

**Open Emissions Model (OPEM)**  
**Re-Thinking the Emissions**  
**Modeling Paradigm**

**Mark Janssen - LADCO**  
**Zion Wang - UC Riverside**

**March 5<sup>th</sup>, 2003**

This document proposes a new construct for emissions modeling. The authors provide a balanced review of the state of the science of emissions modeling and processing and offer a forward thinking design for a new system that will meet future needs. A public domain relational database management system (RDBMS) is used as a backbone for the proposed emissions processing system while all functions currently available in emissions processing systems are retained.

## **Introduction**

Emissions modeling refers to the calculation of complex air pollutant emissions inventories from raw data. The typical calculation is an emission factor multiplied by the activity throughput to obtain an emissions estimate. In reality, emissions models are much more complex than this simple equation. The most common use of emissions modeling is to prepare emissions estimates for photochemical transport modeling. The emissions estimates that are required for this modeling must represent either a specific day or an artificial day in the future. Additionally, the estimates must reflect the spatial resolution of the modeling system and characterize the chemical species required by the chemical mechanism in the photochemical transport model. This preparation involves three main steps in processing emission estimates: spatial allocation, temporal allocation and speciation.

Spatial allocation involves the distribution of emissions onto the Cartesian grid used by the photochemical transport model. Sometimes, emissions are associated with a specific coordinate (this is the case for point sources) and spatial allocation is a matter of translating the coordinate system to the appropriate cell in the Cartesian grid. For other sources, emissions are estimated at an aggregate level such as by county. In these cases, the model uses gridded spatial surrogates, such as population or land use, to assign a fraction of the total emissions to the appropriate cells. The process of creating the gridded spatial surrogates often requires the use of a geographic information system (GIS.) Typical modeling domains contain as many as 3600 counties and 200 to 500 emission inventory categories making spatial allocation a computationally intensive activity for emissions models.

Temporal allocation takes annualized emissions estimates and calculates a representative estimate for a specific day or type of day. There are two methods of temporal allocation: day specific and typical day. Day specific temporal allocation looks at the attributes of a particular real day and accounts for many of the

variables that effect the emissions source. As an example, for biogenic sources these variables might include ambient temperature, solar radiation, cloud cover, and soil moisture. Whereas for on-road mobile sources these variables might include day of the week, mix of cars and trucks, ambient temperature, humidity, frequency of vehicle trip starts and stops, and minimum and maximum temperature for the day. This is the most accurate way to accomplish temporal allocation and it is data intensive and computationally difficult. Typical day temporal allocation is a less complex and more common method that takes a pre-existing annual or seasonal estimate and uses factors to adjust from year to year, from year to month, from average day to specific day of the week, and from day to hour. These factors are often related to activity and do not look at the seasonality or temperature effects on emissions factors. Credible temporal allocation is arguably the most difficult part of emissions modeling.

The third component of emissions modeling is speciation. Speciation is the estimate of emissions for a particular chemical species from more generic emissions estimates using a specific chemical mechanism. One of the most common speciation activities is conversion of VOC (volatile organic compound) emissions estimates to CB-IV (Carbon Bond 4) chemical mechanism species groups. Using this method, each SCC (source category code) in the inventory is cross-referenced to a specific chemical profile. The reference may be county specific, as in the case of automobile fuels, or national, as in the case of residential coal combustion. Each speciation profile contains mass fractions of individual chemical compounds. The mass of each fraction of chemical species is converted to moles of that compound and then to moles of CB-IV species groups. The following two examples illustrate the complexity of this type of application:

1. For auto-body refinishing, 5.67% of VOC emissions are ethylhexane (SAROAD code 90081.)  
Next, for every mole of ethylhexane the model assumes eight moles of the CB-IV group "PAR" (paraffin.)
2. For light duty gasoline vehicles, every gram of gasoline exhaust VOC is assumed to include moles of CB-IV groups "ALD2" (aldehyde, group 2) and "PAR" (paraffin).

In addition to performing this equation for hundreds of species and thousands of SCC codes to create a complete chemical speciation data set, there are several other factors that make speciation difficult. First is the general lack of speciation data available for many types of activities. For example, the speciation of leaf blowers is based on a 1989 study of lawn mowers. Another difficult part of speciation is the cross-reference of individual chemical species to one or more lumped groups. This is confounded by the variety of sources that vary by state, county, and year. Finally, developing a functional speciation processor that is computationally efficient is a challenge.

