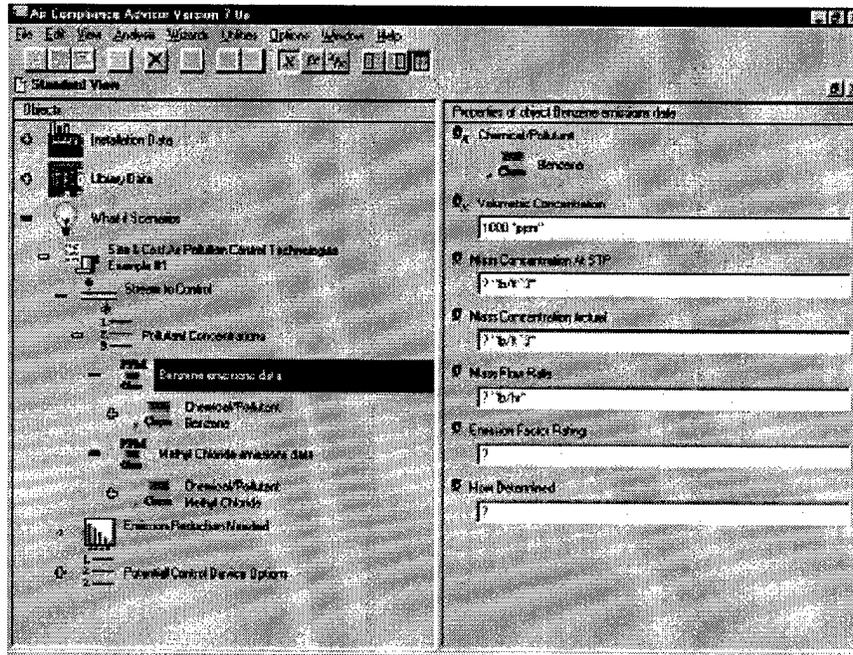
7. Select Analysis | Apply Control Technologies.



Data entry slots for Size & Cost Air Pollution Control Technologies object



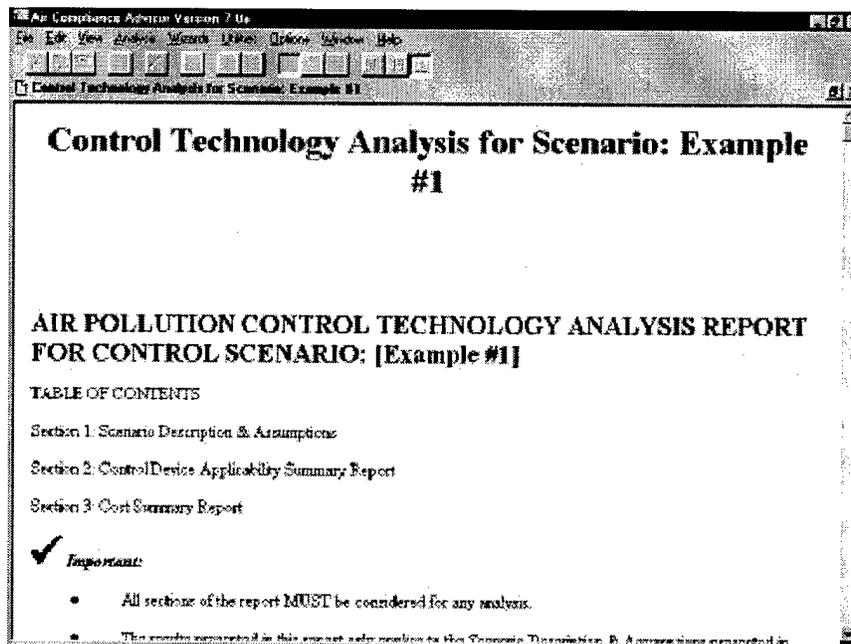
Data entry slots for Stream to Control object



Pollution Concentration Data objects added to Pollutant Concentrations object list

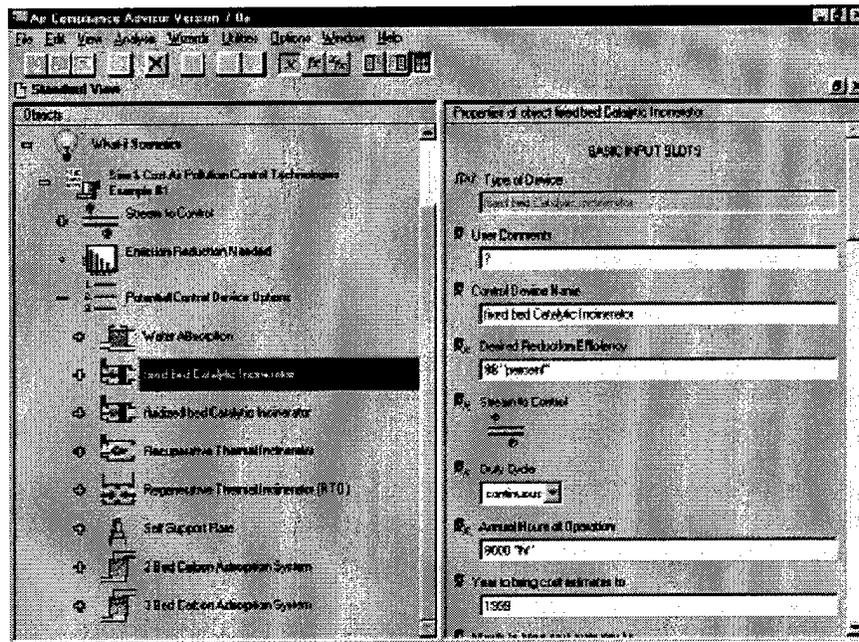
The results of this analysis are two fold:

First, a report will be generated that summarizes: (a) any missing data that is required for any of the potentially applicable control devices, (b) which control devices are potentially applicable, (c) the main reasons why any control devices were determined to be non-applicable, and (d) cost and sizing data for each applicable control device. The figure entitled "Start of Control Device Report" shows an example of the start of this report.



Start of Control Device Report

Secondly, for each control technology that is found to be potentially applicable, an instance of that control device will be placed in the Potential Control Device Options object list contained in the Size & Cost Air Pollution Control Technologies object just below the Emission Reduction Needed object. Users are encouraged to review these instances (as one would a spreadsheet) and modify data entry slots to see how minor modifications affect the size and cost of each control technology. If you would like to get a summary report for the control devices in the Size & Cost Air Pollution Control Technologies/Potential Control Device Options list at any time other than when the analysis option is run, select Run User-Specified Analysis/Action from the "Analysis" Menu and then type in "WriteOutControlDataInWhatIf" at the data entry window. The figure entitled "Slots associated with Potential Control Device Options object" shows an example of a potential control device list.



Slots associated with Potential Control Device Options object

If you would like to view the algorithms for each of the slots, then simply use the "Show slot functions" toolbar button .

More slots, which typically contain more detailed information or intermediate calculations, can be viewed by switching to the "Intermediate" User Level from the Options menu (Options | User Level). You are also able to reprogram any slot (except those that are disabled/grayed out) by going into the slot definition window and reprogramming as desired. If you wish to reprogram any slots, you should first become familiar with the "End User Extendibility in the ACA" — refer to the "Basic End User Extendibility" Section and Appendix C in this User Guide.

In addition, the reader is urged to review the example problems #1 (gaseous emissions) and #2 (PM emissions) in Section 6 to better understand the data required to model these two types of emissions.

## 4.4 Prototype Analyses

### *Check for Applicable Regulations*

The Analysis | Prototype Analyses | Check for Applicable Regulations analysis compares the emission units that the user has defined with the regulations in the ACA Regulations Database. This analysis will rule out those regulations that do not apply to individual emission units. A report is generated that summarizes the findings. The potentially applicable regulations are listed with the emission units under the **Potentially Applicable Regulations** object associated with each emission unit (which can be viewed at the Intermediate user level). The ACA Regulations Database contains all of the NSPS, NESHAP, and MACT standards, as well as the NSR/PSD and NSR/NAA federal regulations.

The ACA defines potentially applicable regulations/emission unit pairs as cases where the applicability of the regulation could not be ruled out. In the case of an emission unit for which little information is entered, many regulations will be listed as being potentially applicable. In order to take advantage of the ACA Regulations Database and comparison checking, it is necessary to describe the emission units in as much detail as possible.

As this analysis is effectively based upon the "ruling out" of regulations, only a single negative comparison is needed to indicate that a regulation is not applicable. While numerous potential comparisons are made, the quickest ways to rule out regulations are to specify:

- the BLIS Process Code (a default code is set for every emission unit)
- SCC values,
- construction, re-construction, modification, and start-up dates, and
- pollution concentration data

To perform this analysis, select from the menu: Analysis | Prototype Analyses | Check for Applicable Regulations. An example using the Check for Applicable Regulations analysis is given in example problem #5 in Section 6 .

### *Check for Applicable Pollution Prevention (P2) Opportunities*

The Check for Applicable Pollution Prevention Opportunities analysis option is similar in nature to the Check for Applicable Regulations analysis. An ACA database of P2 Opportunities is compared with the emission units defined by the user. The ACA reports any P2 Opportunities that could not be ruled out for the emission units specified. A report is generated that summarizes the findings. The potentially applicable P2 Opportunities are listed with the emission units under the **Potentially Applicable Pollution Prevention Opportunities** object list (which can be viewed at the Intermediate user level).

The ACA defines potentially applicable P2 Opportunities/emission unit pairs as cases where the applicability of the P2 Opportunities could not be ruled out. In the case of an emission unit for which little information is entered, a number of P2 Opportunities will be listed as being

potentially applicable. In order to take advantage of the ACA P2 Opportunities database and comparison checking, it is best to describe the emission units as completely as possible.

To perform this analysis, select from the menu: **Analysis | Prototype Analyses | Check for Applicable Pollution Prevention Opportunities**. An example using the **Check for Applicable Pollution Prevention Opportunities** option is presented in example problem #5 in Section 6 .

### ***Check for Applicable Suggestions***

The **Check for Applicable Suggestions** analysis option is similar in nature to the **Check for Applicable Regulations** and the **Check for Applicable Pollution Prevention Opportunities** analyses described in the previous section. An ACA database of Suggestions is compared with the emission units defined by the user. Suggestions that could not be ruled out for the emission units specified are reported by the ACA. A report is generated that summarizes the findings, and potentially applicable Suggestions are listed with the emission units under the **Potentially Applicable Suggestions** object list. The idea behind the Suggestions Library is to provide a database of suggestions/ideas (other than pollution prevention alternatives) that could prove useful to an environmental engineer in the course of managing the emission units at a facility. Currently the Suggestions Library contains "suggestions" on techniques to reduce NOx emissions for external boilers and a few emerging VOC control technologies.

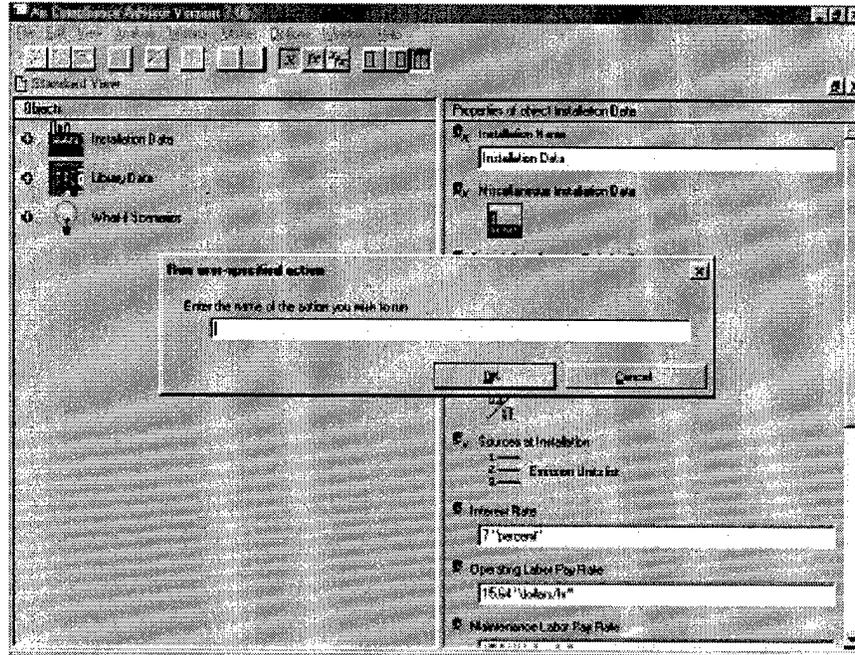
The ACA defines potentially applicable Suggestions/emission unit pairs as cases where the applicability of the Suggestions could not be ruled out. In the case of an emission unit for which very little information is specified, a number of Suggestions will be listed as potentially applicable. To take advantage of the Suggestions database and comparison checking, it is necessary to describe the emission units in as much detail as possible.

To perform this analysis, select from the menu: **Analysis | Prototype Analyses | Check for Applicable Suggestions**. An example using the **Check for Applicable Suggestions** analysis is given in example problem #5 in Section 6 .

If additional detail and guidance is required for any of the prototype analyses, please contact the program developers, as a draft tutorial and additional reports pertaining to these analyses can be made available.

## **4.5 Run User-Specified Analysis/Action**

The Analysis option titled **Run User-Specified Analysis/Action** is used to run any actions that exists in the EXL source code, including those that are not already linked to a specific menu option, as well as any user defined actions. Because the user of the ACA can extend the functionality of this program by adding their own data structures, functions, procedures and actions, this feature is the means by which user-defined actions can be run.



### Run User-Specified Analysis/Action

#### Note:

Refer to Appendix C of this User Guide for guidance on how to add user-defined actions to the ACA.

In order to identify all of the actions that exist in the ACA, you can select the menu View | Advanced Views | Global Actions. Selecting this menu option will provide a long list of all of the actions in the ACA (note that all actions in the ACA are global).

Any actions that are run from Run User-Specified Analysis/Action must be done at the risk of the user. The developers of the ACA have made the tried-and-true actions in the ACA available directly from the menu bar. It should be noted that some of the actions in the ACA are actions that were valid for previous versions of the ACA and are no longer used and could potentially cause havoc with your data. The bottom line is this: Be careful running an action that you are not familiar with. The following are a some useful actions that are not linked to a menu option but can be run from Run User-Specified Analysis/Action and can be used with confidence:

**English** — This action will instruct the ACA to write out all output reports in English units (e.g., "lb", "BTU").

**Metric** — This action will instruct the ACA to write out all output reports in English units (e.g., "kg", "Joule").

**Reset\_ACA\_Control\_Library** — This action will re-initialize the control device instances that reside in the Library Data | Control Devices Library | Standard ACA Control Device Library object. This action would be the shortcut method for resetting all of the control device objects at once. While the user is encouraged to place any customized control device objects in the Library Data | Control Devices Library | User-Defined ACA Control Device Library, it

seems likely that those in the Standard ACA database could actually be versions modified by the end-user. Note that any customized control technologies that are stored in this list prior to running this action will be lost (to prevent this, copy them over to the **User-Defined Library**).

**LoadACA\_Chemicals** — This action will re-load the original Standard ACA Chemicals Library. Like the *Reset\_ACA\_Control\_Library* action listed above, this action is used to get the original ACA version of the database back. Caution should be used in that any chemical properties that have been modified will be lost. Further, any chemicals that have been added to the Standard ACA Chemicals Library will be lost, copy any new chemicals over to the **User-Defined Chemicals Library** to prevent these new chemicals from being lost.

**WriteOutControlDataInWhatIf** — This action will write out a report that summarizes the size and costing data for all of the control technologies that are listed in the **What-If Scenario | Size & Cost Air Pollution Control Technologies | Potential Control Device Options** list.

## 5. Using the ACA Wizards

The information contained in this section, which is also available from the Help | Tips on Using the ACA Wizard menu option, provides help for the three ACA Wizards and the Chemical Properties Conversion Worksheet.

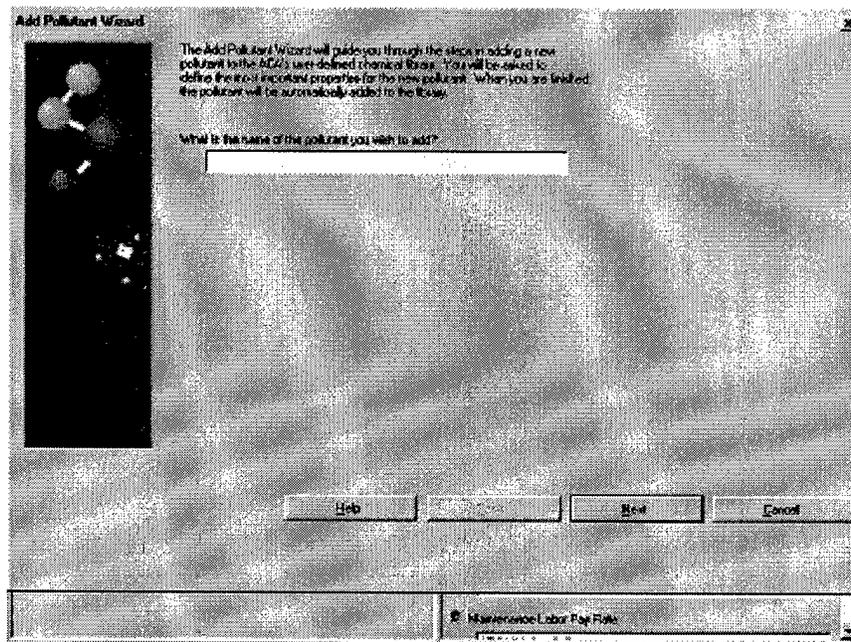
### 5.1 Help for the Add VOC Chemical to User-Defined Chemical Library Wizard

This Wizard allows the user to add VOC chemicals to the User-Defined Chemical Library. The chemical properties that are required for costing and sizing air pollution control technologies are the only chemical parameters that can be entered via this Wizard. Additional chemical properties can be added later by editing the Chemical object directly from the ACA's standard user interface.

For more information on the chemical properties listed here, including definitions, how they are used in the ACA and how to obtain values, see the help option "Tips on Obtaining and Estimating Chemical Property Data" available from the main menu under "Help."

#### *Screen #1: Chemical Name Screen*

The first screen of this Wizard will prompt the user for a name for the new pollutant - please select a unique name.



Screen #1 for Adding a VOC Chemical to the User-Defined Chemical Library Wizard

#### *Screens #2-#4: Enter Chemical Properties Screen*

The chemical properties that are required to calculate the cost and applicability of air pollution control technologies should be entered on screens #2–#4. In order to investigate all of the VOC control technologies, all properties must be entered. Listed below are the various chemical properties that can be entered in these screens, as well as the air pollution control technology calculations that require these parameters. For more information on the chemical properties listed here, including definitions, how they are used in the ACA and how to obtain values, see the help option "Tips on Obtaining and Estimating Chemical Property Data" available from the main menu under "Help."

- Antoine Constants for Vapor Pressure (i.e., A, B, C): Refrigerated Condensers, Carbon Adsorbers
- Boiling Point: Refrigerated Condensers, Carbon Adsorbers
- Critical Temperature: Refrigerated Condensers
- Diffusion Coefficient, Liquid in Water: Gas Absorbers
- Diffusion Coefficient, Vapor in Air: Gas Absorbers
- Gas Phase Heat Capacity: Refrigerated Condensers
- Heat of Combustion: Thermal Incinerators and Flares
- Heat of Condensation: Refrigerated Condensers, Carbon Adsorbers
- Henry's Law Constant: Gas Absorbers
- Index of Refraction: Carbon Adsorbers
- Is Halogenated?: Thermal Incinerators
- LEL (Lower Explosive Limit): Thermal Incinerators, Flares, and Carbon Absorbers
- Liquid Density (at STP): Carbon Adsorbers
- Molecular Weight: All VOC Control Devices
- Vapor Pressure @ 293 K: Carbon Adsorbers

**Add Pollutant Wizard**

Enter values for the following temperature-independent chemical properties of New\_chemical. These properties are required for control technology/analysis.

Molecular Weight:  gram/gram **Required**

Boiling Point (at standard pressure):  K

Liquid Density (at STP):  lb/ft<sup>3</sup>

Critical Temperature:  R

Vapor Pressure (at 25K):  psi

Heat of Combustion (at boiling point):  BTU/lb

Heat of Combustion (at phase, volumetric at STP):  BTU/ft<sup>3</sup>

Help Back Next Cancel

Maintenance Labor Pay Rate

Screen #2 for Adding a VOC Chemical to the User-Defined Chemical Library Wizard

**Add Pollutant Wizard**

Enter values for the following temperature-independent chemical properties of New\_chemical.

Relative Index:

Lower Explosive Limit (LEL):  ppm

Is Flammable?

Is Sublimated?

Is Aqueous?

Is Polymerized?

Help Back Next Cancel

Maintenance Labor Pay Rate

Screen #3 for Adding a VOC Chemical to the User-Defined Chemical Library Wizard

Screen #4 for Adding a VOC Chemical to the User-Defined Chemical Library Wizard

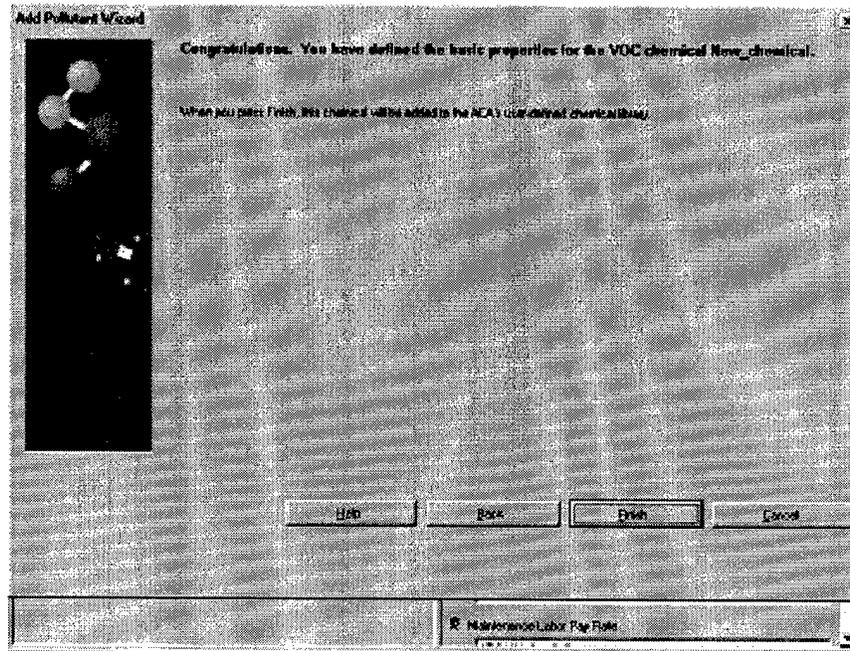
#### Screen #5: Finish Screen

Once you have arrived at this screen, you have entered the data required to add a chemical/pollutant to the User-Defined Chemical Library. Note that the data entered using this Wizard includes only those chemical parameters that are required to perform an air pollution control technology analysis. Additional chemical data can be added for this chemical by directly editing the chemical from the standard ACA interface. For more information on editing data from the standard ACA interface, see Section 3.

By selecting the **Finish** button the ACA will add this chemical to the User-Defined Chemical Library.

By selecting the **Back** button you can review and/or modify the data you have entered before the chemical/pollutant is added to the User-Defined Chemical Library.

By selecting the **Cancel** button you will be returned to the standard ACA interface and the chemical/pollutant will **not** be added to the User-Defined Chemical Library.



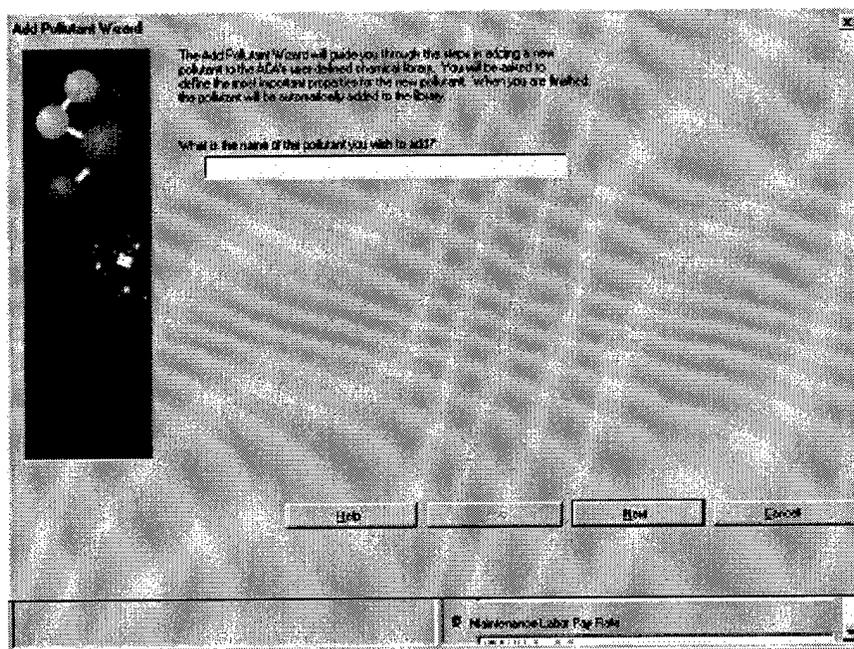
Screen #5 for Adding a VOC Chemical to the User-Defined Chemical Library Wizard

## 5.2 Help for the Add PM Pollutant to User-Defined Chemical Library Wizard

When adding a new Particulate Matter (PM) pollutant to the ACA User-Defined Chemical Library, the "Add PM Pollutant to User-Defined Chemical Library" Wizard can be used.

### *Screen #1: Chemical Name Screen*

The first screen of this Wizard will prompt the user for a name for the new pollutant — please select a unique name.



Screen #1 for Adding a PM Pollutant to the User-Defined Chemical Library Wizard

### ***Screen #2: PM Property Entry Screen***

The second screen of this Wizard, the "PM Pollutant Properties Screen," requires the user to enter data on those properties that are required for determining the cost and applicability of a variety of PM air pollution control technologies. Definitions of the various properties that are required to be end-user supplied are given below.

#### ***Is Water Soluble***

Specify if the PM is soluble in water.

#### ***Particulate Phase***

Specify if the PM is a liquid or a solid.

#### ***PM Density***

Enter the density of the PM.

#### ***PM Size Distribution***

Required for: All PM control technologies.

Definition: This parameter represents the aerodynamic particle size distribution for pollutants that are classified as particulate matter (PM). If a chemical/pollutant is a gas, do NOT enter any data in this slot. For PM data, it is advised that at least five (5) sets of data (particle size vs. percent mass above) be entered. This data will be used to calculate: (1) mass median diameter; (2) cut diameter; and (3) geometric standard deviation. These parameters are typically required to determine the applicability of PM air pollution control devices. The ACA will assume a best-fit

log-normal aerodynamic particle size distribution (this gives a straight line on log-probability paper when plotted) in the calculation of the mass median diameter, the geometric standard deviation, and the cut diameter.

| Aerodynamic diameter, um | Mass Fraction, % |
|--------------------------|------------------|
|                          |                  |
|                          |                  |
|                          |                  |
|                          |                  |
|                          |                  |

Screen #2 for Adding a PM Pollutant to the User-Defined Chemical Library Wizard

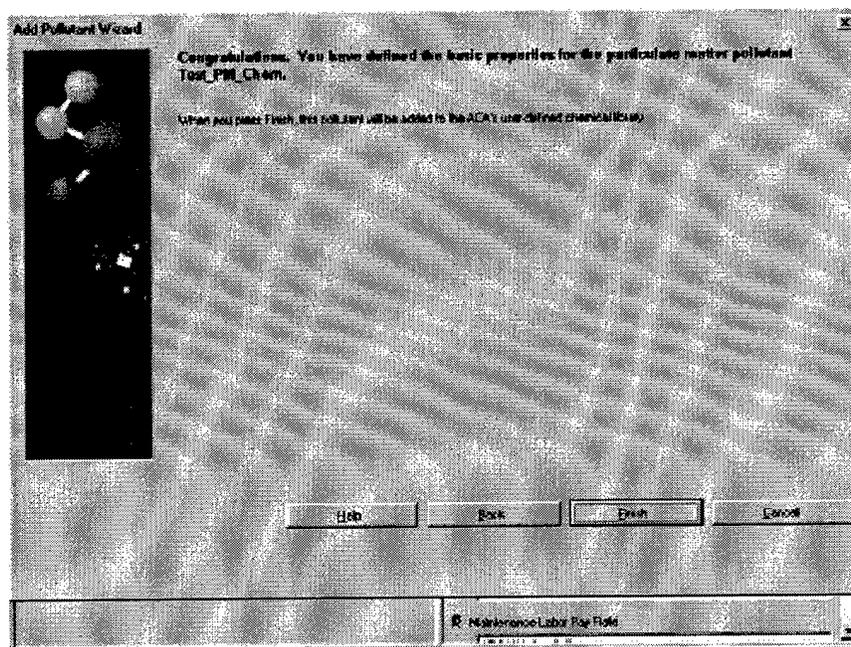
### Screen #3: Finish Screen

Once you have arrived at this screen, you have entered the data required to add a chemical/pollutant to the User-Defined Chemical Library. Note that the data entered using this Wizard includes only those chemical parameters that are required to perform an air pollution control technology analysis. Additional chemical data can be added for this chemical by directly editing the chemical from the standard ACA interface.

By selecting the Finish button the ACA will add this chemical to the User-Defined Chemical Library.

By selecting the Back button you can review and/or modify the data you have entered before the chemical/pollutant is added to the User-Defined Chemical Library.

By selecting the Cancel button you will be returned to the standard ACA interface and the chemical/pollutant will **not** be added to the User-Defined Chemical Library.



Screen #3 for Adding a PM Pollutant to the User-Defined Chemical Library Wizard

## 5.3 Help for the Apply Air Pollution Control Technologies Wizard

The "Apply Air Pollution Control Technologies" Wizard is designed to prompt the user for the minimum data required to investigate potentially applicable air pollution technologies for a user-defined air pollution stream containing either particulate matter (PM) or volatile organic compounds (VOCs).

### *Screen #1: Enter Parameters for an Air Pollution Control Technology Analysis*

The first Wizard screen contains the general parameters associated with the analysis, including:

#### *Case Study Title*

This parameter is used to describe the analysis you are performing. When using the Wizard, the title you enter can be as long as you wish, but it cannot include a carriage return. While this parameter is not required, it can be helpful in documenting your scenarios.

#### *Yearly Hours of Operation*

This parameter is used to describe the total yearly hours of operation that the air pollution stream will be operating. This parameter is used in the calculation of the annual operating costs.

#### *Duty Cycle*

This parameter is a measure of how often the emission stream is operated. This parameter is related to the Yearly Hours of Operation parameter and is used in determining the relative applicability of the various control technologies for a specific pollutant stream (e.g., some

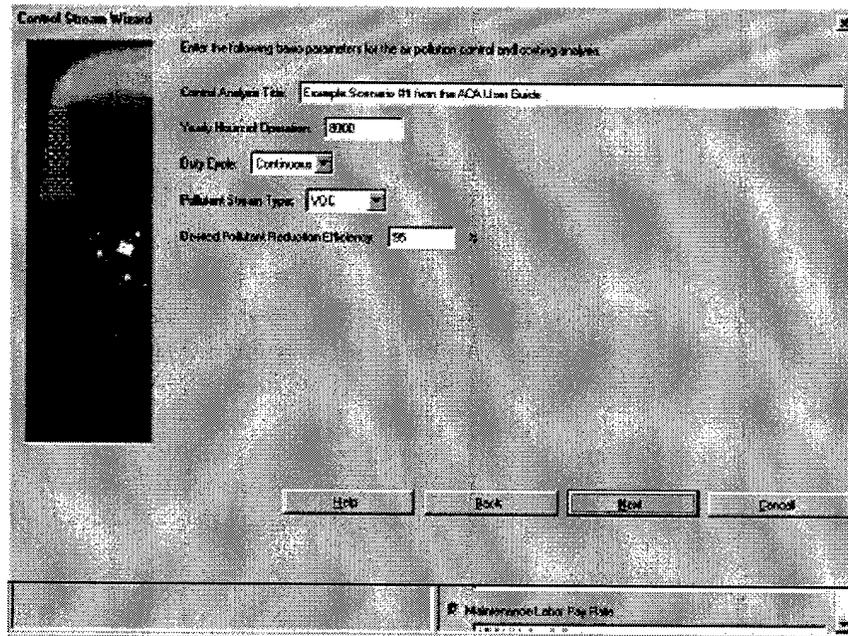
technologies perform better than others when they are operated continuously). As a general rule of thumb, 7000 hours/year or more can be considered "continuous," from 1000 hour/year to 7000 hours/year can be considered "shift," and less than 1000 hours/year can be considered "intermittent."