### **Emissions Processors versus Emissions Models**

It is important to distinguish between emissions models and emissions processors. Emissions processors use spatial, temporal, and speciation programs to modify pre-existing emissions estimates to create emissions data sets that can be used by photochemical transport models. Emissions models create new emissions estimates based on a variety of input factors.

Often the two terms are used interchangeably but the distinction can be important when describing the development of new tools. Common examples of emissions models are MOBILE 6 (for on-road emissions), BEIS-2 and BIOME (for biogenic emissions), and NONROAD (for off-road mobile

emissions.) Examples of emissions processors are the point and area source programs in EMS-2003 that generate model ready emissions estimates based on reported emissions inventories. Both SMOKE and EMS-2003 combine emissions models and emissions processors.

### **History and Critique of EMS-2003**

In the early 1990s the California Air Resource Board (CARB) and The Lake Michigan Air Directors Consortium (LADCO) funded a project to build a new emissions processor to replace the dominant one being used at the time, EPS (Emissions Preprocessor System). The new model, Geocoded Emissions Modeling and Projections (GEMAP), was written in SAS programming language with links to the GIS system ARC/INFO. In 1995 EMS-95 was released. Based on GEMAP, EMS-95 included significant modifications including new area and point processors, speciation processors, growth and control processors, and the incorporation of the Mobile5b emissions model. At the same time, EPA was modifying GEMAP to be part of the MODELS3 system. That modified version of GEMAP was named MEPPS. Most advances in EMS since the mid-nineties have concerned improving the quality assurance features of the processors since many in the field perceive the quality of the raw data to be the biggest issue facing emissions data development. Added features include a tiered reporting structure that prioritizes problems identified in the data and indicates the possible implications of those problems and an expanded library of off-line programs that can be used to identify weaknesses in the input data such as unreasonable stack parameters and speciation. In 2001, a biogenic model, BIOME3, was added to EMS. BIOME3 is capable of running either BEIS2 or BEIS3 formulation scripts. In 2003, the mobile source model in EMS was updated to run the scripts used in USEPA's MOBILE6.

The most current version of EMS available is EMS-2003. The principal difficulty with EMS-2003 is its reliance on SAS. While SAS has powerful tools for data export/import, data visualization, and simple programming, it is very expensive to own and operate. When EMS was first written, a single user could obtain a copy of SAS for \$2000 for many of the computing platforms used to do modeling. The current cost of SAS is in excess of \$20,000 unless a user can obtain a copy from EPA or an affiliated university. This cost is prohibitive for many users. Additionally, because EMS-2003 has very few performance modifications, processing speed is almost directly proportional to hard disk I/O on modern computing systems. To optimize EMS-2003 the developers recommend high-speed SCSI RAID devices for important drives.

### **History and Critique of SMOKE**

MCNC created the Sparse Matrix Operator Kernel Emissions (SMOKE) Modeling System in 1999 to create an emissions data processing method that integrates high-performance-computing (HPC) sparse-matrix algorithms. The SMOKE system supports area, point, and mobile source emissions processing and also includes biogenic emissions modeling based on BEIS2. In 2001, SMOKE was integrated into USEPA's MODELS3.

The SMOKE modeling system provides much flexibility. However, it is a complex system to learn for end users. Most of the complexities in SMOKE are because:

1. Users want to create output for many different photochemical transport models, each with different format requirements (e.g. different elevated/low-level approaches to their emissions inputs)

2. Data formats used by inventory developers continue to evolve making it necessary to convert new data to older formats used by SMOKE
3. Input data are reported in several temporal formats including annual, average-annual-day, average-annual-weekday, average-seasonal-day, average-seasonal-weekday, day-specific, and hour-specific
4. Users want to mix and match the many options for processing on-road mobile sources (e.g. use VMT in some counties and pre-calculated emissions in others.)

In the SMOKE paradigm, the temporal, spatial, speciation, and projection of emissions are “factor based”. This means that these are linear operations that can be represented as multiplication by sparse matrices. The SMOKE developers have discovered that lots of the matrices for the factors are sparse matrices (most of their entries are zeros.) SMOKE is designed to take advantage of this by formulating emissions modeling in terms of sparse matrix operations. SMOKE processes emissions by utilizing optimized sparse matrix libraries to handle sparse matrix computations. The sparse-matrix approach is efficient in terms of computation, but the routines that handle sparse-matrix calculation can be difficult to understand and debug. Very few, if any, end users would be capable of updating or adding a processor to SMOKE.

### **Attributes of a Good Emissions Modeling System**

When assessing the usefulness of emissions modeling systems there are three characteristics that every emissions model should have:

1. *Quality Control:* Inventory quality is highly variable and often of low quality for important parts of the modeling process. The model must be able to identify critical and non-critical errors in the data inputs and report them in an effective way.
2. *Transparency:* In the near future expected models for point, area, on-road mobile, off-road mobile, biogenic, electric utility, livestock, soil, and fires will require models and data sets that are easy to understand and de-construct.
3. *Performance:* Emissions models are part of a larger modeling process and should use the same hardware and software required by the other models in the process.

In terms of quality control, the most important feature for an emissions model is to quickly and efficiently diagnose problems with the inventories and report them in a concise way. In the past, emissions models would identify thousands of potential errors, far more than inventory developers were capable of fixing or even reading. Often the implication of these errors was unclear and, more often than not, unimportant to the overall modeling process. Moreover, important problems were often overlooked, buried in thousands of lines of output. Quality assurance tools must therefore be built that identify significant problems clearly including the implication of found errors. Priority should be given to those issues that are expected to affect the modeled outcome.

Transparency refers to the ability to trace exactly the sources of data and what happens to it during the emission modeling process. This is critical to the success of any emissions model because it allows the end user to understand the reliability and consistency of the data given the many different calculations and variety of data types. This means that the language used to code an emissions model must be readable and easily

learned. Additionally, it must be easy for a user to see the data that the model is using at any step in the process. In order to evaluate output that seems erroneous, users need to be able to run basic queries on data sets in order to trace the exact equations used for a specific value. For example, the user should be able to reference a data set and request only records for a given state, county, and SCC code.

Data transparency is further improved by a system that makes complex ad-hoc report writing possible by the lay emissions modeler after 6 months of experience with the model and its source code. This allows investigations of new data issues as they come up. Processors should also be developed that report emissions estimates through the different stages of emissions modeling such as input processing, spatial processing, temporal processing, and finally speciation processing. These reports would look at state and county totals by major sector of the inventory. Additionally, some method should be devised to store summaries of past modeling runs for easy comparison with current values.

Performance has two distinct components: software and hardware. Satisfactory software compatibility requires the model to run on similar platforms as the photochemical models with minimum modifications required for operation of the model. The current standard operating system for photochemical transport models is Red Hat LINUX along with the Portland Group FORTRAN compiler. Also, any newly designed emissions model needs to create output compatible with the photochemical transport models CMAQ and CAMX. Emissions models should use only software compatible with the Red Hat LINUX platform and with costs that do not exceed \$2000.

Performance can often be improved by the removal of obsolete intermediate data sets. However, these intermediate data sets can be useful for quality assurance of a modeling run. A good emissions model should be designed to allow the user to retain or delete these intermediate data sets depending on quality assurance concerns.

Another software consideration is the link between the meteorological model, MM5, and the emissions model. MM5 is used as an input to both emissions and photochemical models. A strong link between MM5 and the emissions model should be available. For Models3/CMAQ and EMS-2003 this connection is accomplished with external processors. This program should be simple and straightforward and have all the attributes discussed here for a good emissions model.

Finally, user interfaces are, in general, over rated as a practical tool for modeling episodes that span several days and are configured to take advantage of modeling options. The best emissions models have a robust batch processing methodology that allows users to easily configure and run the model and control the numerous options available.

The second aspect of model performance, hardware compatibility, requires that only modest adaptations to the hardware used to run the photochemical transport models need to be made to efficiently run the emissions model. As of December 2002, the hardware recommendations for running a photochemical transport model include a dual 2+ Ghertz Intel Processors, 1 Gbyte of RAM, 800 Gbytes of IDE hard drive, and 200 Gbytes of SCSI hard drive running REDHAT LINUX version 7.3 with the PGI FORTRAN compiler. Often disk input/output (I/O) is the limiting factor in emissions modeling and considering current

options, the best way to resolve disk I/O problems is to use striped RAID SCSI drives. A four drive striped SCSI disk array costs less than \$1500 and should be the only hardware improvements needed for emissions modeling. It is important that the emissions model is at least as fast as the photochemical transport model being used and current emissions models are indeed at least two to 5 times faster.