### *Pollution Type*

This parameter is used to select the type of pollutant you are interested in controlling. Currently, the ACA allows for the control of Volatile Organic Compounds (VOC) or Particulate Matter (PM) pollutants.

### *Desired Pollutant Reduction*

This parameter defines your desired percent reduction in either VOC or PM pollution. For many control technologies this parameter will have an impact on both the capital and operating costs of control technologies (i.e., the higher the desired reduction, the higher the costs to control).



**Screen #1 of the Apply Air Pollution Control Technologies Wizard**

### ***Screen #2: Pollutant Stream Properties Screen***

The layout of this screen will depend upon which pollution type (VOC or PM) was selected in the first Wizard screen. There will be some similarities between the VOC and PM screens, so those parameters that are common to both types will be presented first, followed by those that are just for VOC pollutant stream control and then those that are just for PM pollutant stream control.

## ***Parameters Common to Both PM and VOC Pollutant Stream Properties Screen***

### ***Temperature***

This parameter describes the temperature of the pollution stream. Temperature is required for numerous calculations related to sizing air pollution control equipment, including, for example, converting between actual and standard conditions and the determination of energy required to raise the pollutant stream temperature (for combustion) or lower the pollutant stream temperature (for condensation).

### ***Pressure***

This parameter describes the pressure of the pollution stream. Pressure is required for various calculations related to sizing air pollution control equipment, including, for example, converting between actual and standard conditions.

### ***Volumetric Flow Rate***

This parameter describes the volumetric flow rate of the entire air pollution stream (i.e., pollutants + air + other gases). The volumetric flow rate can be specified at either actual operating temperature and pressure or the equivalent volumetric flow rate at standard temperature and pressure. The volumetric flow rate combined with the pollutant mass concentration data is used to calculate the pollutant mass flow rate for evaluating air pollution control technologies, as well as converting between various measures of pollution concentration.

### ***Moisture Content***

This parameter describes the amount of moisture by volume in the air pollution stream. The moisture content is used in the evaluation of wet scrubbers, carbon adsorbers, and baghouses.

## ***Additional Parameters Applicable Only to VOC Pollutant Stream Properties Screen***

### ***Oxygen Content***

This parameter describes the amount, on a percentage basis, of oxygen in the air pollution stream. The percent of oxygen present in the stream is used to determine if auxiliary air is required to complete combustion when evaluating thermal incineration control devices. Note: Normal outdoor air has approximately 20.9% oxygen.

### ***Particulate Matter in Stream***

This parameter is used to qualitatively specify if any particulate matter (PM) is present in the air pollution stream. This parameter is of primary importance when evaluating the relative applicability and ranking of VOC air pollution control technologies as some VOC air pollution control technologies do not perform well if there is any PM present, while other devices perform fine with some PM present.

***Additional Parameters Applicable Only to PM Pollutant Stream Properties Screen***

*Stream Has...*

- Stream has Acid Gases
- Stream has Alkalis
- Stream has Fluorides
- Stream has Mineral Acids
- Stream has Metals
- Stream has Condensable Metals

This group of parameters that appears when using the ACA Wizard for the control of PM pollutant stream emissions are all used to determine which types of filter bags (for baghouses) would be applicable.

*Stream is Corrosive*

This parameter is used to determine if a stainless steel or fiberglass construction would be required, for those control technologies for which it is an option.

*Dust Type*

This parameter is used to qualitatively describe the PM (e.g., dust) that is present in the air pollution stream. This data is used to determine the air-to-cloth ratio for baghouses control technologies.

Screen #2 of the Apply Air Pollution Control Technologies Wizard (VOC example)

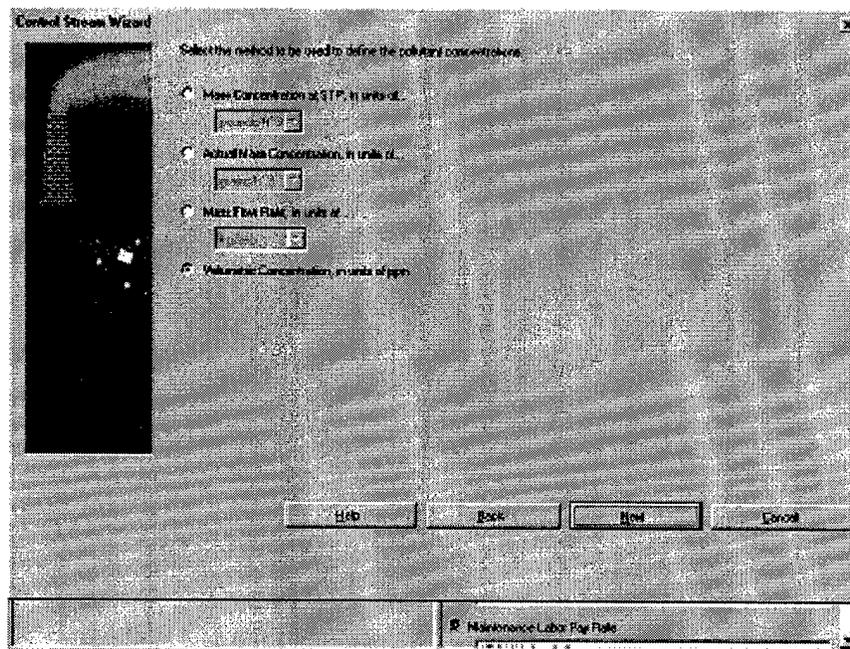
**Screen #3: Enter Method Used to Define Pollution Concentration Data Screen**

This screen gives the user the option of which method to use to define their air pollution concentration data (which will be entered in the next screen). There are either three or four options for specifying the method for entering the pollution concentration data, depending upon the type of pollution you are trying to control. VOC pollution streams can be specified with four methods, while PM pollution streams can be specified with only three methods. The pollution concentration data can be specified for both VOC and PM in the following methods:

- Mass Concentration at STP (with optional units)
- Actual Mass Concentration (with optional units)
- Mass Flow Rate (with optional units)

For VOC pollution streams, the pollution concentration data can also be specified as:

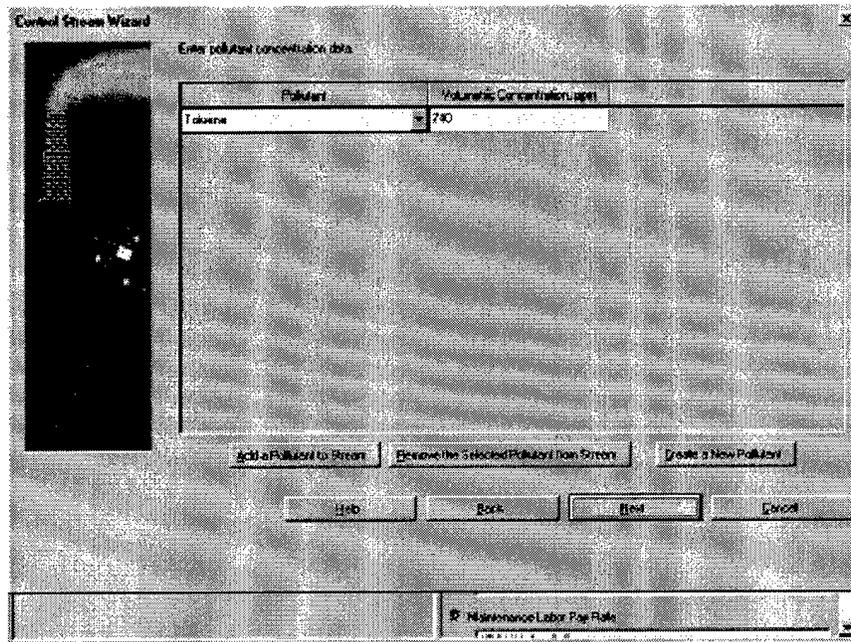
- Volumetric Concentration, in units of ppm (i.e., parts per million)



Screen #3 of the Apply Air Pollution Control Technologies Wizard (VOC example)

***Screen #4: Enter Pollutant Concentration Data Screen***

This screen allows the user to add any number of pollutants to the stream and the associated concentration data (using the method and units specified in the previous screen). Only VOC pollutants can be added when analyzing VOC control technologies and only PM pollutants can be added when analyzing PM control technologies. The pull-down menu with the "Pollutant" parameter allows the user to view all of the appropriate pollutants (i.e., VOC or PM depending upon the type of pollutant you are investigating) in the Standard ACA Chemical Library and in the User-Defined Chemical Library. The user can also add new pollutants to the User-Defined Chemical Library right from this screen, which is described below.



Screen #4 of the Apply Air Pollution Control Technologies Wizard

#### ***Screen #5: Cost Data Screen***

This screen contains the user-supplied data that is used by the ACA to estimate the cost for the various control technologies. A summary of the variables that are required by the ACA and that are presented in this screen are listed below.

#### ***Interest Rate***

The interest rate used in the ACA is the annual "real private rate of return" or the annual pretax marginal rate of return that could be realized on a private investment. The relationship between this interest rate ( $i$ ) and the annual nominal interest rate ( $i_n$ ) is given by the relationship that also includes the annual inflation rate ( $r$ ), as  $(1+i_n) = (1+i)(1+r)$ .

#### ***Year to Bring Cost to & Month/Quarter to Bring Cost to***

These two parameters represent the year and month for which the control device cost estimates will be valid. Data can be escalated or de-escalated to the year and month of interest. Currently the ACA has cost data for most control devices from January, 1989 to Fourth Quarter, 1997. The options for the parameter "Month/Quarter to bring cost to" include:

- January
- February
- March
- First Quarter
- April

- May
- June
- Second Quarter
- July
- August
- September
- Third Quarter
- October
- November
- December
- Fourth Quarter

*Operating Labor Pay Rate*

This is the wage rate that would be paid to personnel who would operate air pollution control equipment.

*Maintenance Labor Pay Rate*

This is the wage rate that would be paid to personnel who would maintain air pollution control equipment. Note that maintenance labor is typically higher than operating labor rates (typically by about 10%).

*Electricity Cost*

This parameter is the cost that the installation pays for electricity.

*Cooling Water Cost*

This parameter is the cost that the installation pays for water.

*Water Disposal Cost*

This parameter is the cost that the installation pays for the disposal of waste water.

*Steam Cost*

This parameter is the cost that the installation pays for steam.

*Dust Disposal Cost*

This parameter is the cost that the installation pays for the disposal of dust.

## Natural Gas Cost

This parameter is the cost that the installation pays for natural gas.

| Parameter                    | Value   | Unit          |
|------------------------------|---------|---------------|
| Interest Rate                | 7       | %             |
| Escalation rate estimates to | 1399    | First Quarter |
| Operating Labor Cost         | 15.64   | \$/hr         |
| Maintenance Labor Cost       | 17.21   | \$/hr         |
| Electricity Cost             | 0.08    | \$/kwh        |
| Cooling Water Cost           | 0.2     | \$/gallon     |
| Waste Water Disposal Cost    | 1.8     | \$/gallon     |
| Steam Cost                   | 4.65    | \$/lb         |
| Dust Disposal Cost           | 25      | \$/ton        |
| Natural Gas Cost             | 0.00000 | \$/M3         |

Screen #5 of the Apply Air Pollution Control Technologies Wizard

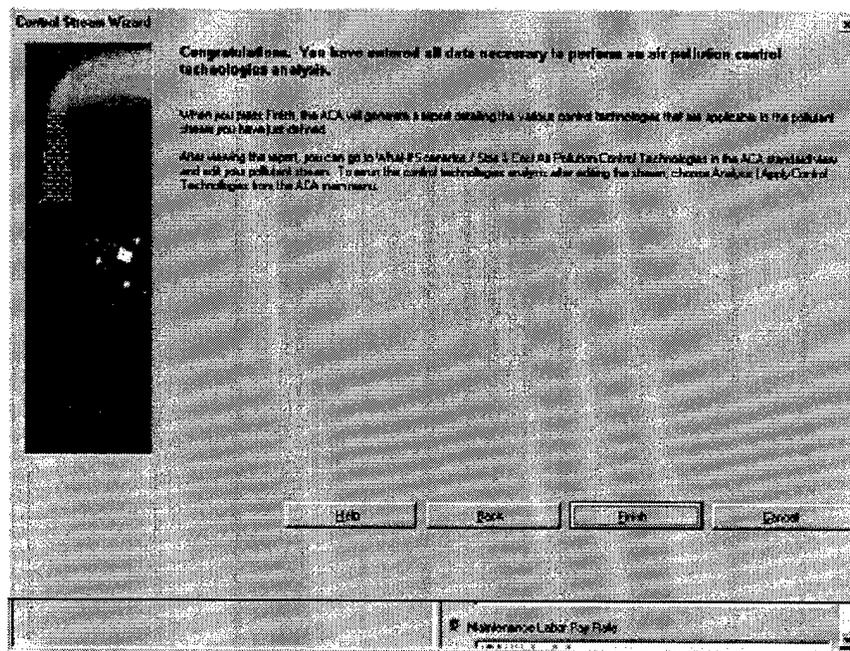
### Screen #6: Finish Screen

Once you have arrived at this screen, you have entered all of the data required to perform an air pollution control technology analysis.

By selecting the **Finish** button, the ACA will perform the air pollution control technology analysis and generate a report summarizing the results of the analysis.

By selecting the **Back** button you can review and/or modify the data you have entered before running the analysis.

By selecting the **Cancel** button you will be returned to the standard ACA interface and the air pollution control technology analysis will **not** be performed.



Screen #6 of the Apply Air Pollution Control Technologies Wizard

## 5.4 Help for the Chemical Properties Conversion Worksheet

Some references (e.g., the NIST www chemical database) provide many chemical properties on a basis that is different than the ACA allows (e.g., molar-basis vs. mass-basis). Therefore, a chemical property conversion worksheet is available in the ACA. To utilize this worksheet you must:

1. Set Options | User Level to "Expert"
2. Open up the Library Data | Chemicals Library | Property Conversion Worksheet

This object will allow you to convert the following chemical parameters to a units basis that is consistent with the ACA chemical properties database:

- Gas Phase Heat Capacity from a data given on a molar-basis to the equivalent volumetric based value that is required by the ACA. A conversion is also made from the molar-based data to the equivalent mass-based data for your reference purposes. Requires that the molecular weight value also be provided.
- Heat of Condensation/Vaporization at Boiling Point from a data given on a molar-basis to the equivalent mass-based value that is required by the ACA. Requires that the molecular weight value also be provided.
- Antoine vapor pressure constants (i.e., A, B, C) from data provided in the format of:
  - $\log_{10}(P) = A - (B / (T + C))$  : with P in "bar" and T in degrees "K" (as given with NIST data)

to the format used in the ACA of:

- $\log_{10}(P) = A - (B / (T + C))$  : with P in "mmHg" and T in degrees "C"
- Heat of Combustion in the gas phase @ STP from data given on molar-basis to the equivalent mass-based value that is required by the ACA. Requires that the molecular weight value also be provided.
- Henry's Law Constant from data given on a mass-basis to the equivalent volume-based value that is required by the ACA. Requires that the liquid density value also be provided.

✓ **Important:**

Molecular Weight is required to convert the Gas Phase Heat Capacity data, the Heat of Condensation/Vaporization data, and the Heat of Combustion data, while the Liquid Density is required to convert Henry's Law Constant data.

## 6. Example Problems

Five different scenarios are presented in this section in order to familiarize the user with the various analysis options available in the ACA and the mechanics of entering data into the ACA. The first four examples are relatively simple, utilizing one or two specific features of the ACA. The final example is more detailed.

For each example a corresponding input file is provided with the installation of the ACA. The names of the input files directly correspond to the number of the example problems in this section (e.g., the input file for example #1 is named `example1.xml`). All input files are located in the inputs folder (e.g., `C:\aca\inputs\example1.xml`) and can be accessed from the menu bar under **File | Open**. Users are encouraged to enter the data as presented in each section without referring to the input files. The completed input files are provided as a reference.

To enter data, start the ACA, or within an existing session start a new session by loading the library data from the main menu (i.e., **File | New**). **Remember to save data from an existing session prior to starting a new session, otherwise this data will be lost.**

The instructions provided for entering data into the ACA are quite detailed. After the first example or two this level of detail may seem excessive. This is a good indication that one understands the data structure of the ACA. While it may take some time to become familiar with the (object oriented) data structure used in this program, once users are familiar with it, data management and data entry become quick and easy.

### 6.1 Example Scenario #1: Costing and Sizing VOC Control Devices

#### *Background/Objective*

The objective of this example is to determine the size and cost of applicable VOC control devices using the **Analysis | Apply Control Technologies** menu option. Users enter emissions data in **What-if Scenarios**. See the "Apply Control Technologies" Section for a generalized discussion of the Apply Control Technologies analysis option.

#### *Input Parameters/Assumptions*

To comply with an operating permit, assume that the emissions from a process need to be reduced by 95%. The following information summarizes the details of the process emissions:

| Parameter                 | Value      |
|---------------------------|------------|
| <b>Operating schedule</b> |            |
| Yearly hours of operation | 8,000 hr   |
| Duty Cycle                | continuous |

### Stream parameters

|                            |                             |
|----------------------------|-----------------------------|
| Volumetric flow rate @ STP | 20,000 ft <sup>3</sup> /min |
| Temperature                | 298 Kelvin                  |
| Percent oxygen             | 21                          |
| % Moisture content         | 1.555                       |

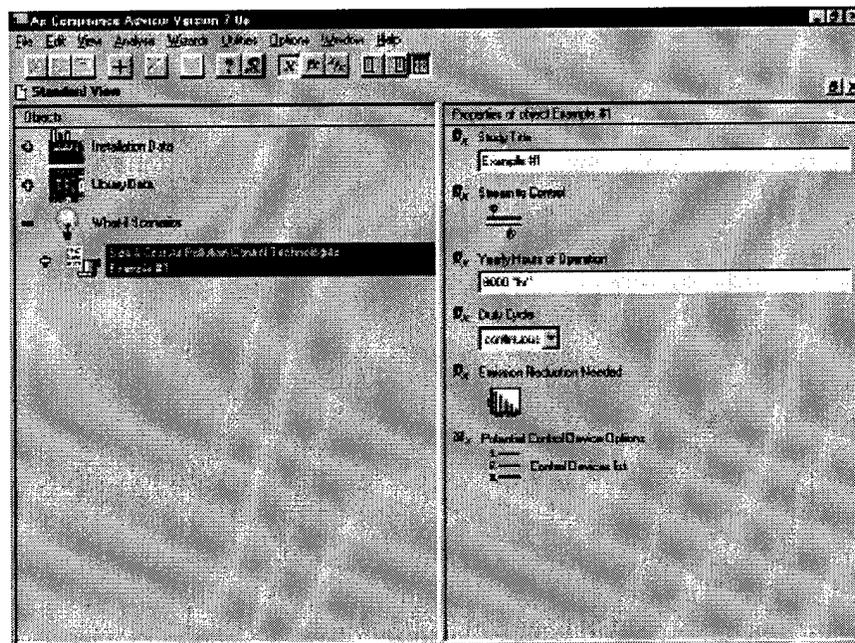
### Pollutant concentrations

|                 |           |
|-----------------|-----------|
| Benzene         | 1,000 ppm |
| Methyl chloride | 1,000 ppm |

### Input parameters for example #1

#### Data Entry Into the ACA

The data provided in the table above is entered into the What-If Scenarios core object under the Size & Cost Air Pollution Control Technologies object. Double click on the Size & Cost Air Pollution Control Technologies object to expand it and then enter the yearly hours of operation and the duty cycle in the appropriate data entry slots in the Properties of Object pane as shown in the figure entitled "Entering data in the Size & Cost Control Technologies object"



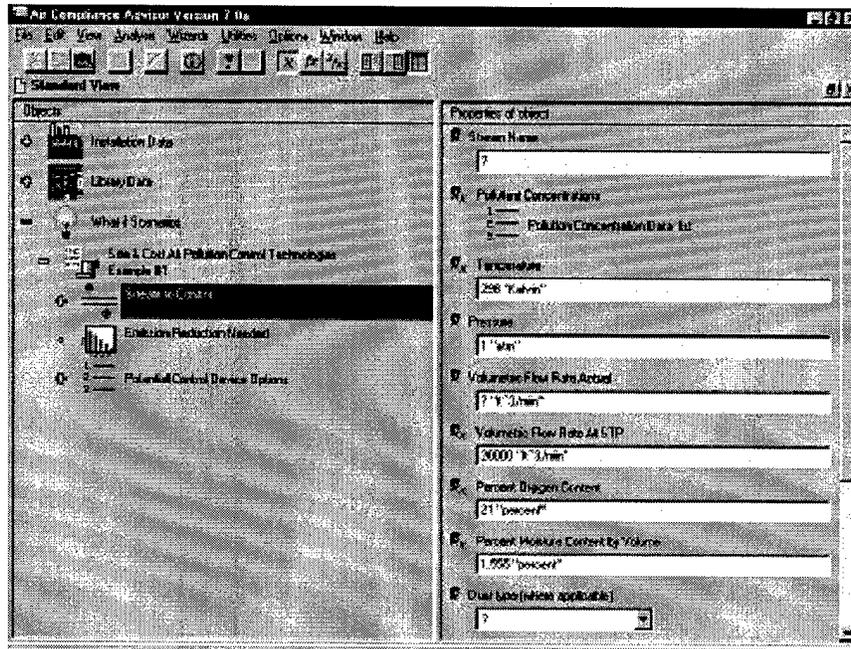
Entering data in the Size & Cost Control Technologies object

Next, enter the stream parameters listed above into their appropriate data entry slots contained in the Stream to Control object in the Properties of Object pane as shown in the figure entitled

"Entering the stream parameters into the Stream to Control object." In order to enter this data you will first need to double click on the Stream to Control object to expand it.

**Warning:**

When entering data, make certain not to use commas in numerical values. For example, enter 20000 without the comma instead of 20,000. If you do use a comma, a warning will appear on your screen.

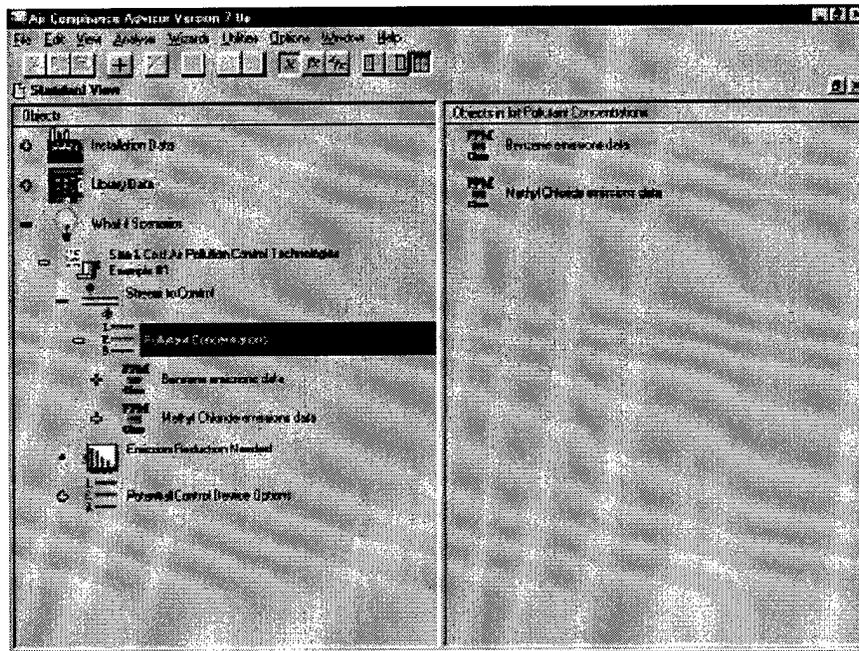


Entering the stream parameters into the Stream to Control object

Next, add the two pollutants in the emission stream and enter their concentrations. To enter this data, a few steps are required:

1. Add two Pollution Concentration Data objects to the Pollutant Concentrations Data object list by following the steps below.
2. Associate the chemicals and enter the volumetric concentrations into the Pollution Concentration Data objects. This procedure is discussed in more detail below.

To add a Pollution Concentration Data object, click on the Stream to Control object in Standard View, and then click on the Pollutant Concentrations object list. Click **+** in the toolbar, and then click **+** in the pop-up window. Repeat this procedure to add the second Pollution Concentration Data objects as shown in the figure entitled "Adding a Pollution Concentration Data object to the Pollutant Concentrations object list."



#### Adding a Pollution Concentration Data object to the Pollutant Concentrations object list

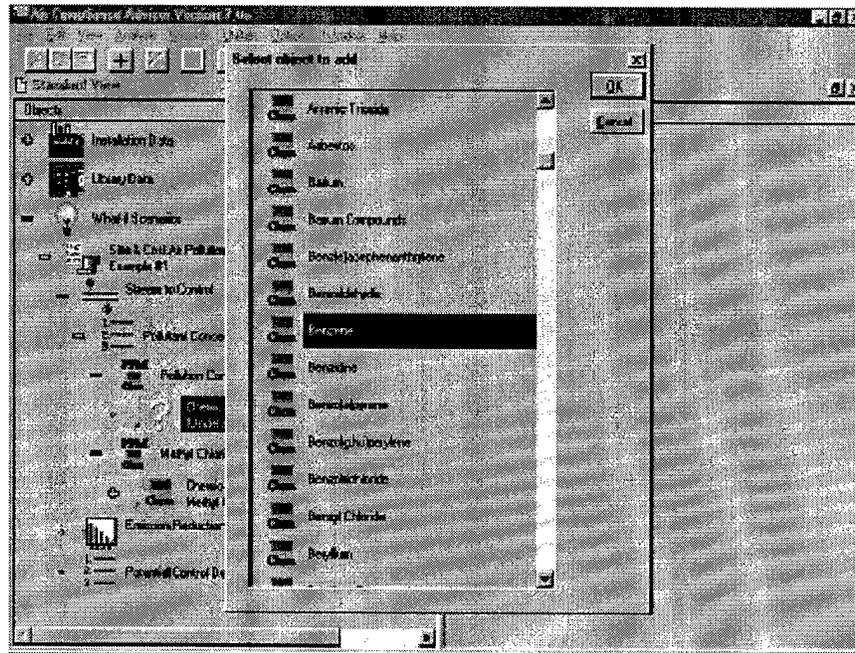
For each Pollution Concentration Data object in the Pollutant Concentrations object list,

- Click on the Pollution Concentration Data object
- Click on the yellow (+) sign to the left to expand that object



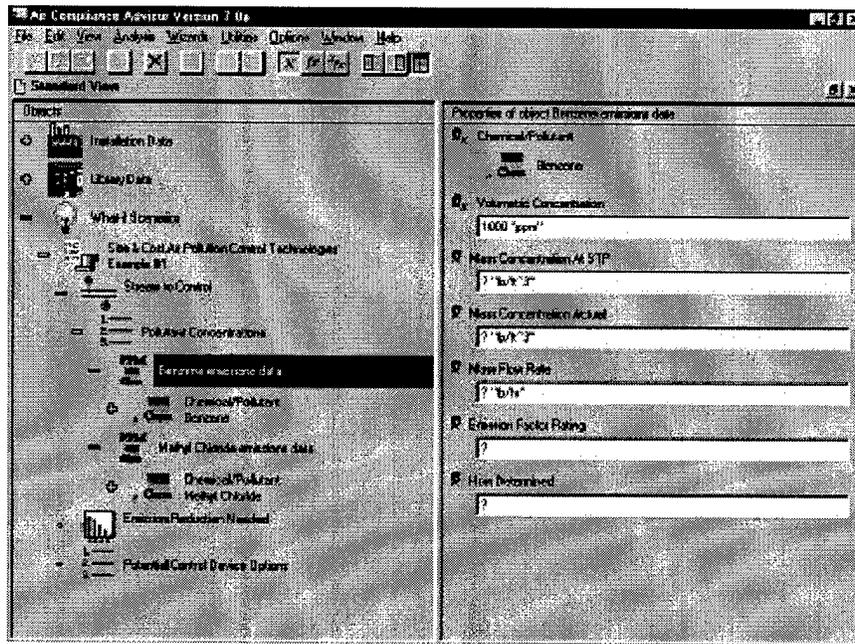
- Click on the large yellow question mark  associated with the Pollution Concentration Data object that was expanded
- Click the add button  on the toolbar Pick the desired chemical from the list that appears, in this case benzene or methyl chloride, and click . This procedure is shown in the figure entitled "Associating a chemical with a Pollution Concentration Data object."

Repeat this procedure for the second Pollution Concentration Data object.



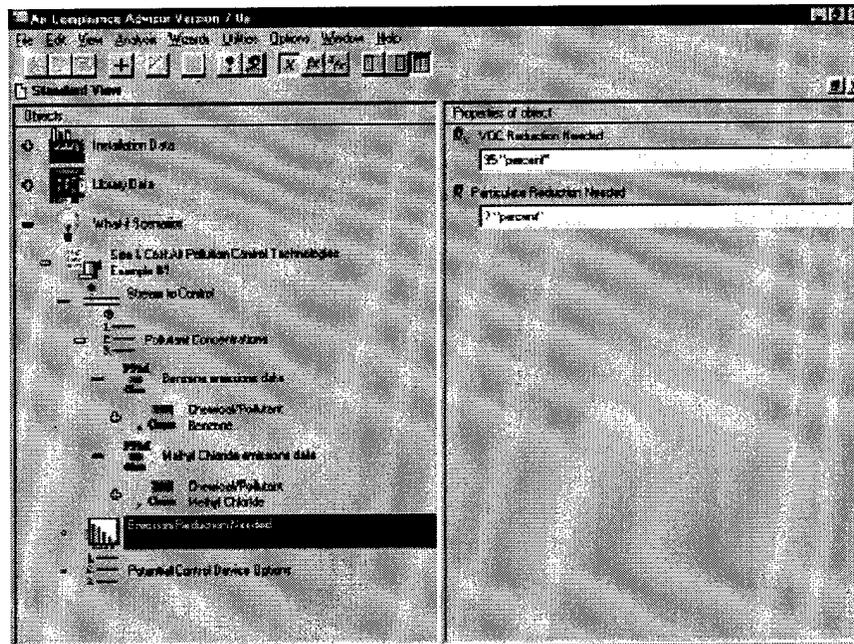
**Associating a chemical with a Pollution Concentration Data object**

Next, enter the volumetric concentrations for each chemical. To do this, select the Pollution Concentration Data objects (one for benzene and one for methyl chloride) and enter the Volumetric Concentration values in the Properties of Object pane. The data entry slots for the benzene Pollution Concentration Data object are shown in the figure entitled "Adding Volumetric Concentration data to the Pollution Concentration Data object." Remember not to use commas when entering numerical data.



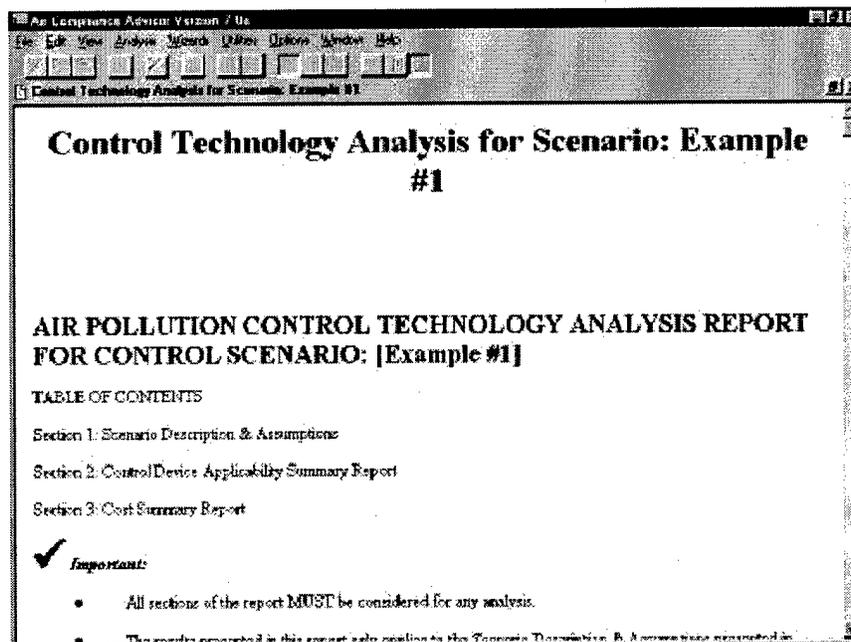
**Adding Volumetric Concentration Data to the Pollution Concentration Data object**

Finally, enter the desired reduction efficiency (i.e., 95%). This datum is entered in the VOC Reduction Needed data entry slot contained in the Emission Reduction Needed object as shown in the figure entitled "Entering in the desired reduction efficiency."



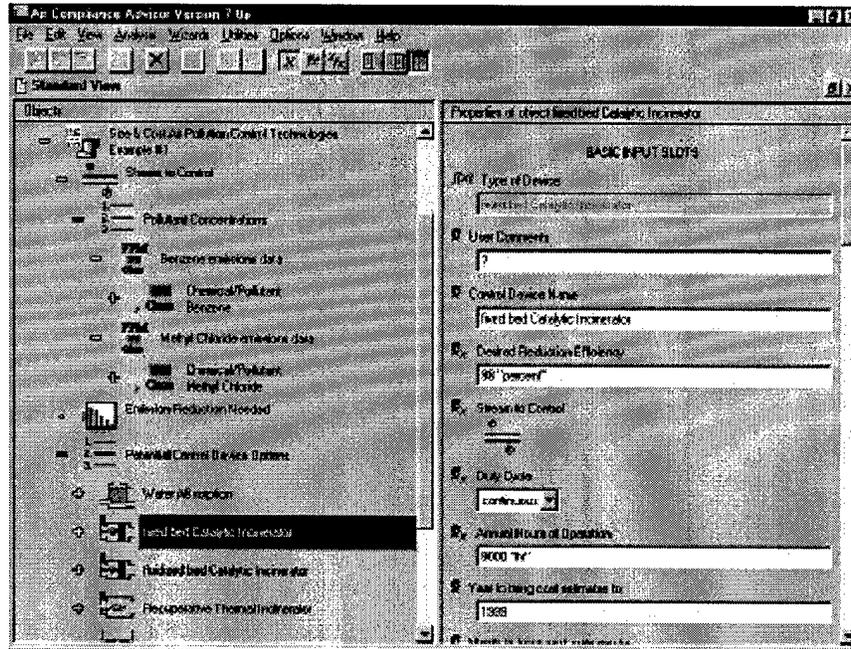
Entering in the desired reduction efficiency

Once the data has been entered, select Analysis | Apply Control Technologies from the main menu. The window "Control Device Report for WHAT-iF Scenario: Example #1" will appear as shown in the figure entitled "Control device summary report for Example #1."



Control device summary report for Example #1

The ACA will create instances of the control devices that are found to be applicable. These instances are placed in the Potential Control Device Options object list below the Emission Reduction Needed object contained in the Size & Cost Air Pollution Control Technologies object as shown in the figure entitled "Listing of potential control devices for Example #1."



Listing of potential control devices for Example #1

### *Extra Things To Try (Optional)*

To refine the cost of applying a control device to the pollutant stream described above, specific cost data can be added to the analysis. To do this:

1. Set the Options | User Level to "Intermediate." (This will provide access to additional data structures and parameters.)
2. Modify any or all of the following slots that are associated with the Installation Data object (at the top-level object) to see how the costs of the control devices (capital and operational) vary:
  - Year to bring cost estimates to
  - Month to bring cost estimates to
  - Interest Rate
  - Operating labor Cost
  - Maintenance labor Cost
  - Electricity Cost
  - Cooling Water Cost

- Water Disposal Cost
- Steam Cost
- Natural Gas Cost

3. Re-run the Apply Control Technologies analysis option to view the results.

## 6.2 Example Scenario #2: Costing & Sizing PM Control Devices

### *Background/Objective*

The goal of this example is to illustrate how to use the ACA to size and cost particulate matter (PM) control devices. Some additional parameters must be considered when defining PM emissions. While gaseous emissions can be described as a mixture of one or more chemicals in the vapor phase, PM pollutants cannot be described this simply. The attributes used to describe PM (e.g., mass distribution, mass median diameter) vary significantly from one emission unit to the next. Because the selection of PM control devices is dependent on these same attributes, it is important that they reflect the emissions of interest. The ACA offers three pre-defined PM emission types: PM10; PM - Liquid Mist, MMD - 1.66  $\mu\text{m}$ ; and PM - Liquid Mist, MMD - 7.7  $\mu\text{m}$ . There are potentially an unlimited number of PM emission types, thus the user should always enter data to define their specific PM pollutants.

Users enter emissions data into the **What-If Scenarios** core object. See the Apply Control Technologies section for a generalized discussion of the Apply Control Technologies analysis option.

### *Input Parameters/Assumptions*

| Parameter                  | Value                       |
|----------------------------|-----------------------------|
| PM Reduction Required      | 98%                         |
| <b>Operating Schedule</b>  |                             |
| Yearly Hours of Operation  | 6,000 hr                    |
| Duty Cycle                 | intermittent                |
| <b>Stream Parameters</b>   |                             |
| Volumetric Flow Rate @ STP | 21,500 ft <sup>3</sup> /min |
| Temperature                | 311 Kelvin                  |
| % Moisture                 | 3.253                       |

## PM Pollutant Data

### Particle Size Distribution

| <u>Aerodynamic Diameter (<math>\mu\text{m}</math>)</u> | <u>% mass above</u> |
|--------------------------------------------------------|---------------------|
| 0                                                      | 100                 |
| 0.5                                                    | 96                  |
| 0.8                                                    | 85                  |
| 1.4                                                    | 60                  |
| 2.5                                                    | 30                  |
| 4.5                                                    | 8                   |

### PM Attributes

|                    |                        |
|--------------------|------------------------|
| Type Pollutant     | criteria air pollutant |
| Corrosive          | false                  |
| Metal              | false                  |
| Condensable Metal  | false                  |
| Water Soluble      | false                  |
| Particulate Matter | true                   |
| Phase              | solid                  |

### PM Concentration Data

|                              |                            |
|------------------------------|----------------------------|
| Mass Concentration at Actual | 4.0 grains/ft <sup>3</sup> |
|------------------------------|----------------------------|

### Input parameters for Example #2

#### *Data Entry Into the ACA*

The data provided in the Input Parameters/Assumptions section above is entered in the What-If Scenario core object under the Size & Cost Air Pollution Control Technologies object.

Using the methods you learned in Example #1, enter the yearly hours of operation and the duty cycle in the appropriate data entry slots for the Size & Cost Air Pollution Control Technologies object in the Properties of Object pane. Refer to the figure entitled "Entering data in the Size & Cost Control Technologies object" as an example of this type of procedure.

Enter the stream level parameters listed above into the appropriate data entry slots contained in the Stream to Control object in the Properties of Object pane. Refer to the figure entitled "Entering the stream parameters into the Stream to Control object" as an example.

Next, add the PM in the emission stream and enter its concentrations. To enter this data, a few steps are required:

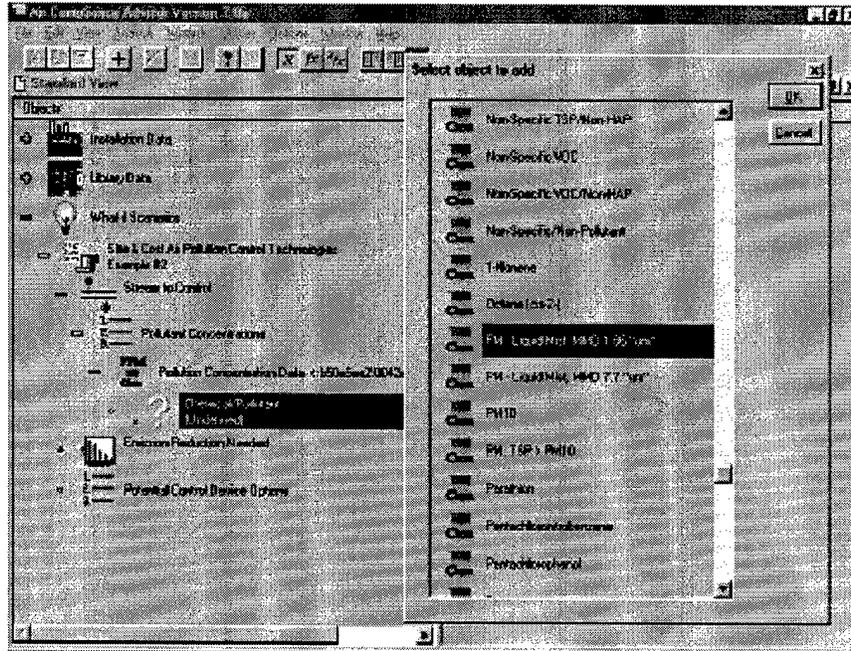
1. Add a Pollution Concentration Data object to the Pollutant Concentrations object list by following the steps below.
2. Associate the PM and enter the volumetric concentrations in the Pollution Concentration Data objects. This procedure is discussed in more detail below.

To add a Pollution Concentration Data object, click on the Pollutant Concentrations object, then click , and click  in the pop-up window. Refer to the figure entitled "Adding a Pollution Concentration Data object to the Pollutant Concentrations object list" for an example.

Perform the following steps to associate the correct Particulate Matter instance to the Pollutant Concentrations object.

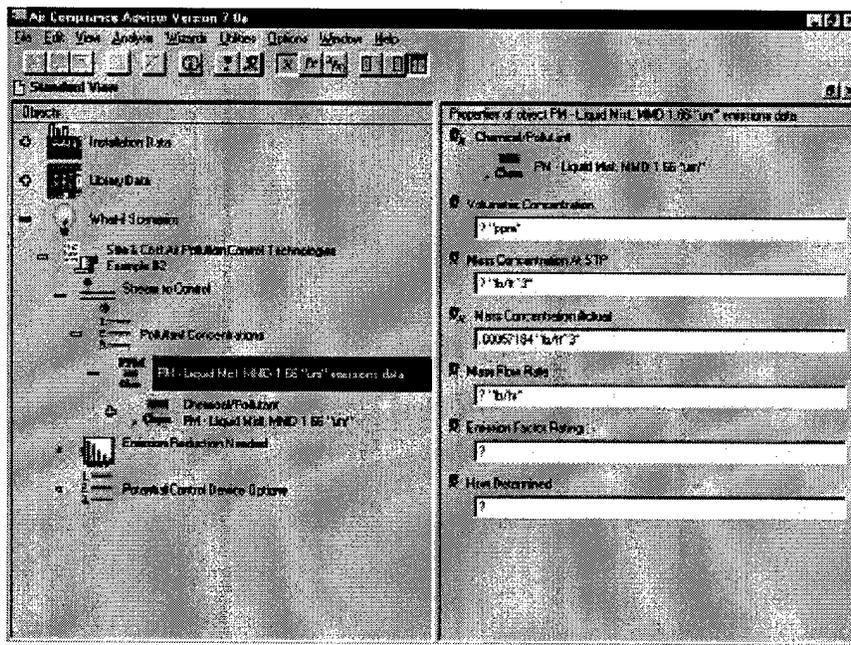
- Click on the Pollution Concentration Data object
- Click on the yellow (+) sign to the left to expand that object 
- Click on the large yellow question mark  associated with the Pollution Concentration Data object that was expanded
- Click the add button  on the toolbar
- Select the PM - Liquid Mist, MMD - 1.66  $\mu\text{m}$  from the list that appears, and click .

This procedure is shown in the figure entitled "Associating the PM type with the Pollution Concentration Data object."



**Associating the PM type with the Pollution Concentration Data object**

Next enter the mass concentration of the PM pollutant (4.0 "grains/ft<sup>3</sup>") into the appropriate data entry slot contained in the Pollution Concentration Data object as shown in the figure entitled "Entering the Mass Concentration at Actual Temperature and pressure to the Pollution Concentration Data object."



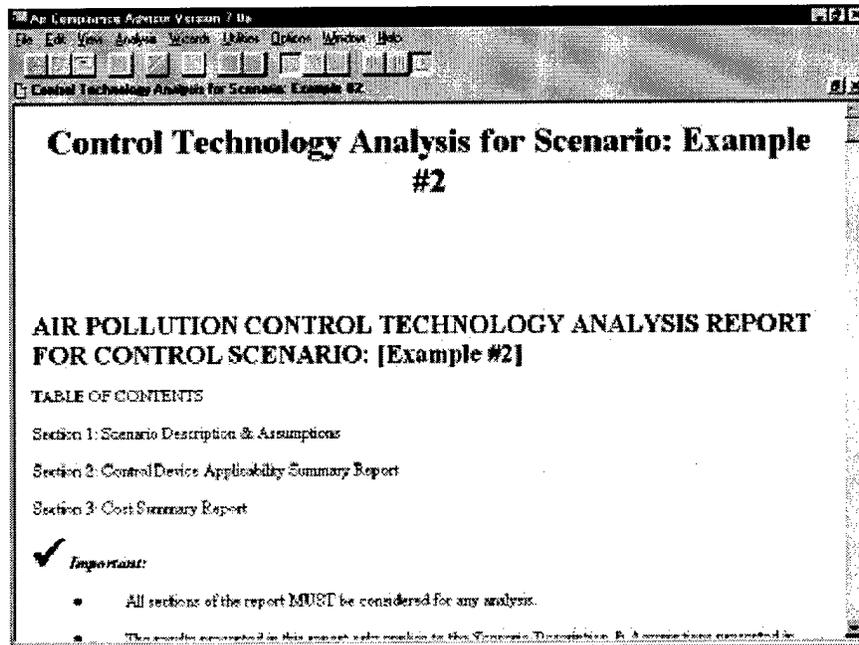
**Entering the Mass Concentration at Actual Temperature and pressure to the Pollution Concentration Data object**

Finally enter the desired PM reduction efficiency. This datum is entered in the Particulate

Reduction Needed data entry slot contained in the Emission Reduction Needed object. Refer to the figure entitled "Entering in the desired reduction efficiency" for an example of this type of procedure.

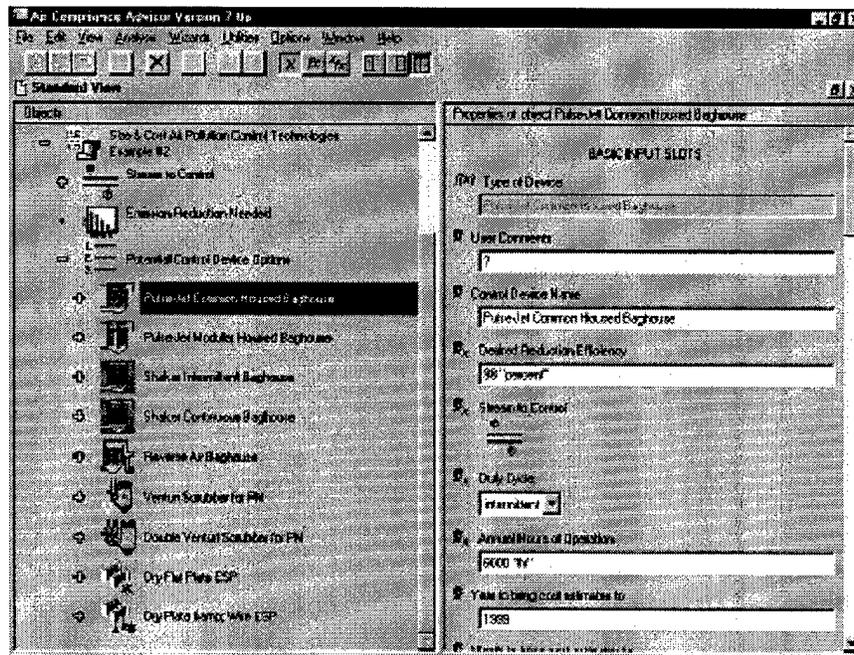
### ***Running the Analyses***

Once the data has been entered, select Analysis | Apply Control Technologies. A report window will appear once the analysis is complete as shown in the figure entitled "Report window for example #2."



**Report window for example #2**

The ACA will create instances of the control devices that are found to be applicable. These instances are placed in the Potential Control Device Options object list contained in the Size & Cost Air Pollution Control Technologies as shown in the figure entitled "List of applicable control devices for example #2".



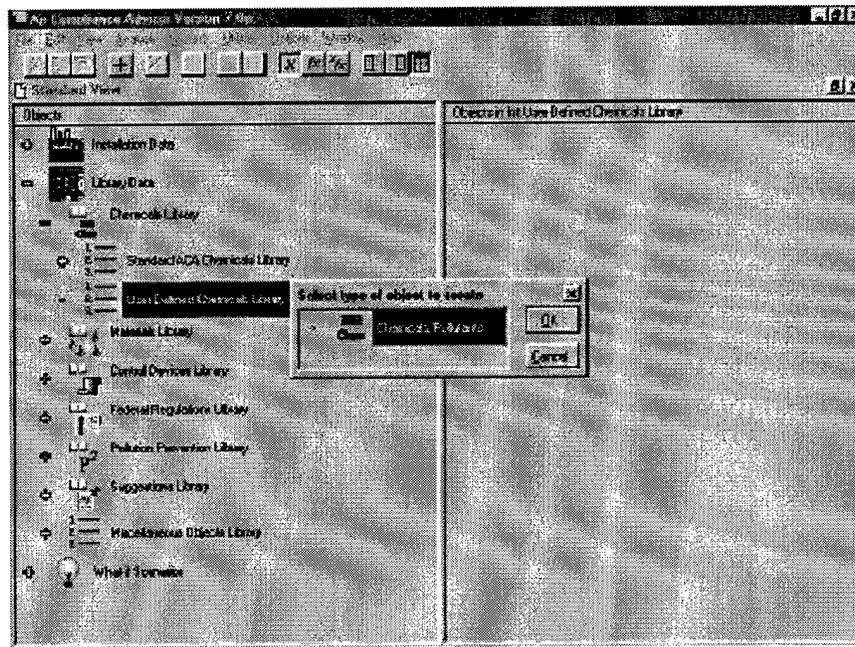
**List of applicable control devices for example #2**

***Extra Things To Try (Optional)***

***Entering Your Own PM Pollutant***

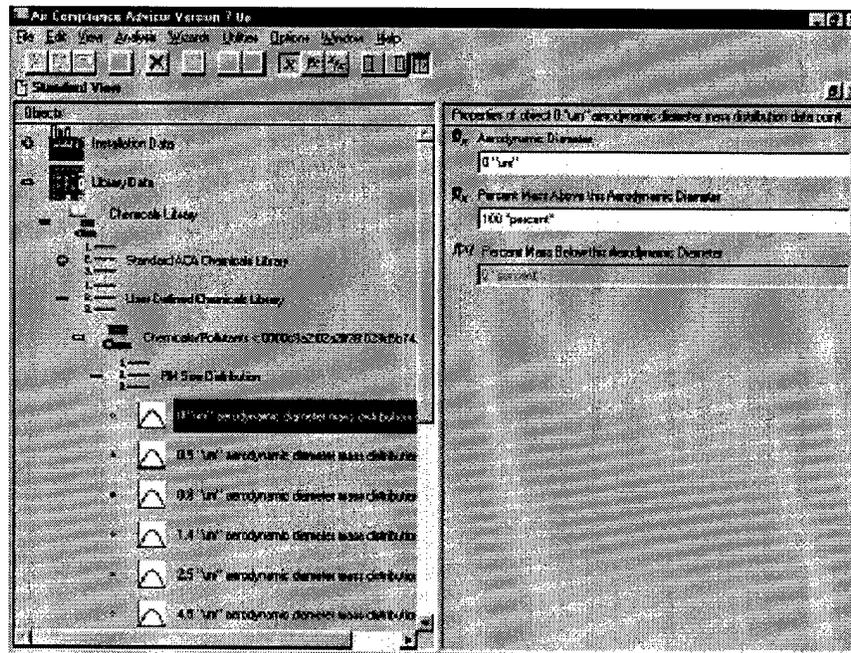
To properly size and cost PM control devices, the qualitative parameters of the PM and its mass distribution need to be accurate. PM data should be entered for each PM emission for which control device applicability, size, and cost estimates are required. This section describes how to define PM mass distribution to be used in place of the pre-defined PM pollutants contained in the ACA chemical library.

To define a new PM pollutant, expand the Library Data core object and the Chemicals Library object, then select the User-Defined Chemicals Library object list. Click on the add button **+** on the toolbar and then click OK to add a Chemicals/Pollutant object to the list as shown in the figure entitled "Adding a Chemicals/Pollutant object."



#### Adding a Chemicals/Pollutant object

This action creates a new Chemical and its properties can now be entered. To completely define the new PM pollutant, it is necessary to specify the mass distribution data listed in the "Input Parameters for Example #2" table in the previous section. To enter this data, expand the newly added Chemicals/Pollutants object and highlight the PM Size Distribution object list. Click the add button  from the toolbar and click OK. Do this six times so that six Mass Distribution Data Type objects are added to the PM Size Distribution object list. Finally, enter the Aerodynamic Diameter ( $\mu\text{m}$ ) and Percent Mass Above data (given in the "Input Parameters for Example #2" table in the previous section) into each of the six Mass Distribution Data Type objects as shown in the figure entitled "Entering in the mass distribution data into the new PM Chemical."



**Entering in the mass distribution data into the new PM Chemical**

### *Sensitivity Studies*

Another method of refining a control device cost study (refer to Example 1 in the previous section) is to modify the parameters of the control devices that the ACA has created. To do this complete the following steps:

1. Set the Options | User Level to "Intermediate." (This yields access to some additional data structures and parameters.)
2. Expand the Potential Control Device Options object list contained in the What-if Scenario object and select any of the control devices in that list.
3. Edit any of the parameters associated with the selected control devices to see how all of the other parameters have changed. An example of some interesting parameters to modify include: year to bring cost estimates to, interest rate, control device life expectancy, auxiliary equipment cost, and pressure drop.

## **6.3 Example Scenario #3: Estimating Emissions (Gas Fired Boiler)**

### *Background/Objective*

The ACA can be used to estimate emissions from many different types of emission units; refer to the Estimate Emission Rates Section for a complete listing. The purpose of Examples 3 and 4 is to illustrate how the ACA can be used to estimate emission rates. In this example, the emission rates from a gas-fired boiler will be calculated.

### ***Input Parameters/Assumptions***

| <b>Parameter</b>                                               | <b>Value</b>               |
|----------------------------------------------------------------|----------------------------|
| Emission Unit Type                                             | Gas Fired boiler           |
| Attempt to Estimate Emissions                                  | true                       |
| Rated Input Capacity                                           | 5 MW                       |
| PM Control Device in Place                                     | none                       |
| SOX Control Device in Place                                    | none                       |
| NOX Control Device in Place                                    | none                       |
| <b>Operating Schedule Data (for both actual and potential)</b> |                            |
| Average Operational Time per Day                               | 24 hr                      |
| Days Operating per Week                                        | 7                          |
| Date the emission unit started operation                       | January 1, 1985            |
| Volumetric Usage Rate (Fuel Usage)                             | 16,980 ft <sup>3</sup> /hr |
| Type Gas Used                                                  | natural gas                |

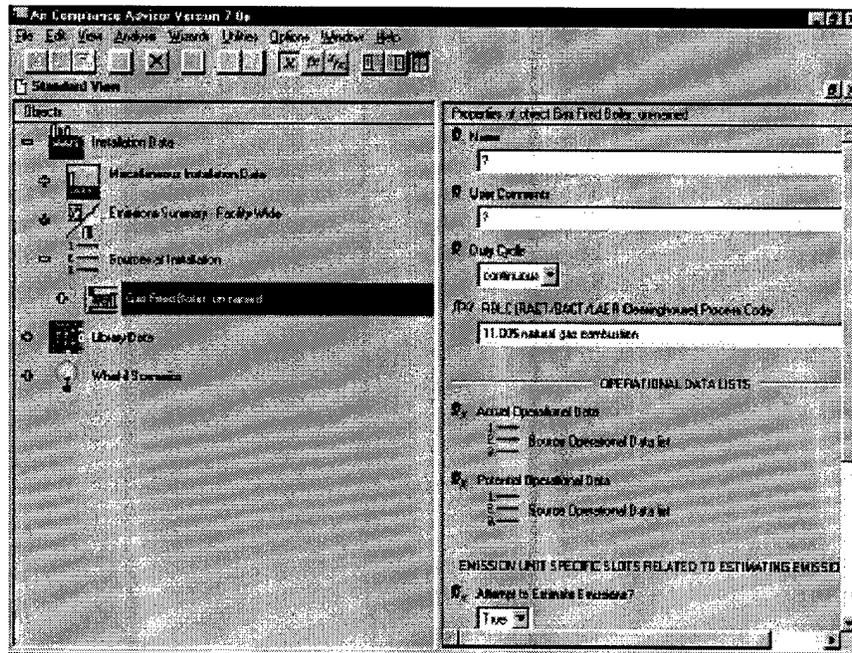
### **Input parameters for Example #3**

Further assume the emission unit is still operating.

### ***Data Entry Into the ACA***

Expand the Installation Data core object and select the Sources at Installation object list. Add a gas-fired boiler emission unit to the Sources at Installation object list by clicking the add button (+) from the toolbar, selecting the gas fired boiler emission unit, and clicking the OK button. Enter the input data provided above into the appropriate data entry slots contained in the Gas Fired Boiler object.

To do so, expand the Gas Fired Boiler object by double clicking on it. Then scroll down to find the appropriate data entry slots. Set the Attempt to Estimate Emissions Rate data entry slot to "True," specify "none" for each control device, and enter the Rated Input Capacity data as shown in the figure entitled "Data entry in the Gas Fired Boiler object."

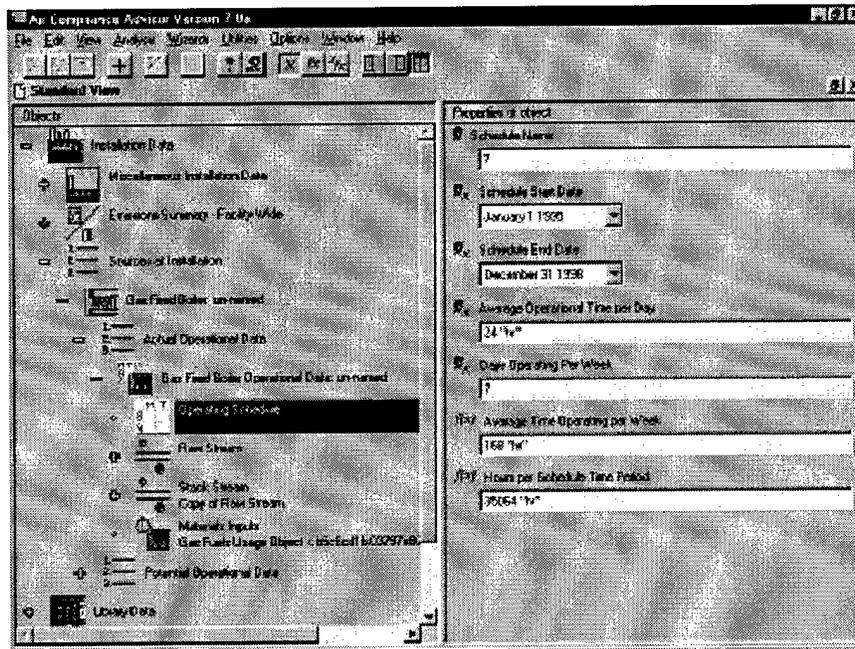


### Data entry in the Gas Fired Boiler object

Select the Actual Operational Data object list, click on the toolbar's add (+) button, and click OK to add an Operational Data Set object. Expand the Gas Fired Boiler Operational Data object, select the Operating Schedule object and enter the data provided for the actual operational data into the appropriate data entry slots contained in the Operating Schedule object as shown in the figure entitled "Data entry for the Actual Operational Data in the Operating Schedule object."

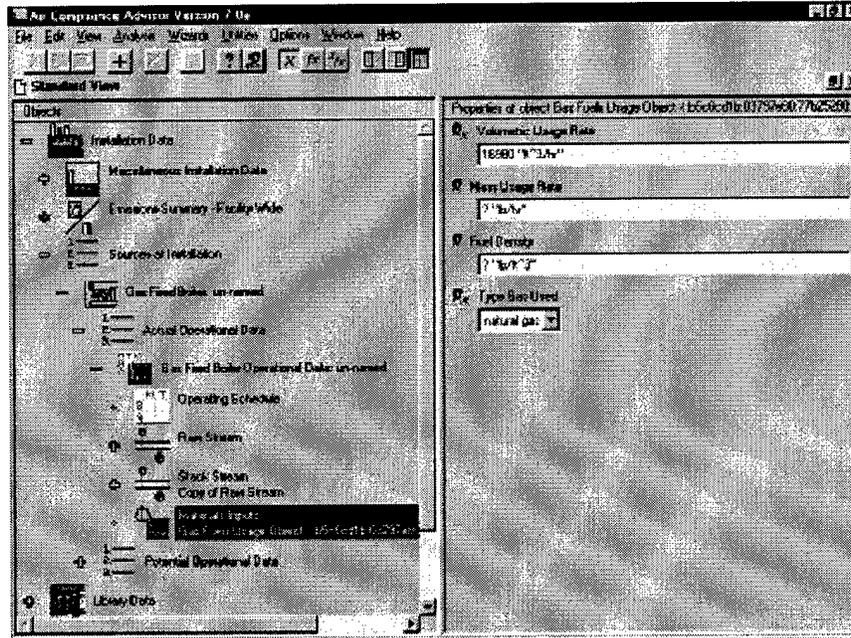
#### Note:

The Average Time Operating Per Week and Hours per Schedule Time Period data slots are automatically calculated.



Data entry for the Actual Operational Data in the Operating Schedule object

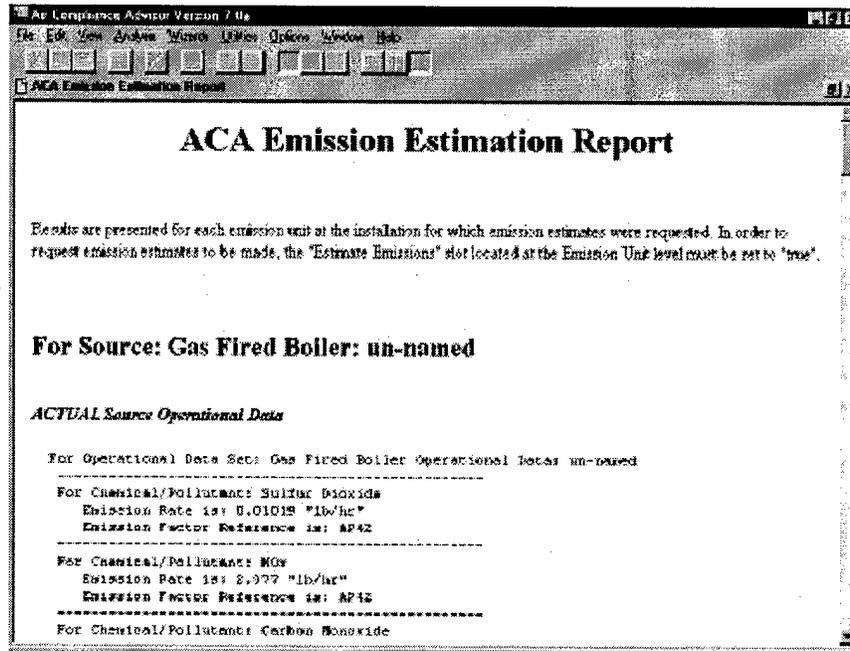
Finally, enter the Volumetric Usage Rate and the Type Gas Used into the appropriate data entry slots contained in the Materials Inputs object as shown in the figure entitled "Entry of fuel usage rate." Scroll through the list of gas types to select "natural gas."



Entry of fuel usage rate

## Running the Analysis

After entering the data, select Analysis | Estimate Emission Rates from the menu bar. An ACA Emission Estimation Report will appear with estimates for each chemical/pollutant as shown in the figure entitled "The emissions estimate report for Example #3."