### **Simplified Explanation of New Paradigm**

The tasks of the emissions processor include reading emissions inventory files, temporal allocation, spatial allocation, speciation, and projection of emissions to future/past years (optional.) Two main paradigms for atmospheric emissions models can be identified: the “network-of-pipes-and-filter” approach and the “factors-based” approach.

In the “network-of-pipes-and-filter” approach, an emissions file contains records that describe each source and all the attributes acquired during each processing stage. As additional data and source information are combined, a new set of records is created. In this approach, all processing is performed one record at a time, without structure or order to the records.

The second approach, the “factor-based” approach, recognizes that the tasks performed during emission processing are linear operations and factors can be developed for each task. The factors can be represented as matrices; and, subsequently, the tasks can be represented by the multiplication of matrices. In addition, the order of multiplication operations can be arranged to increase computational efficiency.

There are pros and cons to each approach. One important area of interest is QA reporting. In EMS-2003, reports are generated at each stage of processing while SMOKE (with the exception of temporal processing) requires the application of additional QA programs to create the report for each stage of processing. Further analysis is needed to determine which approach is better for the new emissions model. However, a relational database management system (RDBMS) is exceptionally suited for emissions processing and will augment either approach.

In the simple paradigm for development of a new emissions model, public domain open source database software will be used as the backbone. This software will be free, readily available on most platforms, and be a RDBMS. Emissions inventory data will be read into the database, all information in the relationship database will be represented as a value in tabular format, and every value will remain accessible by using a combination of the table name, primary key value, and column name. A description of the database and its contents is represented at the logical level in tabular format allowing it to be queried using the database language. Emissions data in the RDBMS can be retrieved, inserted, updated, and deleted. Data rules are defined within the database and stored in an online catalog; therefore, they cannot be bypassed. A variety of “sanity checks” will be implemented to ensure data are within reasonable bounds. In the processing of the emissions data, such as the creation of factors and the merging of factors with the emissions inventory data, the functions, data types, and aggregates available in the RDBMS will be used. Additional functions to create the emissions output for photochemical transport models will be written using GNU C (also free and publicly available.) QA and summary reports will be generated at different stages of processing. The use of a RDBMS for emissions processing will facilitate a transparent process that is easy to understand for both end users and model developers.

## **Improving on Existing Models**

It is logical to ask the following question: *“If good emissions models such as SMOKE or EMS-2003 already exists, why do we need this new model?”* This section explains what has been learned from the existing models and provides examples of how a new Open Emissions Model (OPEM) will improve emissions processing.

### Data Structure

Currently, emissions inventory data are read into an emissions model and stored in inventory files that are sorted in a specific order. These files contain both emissions values and source characteristics (e.g. stack coordinates, stack parameters, SCC codes.) Subsequent operations require a consistent order between related files to enable assignment of factors using subscripts instead of searches. In a RDBMS, once all the data are added to a database, information can be managed and accessed through relational capabilities. Every value in the relational database is accessible by using a combination of the table name, primary key value, and column name. Relationships are defined between tables in a database, so no specific sorted order is necessary. Furthermore, database normalization reduces redundancy by separating data into separate tables within which each type of information is stored only once. This increases efficiency, reduces the size of the database, and makes it easier to update.

### Index

SMOKE employs the “factor-based” approach. Profile and cross-reference tables are used to convert the resolution of emissions for spatial allocation, temporal allocation, speciation, and mobile source emission factor assignment. Each profile entry is assigned a profile number. The cross-reference tables assign the profile to each source and contain source characteristics and the profile numbers to use. The cross-reference tables are applied to the sources in a stepwise manner, such that the most specific entry is always applied. In handling cross-references and profile application, SMOKE first sorts the cross-reference record using the same sort criteria as is used the inventory records. Then, these records are grouped according to the “level” of matching to each of the entries. Once the cross-reference entries are grouped, SMOKE loops through all records in the inventory and attempts to find a matching entry in one of the cross-reference groups. The most specific groups are searched first so, when a match is found, the other groups are not searched and SMOKE continues to the next source in the inventory. SMOKE developers believe this approach is much more efficient than other assignment methods. However, this indexing process is very complex and can easily be solved by storing all data in a RDBMS and using indexes to efficiently retrieve data. This also reduces a potential source of errors.

### File Format

EMS-2003 stores all data in SAS data sets. While SAS has numerous built-in procedures to process, visualize, and provide emissions summaries from the data sets, access to the data requires the purchase of expensive SAS software. SMOKE uses both ASCII and I/O Applications Programming Interface (I/O API) formats to store data. The I/O API files are binary files and can be read and written by a library provided with SMOKE. The I/O API library also provides many quality assurance features useful for all I/O, including I/O for emissions processing. However, any given I/O API library can vary depending on its version and how it is compiled. This lack of consistency can be problematic. Alternatively, all data storage including inventory data, intermediate files, and factors could be efficiently stored and processed in an

RDBMS and accessible to anyone who has obtained the publicly available software used as the backbone.

## **Priorities for Model Development**

Several key attributes need to be kept in mind while building a new emissions model. The first is that the file formats required should be among those already commonly used. These formats include EPA's NIF2.0/3.0 and the Regional Planning Organizations Data Exchange Protocol. The format used is important in order to make it easy for states and local governments to submit their data.

Secondly, OPEM will be created as a community software development project. Model development activities will involve various groups and will not use a hierarchical software development model where there are code validators employed alongside code developers. The challenge of any community software development project is to have a process of model validation that assures emissions calculations are properly coded and that the variety of data sources are handled correctly. Ultimately the modeling community's comfort in OPEM's data transparency, code robustness, stability, and computational accuracy will be the deciding factor in future widespread adoption. The criterion that will drive development of this model is summarized by the simple statement, "Can I easily figure out what the model is doing?"

Finally, development of a user interface should only be pursued when it furthers the ability to understand the model, not as a tool for model execution. Examples of useful user interfaces include:

1. A tool to access any data set in the model and perform queries using nested Boolean equations
2. A tool to trace a single source through the various stages of model processing in order to evaluate the relationship between a specific input record and the consequent output record
3. An online user's guide to direct users through simple problem solving examples and model operation.

## **Selecting the RDBMS**

Several public domain open-source database software packages are currently available. They include MySQL, PostgreSQL, SAP DB, and Firebird. There is a frequently heated debate on the merits of two of the most popular database management systems, namely MySQL and PostgreSQL. Both DBMS are stable systems and capable of being the backbone to OPEM. Both DBMS are available on various platforms including LINUX, Solaris, HP-UX, MacOS, AIX, IRIX, SCO, and Windows NT/2000. After careful comparison, while MySQL and PostgreSQL provide many similar functions, PostgreSQL is recommended for the following reasons:

1. PostgreSQL is more SQL standard compliant than MySQL.
2. PostgreSQL is ACID compliant and MySQL is not. ACID compliance refers to:
  - a. Atomicity is an all-or-none proposition
  - b. Consistency guarantees that a transaction never leaves your database in a half-finished state
  - c. Isolation keeps transactions separated from each other until they are finished
  - d. Durability guarantees that the database will keep track of pending changes in such a way that the server can recover from an abnormal termination.
3. PostgreSQL uses a client/server model allowing the client and the server to be on different hosts and communicate over a TCP/IP connection.
4. The PostgreSQL server can handle multiple concurrent connections from clients.

5. PostgreSQL has a mechanism called Multi-Version Concurrency Control (MVCC) for locking and concurrency support that is comparable or superior to the best commercial databases.
6. PostgreSQL can handle as much as three times the load as MySQL.
7. PostgreSQL forks a new process for each connection making it somewhat slower than MySQL (multi-threaded); however, the difference in speed is not considered significant and there are various ways to speed up PostgreSQL.
8. In the Windows environment, the MySQL server performs better than PostgreSQL; however, OEPM will be used mostly on LINUX platforms.

## **Incremental Development**

The best approach to develop OPEM is to thoroughly examine EPS, EMS, and SMOKE for the creation of a “road map.” This road map will be used to create a logical database design. From this design, OPEM will be implemented in stages. The first stage will lay a foundation with area source emissions. This foundation will set the standards for internal documentation of code, input and output formats, transparency, and programming efficiency. Once the area source emissions processor is finalized, it will be followed by components for projections, point sources, on-road, and biogenics. The entire process will take approximately one to one and a half years to accomplish.

## **Implementation**

Development of a regional emissions model has customarily been an expensive undertaking. Models like EMS and SMOKE can cost millions to develop and implement. However, their development did not rely on the expertise available in the emissions modeling community. Instead single contractors (along with their sub-contractors) were hired at high cost to code the model in a relatively insulated environment. The community based development structure is essential to the development of the LINUX operating system and has been proven to work effectively for models like MM5.

This proposal does not call for a large initial outlay of funding to build OPEM. Instead, it seeks community supporters interested in participating in the development of the model using in-kind resources.