The emissions estimate report for Example #3

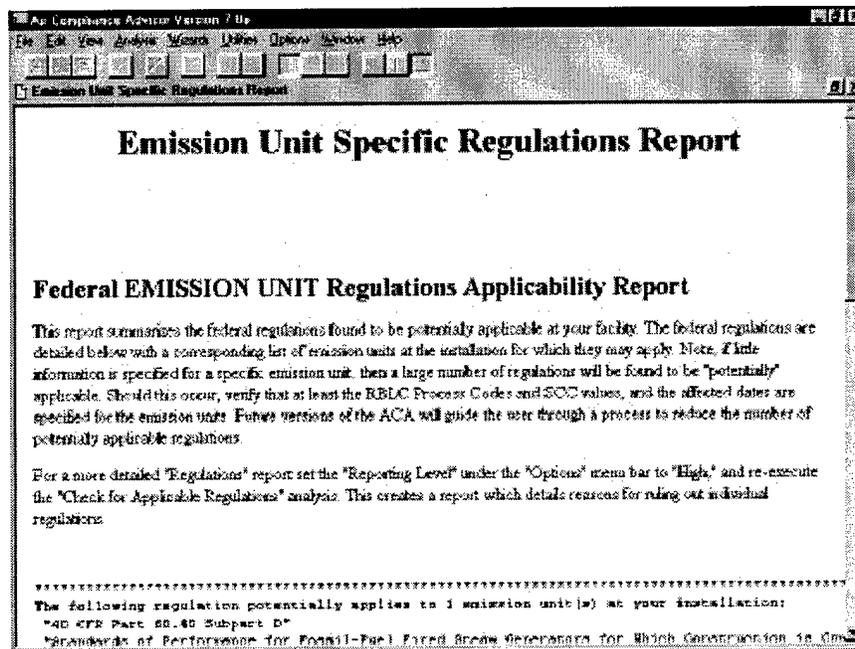
## Extra Things To Try (Optional)

Other options can be applied to the source. These options allow a check for applicable regulations, suggestions, and pollution prevention alternatives.

To check for applicable regulations, select Analysis | Prototype Analyses | Check for Applicable Regulations from the menu bar. The federal regulations may need to be loaded before running this analysis by selecting File | Load | ACA Federal Regulations Database. A federal regulation applicability report will appear in a new view on the screen. The report summarizes all the federal regulations potentially applicable to the gas fired boiler. The start of this report is shown in the figure entitled "Regulations Report for Example #3."

### Note:

You will receive a warning within the report if you need to load the ACA Federal Regulations database. After loading, you will need to run the report again using the same procedure described above.

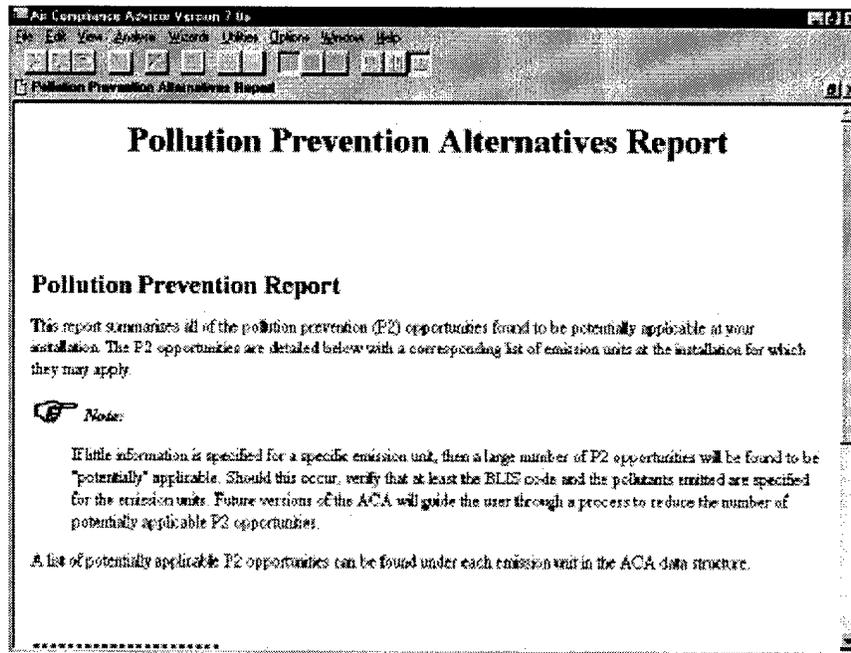


### Regulations Report for Example #3

To check for pollution prevention suggestions, select Analysis | Prototype Analyses | Check for Applicable Pollution Prevention Opportunities from the menu bar. The pollution prevention suggestions library may need to be loaded before running this analysis by selecting File | Load | ACA Pollution Prevention Database. A new view with a report summarizing pollution prevention opportunities will appear on the screen. The start of this report is shown in the figure entitled "Pollution Prevention report for Example #3."

 **Note:**

You will receive a warning within the report if you need to load the ACA Pollution Prevention database. After loading, you will need to run the report again using the same procedure described above.

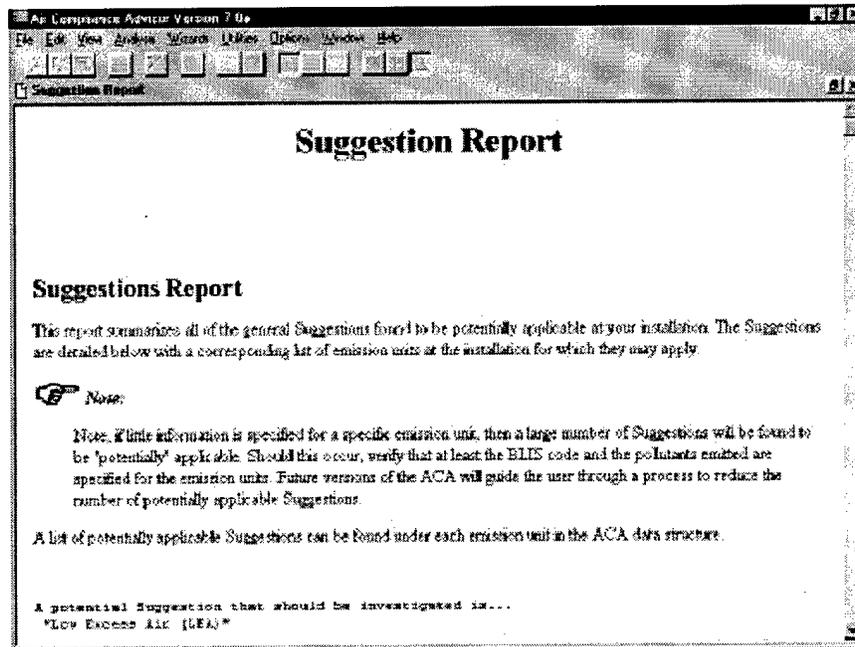


### Pollution Prevention report for Example #3

To check for applicable suggestions, select **Analysis | Prototype Analyses | Check for Applicable Suggestions** from the menu bar. The suggestions library may need to be loaded before running this analysis by selecting **File | Load | ACA Suggestions Database**. A new view with a report summarizing general suggestions for the gas fired boiler will appear on the screen. The start of this report is shown in the figure entitled "Suggestions report for Example #3."

 **Note:**

You will receive a warning within the report if you need to load the ACA Suggestions database. After loading, you will need to run the report again using the same procedure described above.



Suggestions report for Example #3

## 6.4 Example Scenario #4: Estimating Emissions (Waste Water Treatment Facility)

### *Background/Objective*

In this section, the ACA will be used to estimate emission rates from an emission unit. This example uses a more complex emission unit (an "aerated biotreatment" unit at a waste water treatment plant) to illustrate the usefulness of the ACA.

### *Input Parameters/Assumptions*

The input parameters needed for modeling a "mix tank" at a waste water treatment facility are presented in the following table.

| ACA Parameter Name             | Value                |
|--------------------------------|----------------------|
| <b>SOURCE-LEVEL DATA</b>       |                      |
| Attempt to Estimate Emissions? | true                 |
| Water Surface Area             | 2,500 m <sup>2</sup> |
| Waste Water Depth              | 5 m                  |
| Plant Type                     | Treatment            |
| Flow Model                     | Flowthrough          |

|                          |                   |
|--------------------------|-------------------|
| System Type              | aerated treatment |
| Type of Aeration         | Mechanical        |
| Waste Water Is Bioactive | True              |
| Water Has Oil Film       | False             |

**TYPE-DEPENDENT PROPERTIES/MECHANICAL AERATORS**

|                                           |                   |
|-------------------------------------------|-------------------|
| Turbulent Surface Area                    | 47 m <sup>2</sup> |
| Total Power to Aerators                   | 7.5 hp            |
| Number of Aerators                        | 1                 |
| Impeller Diameter                         | 60 cm             |
| Rotational Speed of Impeller              | 1200 rev/min      |
| Oxygen Transfer Rating of Surface Aerator | 3 lb/hp-hr        |
| Oxygen Transfer Correction Factor         | 0.83              |

**TYPE-DEPENDENT PROPERTIES/BIOACTIVITY**

|                       |                      |
|-----------------------|----------------------|
| Biomass Concentration | 300 g/m <sup>3</sup> |
|-----------------------|----------------------|

**OPERATIONAL DATA/MATERIAL USAGE DATA**

|                       |              |
|-----------------------|--------------|
| Volumetric Usage Rate | 40 liter/sec |
|-----------------------|--------------|

**MATERIAL COMPOSITION DATA**

|         |         |
|---------|---------|
| Benzene | 0.001 % |
|---------|---------|

**BENZENE**

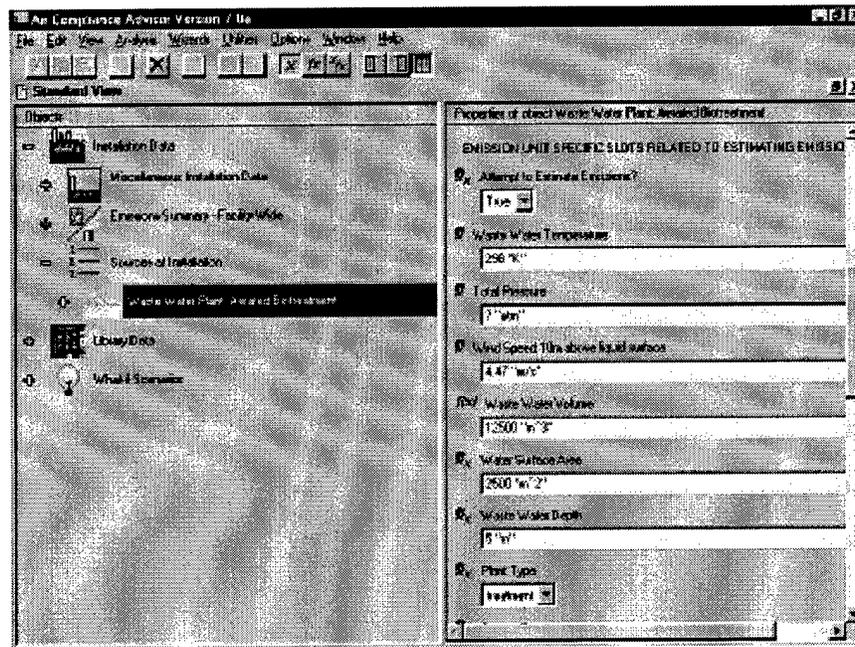
|                                  |                       |
|----------------------------------|-----------------------|
| Half Saturation Biorate Constant | 13.6 g/m <sup>3</sup> |
| Maximum Biorate Constant         | 19 mg/hr-g            |

**Input parameters for Example #4**

***Data Entry into the ACA***

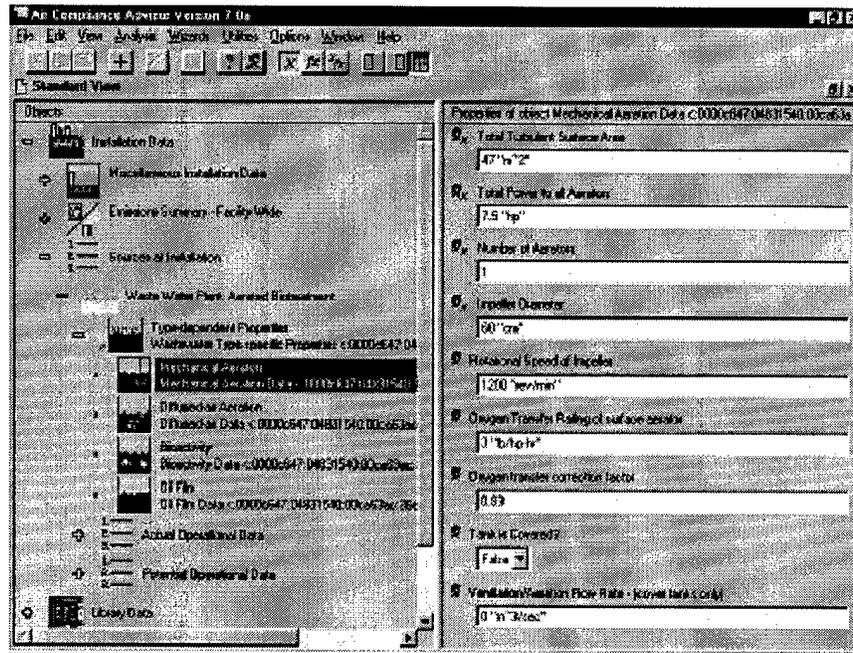
Expand the Installation Data core object and select the Sources at Installation object list. Add a waste water plant emission unit to the Sources at Installation object list by clicking the add button (+) from the toolbar, selecting the waste water plant emission unit, and clicking the OK button. Enter the SOURCE-LEVEL parameters from the "Input parameters for Example #4"

Table into the appropriate data entry slots contained in the Waste Water Plant object as shown in the figure entitled "Parameters associated with a Waste Water Plant object."



Parameters associated with a Waste Water Plant object

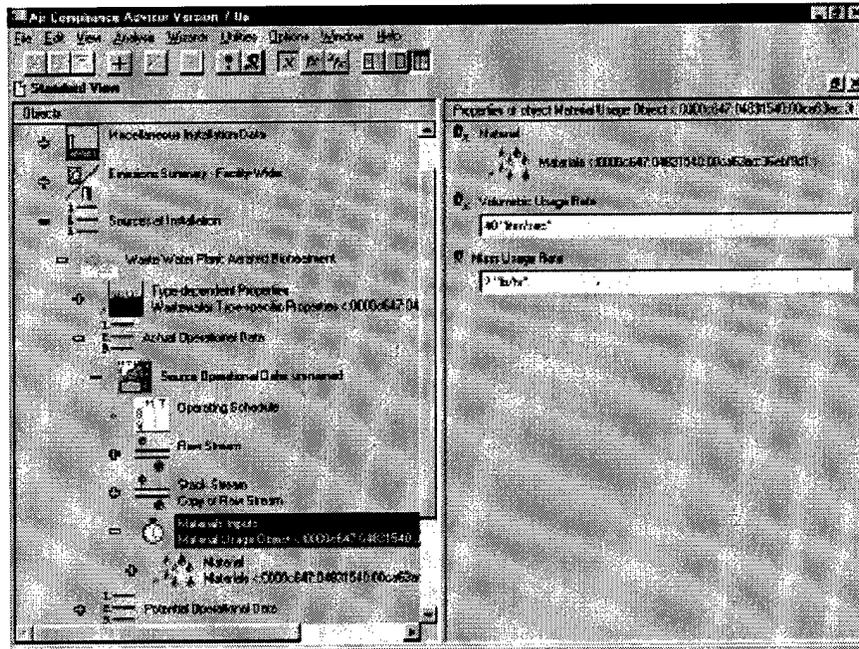
Click on the Waste Water Plant object to expand it and expand the Type-dependent Properties object. Select the Mechanical Aerators object. Enter the TYPE-DEPENDENT PROPERTIES/MECHANICAL AERATORS parameters from the "Input parameters for Example #4" Table into the appropriate data entry slots contained in the Mechanical Aerators object as shown in the figure entitled "Parameters associated with a Mechanical Aerators object."



**Parameters associated with a Mechanical Aerators object**

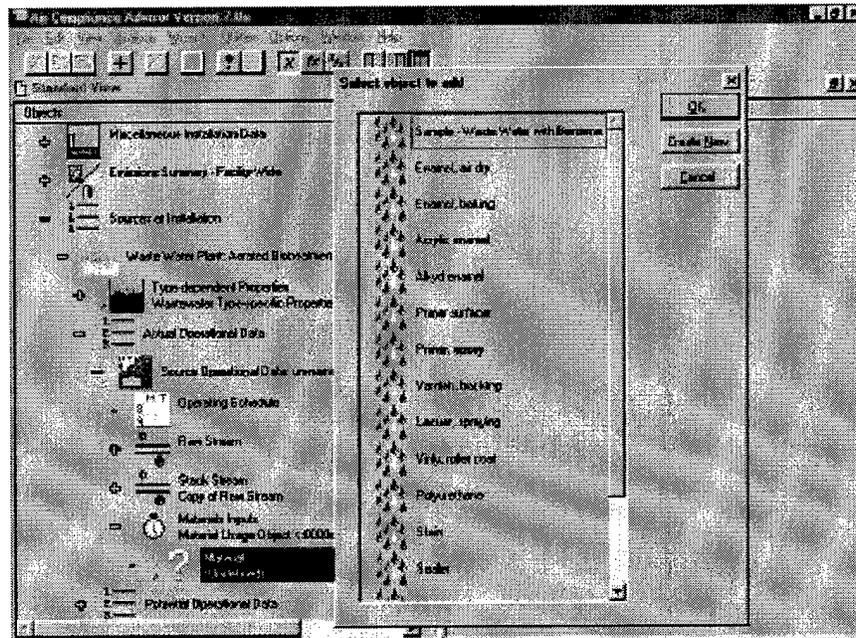
When the mechanical-aerator parameters are entered, select the **Bioactivity** object and enter the **TYPE-DEPENDENT PROPERTIES|BIOACTIVITY** data shown in the "Input parameters for Example #4" Table.

Select the **Actual Operational Data** object, click on the toolbar's add (+) button, and click OK add an **Operational Data Set** object. Expand the **Actual Operational Data** object list, expand the **Source Operational Data** object, and select the **Materials Inputs** object. Enter the value listed the "Input parameters for Example #4" into the **Volumetric Usage Rate** data entry slot as shown in the figure entitled "Entry of the Volumetric Usage Rate for a Materials Inputs object" (note that you will need to change the default units given for this slot of "ft<sup>3</sup>/hr" to "liter/sec").



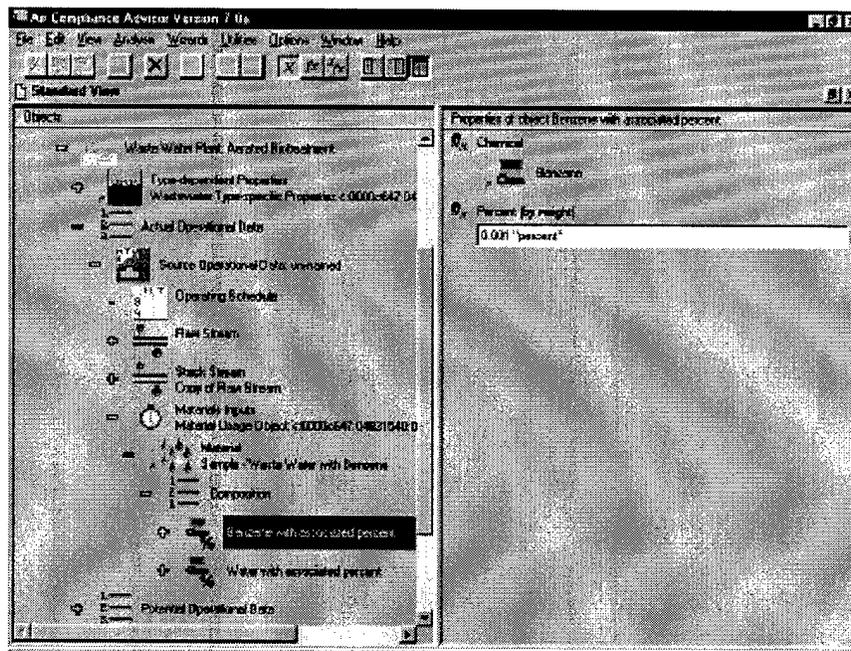
Entry of the Volumetric Usage Rate for a Materials Inputs object

Next a material is associated with the Materials Input object by expanding the Materials Input object, selecting the undefined Material object, and clicking on the toolbar's add button (+) to select a material. However, in this example the material must first be created. A new material can be created from clicking the Create New in the pop-up window containing materials and then clicking OK to add a new Material object as shown in the figure entitled "Creation of a new material from add material pop-up window."



Creation of a new material from add material pop-up window

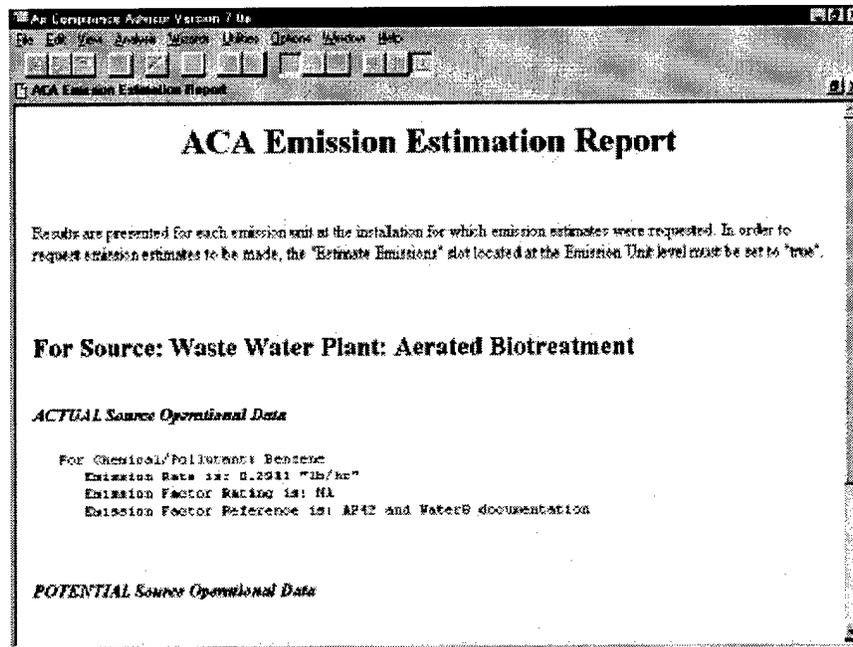
The new material must be defined by specifying its chemical composition. Expand the new Material object contained in the Materials Input object and select the Composition object list. Click on the toolbar's add button **+** and then OK to add a Chemicals with associated percents object to the Composition object list. Expand the Chemicals with associated percents object, highlight the undefined Chemical object, and click on the toolbar's add button to define the undefined chemical as benzene. Finally, click on the Benzene with associated percents object to add the benzene composition information from the "Input parameters for Example #4" Table to the Percent (by weight) data entry slot as shown in the figure entitled "Adding weight percent of benzene." The new material could have also been specified by adding it directly to the User-Defined Materials Library as previously described in Section 3 .



Adding weight percent of benzene

### ***Running the Analyses***

From the Analysis menu, select Estimate Emission Rates. The emission results will be displayed on the screen as shown in the figure entitled "Emission rate summary report for Example #4."



Emission rate summary report for Example #4

## 6.5 Example Scenario #5: Facility-Wide Study

### *Background/Objective*

In this section the ACA will be used to analyze a facility that has two pollutant sources: a paint spray booth (used to apply primer to aerospace equipment) and a waste solvent recovery unit (used to recover spent solvent from the waste stream). This example involves creating sources, entering source data, and performing analyses on sources. In order to simplify this guide, references to figures from the previous examples are used whenever possible.

### *Input Values/Assumptions*

Information regarding the operation of each of the sources is presented in the table that follows. Assume that the "potential" operational data for both sources is the same as the actual data, except that the Paint Spray Booth's potential operations include continuous operations (i.e., 24 hrs/day, 7 days/week).

| Parameter                          | Value          |
|------------------------------------|----------------|
| <b>Source Specific Information</b> |                |
| <b>Paint Spray Booth</b>           |                |
| temperature                        | 20° C (293° K) |
| actual flow rate                   | 180,000 cfm    |

|                                    |                   |
|------------------------------------|-------------------|
| percent relative humidity          | 1.1458 %          |
| duty cycle                         | intermittent      |
| average hours per day of operation | 2.75 hours        |
| days operating per week            | 2 days/week       |
| start date                         | January 1, 1997   |
| end date                           | December 31, 1998 |
| desired VOC control efficiency     | 90%               |

**Composition of the waste gas stream**

|                                |          |
|--------------------------------|----------|
| cyclohexanone                  | 12.6 ppm |
| methyl ethyl ketone            | 29.3 ppm |
| methyl n-amyl ketone           | 8.4 ppm  |
| methylene bisphenyl isocyanate | 4.2 ppm  |

**Waste Solvent Recovery Unit**

|                                    |                   |
|------------------------------------|-------------------|
| temperature                        | 100° F (310.8° K) |
| actual flow rate                   | 20,000 cfm        |
| percent relative humidity          | 3.2175 %          |
| duty cycle                         | continuous        |
| average hours per day of operation | 24 hours          |
| days operating per week            | 7 days/week       |
| start date                         | January 1, 1995   |
| end date                           | December 31, 1995 |
| desired VOC control efficiency     | 90%               |

**Composition of waste gas stream**

|                 |           |
|-----------------|-----------|
| benzene         | 1,000 ppm |
| methyl chloride | 1,000 ppm |

**Input parameters for Example #5**

## Data Entry into the ACA

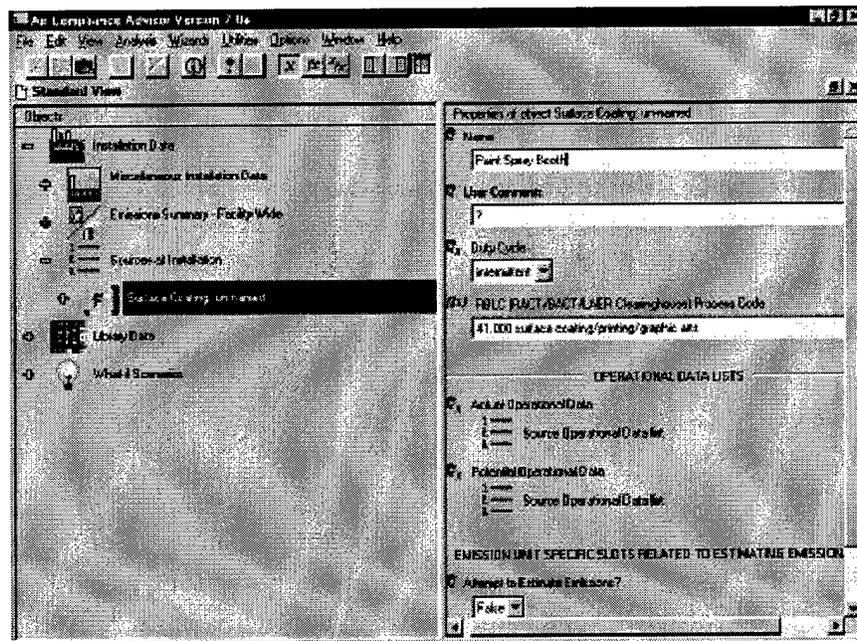
### Data Entry for Spray Paint Booth

Expand the Installation Data core object and select the Sources at Installation object list. Add a surface coating emission unit to the Sources at Installation object list by clicking the add button  from the toolbar, selecting the Surface Coating object, and clicking the OK button.

Open the newly created Surface Coating object and enter the information listed below into the appropriate data entry slots contained in the Surface Coating object as shown in the figure entitled "Entry of data for the surface coating operation."

| Parameter  | Value             |
|------------|-------------------|
| Name       | Paint spray booth |
| Duty cycle | Intermittent      |

Paint spray booth data for example #5



Entry of data for the surface coating operation

Expand the Surface Coating object, select the Actual Operational Data object, click on the toolbar's add button , and click OK to add an Operational Data Set object. Expand the Actual Operational Data object list, expand the Surface Coating Operational Data object, and select the Raw Stream object. In the Properties of Object pane, enter the data listed below to describe the actual operation of the Paint Spray Booth.

| Parameter            | Value                        |
|----------------------|------------------------------|
| Stream name          | Primer coat                  |
| Volumetric flow rate | 180,000 ft <sup>3</sup> /min |
| Temperature          | 293 K                        |

#### Pollutant stream data for the paint spray booth, example #5

Next, enter the actual emissions data. Follow the steps below to do so. The emissions (actual = potential) are repeated here for convenience:

| Pollutant            | Concentration |
|----------------------|---------------|
| cyclohexanone        | 12.6 ppm      |
| methyl ethyl ketone  | 29.3 ppm      |
| methyl n-amyl ketone | 8.4 ppm       |

#### Pollutant concentration data for the paint spray booth, example #5

Add the three pollutants in the emission stream and enter their concentrations. To enter this data, a few steps are required:

1. Add three Pollution Concentration Data objects to the Pollution Concentrations object list.
2. Associate the chemicals and enter the volumetric concentrations into the Pollution Concentration Data objects. This procedure is discussed in more detail below.

To add a Pollution Concentration Data object, click on the Pollutant Concentrations object list, click , and click  in the pop-up window. Repeat this procedure until three Pollution Concentration Data objects have been added. Refer to the figure entitled "Adding a Pollution Concentration Data object to the Pollutant Concentrations object list" for an example of this procedure.

For each Pollution Concentration Data object in the Pollutant Concentrations object list,

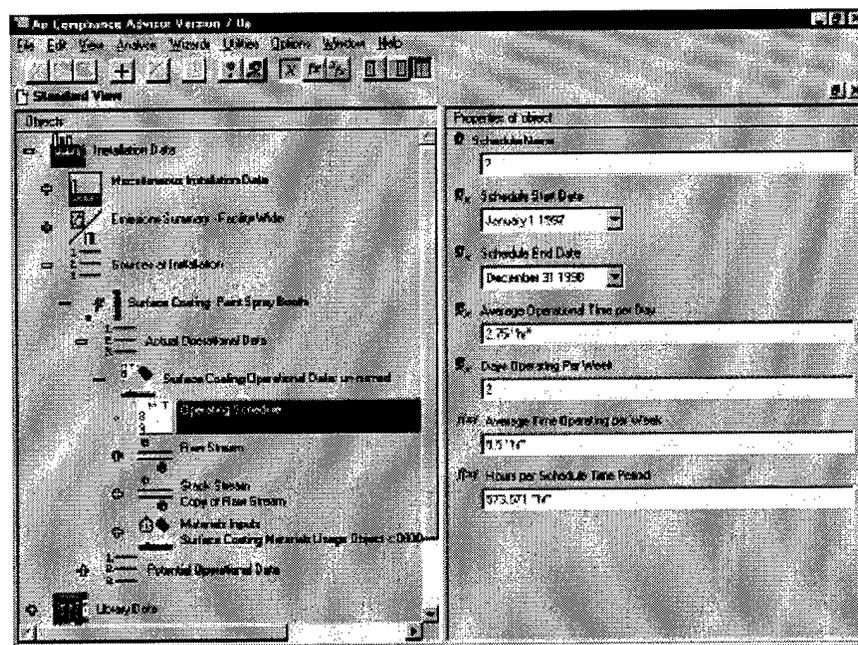
1. Click on it
2. Click on the yellow (+) sign to the left to expand that object 

3. Click on the large yellow question mark 

4. Then click the add button 

5. Select the desired chemical from the list that appears and click . An example of this procedure is shown in the figure entitled "Associating a chemical with a Pollution Concentration Data object."

To describe the operating schedule(s) of the waste solvent recovery unit, select the Operating Schedule object just above the Raw Stream object. Edit the Properties of Object pane using the Source Specific Information found in the beginning of this Example 5 section. The figure entitled "Entry of data for the Operating Schedule object" shows an example of this procedure.



Entry of data for the Operating Schedule object

After all the actual operation data has been entered, drag the Surface Coating Operational Data object from the Actual Operational Data object list and drop it onto the Potential Operational Data object list. This operation will create an exact duplicate of all the operational data.

For this example, the only difference between the actual and potential operational data is the operating schedule. Expand the Potential Operational Data object list, expand the Surface Coating Operational Data object, and select the Operating Schedule object. Edit the appropriate data entry slots to create an operating schedule of 24 hours per day, 7 days per week. The ACA will assume this holds for the entire scheduling period.

### *Data Entry for the Waste Solvent Recovery System*

The procedure for entering data for the Waste Solvent Recovery Unit is similar to the procedure for the Paint Spray Booth outlined in the section above. While the procedure is similar for all emission units, the description of data entry is repeated in detail here to avoid confusion.

Expand the Installation Data core object and select the Sources at Installation object list. Add a waste solvent recovery operation emission unit to the Sources at Installation object list by clicking the add  button from the toolbar, selecting the Waste Solvent Recovery Operation object, and clicking the OK button.

Open the newly created Waste Solvent Recovery Operation object and enter the information listed below into the appropriate data entry slots contained in the Waste Solvent Recovery Operation object. The figure entitled "Entry of data for the surface coating operation" shows an example of this operation for the surface coating emission unit.

| <b>Parameter</b> | <b>Value</b>                |
|------------------|-----------------------------|
| Name             | Waste solvent recovery unit |
| Duty cycle       | Continuous                  |

#### **Waste solvent recovery unit data for example #5**

Expand the Waste Solvent Recovery Operation object, select the Actual Operational Data object list, click on the toolbar's add  button, and click OK to add an Operational Data Set object. Expand the Actual Operational Data object list, expand the Waste Solvent Recovery Operational Data object, and select the Raw Stream object. In the Properties of Object pane, enter the data listed below to describe the actual operation of the waste solvent recovery operation.

| <b>Pollutant</b>     | <b>Concentration</b>        |
|----------------------|-----------------------------|
| Volumetric flow rate | 20,000 ft <sup>3</sup> /min |
| Temperature          | 310.8 K                     |

#### **Actual (and potential) operation data for waste solvent recovery unit, example #5**

Next, enter the actual emissions data using the steps that follow. The actual and potential emissions are repeated here for your convenience:

| Pollutant       | Concentration |
|-----------------|---------------|
| benzene         | 1,000 ppm     |
| methyl chloride | 1,000 ppm     |

#### Composition of waste gas stream from Waste solvent recovery unit, example #5

Add the two pollutants in the emission stream and enter their concentrations. To enter this data, a few steps are required:

1. Add two Pollution Concentration Data objects to the Pollution Concentration Data object list.
2. Associate the chemicals and enter the volumetric concentrations into the Pollution Concentration Data objects. This procedure is discussed in more detail below.

To add a Pollution Concentration Data object, click on the Pollutant Concentrations object list, click , and click  in the pop-up window. Repeat this procedure so that two Pollution Concentration Data objects have been added. Refer to the figure entitled "Adding a Pollution Concentration Data object to the Pollutant Concentrations object list" for an example of this procedure.

For each Pollution Concentration Data object in the Pollutant Concentrations object list,

1. Click on it
2. Click on the yellow (+) sign to the left to expand that object 
3. Click on the large yellow question mark 
4. Then click the add button 
5. Pick the desired chemical from the list that appears and click . An example of this procedure is shown in the figure entitled "Associating a chemical with a Pollution Concentration Data object".

To describe the operating schedule(s) of the waste solvent recovery unit, select the Operating Schedule object just above the Raw Stream object. Edit the Properties of Object pane using the Source Specific Information found in the beginning of this Example 5 section. The figure entitled "Entry of data for the Operating Schedule object" shows an example of this procedure.

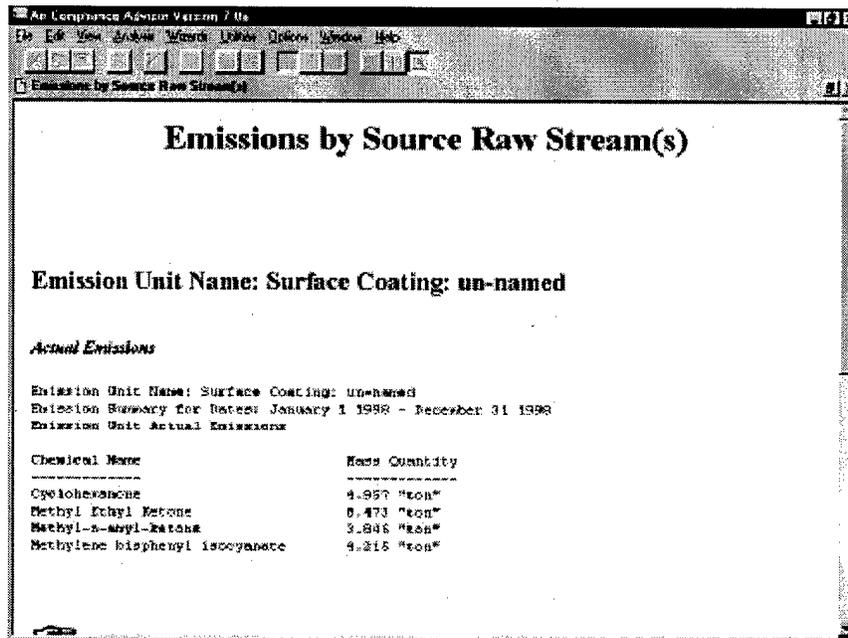
After all the actual operation data has been entered, drag the Waste Solvent Recovery Operation Operational Data object from the Actual Operational Data object list and drop it onto the Potential Operational Data object list. This operation will create an exact duplicate of all the operational data.

For this example, there is no difference between the actual and potential operational, therefore the potential operational data object that was created from the actual does not need to be modified.

### ***Running the Summarize Emissions Analyses***

After entering the information for the two sources and their actual and potential operating parameters, the user can run the analyses for summarizing emissions. This analysis calculates facility-wide emissions. In order for the ACA to summarize facility-wide emissions, the time period of interest must be specified by the user. Open the **Emissions Summary-Facility Wide** object and enter the start and stop dates for the analysis period in the **Properties of Object** pane. For this example a one-year period will be used, assuming a start date of January 1, 1998 and a stop date of December 31, 1998. The analysis time period can be longer than one year or can be a fraction of one year, depending upon the needs of the user. It should be noted that these dates do not have to match the operational dates of any of the sources at the facility. The ACA program determines which sources are in operation during the analysis period; considers how these sources operate (based upon the individual source operating schedules); and uses this information to determine the actual and potential emissions for the facility as a whole, on a chemical-by-chemical basis.

To perform this analysis select **Analysis | Summarize Emissions** and then either **Directly out of Emission Units** or **From Stack (to Atmosphere)** from the main menu. Please refer to the "Summarize Emissions" Section in Section 4 for a general discussion of these two analysis options. These analyses will both produce three reports with each report in its own window. The three figures shown below show the report windows for this example after the **Directly out of Emission Units** analysis is run.



**Report of actual and potential emissions on an emission unit-by-emission unit basis for example #5**

Ap Compliance Advisor Version 7.0e

File Edit View Actions Windows Utilities Options Windows Help

Facility-Wide Potential Emissions Stack Stream(s)

### Facility-Wide Potential Emissions Stack Stream(s)

Please check the bottom of this report for any notes, warnings or errors!

This section computes the facility-wide Potential Emissions. If Source Operational Data does NOT exist for ANY emission unit, the facility-wide Potential Emissions cannot be computed.

---

Emission Summary for the Dates: January 1 1998 - December 31 1998  
 Facility-Wide Potential Emissions  
 to the atmosphere. This information will be summarized in the object (data structure) entitled: "Installation | Emissions Summary Facility-Wide".

| Chemical Name                  | Mass Quantity |
|--------------------------------|---------------|
| Cyclohexanone                  | 151.4 "ton"   |
| Methyl Ethyl Ketone            | 258.8 "ton"   |
| Methyl-n-ethyl-ketone          | 117.5 "ton"   |
| Methylene bisphenyl isocyanate | 128.9 "ton"   |
| Benzene                        | 1,082 "ton"   |
| Methyl Chloride                | 647.8 "ton"   |

Facility-Wide Potential Emissions report for example #5

Ap Compliance Advisor Version 7.0e

File Edit View Actions Windows Utilities Options Windows Help

Facility-Wide Actual Emissions Stack Stream(s)

### Facility-Wide Actual Emissions Stack Stream(s)

Please check the bottom of this report for any notes, warnings or errors!

This section computes the facility-wide Actual Emissions. If Source Operational Data does NOT exist for ANY emission unit, the facility-wide Actual Emissions cannot be computed.

---

Emission Summary for the Dates: January 1 1998 - December 31 1998  
 Facility-Wide Actual Emissions  
 to the atmosphere. This information will be summarized in the object (data structure) entitled: "Installation | Emissions Summary Facility-Wide".

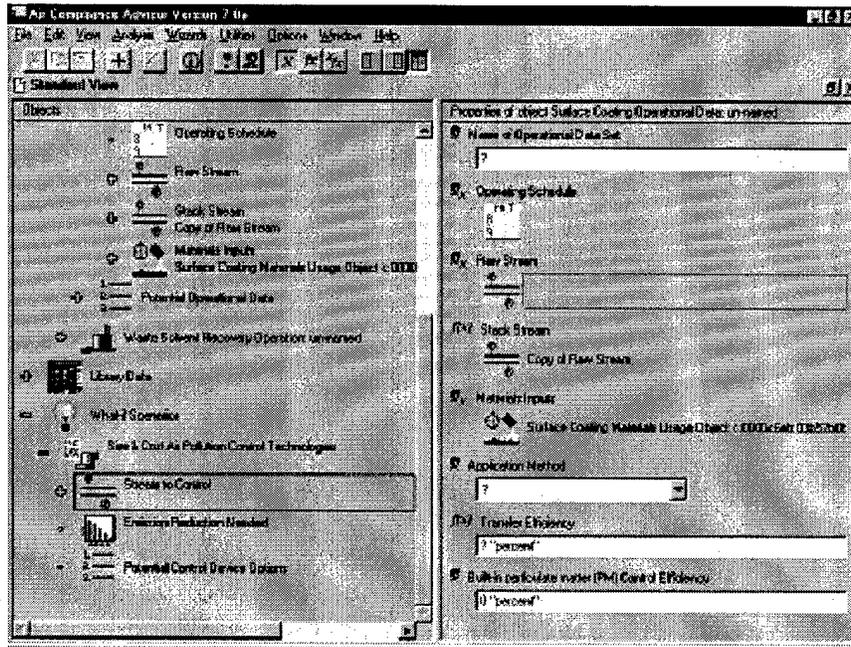
| Chemical Name                  | Mass Quantity |
|--------------------------------|---------------|
| Cyclohexanone                  | 4.937 "ton"   |
| Methyl Ethyl Ketone            | 8.471 "ton"   |
| Methyl-n-ethyl-ketone          | 3.646 "ton"   |
| Methylene bisphenyl isocyanate | 4.215 "ton"   |
| Benzene                        | 1,082 "ton"   |
| Methyl Chloride                | 647.8 "ton"   |

Facility-Wide Actual Emissions report for example #5

### Running the Apply Control Technologies Analysis

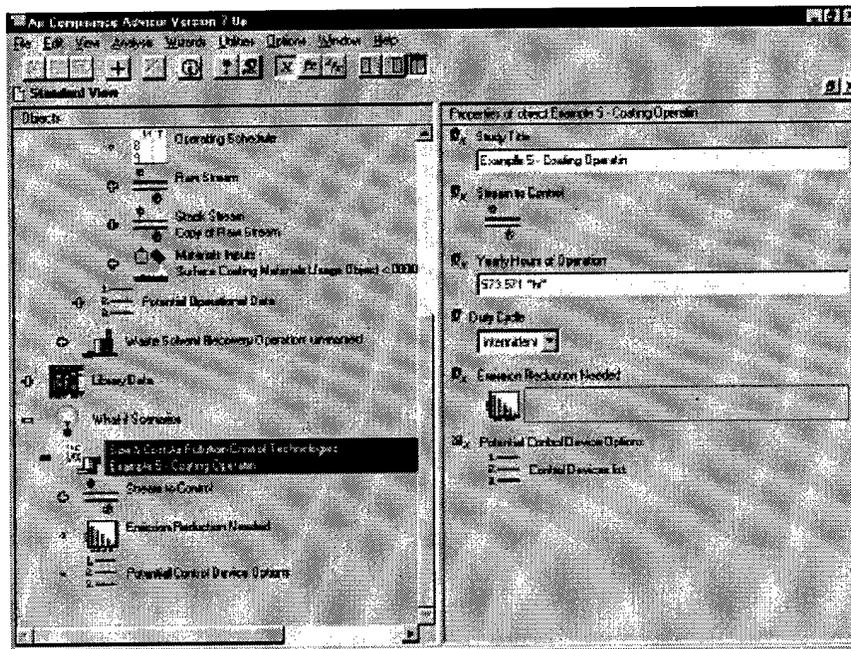
The ACA can be used to investigate the cost of control technologies. However this analysis is always based on information provided as part of the What-if Scenarios core object. Therefore a what-if scenario must be created before this analysis can be run. Because of the drag and drop feature of the ACA, a what-if scenario can be created in just a few steps.

First expand the Surface Coating object until the Raw Stream object is visible. Expand the What-if Scenarios object and then the Size & Cost Air Pollution Control Technologies object so that the Stream to Control object is visible. Then drag the Raw Stream object and drop it onto the Stream to Control object. The figure entitled "Drag and drop of Raw Stream object onto Stream to Control object" shows this procedure. An independent copy of the surface coating stream is now contained in the Stream to Control object.



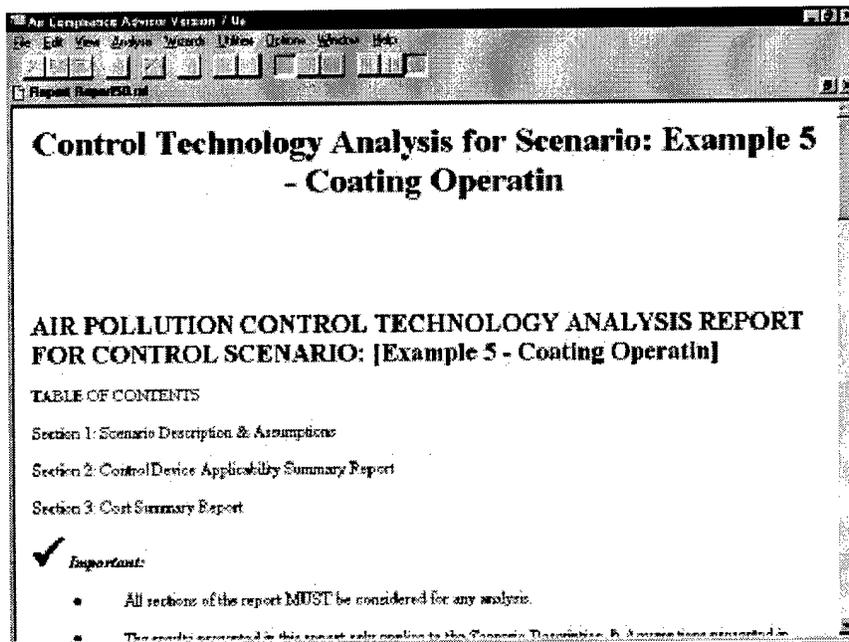
**Drag and drop of Raw Stream object onto Stream to Control object**

A few other data entry slots must be filled in before the Apply Control Technologies analysis can be run. First highlight the Size & Cost Air Pollution Control Technologies object and fill in the data entry slots for Study Title, Yearly Hours of Operation, and Duty Cycle as shown in the figure entitled "Entering What-if Scenario information for example #5". The duty cycle information was presented in the "Input Values for Example #5" Table and the yearly hours of operation were calculated in the Operating Schedule object. Finally, select the Emission Reduction Needed object and enter 90 "percent" into the VOC Reduction Needed data entry slot.



Entering What-if Scenario information for example #5

Select Analysis | Apply Control Technologies to run the analysis. Various control technologies will be applied to the waste gas streams from both of the emission units in an attempt to meet the desired reduction efficiencies. A new view with a report will be displayed with the analytical results. The figure entitled "Control device report for example #5" shows this report for the surface coating emission unit.



Control device report for example #5

The same steps could be applied to run this analysis for the waste solvent recovery unit.

### ***Running Applicability Analyses***

Finally, from the main menu select:

- Analysis | Prototype Analyses | Check for Applicable Regulations
- Analysis | Prototype Analyses | Check for Applicable Pollution Prevention Opportunities
- Analysis | Prototype Analyses | Check for Applicable Suggestions

The ACA will compare each of the sources in this example with ACA library data for federal regulations, pollution prevention opportunities, and suggestions. All results are saved to text files, which the user can access from outside the ACA. Again, file names are listed at the bottom of each of the corresponding "results" windows.

### ***Additional Things To Try***

For the waste solvent recovery unit, see how applying a control technology will effect facility-wide emissions. To do this, expand the **Size & Cost Air Pollution Control Technologies** object contained in the **What-if Scenarios** object in order to view the control technologies that were found to be applicable in the **Potential Control Device Options** object list. Select one of these control devices and locate the "After Control Output Stream." This stream contains an estimate of the air pollution emission rates **after** the associated control technology is applied. Drag and drop the "After Control Output Stream" stream to the Stack Stream associated with the actual operational data set for the waste solvent recovery unit. This process will override the Stack Stream's dependency on the Raw Stream and the **Stack Stream** object will now contain the emissions that were the result of the what-if scenario analysis. Finally, re-run the **Analysis | Summarize Emissions | From Stack (to Atmosphere)** from the main menu. Note how the actual emissions have been reduced.

# Appendix A — Tips on Using Units in the ACA

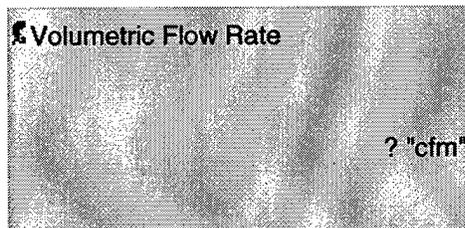
As you use the ACA, you will notice that many of the slots are floating point numbers with associated physical units. The information in this help file is to provide help in entering data that has associated units.

## ✓ *Important:*

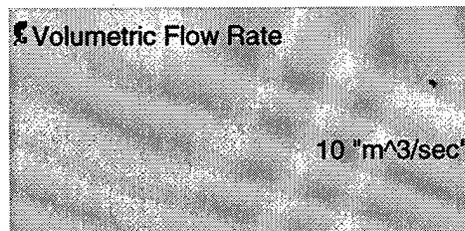
If you are interested in programming with units, while it is quite easy and intuitive, you should review the EXLGUI programmer's manual, which is available on the Internet at the D & E Technical Web site. [Click here to go there now.](#)

## General Rules for Entering Associated Units

Undefined slots that are floating point numbers with associated physical units will be presented in the slot window with a "?" and the default units (e.g., "m/s"). The default set of units should be used as a guide for the required dimensions of the units, and not as a requirement that the data be entered in those specific units. For example, let's assume that a slot in the ACA is requiring the volumetric flow-rate gas (i.e., volume of gas per unit time), the slot window (before being defined) might appear as:



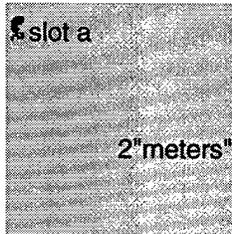
If the volumetric flow-rate was known to the user in "m<sup>3</sup>/sec" then the user could fill in the slot window with those units, e.g.,



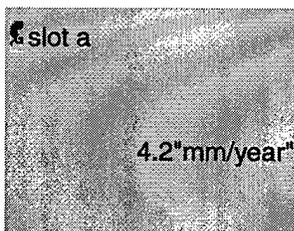
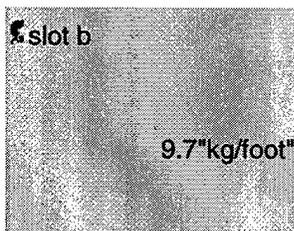
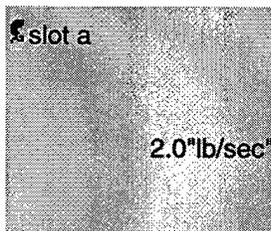
If, on the other hand, the user entered units of "kg/sec" a data-entry error would result.

The following "Tips" should help to clarify the remaining rules of using physical units when entering data.

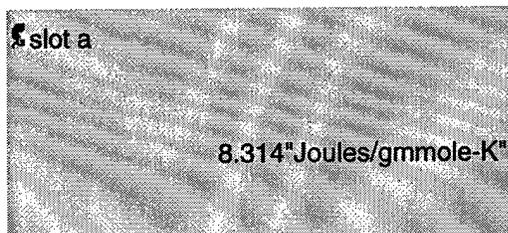
1. Units are separated from the floating point number in the slot window by at least one space and must be enclosed in double quotes, for example:



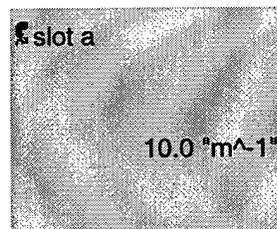
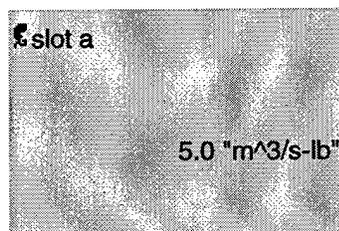
2. The units from one slot to another do **not** have to be in a consistent set of units, for example:



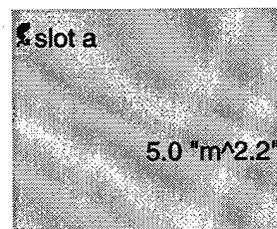
3. When entering units that contain more than one type of unit in the numerator or denominator of a slots units, then a "-" is used to separate the units. For example, the universal gas constant could be entered in to a slot as:



4. The ACA limits the raising of units to integer powers. So, for example, the following units are allowed:



The following would **not** be allowed:



5. Use only those base units and prefixes to units that are allowed in the ACA database. While the allowable list is fairly extensive, it is important to know what is available to take the greatest advantage of the ACA. A review of the following two tables ("Base Units" and "Prefixes To Units") is helpful.

⚠ **Caution:**

When raising units to a power, prefixes are considered first.

## Table of Base Units Available in the ACA

| Quantity           | Primary Unit | Other Acceptable Units            |
|--------------------|--------------|-----------------------------------|
| <b>Length</b>      | meter        | meters, m, Meters                 |
|                    | foot         | feet, ft, Feet                    |
|                    | inch         | inches, in, Inches                |
|                    | yard         | yards, yd, Yards                  |
|                    | mile         | miles, mi, Miles                  |
|                    | micron       | microns                           |
|                    | angstrom     | angstroms, Angstrom, Angstroms, A |
| <b>Mass</b>        | gram         | gm, grams, g, Grams               |
|                    | pound        | pounds, lb, lbs, Pounds           |
|                    | grain        | grains, gr                        |
|                    | ton          | tons, t                           |
| <b>Time</b>        | second       | seconds, sec, secs, Seconds, s    |
|                    | minute       | minutes, min, mins, Minutes       |
|                    | hour         | hours, hr, hrs, Hours             |
|                    | day          | days, Days                        |
|                    | week         | weeks, wk, Weeks                  |
|                    | year         | years, yr, Years                  |
| <b>Temperature</b> | kelvin       | Kelvin, K, kelvins, Kelvins       |
|                    | rankine      | Rankine, R, rankines, Rankines    |

|                    |                                                    |                                         |
|--------------------|----------------------------------------------------|-----------------------------------------|
|                    | C (not allowed in programming,<br>only data entry) |                                         |
|                    | F (not allowed in programming,<br>only data entry) |                                         |
| <b>Plane Angle</b> | radian                                             | radians, Radians                        |
|                    | degree                                             | degrees, deg, Degrees                   |
|                    | revolution                                         | revolutions, rev, Revolutions, Revs     |
| <b>Area</b>        | acre                                               | acres, Acres                            |
|                    | Hectares                                           |                                         |
| <b>Volume</b>      | liter                                              | liters, l, Liters                       |
|                    | gallon                                             | gallons, gal, Gallons                   |
|                    | pint                                               | pints, Pints                            |
|                    | quart                                              | quarts, Quarts                          |
|                    | cf                                                 | ft <sup>3</sup>                         |
| <b>Velocity</b>    | knot                                               | knots, Knots                            |
|                    | mph                                                |                                         |
| <b>Frequency</b>   | hertz                                              | Hertz, hz, Hz                           |
| <b>Force</b>       | newton                                             | newtons, Newton, Newtons, Neutons,<br>N |
|                    | dyne                                               | dynes, dyn, Dynes                       |
|                    | lbf                                                |                                         |
| <b>Energy</b>      | joule                                              | joules, Joule, Joules, J                |
|                    | WH                                                 | Watt-hours                              |
|                    | erg                                                | ergs, Ergs                              |
|                    | btu                                                | btus, Btu, BTUS, BTUs, btu              |
|                    | calorie                                            | calories, cal                           |
| <b>Power</b>       | watt                                               | watts, Watt, Watts, W, w                |
|                    | horsepower                                         | hp, BHP, Horsepower, horsepower         |

|                                     |            |                                                         |
|-------------------------------------|------------|---------------------------------------------------------|
| <b>Pressure</b>                     | pascal     | pascals, Pascal, Pascals, Pa                            |
|                                     | bar        | bars, Bars                                              |
|                                     | atmosphere | atmospheres, Atmosphere, Atmospheres, atm               |
|                                     | mmHg       | mmMercury, mmHg                                         |
|                                     | torr       | torrs, Torr, Torrs, torr                                |
|                                     | inchwater  | incheswater, inches_water, inwater, in_water, inchwater |
|                                     | footwater  | feetwater, footwater, ftwater, ft_water, feet_water     |
| <b>Value</b>                        | dollar     | dollars, DOLLAR, DOLLARS                                |
| <b>Volume per Time</b>              | cfm        | cubic(foot)/min, CFM                                    |
| <b>Current</b>                      | ampere     | Ampere, Amperes, Amp, amp, amps, Amps                   |
| <b>Voltage</b>                      | volt       | V, Volt, Volts, volts                                   |
| <b>Amount of Substance</b>          | mole       | gmole, gmmole                                           |
|                                     | lbmole     |                                                         |
| <b>Luminosity</b>                   | candela    |                                                         |
| <b>Concentration</b>                | ppm        |                                                         |
| <b>Relative Amount of Substance</b> | percent    |                                                         |

⚠ **Warning:**

When raising units to a power, prefixes are considered first.

## Table of Prefixes to Units

It is customary, especially in the metric system of units, to modify a unit name with a prefix, such as "kilo" or "milli." The unit conversion database contains a section where prefixes are specified. The prefixes in the default EXLGUI units conversion database are listed below.

| Prefix Name | Abbreviation | Value      |
|-------------|--------------|------------|
| atto        | a            | $10^{-18}$ |
| femto       | f            | $10^{-15}$ |
| pico        | p            | $10^{-12}$ |
| nano        | n            | $10^{-9}$  |
| micro       | u            | $10^{-6}$  |
| milli       | m            | $10^{-3}$  |
| centi       | c            | $10^{-2}$  |
| deci        | d            | $10^{-1}$  |
| deca        | ---          | $10^1$     |
| hecto       | h            | $10^2$     |
| kilo        | k            | $10^3$     |
| mega        | M            | $10^6$     |
| giga        | G            | $10^9$     |
| tera        | T            | $10^{12}$  |
| peta        | P            | $10^{15}$  |
| exa         | E            | $10^{18}$  |

### *Warning:*

When raising units to a power, prefixes are considered first.

## Examples With Units

### *Units Example 1*

"km<sup>3</sup>" is equivalent to "(km)<sup>3</sup>"

### *Units Example 2*

Assume that you are told that the cost of a material is \$100 per thousand (1000) cubic feet. This material cost would first be defined as:

```
var cost: float "dollar/ft^3";
```

The following ways would be correct ways to specific the value:

```
cost:= 100 "dollar"/ 1000 "cf";  
cost:= 0.1 "dollar/ft^3";  
cost:= 0.1 "dollar/cf";  
cost:= 100 "dollar/kcf";
```

The following ways would **not** be a correct way to specific the value:

```
cost:= 100 "dollar/kft^3";  
// this value is equivalent to $100,000,000. per thousand cubic  
feet!
```

# Appendix B — Tips on Obtaining and Estimating Chemical Property Data

The standard ACA chemical database contains the commonly available chemical properties for a wide range of common volatile organic compounds (VOCs) and hazardous air pollutants (HAPs). To date, estimated chemical properties have not been used to "fill out" the ACA chemical database. The end user may need to either use estimated chemical properties (for those chemicals already listed in the ACA's chemical database) and/or obtain measured values from the literature for chemicals that are not yet in the ACA database that they choose to add. The guidance in this help menu will help to point the user in the right direction for obtaining and estimating the chemical properties that are need for costing and sizing air pollution control technologies.

## Background on the Chemical/Pollutant Properties Required By the ACA

The following chemical properties are used by the ACA in determining the size and cost of air pollution control technologies (along with the associated control technology calculations that require the properties).

### *Background on the VOC Chemical Properties Required By the ACA*

#### *VOC Chemical Properties Required By the ACA — Listed By Property*

| <b>Chemical Property</b>               | <b>Required for Analysis of...</b>         |
|----------------------------------------|--------------------------------------------|
| Antoine Constants for Vapor Pressure   | Refrigerated condensers, carbon adsorbers* |
| Boiling Point                          | Refrigerated condensers, carbon adsorbers  |
| Critical Temperature                   | Refrigerated condensers                    |
| Diffusion Coefficient, Liquid in Water | Gas absorbers                              |
| Diffusion Coefficient, Vapor in Air    | Gas absorbers                              |
| Gas Phase Heat Capacity                | Refrigerated condensers                    |
| Heat of Combustion                     | Thermal incinerators and flares            |
| Heat of Condensation                   | Refrigerated condensers, carbon adsorbers  |
| Henry's Law Constant                   | Gas absorbers                              |
| Index of Refraction                    | Carbon adsorbers*                          |

|                             |                                                    |
|-----------------------------|----------------------------------------------------|
| Is Halogenated?             | Thermal incinerators                               |
| LEL (Lower Explosive Limit) | Thermal incinerators, flares, and carbon adsorbers |
| Liquid Density (at STP)     | Carbon adsorbers*                                  |
| Molecular Weight            | All VOC control devices                            |
| Vapor Pressure @ 293 K      | Carbon adsorbers*                                  |
| Yaws Coefficients           | Carbon adsorbers*                                  |

 **Note:**

(\*) The analysis for Carbon adsorbers requires either just the Yaws Coefficients or Antoine Constants / Vapor Pressure @ 293 K, Boiling Point, Heat of Condensation, Index of Refraction, LEL, Liquid Density, and Molecular Weight.

*VOC Chemical Properties Required By the ACA — Listed By Control Technology*

| <b>Control Technology</b>               | <b>Required Chemical Properties</b>                                                                                                              |
|-----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Carbon adsorbers (Option #1 parameters) | Antoine Constants or Vapor Pressure @ 293 K, Boiling Point, Heat of Condensation, Index of Refraction, LEL, Liquid Density, and Molecular Weight |
| Carbon adsorbers (Option #2 parameters) | Yaw's Coefficients and Molecular Weight                                                                                                          |
| Flares                                  | Heat of combustion, LEL, and Molecular Weight                                                                                                    |
| Gas absorbers                           | Diffusion coefficient — liquid in water, Diffusion coefficient — vapor in air, Henry's Law Constant, Molecular Weight                            |
| Refrigerated condensers                 | Antoine Constants for Vapor Pressure, Boiling Point, Critical Temperature, Gas Phase Heat Capacity, Heat of Condensation, Molecular Weight       |
| Thermal Incinerators                    | Heat of Combustion, Is Halogenated?, LEL, and Molecular Weight                                                                                   |

*VOC Chemical Properties Definitions*

*Antoine Constants for Vapor Pressure (i.e., A, B, C)*

The Antoine constants are for equations calculating vapor pressure (P) as a function of temperature (T). In the ACA, the Antoine constants must be in the form that will calculate pressure in units of mmHg, give temperature in units of "C." The form of the equation is:  $\log(P) = A - (B/(T+C))$ . This parameter IS Temperature dependent. This parameter is required for: refrigerated condensers and carbon adsorbers\*.

#### *Boiling Point*

This parameter gives the temperature at which a liquid will boil. While the boiling point can be measured at a variety of pressures, it is typically at standard pressure, unless otherwise noted. In the ACA, the boiling point is for standard pressure. This parameter is NOT temperature dependent. This parameter is required for: refrigerated condensers and carbon adsorbers.

#### *Critical Temperature*

The critical temperature is the temperature above which a compound in the gas phase cannot become liquefied with an increase of pressure. This parameter is NOT temperature dependent. This parameter is required for: refrigerated condensers.

#### *Diffusion Coefficient, Liquid in Water*

This parameter is a measure of the diffusivity of the chemical in water. This parameter IS temperature dependent. This parameter is required for: gas absorbers.

#### *Diffusion Coefficient, Vapor in Air*

This parameter is a measure of the diffusivity of the chemical in air. This parameter IS temperature dependent. This parameter is required for: gas absorbers.

#### *Gas Phase Heat Capacity*

This parameter is a measure of the amount of heat required to raise the temperature of a unit mass of a compound (in the gas phase) one degree. This parameter IS temperature dependent. This parameter is required for: refrigerated condensers.

#### *Heat of Combustion*

This parameter is a measure of the energy that is given off when the chemical (in the gas phase) is completely oxidized. Data for this parameter may be available on a mass-basis, on a volume-basis, or on a molar-basis. This property should be given for STP, but it is not strongly temperature dependent. This parameter is required for: thermal incinerators and flares.

#### *Heat of Condensation at Boiling Point*

This parameter is a measure of the amount of energy that is given off when a vapor condenses to a liquid at its normal boiling point. It should be entered into the ACA as a positive number and is equal in magnitude to the enthalpy (heat) of vaporization. The heat of condensation at a given condensation temperature other than the boiling point will be estimated directly in the ACA using the Watson equation (OAQPS, equation 8.13). This parameter is NOT temperature dependent. This parameter is required for: refrigerated condensers and carbon adsorbers.

### *Henry's Law Constant*

This parameter is a measure of the slope of the gas absorption equilibrium curve over a small temperature range. The ACA offers two ways to model gas absorbers: 1) a graphical approach; and 2) an approach that uses the Henry's Law Coefficient. The default approach is using the Henry's Law Coefficient since this can be done using chemical properties that are readily available in the ACA database and are not dependent on the specifics of the pollutant stream concentration and removal efficiency desired. The Graphical Method is more accurate, but requires a graph of the equilibrium curve and operating line. Obtaining the data required for the Graphical Method is more difficult than the more simplistic approach of using the Henry's Law Approach. The Graphical Approach requires that the following two additional parameters: 1) the Outlet equilibrium pollution concentration in solvent,  $X_o^*$  (on the op. line); and 2) the Gas pollution concentration in equilibrium with  $X_o$  ( $Y_o^*$ ). The Henry's Law Approach requires the Henry's Law Coefficient to be entered. This parameter IS temperature dependent. This parameter is required for: gas absorbers.

### *Index of Refraction*

The index of refraction provides information on how a chemical interacts with light and is the ratio of the speed that light travels in a perfect vacuum to the speed that light will travel through the chemical. This parameter is used in the Calgon Corporation's modified Dubinin-Radushkevich equation to calculate the polarizability used in estimating the adsorptivity of a compound on activated carbon. This parameter is NOT temperature dependent. This parameter is required for: carbon adsorbers.

### *Is Halogenated?*

Halogenated compounds include those compounds that contain group VIIA elements from the periodic table, namely: fluorine (F), chlorine (Cl), bromine (Br), iodine (I), and astatine (At). This parameter is used to determine the required combustion temperature for thermal incineration — halogenated compounds require a higher combustion temperature. This parameter is NOT temperature dependent. This parameter is required for: thermal incinerators.

### *LEL (Lower Explosive Limit)*

This parameter is a measure of the minimum concentration in air that will support combustion. Concentrations greater than 25% of the LEL are typically considered to be unsafe for standard air pollution control technologies. This parameter is NOT temperature dependent. This parameter is required for: thermal incinerators, flares, and carbon adsorbers.

### *Liquid Density (at STP)*

This parameter is a measure of the density of the liquid at standard temperature and pressure. The liquid density is equal to the (molecular weight)/(liquid molar volume). The liquid density is used to calculate the mass loading for the carbon adsorbers calculations. This parameter is NOT temperature dependent. This parameter is required for: carbon adsorbers.

### *Molecular Weight*

This parameter is NOT temperature dependent. This parameter is required for: all VOC control devices.

### *Vapor Pressure @ 293 K*

This parameter is the specification of a compound's vapor pressure at standard temperature (i.e., 293 K). If the Antoine constants are known, then they can be used to estimate this parameter, but if actual data is available it is preferred. DATA VALID AT 293 K, 1.0 atm. This parameter is required for: carbon adsorbers\*.

### *Yaws Coefficients*

The Yaws Coefficients (i.e., A, B and C) are used as part of an alternate approach to calculating the adsorptivity of a compound on activated carbon. These coefficients will be used/requested if all of the chemical parameters that are required for the Calgon Corporation's modified Dubinin-Radushkevich equation are not available. Pollution Engineering presented an article by Dr. Charles Yaws that provides a carbon adsorption equilibrium concentration equation and associated coefficients for a long list of VOC chemicals. The equation used to calculate the adsorption capacity (Q, in units of "cm<sup>3</sup>/hg") is given as:  $\log_{10}(Q) = A + B \cdot \log_{10}(Y) + C \cdot (\log_{10}(Y))^2$ , where Y is the VOC concentration in units of "ppm." Based upon an agreement that the ACA group and the US EPA / OAQPS / ISEG group has with Dr. Yaws, a limited number of chemicals (i.e., 30) with associated Yaws coefficients are given in the ACA database. The remaining chemicals in the ACA database require the user of the ACA to input the equation coefficients into their own personalized copy of the ACA themselves from either the table in the Pollution Engineering journal article or by buying Dr. Yaws' database (which is approximately \$40). Professor Carl Yaws can be reached at: P.O. Box 10053, Beaumont, Texas, 77710, 409-880-8787. DATA VALID AT 298 K.

Required for: Carbon adsorbers.

### ***Background on the Particulate Matter (PM) pollutant properties required by the ACA***

The properties listed on the PM Pollutant Properties Screen are those properties that are required for determining the cost and applicability of a variety of PM air pollution control technologies. Definitions of the various properties that are required and are to be supplied by the end-user are given below.

#### *Is Water Soluble*

Specify if the PM is soluble in water.

#### *Particulate Phase*

Specify if the PM is a liquid or a solid.

#### *PM Density*

This is a measure of the density of the PM.

### *PM Size Distribution*

Required for: All PM control technologies.

This parameter represents the aerodynamic particle size distribution for pollutants that are classified as particulate matter (PM). If a chemical/pollutant is a gas, do NOT enter any data in this slot. For PM data, it is advised that at least five sets of data (particle size versus percent mass above) be entered. This data will be used to calculate: (1) mass median diameter; (2) cut diameter; and (3) geometric standard deviation — these parameters are typically required to determine the applicability of PM air pollution control devices. The ACA will assume a best-fit log-normal aerodynamic particle size distribution (this gives a straight line on log-probability paper when plotted) in the calculation of the mass median diameter, the geometric standard deviation, and the cut diameter.

## **Literature References for Obtaining VOC Chemical Properties**

There are numerous resources available for obtaining the chemical properties required by the ACA. Some of the more commonly available references that have been used to date in obtaining chemical properties for the ACA chemical database are listed below. With each reference listed, the relevant chemical properties that can potentially be estimated are listed.

***Perry, R.H., and D. W. Green, eds., Perry's Chemical Engineers Handbook, 6th ed., McGraw-Hill, New York, 1984.***

- Heat of combustion — values
- Liquid Density @ STP — estimation method
- Gas Phase Heat Capacity — estimation method
- Critical Temperature — values
- Heat of Condensation — values

***Lide, D. R., ed., CRC Handbook of Chemistry and Physics, 72nd ed., CRC Press, Inc, Cleveland, OH, 1977.***

- Antoine Constants for Vapor Pressure (i.e., A, B, C)
- Boiling Point
- Critical Temperature
- Diffusion Coefficient, Liquid in Water
- Gas Phase Heat Capacity
- Heat of Combustion

- Heat of Condensation
- Henry's Law Constant
- Index of Refraction
- LEL (Lower Explosive Limit)
- Liquid Density (at STP)
- Molecular Weight

***Dean, J. A., ed., Lange's Handbook of Chemistry, 14th ed., McGraw-Hill, Inc, New York, 1992.***

- Antoine Constants for Vapor Pressure (i.e., A, B, C)
- Boiling Point
- Critical Temperature
- Gas Phase Heat Capacity
- Heat of Condensation
- Index of Refraction
- Liquid Density (at STP)
- Molecular Weight
- Vapor Pressure @ 293 K

***Yaws, Carl L., Chemical Properties Handbook: Physical, Thermodynamic, Environmental, Transport, Safety, and Health Related Properties, McGraw-Hill, Inc, New York, 1999.***

- Vapor Pressure Equations — Non-Antoines Equation
- Boiling Point
- Critical Temperature
- Gas Phase Heat Capacity
- Heat of Combustion
- Heat of Condensation
- Henry's Law Constant
- Index of Refraction

- LEL (Lower Explosive Limit)
- Liquid Density (at STP)
- Molecular Weight

***Yaws, Carl L. et al., "Determining VOC Adsorption Capacity," Pollution Engineering, February 1995, pp. 34-37.***

The Yaws Coefficients (i.e., A, B and C) are used as part of an alternate approach to calculating the adsorptivity of a compound on activated carbon. These coefficients will be used/requested if the chemical parameters that are required for the Calgon Corporation's modified Dubinin-Radushkevich equation are not all available. Pollution Engineering presented an article by Dr. Charles Yaws that provides a carbon adsorption equilibrium concentration equation and associated coefficients for a long list of VOC chemicals. The equation used to calculate the adsorption capacity (Q, in units of "cm<sup>3</sup>/hg") is given as:  $\log_{10}(Q) = A + B \cdot \log_{10}(Y) + C \cdot (\log_{10}(Y))^2$ , where Y is the VOC concentration in units of "ppm." Based upon an agreement that the ACA group and the US EPA / OAQPS / ISEG group has with Dr. Yaws, a limited number of chemicals (i.e., 30) with associated Yaws Coefficients are given in the ACA database. The remaining chemicals in the ACA database require the user of the ACA to input the equation coefficients into their own personalize copy of the ACA themselves from either the table in the Pollution Engineering journal article or by buying Dr. Yaws' database (which is approximately \$40). Professor Carl Yaws can be reached at: P.O. Box 10053, Beaumont, Texas, 77710, 409-880-8787.

#### ***NIST WWW Database***

The following Internet site contains the NIST electronic chemical database:  
<http://webbook.nist.gov/chemistry/>

Relevant properties that are available (combination of measured and estimated values) include:

- Chemical Structures\*
- Molecular Weight
- Heat Capacity — Gas Phase
- Boiling Point
- Critical Temperature
- Heat of Vaporization // Heat of Condensation
- Antoine Equation Constants:  $\log_{10}(P) = A - (B / (T + C))$

- Vapor Pressure Data
- Heat of Combustion
- Henry's Law Constant (Water Solution)

 **Note:**

\*The chemical structure can be of value when using some chemical property estimation methods.

## Literature References for Estimating Chemical Properties

The following references have chemical property estimation techniques that may be of value. With each reference listed, the relevant chemical properties that can potentially be estimated are listed.

***Edward J. Baum "Chemical Property Estimation — Theory and Application," Lewis Publishers, New York, 1998***

- Boiling Point
- Diffusion Coefficient, Liquid in Water
- Diffusion Coefficient, Vapor in Air
- Henry's Law Constant
- Absorbers
- Liquid Density (at STP)
- Vapor Pressure Data

***Lyman, W.J., W. F. Reehl, and D. H. Rosenblatt, eds., Handbook of Chemical Property Estimation Methods, McGraw-Hill, New York, 1982.***

- Boiling
- Critical Temperature
- Diffusion Coefficient, Liquid in Water
- Diffusion Coefficient, Vapor in Air
- Gas Phase Heat Capacity
- Heat of Condensation
- Henry's Law Constant

- Index of Refraction
- Liquid Density (at STP)
- Vapor Pressure Data

***Douglas A. Logan, "Estimating Physical Properties for Control Equipment Design,"  
Environmental Progress, Winter, 1997, Vol. 16, No. 4.***

- Heat of Combustion
- Vapor Pressure
- Heat of Vaporization
- LEL
- Liquid Heat Capacity
- Gas Phase Diffusion Coefficients
- Molecular Volume
- Liquid Phase diffusivity
- Vapor Liquid Equilibrium (Henry's Law Constant)

## Appendix C — End-User Extensibility

Basic end-user extensibility of the ACA deals with the ability of the user to modify (override) the functionality of slots directly from the graphical user interface by using the **Show slot functions** button. The details of basic end-user extensibility are limited to using the **View detailed information** button to identify the variable names of other slots that belong to the same class. This is very much like programming a spreadsheet.

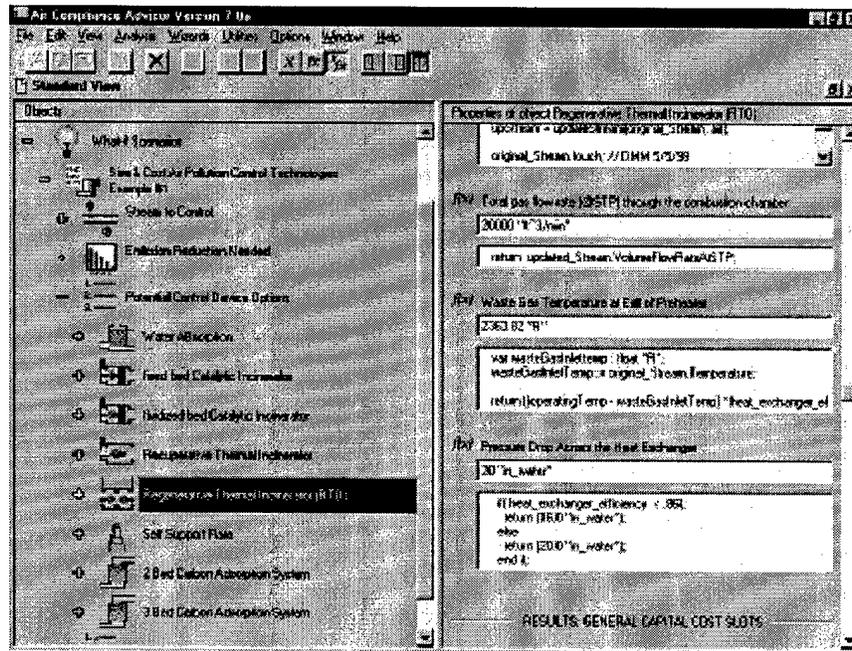
Like basic end-user extensibility, intermediate end-user extensibility replaces/adds equations to the user's data, but does not add new source code to the core ACA program. In other words, all basic and intermediate end-user extensions (see below) will "reside" in the data files for the ACA (e.g., \*.oxl files) and will be associated with a single object. The fact that the extension resides in the data files should not lead one to believe that extensions at these two levels (basic and intermediate) need be trivial. Entire complex algorithms can be changed.

Advanced end-user extensibility, on the other hand, considers those additions to the ACA source code that reside outside of the data files. They are compiled when the ACA is started and become part of the core system. For example, a user can define a new EXLGUI class representing a new type of air pollution source. This new source type can then be used exactly like the source types that are included in the standard ACA library.

Note that the information presented here does **not** address the details of the EXLGUI programming language. It is assumed that the reader of this appendix has a basic understanding of the features of EXLGUI. If the reader is not familiar with the EXLGUI programming language, then they should refer to the EXLGUI Programmer's Manual, Version 3.1.

### Basic End-User Extensibility

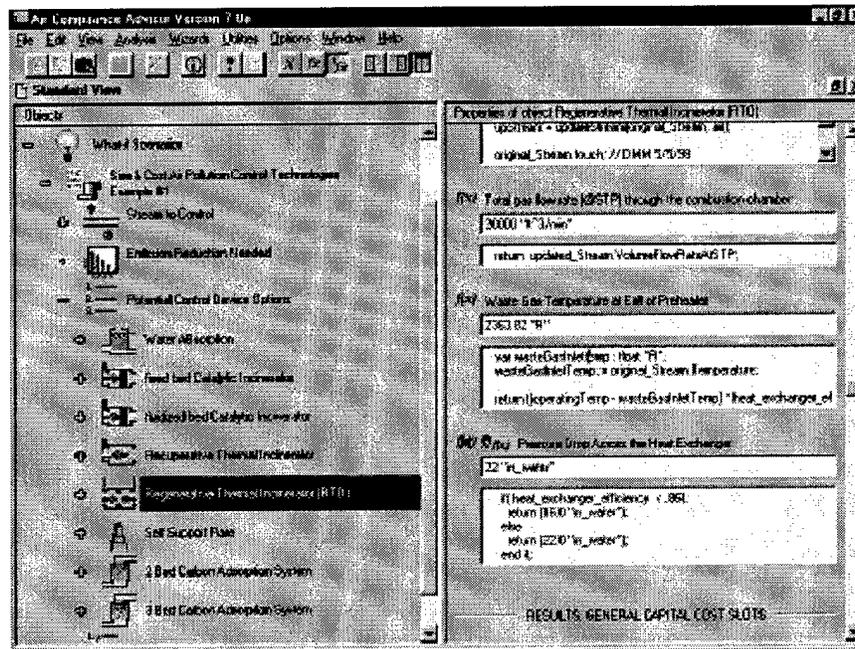
Users can edit existing functions or create new functions in the function view. Users can view the underlying functions of data entry slots by selecting the **fx** button to view both the functions and the values or the **fx** button to view the functions alone. The **fx** button will toggle the view back to just values. The figure below shows an example of the right pane of the **Standard View** displaying both values and functions. The list of potential control device options is generated after running the **Apply Control Technologies** analysis for a **What-if Scenario**. Details of this analysis is discussed in the "Apply Control Technologies" Section of this manual. The data entry slot for **Pressure drop** will be used for a simple example of how users can modify functions.



### Data entry slot values and functions

The function view (i.e., using the "Show slot functions" button ) for the Pressure drop data entry slot shows a small section of computer code. This code is the function for pressure drop. In this case the function is a simple if-then type statement and the value of the slot is dependent on the slot name `heat_exchanger_efficiency`. If a user had updated information from a vendor indicating that the pressure drop should really be 22 inches of water for heat exchanger efficiencies greater than or equal to 0.86, the user could modify the function to reflect this new information. The figure below shows an edited function and the result this change had on the value of the Pressure drop data entry slot. The icon showing the slot type also changed to indicate that a user-supplied function had been entered.

This brief example and discussion only covers a very small portion of the user extensibility options within the ACA.



### Results from editing pressure drop function

Users can obtain detailed information about a data entry slot by clicking on the toolbar button . The information includes the slot name of the data entry slot. The slot name is required to create user-supplied functions that depend on that slot.

The "Set slot to unknown" toolbar button  and the "Reset slot to default value" toolbar button  are used to set slots to unknown and to reset slots to the default value respectively. For data entry slots, setting a slot to unknown sets the value to unknown and also removes any underlying function. For data entry slots with a function, resetting the slot to the default value will regenerate the function and the value calculated by the function.

#### Note:

If a data entry slot without an underlying function is first set to unknown and then reset to the default value, the original value will not be recovered since there is no function to calculate the value.

| Icon                                                                              | Description                              |
|-----------------------------------------------------------------------------------|------------------------------------------|
|  | Reveal detailed information about a slot |
|  | Show slot values                         |
|  | Show slot functions                      |
|  | Show slot values and functions           |

#### Toolbar buttons

## Intermediate End-User Extensibility

Intermediate end-user extensibility can be considered the ability for the end-user to reprogram a slot by using global declarations (i.e., constants, procedures, functions, and types), as well as by referencing slots, procedures and functions of other objects in the ACA's data containment hierarchy. These two forms of intermediate end-user extensibility will be addressed separately.

### *The Types of Objects view*

To determine how a particular object is defined, it is best to use the **Types of Objects** window. Choosing any of the available object types, one will see a list of the properties associated with the object in the **Properties of Objects** pane. By selecting a particular property, the EXLGUI source code associated with that property will appear in the **Definition of Property** pane located below the **Types of Objects** and **Properties of Objects** panes. The user can then determine how a particular slot, function, or procedure is defined, and decide whether to modify its definition by overriding it with a particular value, modifying an existing equation, or entering a new equation.

The **Types of Objects** view will be most useful for advanced users who wish to extend the capabilities of the ACA. The **Types of Objects** view includes all of the various data types that are available in the ACA. In many cases the data types are arranged hierarchically. The parameters (or "slots"), functions, and procedures associated with each of these data types can be viewed from this view, as can the source code that makes up each slot. The first pane in the **Types of Objects** view, the upper-left pane, is titled **Available Object Types** and includes all of the data types that exist in the ACA. These are the hierarchically arranged data types that have been defined in the ACA. This pane is the same as the **Types of Objects** view in previous versions of the ACA. The second pane, the upper-right pane, of the **Types of Objects** view is titled **Properties of Object**. These are the slots (i.e., parameters), functions, and procedures associated with the "object type" that is selected in the upper-left view. The last pane, the bottom pane, in the **Types of Objects** view allows the user to view the source code that describes the slot, function, or procedure selected in the **Properties of type ...** pane. The title of this pane changes and is based upon the selected slot, function, or procedure.

### ***Referencing global declarations***

Global declarations can be used at any point in the ACA source code. Many slots within the ACA are described by using a global declaration of one type or another. For example, the global constant "GasLawConstant" is a global constant that is used in the calculation of many slots.

An EXLGUI identifier can reference either a local, class-local, or global declaration. For example, consider the following source code fragment:

```
var G: integer;
-----
class C;
    slot S: integer;
    function F: integer;
end class;
-----
slot C.S: integer;
    const Z = 3;
    return G + F + Z;
end slot;
```

The return statement in slot S references three declarations: G (a global variable), F (a class-local function), and Z (a local constant). When the user wishes to find the declaration of an identifier that is encountered in a slot's source code, they should first search for the identifier's declaration in the current slot's source code, then in the current slot's class, then in the list of global declarations.

When searching for an identifier's declaration in a class, either the **Standard View** or the **Type of Objects** view can be used. Global declarations can be searched for in the **All Global Identifiers** view, which can be displayed by choosing **View|Advanced Views|All Global Declarations** from the menu.

### ***Referencing slots, procedures and functions of other objects in the ACA's data containment hierarchy***

The three core objects in the ACA are the reference points for identifying slots, functions, and procedure used in other calculations. The three core objects and their programming name are:

```
Library Data = library
Installation Data = installation
What-if Scenarios = worksheet
```

With this bit of knowledge, the end user can reference slots, procedures, and functions of any other instance contained in the ACA data hierarchy. For example, the stream that is associated with the **Control Technology What-if Scenario** can be referenced like this:  
**Worksheet.SingleControlScenario.Stream**

The programming names for the other slots following the core object were found by selecting those slots in the ACA and then pressing the "View detailed information" toolbar button .

# Advanced End-User Extensibility

Advanced end-user extensibility can be considered the ability for the end-user to add new source code to the ACA. New actions, classes, subclasses, global variables, global functions, and global procedures can be added to the ACA.

## *The basics of adding new source code to the ACA*

In order to add new source code to the ACA, the user must perform three steps.

### *Step 1: Create one or more text files to hold the source code.*

All EXLGUI source code must be stored in text files. These files can be created by any text editor. The Notepad® utility that ships with the Microsoft Windows® operating systems works fine. Word processors such as Microsoft Word® should not be used, as they store documents in a proprietary binary format that cannot be read by the EXLGUI.

Files that hold EXLGUI source code should be given the extension .exl.

As an aid to the user, the ACA ships with a sample source code file called `User_Defined_Code.exl`. This file contains sample EXLGUI declarations. The user can add their own declarations to this file, using the existing sample code as a template for their own.

The actual source code for the ACA is stored in the file `ACA_Version_6_0_Source_code.exl`. This file is shipped in an encrypted format to protect it from user modifications. This code can be view through the EXLGUI (See Other Notes below).

### *Step 2: Insure that the files are included when the ACA is compiled*

When the ACA is first started, it reads a file called `exlgui.fx1`. This file contains a list of all of the EXLGUI source code files that are compiled as part of the ACA. The version of this file that ships with the ACA contains the following two lines.

```
uses 'ACA_Version_6_0_Source_code.exl';  
uses 'User_Defined_Code.exl';
```

These lines tell the EXLGUI that all of the ACA source code is contained in the two files: `ACA_Version_6_0_Source_code.exl` and `User_Defined_Code.exl`. The user can add lines to `exlgui.fx1` to reference other source code files if desired.

### *Step 3: Add the source code to the files.*

New EXLGUI source code should be added to an .exl file in the form of one or more EXLGUI declarations, separated by lines of 5 dashes. The sample source code in `User_Defined_Code.exl` can be used as a model. The EXLGUI Programmer's Manual should be consulted for details.

## *Other Notes Concerning End-User Extensibility*

- All ACA source code, including any user-supplied code, is compiled each time the ACA is

started. If there are any errors in the code, a message will be displayed at this time. If the user edits user-supplied code, the changes will not take effect until the ACA is restarted.

- All declarations (including user-supplied ones) can be seen directly from the graphical user interface (GUI) of the ACA, including:

- **Classes**

- Seen in the upper left pane when you select **View | Advanced Views | Types of Objects** from the GUI. This pane is called the **Available Object Types**. Note that the objects are listed in a hierarchy (e.g., **Emission Units** can be found by expanding the **Physical Objects**). This hierarchy represents the class inheritance hierarchy (i.e., base classes appear at the roots of this hierarchy, and their derived classes appear as children).

- The class definition source code (i.e., class name in the code; **ClassOptions**; its super-class; and all slots, procedures & functions for the class) can all be seen in the pane at the bottom of the **Types of Objects** window when a specific class is selected in the upper left pane.

 **Note:**

See the "EXLGUI Programmer's Manual," Version 3.1, page 29 for details.

- **Class declarations: i.e., slots, member functions, and member procedures**

- Seen in the upper right pane when you select a specific object in the **Types of Objects** view. This pane is called the **Properties of type....** When selecting a specific property in the upper right pane, the code that describes that property will be displayed in the pane at the bottom of the window called **Default definition of .....** This is where the source code of class functions and procedures can be seen. (Only slots source code can be seen from the **Standard View**)

- **Global declarations: i.e., constants, variables, types, functions, procedures, and actions**

- Currently, a subset of the global declarations can be seen when you select the option **View | Advanced | All Global Declarations**. In the near future, there will also be options for viewing a subset of the global declarations (e.g., "Global Actions").

- The code associated with the various global declarations will be displayed. Currently, this data appears as a simple text listing. In the future, it will appear as a two-paned window that operates similarly to the **Types of Objects** window.

# Appendix D: Generating an ACA Control Device Report from the Command Line

## Overview

The Air Compliance Advisor's (ACA's) Control Technology Sizing and Costing functionality can be accessed from external programs by running the ACA in a special command line mode. When run in this mode, the ACA will initialize itself, read a file that contains the same information as is contained in the "Size and Cost Air Pollution Control Technologies" object, run the "Apply Control Technology" action, write the control device report to a file, and then exit. The external application can then read the report file and process it as needed.

To access this functionality, run the ACA using the following command line option...

```
exlgui -controlstream [controlstreamfile]
```

...where [controlstreamfile] is the file that contains a description of the stream to control, which is described below.

The ACA will write the control device report to the file "report1.txt".

A sample control stream file, named SampleControlStreamFile.dat, is shipped with the ACA.

Important Notes:

- 1) The Control Stream File Format is also referred to as the ACA Control Technology Analysis Engine Interface Input File (i.e., ACTAEIIF) format.
- 2) Control Stream files can also be imported directly from the ACA by selecting  
the menu option:  
File | Import | Air Pollution Control Technology Data (ACTAEIIF format)

## Overview of the Format of the Control Stream File

The Control Stream File is an ASCII file consisting of a sequence of data items (i.e., integers, floating-point numbers, and text strings) in free format. One or more characters of white space must separate two consecutive items. White space consists of blanks, tabs, and newline characters.

Every data item has a type. The three types of data items are:

Integer - Cannot contain a decimal point. Example of integers: 0, -100, 23456

Floating Point Number - Can optionally contain a decimal. Exponential notation can be specified using 'e' or 'E'. Examples of floating point numbers: 0 1.0, -2.3e10, 0.4E-5

Text String - Delimited by double quotation marks ". To put a literal double-quote in a string, use the three-character sequence {\"}. To put a literal opening curly bracket { in the string, use the three-character sequence {{}. To put a literal closing curly bracket into the string, use the three-character sequence }}. To put a line break in the string, use the three-character sequence {n}.

Comments can appear in the file. Comments start with the pound sign #, and go to the end of the line.

The phrases "For each" or "For every" in the description of the file format imply a list of items. Lists are terminated by the word END with no surrounding quotation marks. Note that the END marker must be present even if the list is empty.

For example, a list of pollutants and associated flow rates in a pollutant stream is described like this in the file format description:

For each pollutant in the pollutant stream:  
Name of the pollutant (text)  
Flow rate of the pollutant (float)

An example of a list conforming to the above description would be:

```
"Benzene"          1320.0
"Methyl Chloride"  2540.0
END
```

## Format of the Control Stream File

For each pollutant to be defined:

Pollutant Name (text)

Note: This must be unique, and must not be the same as any existing pollutant in the ACA chemical library.

Code for type of pollutant (integer)

Note: Use 0 for VOC, 1 for Particulate

if (type of pollutant is VOC):

Molecular weight, in gram/gmmole (float)

(float)

Heat of combustion, gas phase, volumetric at STP, in BTU/ft<sup>3</sup>

Note: Specify -9.9e99 if this parameter is not known

Boiling point, in degrees K (float)

Note: Specify -9.9e99 if this parameter is not known

Critical Temperature, in degrees K (float)

Note: Specify -9.9e99 if this parameter is not known

Antoine Constant A (float)

Antoine Constant B (float)

Antoine Constant C (float)

Note: The Antoine constants are for equation  
calculating P in mmHg using the equation  
 $\log(P) = A - (B/(T + C))$ , with T in deg C

Note: Specify -9.9e99 for any of these parameter that are not

known

Liquid Density, in lb/ft<sup>3</sup> (float)

Note: Specify -9.9e99 if this parameter is not known

Heat Of Condensation, in BTU/lb (float)

Note: Specify -9.9e99 if this parameter is not known

Refractive Index (float)

Note: Specify -9.9e99 if this parameter is not known

Henry's Law Constant, in gmmole/l atm (float)

Note: Specify -9.9e99 if this parameter is not known

Diffusion Coefficient, Liquid in Water, in cm<sup>2</sup>/sec (float)

Note: Specify -9.9e99 if this parameter is not known

Diffusion Coefficient, Vapor in Air, in cm<sup>2</sup>/sec (float)

Note: Specify -9.9e99 if this parameter is not known

Heat capacity, in BTU/lb-R (float)

Note: Specify -9.9e99 if this parameter is not known

Lower Explosive Limit (LEL), in ppm (float)

Note: Specify -9.9e99 if this parameter is not known

Vapor Pressure, in psi (float)

Note: Specify -9.9e99 if this parameter is not known

Lowest Temperature for which Temperature-Dependent Parameters  
Apply, in degrees K (float)

Highest Temperature for which Temperature-Dependent Parameters  
Apply, in degrees K (float)

Note: Specify -9.9e99 for two these parameters if they are  
not known

Note: The following parameters are temperature-dependent:

- 1) Antoine Constants
- 2) Gas-Phase Heat Capacity
- 3) Henry's Law Constant
- 4) Liquid in Water and Vapor in Air Diffusion

Coefficients

Is Pollutant Halogenated? Use 0 for no, 1 for yes, 2 for unknown  
(integer)  
Is Pollutant Sulfonated? Use 0 for no, 1 for yes, 2 for unknown  
(integer)  
Is Pollutant Animated? Use 0 for no, 1 for yes, 2 for unknown  
(integer)  
Is Pollutant Polyelemental? Use 0 for no, 1 for yes, 2 for  
unknown (integer)

...else if type of pollutant is Particulate...

Is Pollutant Water Soluble? Use 0 for no, 1 for yes (integer)  
Phase at STP: use 0 for solid, 1 for liquid (integer)  
PM Density, in lb/ft<sup>3</sup> (float)  
Note: Now specify pollutant's particle size distribution as  
a list of particle aerodynamic diameters, in um, and  
percent  
mass above that diameter

For each diameter/mass above pair in the particle size  
distribution table...

Particle aerodynamic diameters, in um (float)  
Percent mass above this diameter (float)

Control Device Case Study Title (text)  
Hours of Pollutant Stream Operation Per Year (float)  
Stream Duty cycle: use 0 for Intermittent, 1 for Shift, 2 for Continuous  
(integer)  
Stream Type: use 0 for VOC, 1 for Particulate (integer)  
Desired % Pollutant Reduction (float)  
Stream Temperature, in degrees K (float)  
Stream Pressure, in atm (float)  
Stream Volumetric Flow Rate, in ft<sup>3</sup>/min (float)  
Code that specifies if previous parameter specifies actual flow rate  
or flow rate at STP. Use 0 for actual volumetric flow rate, 1 for  
volumetric flow rate at STP (integer)  
Does stream contain particulate matter? Use 0 for no, 1 for yes  
(integer)  
Note: Only used for VOC streams

Stream Oxygen Content, in % (float)

Note: Only used for VOC streams

Stream Moisture Content, by volume, in % (float)

Note: For the following seven parameters, Use 0 for no, 1 for yes.

These seven parameters are only used for Particulate streams.

Stream has Acid Gases? (integer)

Stream has Alkalis? (integer)

Stream has Fluorides? (integer)

Stream has Mineral Acids? (integer)

Stream is Corrosive? (integer)

Stream has Metal? (integer)

Stream has Condensable Metal? (integer)

Stream Dust Type (integer)

Note: Only used for Particulate streams

Note: Use one of the following codes

- 0 = Unknown
- 1 = activated charcoal
- 2 = alumina
- 3 = aluminum dust
- 4 = ammonium phosphate fertilizer
- 5 = asbestos
- 6 = bauxite
- 7 = carbon black
- 8 = cardboard
- 9 = cellulose material
- 10 = cement
- 11 = ceramic pigment
- 12 = clay
- 13 = coal
- 14 = cocoa
- 15 = chocolate
- 16 = coke
- 17 = cosmetics
- 18 = detergents
- 19 = diatomaceous earth
- 20 = dry petrochemical
- 21 = dyes
- 22 = enamel frit
- 23 = feed
- 24 = feldspar
- 25 = fertilizer
- 26 = fibrous material
- 27 = flour
- 28 = fly ash
- 29 = grain
- 30 = graphite
- 31 = gypsum
- 32 = iron ore
- 33 = iron oxide
- 34 = iron sulfate
- 35 = kaolin
- 36 = lead oxide
- 37 = leather dust

38 = lime  
39 = limestone  
40 = machining operations  
41 = metal powders  
42 = mica  
43 = paint pigments  
44 = paper  
45 = perlite  
46 = plastics  
47 = quartz  
48 = resins  
49 = rock dust  
50 = rubber chemical  
51 = salt  
52 = sand  
53 = sawdust  
54 = silica  
55 = silicates  
56 = slate  
57 = soap  
58 = soda ash  
59 = spices  
60 = starch  
61 = sugar  
62 = supply air  
63 = talc  
64 = tobacco  
65 = wood products

Note: The following 11 parameters are used in estimating the costs of the various pollution control options.

Yearly Interest Rate, in % (float)

Year to bring cost to (1900 to 2100) (integer)

Month or quarter to bring cost to (text)

Note: Must be one of following values

"January"  
"February"  
"March"  
"First Quarter"  
"April"  
"May"  
"June"  
"Second Quarter"  
"July"  
"August"  
"September"  
"Third Quarter"  
"October"  
"November"  
"December"  
"Fourth Quarter"

Operating Labor Cost, in \$/hr (float)

Maintenance Labor Cost, in \$/hr (float)

Electricity Cost, in \$/kilowatt-hr (float)

Cooling Water Cost, in \$/kgallon (float)

Waste Water Disposal Cost, in \$/kgallon (float)

Steam Price, in \$/klb (float)

Dust Disposal Cost, in \$/ton (float)

Natural Gas Price, in \$/ft<sup>3</sup> (float)

Method used to specify pollutant flow rates in stream (integer)

This is the method used to specify pollutant flow rates below.

Use 0 for Mass Concentration at STP, in lb/ft<sup>3</sup>

Use 1 for Actual Mass Concentration, in lb/ft<sup>3</sup>

Use 2 for Mass Flow Rate, in lb/hr

Use 3 for Volumetric Concentration, in ppm

Note: Volumetric Concentration in ppm can only be used for VOC streams

For each pollutant in the pollutant stream:

Pollutant name (text)

Note: The pollutant name must exactly match either 1) a new pollutant specified earlier in this file, or 2) a pollutant already in the ACA chemical library.

Pollutant flow rate (float)

Note: The pollutant flow rate is specified using the method and units specified above.

## **APPENDIX C: ACA TRAINING MATERIALS**

# ACA TRAINING MATERIAL

## PART I - BASIC TRAINING

### Overview of the ACA Software

#### Session #1

- **Install the ACA on your PCs:**

step 1: copy aca.exe to your hard drive

step 2: double click on this self-extracting zip file

step 3: go to the directory that the file was unzipped to

step 4: double click on the exlgui.exe file (or shortcut) to start the ACA. *Note that a shortcut may be created by dragging the exlgui.exe file onto the desktop.*

- **Basics of getting around the ACA data structure**

First, a word about objects...

*"Objects are data structures that typically represent something in the real world."*

---

NOTES

*Objects can contain data elements (slots) as well as other objects. Objects that contain other objects make up a data containment hierarchy."*

**Tip: In the ACA, objects are visually represented by icons.**

- Data/Data structures in the ACA are in one of 3 forms:
  - Objects
  - Lists (of Objects)
  - Text Edit Slots
  
- The overall ACA data structure is based on 3 core objects which appear only in the left pane:
  - Installation Data
  - Library Data
  - What-if Scenarios
  
- In the ACA the data/data structures are presented in a format similar to "files & folders"
  - Objects ~ folders
  - Lists (of objects) ~ folders
  - Text Edit Slots ~ files

---

*NOTES*

*The left pane contains: Objects & Lists (of objects)*

Right pane contains: Objects, Lists (of objects) & Text Edit Slots

- *The term "slot" refers to a piece of data associated with an object. Slots can include other objects, lists (of objects), and Text Edit Slots.*

Lists of objects contain **list elements**.

- In summary, there are 5 different classifications of data
  1. Core objects
  2. Slots that contain an object
  3. Slots that contain a list of objects
  4. Objects that are elements of a list
  5. Text edit slots
- Left clicking on the "+" located to the left of an object will open the **data containment** structure for that object. The data containment structure includes all of the contained data from the three core objects.

Convention when referencing a location of an object, a list, a list element and a Text Edit Slot:

| <i>Library Data</i> | <i>Chemicals Library</i> | <i>Standard ACA Chemicals Library</i> | <i>Acetone</i> | <i>Boiling Point</i> |
|---------------------|--------------------------|---------------------------------------|----------------|----------------------|
| ^                   | ^                        | ^                                     | ^              | ^                    |
| <i>one of three</i> |                          |                                       |                |                      |
| <i>core objects</i> |                          |                                       |                | ----                 |
|                     | <i>slot that contain</i> |                                       |                |                      |

---

*NOTES*

|                  |                             |                     |                  |
|------------------|-----------------------------|---------------------|------------------|
| <i>an object</i> | /                           | /                   | /                |
|                  | <i>slot that contains a</i> | /                   | /                |
|                  | <i>list of objects</i>      | <i>list element</i> | /                |
|                  |                             |                     | <i>slot that</i> |
|                  |                             |                     | <i>is a Text</i> |
|                  |                             |                     | <i>Edit Slot</i> |

- **The ACA toolbar**

- Enabled buttons on the toolbar effect the selected slot (i.e., object, list of objects, text edit slot).
- Disabled toolbar buttons are not applicable to the selected slots.
- Holding the mouse cursor over a toolbar button provides a brief description of that button's function.

- **Entering data ("+" button, drag and drop)**

- Objects can be added using the "+" tool bar button.
- The "+" button will add objects to a list, replace an object or "fill in" an undefined object.
- **Tip:** You can perform these same operations using "drag and drop".
- **Tip:** When creating new objects (e.g., emission units) it is a good practice to give them names (e.g., "Coal-fired boiler near Building 201").

---

*NOTES*

## • The Units Conversion Utility

- The Units Conversion Utility is useful when programming in the ACA, QA of input data, general conversion of units.
- Irony is that the Units Conversion Utility is NEVER needed to enter data into ACA because slots in the ACA are "united" and allows any dimensionally correct set of units to be used.
- When using the ACA, the Units Conversion Utility is useful for:
  - 1) converting raw data into units that you are more familiar with
  - 2) making certain customized extensions

### **EXAMPLE PROBLEM #1: Using the Units Conversion Utility**

**Purpose:** To learn about the ACA Units Conversion Utility, and to learn about which units are available in the ACA.

**Problem:** Select the menu option "Utilities | Units Conversion Utility" and convert the following:

- #1) 52 "kg" = \_\_\_\_\_ "lbs"
- #2) 300 "dollars/cfm" = \_\_\_\_\_ "dollars/(m<sup>3</sup>/s)"
- #3) 1985.77 "mBTU/lbmole-R" = \_\_\_\_\_ "Joules/gmmole-K"  
= \_\_\_\_\_ "liter-atm/gmmole-K"  
= \_\_\_\_\_ "cal/gmmole-K"

*For help with units, see the ACA help menu: Help | Tips on Using Units*

---

*NOTES*

---

*NOTES*

## **EXAMPLE PROBLEM #2: Entering Data**

**Purpose:** To learn the basics of entering data.

**Problem:** Enter the following two pollutants into the "User-Defined Chemicals Library"

**Step #1)** Select the list *Library Data | Chemicals Library | User-Defined Chemicals Library* and then select the [+] toolbar button. You will only have the choice to add a "new" chemical - accept that option. (Repeat this step twice, once for both chemicals).

**Step #2)** Expand the *Library Data | Chemicals Library | User-Defined Chemicals Library* list and select one of the elements that you have just added.

**Step #3)** Enter the following parameters for the first chemical/pollutant

|                                        |                                |
|----------------------------------------|--------------------------------|
| Primary Name                           | Isopropyl Acetate              |
| CAS Number                             | 108-21-4                       |
| Hazardous Air Pollutant (list of 189)  | False                          |
| Emitted as Particulate Matter (PM)     | False                          |
| Phase at STP                           | liquid                         |
| Molecular Weight                       | 102.13 "gm/gmmole"             |
| Heat of Combustion (gas phase)         | 3207.46 "BTU/ft <sup>3</sup> " |
| Boiling Point                          | 361.8 "K"                      |
| Critical Temperature                   | 955.8 "R"                      |
| Antoine Constant A                     | 7.330018                       |
| Antoine Constant B                     | 1437.16                        |
| Antoine Constant C                     | -38.7851                       |
| Liquid Density @STP                    | 0.8711 "gram/ml"               |
| Heat of Condensation                   | 3.6E5 "J/kg"                   |
| Refractive Index                       | 1.3770                         |
| Diffusion Coefficient, liquid in water | 9.63E-6 "cm <sup>2</sup> /sec" |
| Diffusion Coefficient, vapor in air    | 0.081 "cm <sup>2</sup> /sec"   |
| Heat Capacity                          | 1.95 "J/gram-K"                |

---

### **NOTES**

|                             |             |
|-----------------------------|-------------|
| Lower Explosive Limit (LEL) | 17600 "ppm" |
| Vapor Pressure @ 293 K      | 47.5 "mmHg" |
| Organic                     | TRUE        |
| Halogenated                 | FALSE       |

**Step #4)** Enter the following Text Edit Slots for the second chemical/pollutant

|                                                  |                                     |
|--------------------------------------------------|-------------------------------------|
| Primary Name                                     | Sample Particulate Matter (PM) Data |
| Hazardous Air Pollutant (list of 189)?           | False                               |
| Criteria Air Pollutant?                          | True                                |
| Emitted as Particulate Matter (PM)?              | True                                |
| Phase at Standard Pressure and Temperature (STP) | Solid                               |
| Solid Density                                    | 90 "lb/ft <sup>3</sup> "            |

**Step #5)** For the "Particulate Matter (PM) Data" pollutant, expand object so that you see the list titled "PM Size Distribution." Select this list and then add 6 Mass Distribution Data objects to this list. Elements are added to the list in the same manner in which chemicals/pollutants were added to the *Library Data | Chemicals Library | User-Defined Chemicals Library* list in Step #1 above.

**Step #6)** For each of the 6 Mass Distribution Data objects that you added in Step #4b edit the "Aerodynamic Diameter" and the "Percent Mass Above" Text Edit Slots with the following data:

---

*NOTES*

### PM Size Distribution

| <u>aerodynamic diameter</u> | <u>percent mass above</u> |
|-----------------------------|---------------------------|
| 0 "um"                      | 100 "percent"             |
| 0.5 "um"                    | 96 "percent"              |
| 2 "um"                      | 85 "percent"              |
| 5 "um"                      | 60 "percent"              |
| 8 "um"                      | 30 "percent"              |
| 9 "um"                      | 8 "percent"               |

**Step #7)** Save your file, using the menu: **File | Save as** option. Default file extension for ACA files is "OXL". Suggest saving file as "WorkShop\_Problem\_2.oxl".

---

*NOTES*

# Overview of the ACA Data Structure

## Session #2

- **How the ACA data structure is unique**
  - Standard View: what we will cover in the "Basics" course
  - Advanced Views: what will be covered in the "Advanced" course
    - Contains detailed information on objects
    - Can view global declarations
    - Can see all code that describes objects
- **Specific data structures to consider**
  - Chemicals
  - Pollution Concentration Data
  - Operational Data
  - Emission Summary
  - Libraries
  - Installation
  - Size and Cost Air Pollution Control Technologies (a.k.a. Control Data Packets)
- **Links**

*"A 'link' is a reference to an object that resides somewhere else"*

  - Only objects can be linked.

---

*NOTES*

- Objects that are linked are noted with a small arrow (similar to the MS Windows shortcut symbol).
- Lists that contain objects that are linked will also be noted with the shortcut symbol.
- Objects that are typically linked are those that are "Library Data" items, such as chemicals, materials, etc. These objects are not linked in the "Library Data" containment structure, but they are linked elsewhere.

*Note that (currently) you can edit a "linked" object and its modifications will be made to the core instance of the object.*

- **User- Levels**

**Novice:** presents the most critical "input" data and most significant "output" data

**Intermediate:** presents more specific input and output data

**Expert:** presents remaining data - typically only need to view in this mode if programming or verifying algorithms.

**EXAMPLE PROBLEM #3:**

**Purpose:** Become comfortable with the ACA data structure and entering slightly more complex data than in the previous example.

**Problem Statement:** Define a un-permitted printing operation that operates 16 hrs/day, 5 days/week. Assume the source has been operating this way since January 15, 1990 and will continue to operate this way until the end of the year 2005.

---

*NOTES*

**Step #1)** Open the file you started in Problem #2 (e.g., WorkShop\_Problem\_2.oxl), using the menu: **File | Open...** option.

**Step #2)** Add a emission unit to the **Installation Data | Sources at Installation** list. To do this, select the **Sources at Installation** list and then select the [+] toolbar button. Pick the **Graphic Arts/Printing** icon which is found at...

Emission Units  
Organic Evaporative Losses  
Surface Coating/Printing/Graphic Arts  
**Graphic Arts/Printing**

**Step #3)** Give the emission unit a name of your choice.

**Step #4)** Expand the emission unit and then add an element (object) to the "Actual Operational Data" list (select the list and then select the [+] toolbar button).

**Step #5)** Expand the "Actual Operational Data" list.

**Step #6)** Next, select the new "Source Operational Data" element (object) that you just created.

**Step #7)** Enter a name for the "Name of Operational Data Set" in the right window (e.g., "Standard operating mode" - just for book keeping purposes)

**Step #8)** Expand the "Source Operational Data" object.

**Step #9)** Select the Operating Schedule icon in the left window.

**Step #10)** Enter data for the operating schedule, including the following slots:

---

*NOTES*

|                                    |                                    |
|------------------------------------|------------------------------------|
| Schedule Name                      | 'Two shifts per week day schedule' |
| Schedule Start Date                | January 15, 1990                   |
| Schedule End Date                  | December 31, 2005                  |
| Average Operational Time per Day * | 16 "hrs"                           |
| Days Operating per Week *          | 5                                  |

**\*Note** that by going to the "Intermediate" User Level you are given the opportunity to enter daily operating schedule. In this case, the "Average Operational Time per Day" and the "Days Operating per Week" are calculated by the ACA.

**Step #11)** Run the menu: **Utilities | Check Schedules for Consistency** option to ensure that your operating schedule(s) are consistent.

**Step #12)** Save your file, using the menu: **File | Save as** option. Default file extension for ACA files is "OXL". Suggest saving file as "WorkShop\_Problem\_3.oxl".

### ***EXTRA THINGS TO TRY...***

**Optional problem #3.a)** Switch to the intermediate level mode and enter daily schedule information (e.g., Monday - Friday 6:00a.m. to 4:00p.m. and 5:15p.m. to 11:15 p.m.).

**Optional problem #3.b)** Define the schedule for the "Potential Operational Data" list (i.e., continuous operation).

---

### ***NOTES***

**Optional problem #3.c)** Add a second Operational Data set to the "Actual Operational Data" list. Enter similar data as before (Step #9 above), except assume that starting January, 1 2001, operations will increase by 4 hour per day (total of 20 hrs/day). Note this will require that the schedule for the first Operational Data Set be set to only be valid until December 31, 2000, so make sure to modify as needed.

---

*NOTES*

# Overview of the Built-in Analyses in the ACA

## Session #3

- **Minimum data requirements:**

- The "?" symbol marks data (i.e., an object or a Text Edit Slot ) that has not been defined
- Undefined variables that are critical to analyses are reported
- Some analyses can use be made with minimal defined variables

"EvaluateIfPossible"

```
LS1 = True;  
LS2 = False;  
LS3 = ?;  
EvaluateIfPossible(LS1 or LS2 or LS3) ---> True  
EvaluateIfPossible(LS1 and LS2 and LS3) ---> False
```

"IsDefined"

```
IsDefined (LS1) => true  
IsDefined (LS2) => true  
IsDefined (LS3) => false
```

---

*NOTES*

- **Mature analyses:**

- Estimate Emission Rates

- 36 emission units in ACA have associated algorithms

Aboveground Tanks  
Abrasive blasting #  
Asphalt cutbacks \*  
Carpentry operations #  
Clean up Materials  
Coal fired boilers  
Degreaser operations \*  
Electroplating #  
External Floating Roof Tanks  
Fuel Dispensing Facility  
Gas Turbine  
Gas fired boilers  
Graphic Arts / Printing  
Industrial Engine  
Internal Floating Roof Tanks  
Jet Engine Test Stands  
Loading Racks  
Medical waste incinerators #  
Multi-chamber incinerators \*  
Natural Gas Prime Mover  
Oil fired boilers  
Petroleum Spill  
Single-chamber incinerators \*  
Solvent Spill  
Spray Cleaner  
Surface Coating  
Tactical Haulers  
Tank and Drum Cleaning  
Thermal Metal Sprayers  
Underground Tanks  
Waste Solvent Recovery Operation

Waste oil fired boilers  
Wastewater treatment units  
Water Solvent Recovery Operation  
Welding operations #  
Wood-fired boilers

*# Not fully tested*

*\* Makes calculations, but not via the "easily viewable" slot format.*

---

*NOTES*

- AP42, mass balance, AQUIS, Water 8 documentation, etc...
- numerous build-in helper functions to assist in adding additional emission units
- user can customize/modify algorithms
- Summarize Emissions (2 options)
  - Directly out of Emission Units
    - summary of the "Raw Emission Streams"
    - "Raw Emission Stream" = results of "Estimate Emission Rates" or user defined, what comes out of the emission units as described (includes controls considered in estimating emissions)
  - From Stack (to Atmosphere)
    - summary of "Stack Emission Streams"
    - Results will equal the "Directly out of Emission Units" summary, unless user has modified a "Stack Emission Streams". The "Stack Emission Streams" is a copy of the "Raw Emission Stream" by default, but can be modified by the user. Ability to modify the "Stack Emission Streams" allows the user (among other things) to define an add-on control technology not considered in estimating emissions
- Apply Control Technologies
  - Cost level estimates for control of particulate matter (PM) and volatile organic compounds (VOC).

---

*NOTES*

- The user supplies data related to:
  - the air pollution stream to control,
  - the annual hours of operation, and
  - the desired reduction efficiency.
- Models are based primarily upon guidance provided in :
  - "OAQPS Control Cost Manual," US EPA OAQPS,
  - "Estimating Costs of Air Pollution Control," William M. Vataavuk,
  - "Air Pollution Control Equipment - Volumes I & II," Louis Theodore and Anthony J. Buonicore's , and
  - "Air Pollution Engineering Manual," Air & Waste Management Association.

•VOC Control Devices:

1. Thermal Incinerators
2. Flares
3. Carbon Adsorbers
4. Gas Absorbers
5. Refrigerated Condensers

•Particulate Matter (PM) Control Devices:

1. Baghouses
2. Electrostatic precipitators (ESP)
3. Cyclones
4. Mist Eliminators
5. Wet Scrubbers

---

*NOTES*

- **Prototype analyses:**

- Applicability matching engine

- Checks for Applicable Regulations:

The ACA compares the emission units that the user has defined with the regulations in the ACA Regulations Database.

- Checks for Applicable Pollution Prevention Opportunities:

The ACA reports any P2 Opportunities that could not be ruled out for the emission units specified.

- Checks for Applicable Suggestions:

Suggestions that could not be ruled out for the emission units specified are reported by the ACA.

---

*NOTES*

- Why the applicability analyses are "prototype"..
  - rules out non-applicable, leave "potentially" applicable
  - recent additions to ACA allow for significantly more powerful tool (customize functions)
- Strengths of ACA approach
  - ACA data structure
  - built-in matching utilities
  - existing databases as a baseline for additions

*EPA input/involvement is key to improve the regulations applicability analysis*

- **Two more useful "Utilities":**

**Check Schedules for Consistency:** make sure operating schedules within an operational data set and between operational data sets do not overlap or are inconsistent (e.g., more than 24 hours in one day, more than two modes of operation at the same time).

**Set Object Names:** will assign names to pollutant streams & operational data sets if the user has not done so.

---

*NOTES*

### **EXAMPLE PROBLEM #4 Running Analyses with pre-defined data**

**Purpose:** to become familiar with running the analyses, using pre-entered data.

**Problem Statement:** Data has been entered into the ACA input file "inputs/WorkShop\_problem\_4\_in.oxl". It contains a surface coating operation that operates 8 hrs/day (actual), 24 hrs/day (potential).

**Step #1)** load the file "inputs/WorkShop\_problem4\_in.oxl"

**Step #2)** run "Utilities | Check Schedules for Consistency"

**Step #3)** run "Utilities | Set Object Names"

**Step #4)** run "Analysis | Estimate Emission Rates"

**Step #5)** run "Analysis | Summarize Emissions > Directly out of Emission Units "

**Step #6)** run "Analysis | Apply Control Technologies"

**Step #7)** "fileload! ACA Federal Regulations Database"

**Step #8)** "fileload! ACA Pollution Prevention Opportunities Database"

**Step #9)** "fileload! ACA Suggestions Database"

**Step #10)** run "Analysis | Prototype Analyses | Check for Applicable Regulations"

**Step #11)** run "Analysis | Prototype Analyses | Check for Applicable Pollution Prevention Opportunities "

**Step #12)** run "Analysis | Prototype Analyses | Check for Applicable Suggestions"

---

### **NOTES**

**Step #13)** save the data to the file "inputs/WorkShop\_problem4.xml"

**Optional Problem #4.a)** Search through the ACA data structure to answer the following questions:

- What is the method of paint application (actual) in 1997?
- What changes in the daily operating schedule happens in 1998 on Fridays? (Hint: will need to be in intermediate mode)
- What other modifications to (actual) operations happen in 1998? (Hint: look at material usage data, materials used, application method, etc.)
- Can you think of any reasons that the potential operational data differs from the actual?

**Optional Problem #4.b)** Create a new potential operational data set that assumes a limitation on the hours of operation to be 40 hours/week. How does this affect the facility-wide potential emissions?

---

*NOTES*

# Sensitivity Analyses

## Session #4

- **Spreadsheet-like functionality:**

Numerous calculations in the ACA are made "on the fly" (similar to a spreadsheet that automatically re-calculates as data is entered/modified).

Whenever the value of a slot is changed, whether to a particular value or to a function that is dependent on other parameters, all of the parameters which depend on that slot will be updated automatically (just like in a spreadsheet, when a particular cell is updated, all of the cell values which depend on the value of that cell will be updated automatically as well).

**A step beyond standard "spreadsheet-like" functionality with ACA...**

Slots are defined as being "User" defined, defined by a "Calculation" or "Externally" defined. This gives user an idea which slots are intended to be entered/modified.

Calculation slots can be over-ridden with a user-supplied value or equation - this modification will be visually marked on ACA's user interface. Can revert back to original slot definition.

---

*NOTES*

- **The (I) toolbar option:** 

Detailed information about a slot, including its variable name, can be found by clicking on the (I) button from the main menu bar.

- **The "Fx" feature:** 

The  button shows only the value of the parameter, and the  button shows only the functions on which the property is based. The user can view both the value and the functions at the same time by clicking on the  button from the main menu bar.

This toolbar button shows slot functions. When the user wants to view the definition of a slot, or change the definition of a slot, the "Fx" button shows only the functions on which the slot is based.

- **A word about the three core objects**

The three core objects in the ACA are the reference point for identifying slots used in other calculations

Library Data = library.

Installation Data == installation.

What If Scenarios == worksheet.

- GUI:

*Library Data | Chemicals Library | Standard ACA Chemicals Library*

- Source Code or Fx view:

*Library.ChemicalLibrary.aca\_chemicallist*

These slot names and locations can be determined with a combination of looking at slot definitions in the  and using the  tool bar option.

---

*NOTES*

Any "identifier" that you see in the code will be either:

a **slot**, procedure, or function **that belongs to the same object**

a **slot**, procedure, or function **that belongs to one of the 3 core objects**

a global variable

a global constant

a global type

a global procedure

a global function

*Only those identifiers that are bolded and underlined will be considered in the Basic Course - the Advanced Course will consider the others.*

### **EXAMPLE PROBLEM #5 Sensitivity Analysis - basic end-user extendibility**

**Purpose:** to become familiar with the basics of "end-user" extendibility.

**Problem Statement:** The "Regenerative Thermal Incinerator (RTO)" from Example Problem #4 will be fine-tuned to illustrate basic end-user extendibility. Simply follow the steps outlined below:

**Step #1)** load the file "inputs/WorkShop\_problem4.oxl"

**Step #2)** Expand the core "What-If Scenario" core object.

**Step #3)** Expand the "Size & Cost Air Pollution Control Technologies" and select the list titled "Potential Control Device Options".

**Step #4)** Select the RTO (i.e., "Regenerative Thermal Incinerator" element/object) from the "Potential Control Device Options" list and review the slots in the right pane associated with this object.

---

### **NOTES**

**Step #5)** Locate the following slots of the RTO, and pay attention to their value as we make various modifications:

Year to Bring Cost Estimates to  
Month to Bring Cost Estimates to  
Pressure Drop  
Hours of maintenance labor per shift\*  
Annual Maintenance Labor Costs\*  
Total Capital Investment  
Total Annualized Costs

*\*You will need to be in the Intermediate User Level to see these slots.*

***Changing a User-Defined slot:***

Assume that you have some estimates from a vendor of RTOs from May of 1995 and you want to compare the estimates that they provide with the ACA.

**Step #6)** Change the values of the following two slots:

Year to Bring Cost Estimates to      1995  
Month to Bring Cost Estimates to      May

*Note how the icon associated with these two slots has changed as well as the calculated values of the Total Capital Investment and the Annualized Costs.*

---

**NOTES**

***Overriding a Calculation slot with a user defined value:***

Further assume that the information from the vendor guarantees a pressure drop of no more than 0.5 "psi".

**Step #7)** Change the values of the following slot:

Pressure Drop            0.5 "psi"

*Note how the icon associated with this slot has changed as well as the calculated values of the Annualized Costs.*

***Changing slots that belongs to a core object:***

You know that the maintenance crew that your facility uses are very inefficient, but since they bid was the lowest you are stuck with them. You want to make sure that the costs with the ACA account for this, so you decide to investigate how the Maintenance Labor Cost slot is calculated.

**Step #8)** Click on the  button from the main menu bar. Notice how there are now 2 windows for each slot: one contains the data and the other contains the function that describes that slot.

**Step #9)** Select the "Annual Maintenance Labor Costs" slot of the RTO and then click on the  button from the toolbar. Note that the slot is given as a function of the following variables:

**Installation.maintenance\_labor\_cost**  
**maintenance\_labor\_per\_shift**  
shifts\_per\_year  
systemOperatingHours

---

***NOTES***

We will look at the first two (bolded) variables. Note that the first variable starts off with the tag "Installation" which indicates it belongs to the "Installation Data" object.

**Step #10)** Modify the **Installation.maintenance\_labor\_cost** slot. Select the "Installation Data" object and notice all of the default data that can be customized for your facility, including "Maintenance Labor Cost". Change the following slots that belong to the "Installation Data" object:

Maintenance Labor Cost      14.0 "DOLLAR/hr"

**Step #11)** Switch back to your RTO and notice how values have changed.

***Replacing a calculation slots with a user-defined calculation***

Changing the hourly cost of the maintenance labor did not account for the fact that they are not efficient works. In order to account for that, we will change the function for the "Hours of maintenance labor per shift" slot (variable name: **maintenance\_labor\_per\_shift**). Assume that you have observed that your maintenance workers are twice as slow as other/typical maintenance workers.

**Step #12)** Place your cursor into the second cell associated with the "Hours of maintenance labor per shift" slot (this second cell is where the function is defined). Then, replace the existing contents of the cell, namely:

```
return thermal_maintenance_labor;
```

with:

```
return 2.0 * thermal_maintenance_labor;
```

*Note how the icon associated with this slot has changed as well as the calculated values of the slot.*

---

***NOTES***

# Overview of the Built-in ACA Libraries

## Session #5

- **ACA Library databases**
  - Chemicals Library
    - Contains 189 HAPs
    - Criteria pollutants
    - Sample particulate matter (PM) data
  - Materials Library
    - a few sample paints with default properties from AP-42
    - sample waste water "material"
  - Control Devices Library
    - 8 unique types of control devices for VOCs and PM control
    - 30 different entries in library to account for differences within a category (e.g., pulse-jet vs. reverse air baghouses):
  - Federal Regulations Library
    - NESHAP
    - NSPS
    - MACT Standards
    - Other Federal Regulations

---

*NOTES*

- Facility Wide Federal Regulations (NSR)
- Pollution Prevention Libraries
  - Data from the Tri-Services P2 database
- Suggestions Libraries
  - 10 suggestions related to emissions control options (NO<sub>x</sub>, VOC, SO<sub>x</sub>)
  - a potential means by which different users can share knowledge
- Miscellaneous Objects Library
  - Currently has two operating schedules stored there
  - Can store anything here, except a copy of the core Library Data object since the Miscellaneous Objects Library is contained in the Library data
  - A good place to "store" items you may want to possibly reuse later
- Materials used for Fabric Filtration
  - 34 different materials/bag configuration choices
  - cost data
  - applicability data

---

*NOTES*

- **Adding data to the libraries**

The user can add/delete to the "User-defined" library lists for

- Chemicals Library | User-Defined Chemicals Library
- Materials Library | User-Defined Materials Library
- Control Devices Library | User-Defined Control Devices Library
- Federal Regulations Library | User-Defined Regulations (emission unit-type)
- Federal Regulations Library | User-Defined Regulations (facility-wide type)
- Pollution Prevention Libraries | User-Defined Pollution Prevention Opportunities
- Suggestions Libraries | User-Defined Suggestions Library
- Miscellaneous Objects Library
- Materials used for Fabric Filtration

The user can also edit:

- *Chemicals Library | Standard ACA Chemicals Library*

The user can only modify objects (can't add or delete) in the following "ACA Standard" library lists

- Materials Library | Standard ACA Materials Library
- Control Devices Library | Standard ACA Control Devices Library
- Federal Regulations Library | Federal NESHAP Regulations Library
- Federal Regulations Library | Federal NSPS Regulations Library
- Federal Regulations Library | Other Federal Regulations Library
- Federal Regulations Library | Facility-Wide Federal Regulations Library
- Pollution Prevention Libraries | Standard ACA Pollution Prevention Opportunities
- Suggestions Libraries | Standard ACA Suggestions Library

---

*NOTES*

**EXAMPLE PROBLEM #6: Library data entry problem**

**Purpose:** to become familiar with the ACA library databases.

**Problem Statement:** Create a "User-Defined" ink (material) with the following properties:

density: 9.2 "lbs/gallon"

phase: liquid

VOC contents

toluene: 15 "percent" (by weight)

isopropyl acetate 10 "percent" (by weight)

**Step #1)** load the file "inputs/WorkShop\_problem3.oxl"

**Step #2)** Expand the core "Library Data | Materials Library" object, and select the "User-Defined Materials Library" list

**Step #3)** Add a new material to the list with the "+" button.

**Step #4)** Enter in the values for the following parameters:

Name (call it "Ink #7")

Phase (given above)

Density (given above)

**Step #4)** Select the "Composition" list and add two elements ("chemicals with associated percents" objects).

**Step #5)** For each of the "chemicals with associated percents" objects that you have created, select the undefined "?" chemical icon and use the "+" button to link the appropriate chemical and enter the corresponding "Percent (by weight)" value.

---

*NOTES*

**Step #6)** Add a suggestion to the Suggestions Libraries | User-Defined Suggestions Library that would "warn" that the use of materials with "isopropyl acetate" with graphic art emission units should consider contacting the ACME company to learn about their new "Super Ink" that is cheaper and more efficient than inks that use "isopropyl acetate".

**Step #7)** Save your case as "Workshop\_Problem\_6.oxl".

---

*NOTES*

# Overview of the Estimating Emissions with the ACA

## Session #6

- **ACA Estimates emission rates for each operational data set**

- **Estimate Emission Rate Analysis available for...**

|                              |                                  |
|------------------------------|----------------------------------|
| Aboveground Tanks            | Multi-chamber incinerators *     |
| Abrasive blasting #          | Natural Gas Prime Mover          |
| Asphalt cutbacks *           | Oil fired boilers                |
| Carpentry operations #       | Petroleum Spill                  |
| Clean up Materials           | Single-chamber incinerators *    |
| Coal fired boilers           | Solvent Spill                    |
| Degreaser operations *       | Spray Cleaner                    |
| Electroplating #             | Surface Coating                  |
| External Floating Roof Tanks | Tactical Haulers                 |
| Fuel Dispensing Facility     | Tank and Drum Cleaning           |
| Gas Turbine                  | Thermal Metal Sprayers           |
| Gas fired boilers            | Underground Tanks                |
| Graphic Arts / Printing      | Waste Solvent Recovery Operation |
| Industrial Engine            | Waste oil fired boilers          |
| Internal Floating Roof Tanks | Wastewater treatment units       |
| Jet Engine Test Stands       | Water Solvent Recovery Operation |
| Loading Racks                | Welding operations #             |
| Medical waste incinerators # | Wood-fired boilers               |

*# Not fully tested*

*\* Makes calculations, but not via the "easily viewable" slot format.*

- AP42, mass balance, AQUIS, Water 8 documentation, etc...

---

*NOTES*

- **Data requirements**

Since the data requirements for estimating emission rates are dependent on the type of emission unit considered (e.g., gas-fired boiler vs. wastewater treatment plant), the exact procedure for entering data into the ACA will be a function of the emission unit.

There are a few steps that are common to all emission units:

1. Define an emission unit. There are typically a few slots at the emission unit level that need to be set - these slots all start after the slot-divider line titled: "EMISSION UNIT SPECIFIC SLOTS RELATED TO ESTIMATING EMISSION RATES".

2. Add the number of actual and potential operational data sets required to define the operation of the emission unit. The operational data sets for some of the emission units is specialized and may have slots that are required to be specified in order to estimate emissions.

3. Define the material usage information for each operational data set associated with the emission unit. Typically most of the slots that are located at the material usage level are required in order to estimate emissions.

4. When in doubt, attempt to estimate the emissions and if you have not provided enough information, the ACA will let you know which parameters are still required.

- **Viewing the algorithms**

To view the calculations for estimating emissions from the majority of emission units listed above, you can do the following:

1. Create an instance of the emission unit of interest
2. Switch to the intermediate user level
3. View the slots associated with the emission unit, while in the  viewing mode

---

*NOTES*

4. Locate the slot titled: **"Estimate Emission Rates Calculation Slot"**

- This slot contains the calculations required to estimate the emissions for **each operational data set** associated with the emission unit.
- This slot can be edited and therefore it is a means by which the end-user can "extend" the ACA. Note that this slot is a "Calculation" slot, so if you edit it, it will be marked as such.

- **Viewing the results**

A report will be generated when the "Estimate Emission Rates" Analysis is executed. This report will contain the emissions for each operational data in each emission unit for which you are interested in estimating emissions. Any errors, warnings or notes will be also presented in the report.

Once the analysis has successfully been performed for a specific operational data set, the "Raw Stream" associated with that operational data set will have the appropriate pollution concentration data sets (with the appropriate chemicals/pollutants and their emission rates)Each operational data set.

Any errors, warnings or notes that were generated for each operational data set (during the analysis) will be recorded at in the operational data set in a slot called **"Errors, Warnings, and Notes Report Generated When Estimate Emission Rates was Last Run "**. This slot can be seen at the Intermediate User Level.

---

*NOTES*

**EXAMPLE PROBLEM #7: Estimate emissions**

**Purpose:** to become familiar with estimating emission rates with the ACA.

**Problem Statement:** Estimate emission rates for a coal-fired boiler that is a 20 "Mwatt" cyclone furnace, with fly ash reinjection and no pollution control equipment currently installed. Further, the boiler consumes 3.2 "tons/hr" of bituminous coal that contains 0.3 "percent" sulfur and 6 "percent" ash.

**Optional Problem 7.a)** Enter the data for Example #3 in the ACA User's Guide

**Optional Problem 7.b)** Enter the data for Example #4 in the ACA User's Guide

---

*NOTES*

# Overview of the Evaluating Control Technologies with the ACA

## Session #7

- **Overview of controlling VOC and PM**

- VOC Control Device (see ACA Control Devices Library)

1. Thermal Incinerators
2. Flares
3. Carbon Adsorbers
4. Gas Absorbers
5. Refrigerated Condensers

- Particulate Matter (PM) Control Device (see ACA Control Devices Library)

6. Baghouses
7. Electrostatic precipitators (ESP)
8. Cyclones
9. Mist Eliminators
10. Wet Scrubbers

- **Chemical/Pollutant data requirements**

- VOC Control technologies

Molecular Weight

(all VOC controls)

---

*NOTES*

|                                           |                                                  |
|-------------------------------------------|--------------------------------------------------|
| Heat of Combustion (gas phase)            | (thermal incinerators, flares)                   |
| Boiling Point*                            | (refrigerated condensers, carbon adsorbers)      |
| Critical Temperature                      | (refrigerated condensers)                        |
| Antoine Constant A, B, and C              | (refrigerated condensers)                        |
| Liquid Density @STP                       | (carbon adsorbers)                               |
| Heat of Condensation*                     | (refrigerated condensers, carbon adsorbers)      |
| Refractive Index*                         | (carbon adsorbers)                               |
| Henry's Law Constant or equilibrium curve | (absorbers)                                      |
| Diffusion Coefficient, liquid in water    | (absorbers)                                      |
| Diffusion Coefficient, vapor in air       | (absorbers)                                      |
| Heat Capacity                             | (refrigerated condensers)                        |
| Lower Explosive Limit (LEL)               | (thermal incinerators, flares, carbon adsorbers) |
| Halogenated                               | (thermal incinerators)                           |

*\*Alternate method for carbon adsorbers does NOT require these chemical parameters. Instead, coefficients from Prof. Carl Yaw's can be entered.*

#### PM Control Technologies

PM Size Distribution (all but baghouses)

**User Input:** aerodynamic diameter vs. % mass (min 5 points)

ACA assumes a log-normal distribution

ACA calculates: mass median diameter, cut diameter, critical particle size, etc...

Solid density (cyclones)

Resistivity (ESPs - missing from EPA and ACA methodologies)

---

#### NOTES

- **Modifying control technology models in the ACA**

Range for what types of changes can be made:

User defined control technology that is stored in the User-Defined Control Device Library (basically a copy of standard ACA control device with set user-supplied slots)

User defined control technology that is stored in the User-Defined Control Device Library (basically a copy of standard ACA control device with set over-rides of functions)

Brand new control technology, either a customized version of a standard control device (additional slots and different algorithms - maybe more detailed) or an entirely new type of device with a complete set of unique algorithms.

- **General steps required to use the analysis of Size and Cost Air Pollution Control Technologies**

**Step #1)** Open up the "What-If Scenario | Size & Cost Air Pollution Control Technologies" object. The "What-If Scenario" object is one of the 3 root-level objects in the ACA's Standard View.

**Step #2)** Enter data for the following text edit slots for the "Size & Cost Air Pollution Control Technologies" object:

Study Title           *(optional, but good for bookkeeping purposes)*

Yearly Hours of Operation

Duty Cycle

---

*NOTES*

**Step #3)** Open up the "Size & Cost Air Pollution Control Technologies | Stream to Control" object and enter the all data for:

Temperature

Pressure

'Volumetric Flow Rate Actual' **OR** 'Volumetric Flow Rate At STP' (**but not both**) .

*Note, it is also a good idea to enter data for all of the other text edit slots at this level for which you have available data. Depending on the control technology under consideration, some or all of these other slots may be required (you will be notified if additional "undefined" slot require data when you attempt to run the analysis).*

**Step #4)** Next, you will need to add chemicals/pollutants and their associated concentrations (these are called "Pollution Concentration Data" objects in the ACA) to your "Stream to Control" object for each chemical/pollutant that you want to consider in your stream. This is done by selecting the "Pollution Concentrations" list and selecting the "+" button on the tool bar.

**Step #5)** Describe the "Pollution Concentration Data" set. To do this you will need to:

(a) associate it with a chemical/pollutant. Select the undefined object "Chemical/Pollutant" and then select the "+" button on the tool bar. This will allow you pick one of the chemicals that are stored in the ACA's chemical database (i.e., "Library Data | Chemicals Library").

---

*NOTES*

(b) enter one (and only one) measure of the pollutant's concentration \*, namely one of the following slots at the "Pollution Concentration Data" level:

Volumetric Concentration,  
Mass Concentration At STP,  
Mass Concentration, Actual, or  
Mass Flow Rate

*\*Note: if you enter more than one measure of pollutant concentration you risk the possibility of these "co-dependent" variables becoming inconsistent. The ACA will by default calculate the 3 measures of concentration from the one entered (provided that the pollutant streams temperature, pressure and volumetric flow rate are known as well as the molecular weight of the chemical/pollutant). To see the ACA calculated versions of all of the measures of concentration, go to the "Intermediate" User Level and you can view disabled (grayed out) versions of these slots with all of the measures of concentration that will be used in any further calculations.*

REPEAT THIS STEP (5) FOR EACH POLLUTION CONCENTRATION DATA OBJECT THAT YOU ADDED IN STEP 4.

**Step #6)** Open up the "Size & Cost Air Pollution Control Technologies | Emission Reduction Needed" object and select the percentage of VOC or PM control that you desire.

**Step #7)** Select "Apply Control Technologies" from the "Analysis" Menu.

---

### NOTES

- **The results of this analysis**

The results from the Apply Control Technologies are two-fold:

First, a report will be generated that summarizes: (a) any missing data that is required for any of the potentially applicable control devices; (b) which control devices are potentially applicable; (c) the main reasons why any control devices were determined to be non-applicable; (d) cost and sizing data for each applicable control device. The menu selection "Options : Report Level" can be selected to be "Low", "Medium" or "High" in order to control the amount of detail in the output report.

Secondly, for each control technology that is found to be potentially applicable, an instance (object) of that control device will be placed in the "Size & Cost Air Pollution Control Technologies | Potential Control Device Options" list. The user is encouraged to review these instances (as one would a spreadsheet) and modify any user-defined slots to see how minor modifications affect the size and cost of each control technology. If you would like to get a summary report for the control devices in the "Size & Cost Air Pollution Control Technologies | Potential Control Device Options" list at any time other than when the analysis option is run, then select "Run User-Specified Analysis/Action" from the "Analysis" Menu and then type in "WriteOutControlDataInWhatIf" at the data entry window.

---

*NOTES*

# VOC Control Technology Example Problem

## Session #8

### **EXAMPLE PROBLEM #8: VOC control technology problem**

**Purpose:** to become familiar with the estimating costs for VOC control technologies with the ACA.

**Problem Statement:** Determine the cost to control 98 % of the emissions from a process that operates 8500 hours/year and emits 100 "lbs/hour" of benzene. The pollutant stream from the emission units has the following properties:

|                                           |               |
|-------------------------------------------|---------------|
| Volumetric Flow Rate @ actual conditions: | 20,000 cfm    |
| Temperature:                              | 75 "F"        |
| Percent Moisture Content                  | 1.2 "percent" |

**Step #1)** using the steps outlined in the previous session "**General steps required to use the analysis of Size and Cost Air Pollution Control Technologies,**" determine which control technologies are potentially applicable and which has the lowest capital cost and which has the lowest annualized cost.

---

*NOTES*

**Step #2)** Enter in the following cost data (located at the core "Installation Data" object) to over-ride the defaults in the ACA.

|                         |                                  |
|-------------------------|----------------------------------|
| Operating Labor Cost:   | 17.00 "dollars/hr"               |
| Maintenance Labor Cost: | 18.00 "dollars/hr"               |
| Electricity cost:       | 0.05 "dollars/kilowatt-hr"       |
| Natural Gas Cost:       | 0.004 "dollars/ft <sup>3</sup> " |

How does this change the relative cost of the various control technologies?

**Optional #8.a)** assume you call a vendor and they have a 3 bed carbon adsorber available, but they can guarantee a pressure drop of 6 "in-water". How does this change the annual operating costs?

**Optional #8.b)** How much would it have cost you to purchase a Regenerative Thermal Incinerator, with the same specifications as given this problem, in May of 1992?

**Optional #8.c)** Enter the data in Example problem #1 of the ACA User Guide.

---

*NOTES*

# PM Control Technology Example Problem

## Session #9

### **EXAMPLE PROBLEM #9: PM control technology problem**

**Purpose:** to become familiar with estimating costs for PM control technologies with the ACA.

**Problem Statement :** Determine the cost to control 97 % of the emissions from a process that operates 6000 hours/year and emits 4.2 "grains/hour" of particulate matter (PM). Use the PM distribution defined in Example Problem #2 of this workshop (i.e., you can take a shortcut by loading the file WorkShop\_Problem\_2.oxl). The pollutant stream from the emission units has the following properties:

|                             |               |
|-----------------------------|---------------|
| Volumetric Flow Rate @ STP: | 15,000 cfm    |
| Temperature:                | 400 "F"       |
| Percent Moisture Content    | 0.1 "percent" |

**Step #1)** using the steps outlined in the previous session "**General steps required to use the analysis of Size and Cost Air Pollution Control Technologies,**" determine which control technologies are potentially applicable and which has the lowest capital cost and which has the lowest annualized cost.

**Step #2)** Enter in the following cost data (located at the core "Installation Data" object) to over-ride the defaults in the ACA.

---

*NOTES*

|                             |                            |
|-----------------------------|----------------------------|
| Operating Labor Cost:       | 17.00 "dollars/hr"         |
| Maintenance Labor Cost:     | 18.00 "dollars/hr"         |
| Electricity cost:           | 0.05 "dollars/kilowatt-hr" |
| Default Dust Disposal Cost: | 20 "dollars/ton"           |

How does this change the relative cost of the various control technologies?

**Optional #8.a)** Enter data the data in Example problem #2 of the ACA User Guide.

---

*NOTES*

# Comprehensive Example Problem

## Session #10

### **EXAMPLE PROBLEM #10: Comprehensive problem**

**Purpose:** to consider many of the ACA features together to solve a more complicated problem.

**Problem Statement :** Determine the actual and potential emissions for an un-permitted printing operation that operates 2 shifts per day, 6 days/week, all year long (except for the month of June when the plant shuts down for maintenance). Assume that the printing operation is a Rotogravure printing operation with a hot air dryer. Records indicate that, on average, 350 "lbs/hr" of ink #7 (from Example #6) is used.

What are the annual actual emissions?

What are the potential emissions?

What percentage of the federal regulations does the ACA rule out?

Check for any applicable suggestions or pollution prevention opportunities.

Suppose you need to reduce your emissions from this facility by 97%. Which control options are applicable to this emission unit? Which one would you pick?

What are the emissions after control? Does this effect the potential emissions?

---

*NOTES*



- the **ToFloat** operator can be used to force a variable with associated units to a plain floating point value

### Examples

```
var Cross_Sectional_Area: float "inch^2";
Cross_Sectional_Area:= 200.0 "mm^2";
var Tank_height: float "meter";
Tank_height:= 100 "ft";

var volume: float "ft^3";
volume:= Cross_Sectional_Area * Tank_height; // OK
volume:= 500.0 "dollars" * 5 "gallon/dollar"; // OK
volume:= Cross_Sectional_Area / Tank_height; // Compile-time ERROR
volume:= Cross_Sectional_Area * 10.0 "kg"; // Compile-time ERROR
volume:= (Cross_Sectional_Area * Tank_height)^1.3; // Compile-time ERROR
volume:= (ToFloat(Cross_Sectional_Area * Tank_height, "m^3")^0.9) "m^3"; // OK
```

- **Basic programming operators in EXLGUI:**

See EXLGUI Programmer's Manual - we will go through the manual and comment as we go

---

*NOTES*

## **ADVANCED EXAMPLE PROBLEM #1: Programming with units**

**Purpose:** To learn the basics of programming with units.

**Problem:**

Starting with the following snippet of code....

```
var TankCost: float "dollars";
var Tank_Volume: float "meter^3";
var TankCost_per_Volume: float "dollars/m^3";
TankCost_per_Volume:= 50.0 "dollars/ft^3";
Tank_Volume:= 20 "m^3";
```

which of the following will compile?

```
TankCost:= TankCost_per_Volume / Tank_Volume;
```

```
TankCost:= TankCost_per_Volume * Tank_Volume;
```

```
TankCost:= ToFloat(TankCost_per_Volume, "dollar/m^3") *
ToFloat(Tank_Volume, "m^3");
```

```
TankCost:= TankCost_per_Volume * 8.0 "m^3/kg" * 10 "lb/hr" * 8 "hr";
```

```
TankCost:= ToFloat(TankCost_per_Volume, "dollar/ft^3") *
ToFloat(Tank_Volume, "ft^3") "dollar";
```

what will the resultant value be of those that will compile?

*Hint, the units conversion utility may be helpful if you are not sure of the equivalent dimensions of a variable.*

---

**NOTES**

# Working with Global Declarations

## Session #12

- **Overview of advanced end-user extensibility:**
  - The main ACA source code is encrypted in the file "ACA\_Version\_6\_0\_Source\_code\_1.ex1" and "ACA\_Version\_6\_0\_Source\_code\_2.ex1".
  - All files listed in the file exlgui.fxl are compiled when starting the ACA. Currently, three files are listed.
  - All declarations can be seen from directly from the graphical user interface (GUI) of the ACA, including:
    - **classes**
      - Seen in the upper left pane when you select "View | Advanced > Types of Objects" from the GUI. This pane is called the "Available Object Types." Note that the objects are listed in a hierarchy (e.g., "Emission Units" can be found by expanding the "Physical Objects")
      - The class definition source code (i.e., class name in the code; ClassOptions\*; its super-class; and all slots, procedures & functions for the class) can all be seen in the pane at the bottom of the "Types of Objects" window when a specific class is selected in the upper left pane.

*\*Note, see the "EXLGUI Programmer's Manual," Version 3.1, page 29 for detail.*

---

### NOTES

- **class declarations:** i.e., slots, functions, and procedures
  - Seen in the upper right pane when you select a specific object in the "Types of Objects" view. This pane is called the "Properties of type...". When selecting a specific property in the upper right pane, the code that describes that property will be displayed in the pane at the bottom of the window called "Default definition of ...". This is where the source code of class functions and procedures can be seen. (Only slots source code can be seen from Standard View)
- **global declarations:** i.e., constants, variables, types, functions, procedures, and actions
  - Seen when you select the option "View | Advanced > All Global Declarations". There are also options for viewing a subset of the global declarations (e.g., "Global Actions"). **\*\* VERSION 6.1 UPGRADE \*\***
  - The code associated with the various global declarations will be displayed.
- New actions, classes, subclasses, global variables, global functions, global procedures can be added to the ACA.
- **Extra things you need to know to add new code:**
  - New code can be added to the file " User\_Defined\_Code.ex1" - when recompiling the ACA, this will be included
  - entirely new files can also be included by editing the exlgui.fxl file
    - uses 'NewCodeFileName.ex1'; // **Example of new line in exlgui.fxl file for new file**

---

*NOTES*

## **ADVANCED EXAMPLE PROBLEM #2: Programming with units**

**Purpose:** To learn the basics of working with global declarations.

### **Problem:**

**Step #1)** Open up a text editor, such as Notepad, and open the file "User\_Defined\_Code.exl".

*Note that the use of an editor (such as CodeWright) which performs syntax highlighting is useful if you plan to make significant additions to the ACA (beyond this workshop).*

*Note that you should not use a word processor such as Microsoft Word. Word processors save files in a binary format that the EXLGUI does not read.*

**Step #2)** Using the notes from above, and the EXLGUI Programmer's Manual, review the declarations in this file.

**Step #3)** Tweak the declarations. Suggestions...

- Change the name of the person extending the code
- Add a choice to the End\_User\_Distribution\_Type
- Change what is added to Who\_I\_AM in function WhoWroteThisExtension
- Modify SourceCount so that the name of each source is printed in the report
- Hint: The programming name of the slot that contains a source's name is Name.
- Hint: You can run any action by choosing "Analysis/Run User-Specified Analysis/Action" from the menu, and then entering the name of the action to run in the dialog box.

**Step #3)** Save the file "User\_Defined\_Code.exl" and re-start the ACA. If there are any syntax errors, you will be notified of the problem.

---

*NOTES*

# Creating New Data Types in the ACA

## Session #13

### ***ADVANCED EXAMPLE PROBLEM #2: Working with Classes***

**Purpose:** To gain experience with the Type of Objects view.

**Problem:** Locate the slot that describes the equipment cost of a "Double Venture Scrubber." If you have difficulty, use the screen capture on the following page as a guide to locate this property.

### ***ADVANCED EXAMPLE PROBLEM #3: Creating a new sub-class***

**Purpose:** To gain experience with creating classes

**Problem:** Define a new sub-class of the "Double Venture Scrubber". Do not, at this point, define or redefine any of the properties.

**Tip:** The amount of source code that you need to add to create the new sub-class can be done in under 6 lines of code. This new class definition can be put directly into the file "User\_Defined\_Code.exl".

**Step #1)** Open up a text editor, such as Notepad, and open the file "User\_Defined\_Code.exl".

**Step #2)** Using the notes from above, and the EXLGUI Programmer's Manual, define the new sub-class.

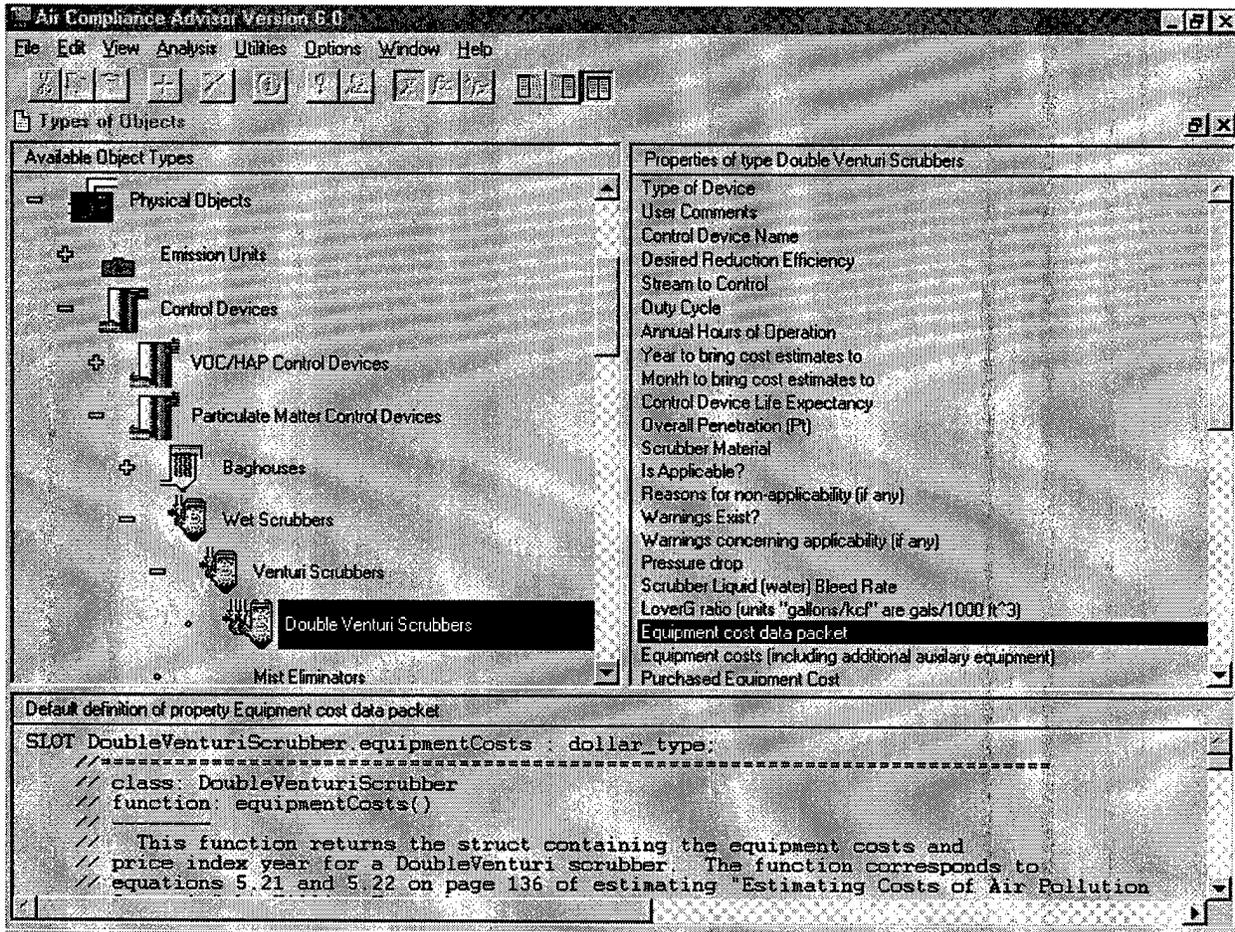
**Step #3)** Save the file "User\_Defined\_Code.exl" and re-start the ACA. If there are any syntax errors, you will be notified of the problem.

---

*NOTES*

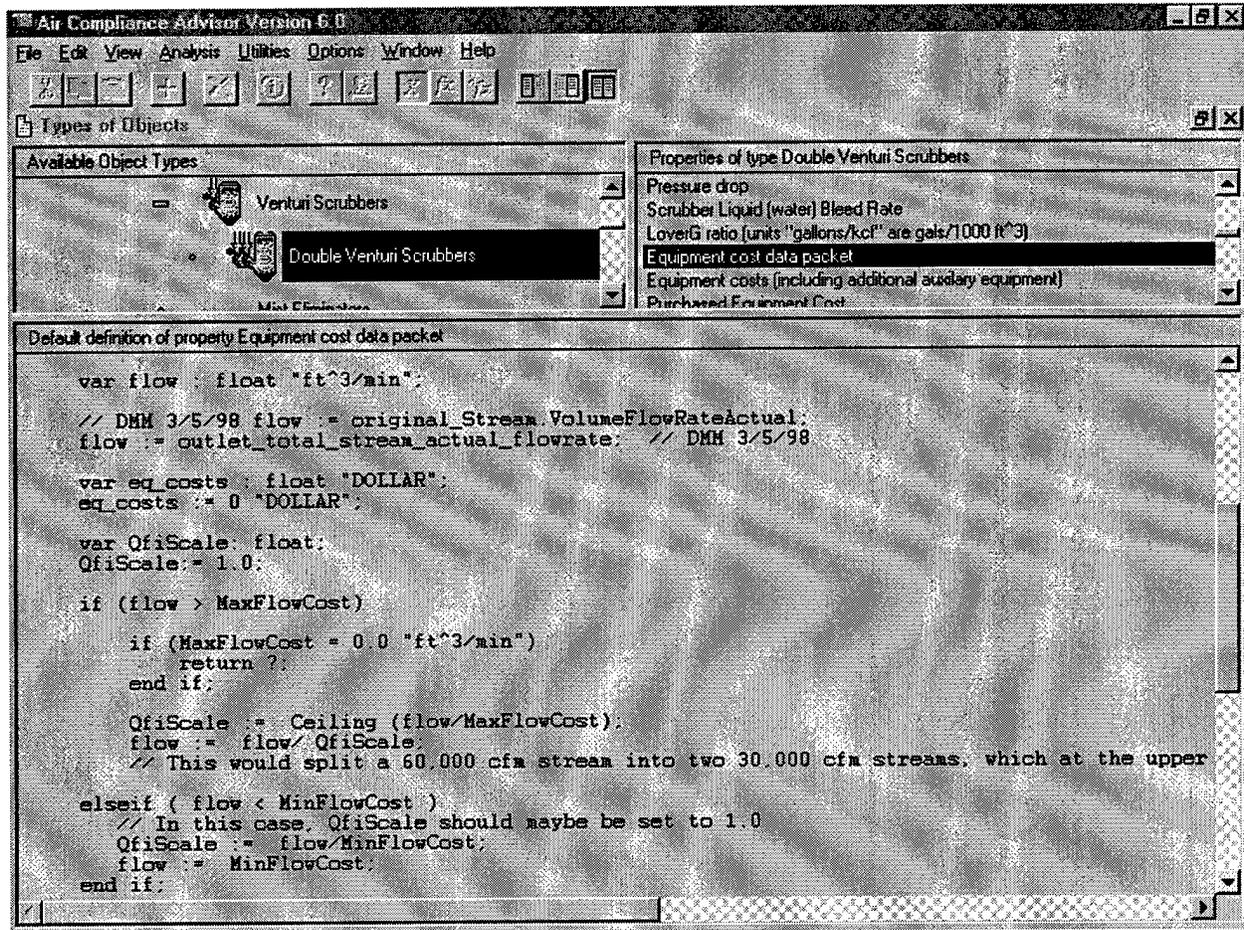
**Step #4)** Locate your new sub-class from the ACA's "Types of Objects" view.

Once you have located your new sub-class, look at the associated properties window. Can you figure out where all of the properties that are defined there came from? (**Answer:** the were "inherited from its super-class, the DoubleVenturiScrubber class).




---

*NOTES*



Screen Captures to Assist with Problem #2.

---

*NOTES*

# Creating a Customized Source - Part I

## Session #14

In the next two sessions, you will create a customized version of a class "Venturi Double Scrubber control device. The customized scrubber will allow the user to select whether the costing functions should be based upon the minimum or maximum of the inlet and outlet gas flow rates.

**Step #1)** Create a new global oneof type with the choices of MinimumFlowRate and MaximumFlowRate. This type should be called FlowRateSelectorType. (Hint: See End\_User\_Distribution\_Type for an example).

**Step #2)** Create a new global function. This function should take three arguments. The first two arguments should be mass flow rates, and the last argument should be of type FlowRateSelectorType. The function should return a mass flow rate. The function should return the proper flow rate depending on the value of the FlowRateSelectorType parameter.

Hint: You can perform branching in EXLGUI using the If statement, like this...

```
if (A = B)
  // Do something
else
  // Do something else
end if;
```

---

*NOTES*

# Creating a Customized Source - Part II

## Session #15

(Problem continued from previous session)

**Step #3)** Add a new slot to the new class you created in Advanced Example Problem #3. It should be a user-defined slot of type `FlowRateSelectorType`. By default, this slot should return `MaximumFlowRate`.

**Step #4)** Add a new function to the new class you created in Advanced Example Problem #3. The function should not take any parameters, and should return a mass flow rate. It should use the slot just added in step 3, the global function you created in step 2, and inlet and outlet flowrates.

Hint: The inlet flowrate can be accessed like this: `original_Stream.VolumeFlowRateActual`.  
Hint: The outlet flowrate can be accessed like this: `outlet_total_stream_actual_flowrate`

**Step #5)** Override the code for a member slot `Name`, so that it returns a new title, such as "My Venturi Double Scrubber".

Hint:

**Step #6)** Override the slot `EquipmentCosts` so that uses the flowrate returned from the function that you defined in step #4. **Hint:** You can find the inherited declaration of this slot by using the `Types of Objects` window in the ACA. Select `Double Venturi Scrubber` (the class that your new class is based on) in the `Available Object Types` pane.

**Step #7)** Your new scrubber is ready for use. Start the ACA. Go the `Library/Control Device Library/User-defined Control Device Library`, and add a new instance of your new scrubber to the list. Your new scrubber will now be considered whenever the `Apply Control Technologies` analysis is run.

---

### NOTES

# Creating a New Analysis

## Session #16

**ADVANCED EXAMPLE PROBLEM #7:** *Write an action that reports the number of HAPs in the chemical database.*

**Purpose:** The gain experience writing actions and analysis

**Problem:** Create an action that will create a report listing the number of HAPS in the ACA chemical database. The report should contain the name of each HAP, and also list the total number of HAPs.

**Hints:** Study how procedure SourceCount loops over all the sources in the installation source list. You want to do the same on the two lists in the Library.ChemicalLibrary object.

---

*NOTES*