



LABORATORY OF APPLIED THERMODYNAMICS  
MECHANICAL ENGINEERING DEPARTMENT  
ARISTOTLE UNIVERSITY THESSALONIKI  
P.O. BOX 458 GR 541 24 THESSALONIKI GREECE

---

# ***COPERT 4***

## ***Beta Version Software Description***

Santiago Bel  
Charis Kouridis  
Leonidas Ntziachristos

Thessaloniki  
December 2005





LABORATORY OF APPLIED THERMODYNAMICS  
MECHANICAL ENGINEERING DEPARTMENT  
ARISTOTLE UNIVERSITY THESSALONIKI  
P.O.BOX 458 GR-54124 THESSALONIKI GREECE

tel: +30 2310 996047 fax: + 30 2310 996019  
<http://lat.eng.auth.gr/>

<b>Project Title</b> COPERT 4	<b>Contract No</b> 3331/B2005.EEA-ETC/ACC
<b>Report Title</b> Beta Version Software Description	<b>Reference No</b> Web version
<b>Project Manager</b> Prof. Zisis Samaras (zisis@auth.gr)	
<b>Author(s)</b> Santiago Bel, Charis Kouridis (hkouridi@auth.gr), Leonidas Ntziachristos (leon@auth.gr)	
<b>Summary</b>  This report describes the development and the characteristics of the beta version of Copert 4. It justifies the selection of .NET as the software development platform and analyses the structure of the new software. This report is intents to inform the national experts and Copert users of the upcoming software characteristics and to introduce them into the new specifications. Any observations, requests or general feedback should be addressed to Leonidas Ntziachristos ( <a href="mailto:leon@auth.gr">leon@auth.gr</a> ) and Charis Kouridis ( <a href="mailto:hkouridi@auth.gr">hkouridi@auth.gr</a> ).	
<b>Keywords</b> Software, Copert, emission factor, .NET	

<b>Internet reference</b> <a href="http://vergina.eng.auth.gr/mech0/lat/copert/copert.htm">http://vergina.eng.auth.gr/mech0/lat/copert/copert.htm</a>			
<b>Version / Date</b> 1.0/December 2005		<b>Classification statement</b> FOR EVALUATION	
<b>No of Pages</b> 52	<b>Price</b> FREE	<b>Declassification date</b> Summer 2005	<b>Bibliography</b> Yes

# Contents

<b>1. Introduction .....</b>	<b>5</b>
1.1 Introduction of the methodology used by COPERT .....	5
1.2 COPERT 4 prototype and development objectives .....	6
1.3 Technical Solution for the prototype development .....	7
<b>2. Methodology for calculation of COPERT .....</b>	<b>8</b>
<b>3. Detailed prototype requirements analysis.....</b>	<b>11</b>
<b>4. Diagram and Specification.....</b>	<b>13</b>
4.1 Diagram .....	13
4.2 Specification .....	14
<b>5. Kernel and Database programming .....</b>	<b>20</b>
5.1 Database Technologies .....	20
5.2 Kernel Technologies.....	21
<b>6. Database design and construction.....</b>	<b>23</b>
6.1 Table Explanation.....	25
6.1.4. A_POLLUTANTS .....	26
<b>7. Calculation system design .....</b>	<b>34</b>
7.1 Class Diagram .....	34
7.2 Operations Specification .....	36
<b>8. Summary &amp; Status of the software .....</b>	<b>51</b>
<b>9. References .....</b>	<b>52</b>

# 1. Introduction

The software COPERT (COMputer Programme to calculate Emissions from Road Transport) estimates emissions of all regulated air pollutants (CO, NO<sub>x</sub>, VOC, PM) produced by different vehicle categories (passenger cars, light duty vehicles, heavy duty vehicles, mopeds and motorcycles) as well as CO<sub>2</sub> emissions on the basis of fuel consumption. Furthermore, emissions are calculated for an extended list of non regulated pollutants, including CH<sub>4</sub>, N<sub>2</sub>O, NH<sub>3</sub>, SO<sub>2</sub>. Emissions estimated are generally distinguished in three sources: Emissions produced during thermally stabilized engine operation (hot emissions), emissions occurring during engine start from ambient temperature (cold-start and warming-up effects) and NMVOC emissions due to fuel evaporation. The total emissions are calculated as a product of activity data provided by the user and speed-dependent emission factors calculated by the software.

The objective of this report is to present the software characteristics of the beta version of COPERT 4. This will be the fourth update of the initial methodology developed on the basis of the work of a working group which was set up for this purpose (the initial version was COPERT 85 (1989) and then followed COPERT 90 (1993) and COPERT II (1997) and COPERT III (2000)). The current software however does not include any of the new methodological elements of Copert 4 but it is just a transfer of Copert III methodology into the new platform for evaluation.

The technology used for the new Copert version is completely new. Previous versions were developed using MS Access and this new version will be developed using MS Visual Studio .NET. This change of technology implies a completely new software design and coding work.

**Note:** It should be made clear that this report and the beta version of the software should be only treated as evaluation versions. The software is supposed to be used by national experts and interested parties in an effort to collect comments about its compatibility with different operation systems, speed of calculations and ease of operation. It should by no means be used as a tool for official data submission in the framework of CLRTAP, UNFCCC or other activities. Additionally, the development team shall not be liable for any damages whatsoever arising out of the use or inability to use the Copert 4 beta version software.

**Note:** For any requests, feedback and general questions, users should contact Leonidas Ntziachristos ([leon@auth.gr](mailto:leon@auth.gr)) or Charis Kouridis ([hkouridi@auth.gr](mailto:hkouridi@auth.gr)).

## 1.1 Introduction of the methodology used by COPERT

COPERT calculates the pollution produced by one (or more) vehicle in one year. For this objective, the COPERT methodology [2] bases the calculations in the called Emission Factors. These factors describe how much quantity of a pollutant is produced by a vehicle per Km.

Basically there are two Emission Factors,

- a. Hot Emission Factors
- b. Cold Emission Factors

This differentiation is made because the pollutant produced by a vehicle depends highly on the temperature of the engine. Hot Emission factors are related to the quantity of pollutant produced when the engine is in normal temperature conditions. In the other hand, Cold Emission Factors are related to the quantity of pollutant produced when the engine is not still in the proper warm temperature.

Once the Emission Factors are calculated it is possible to calculate the Cold and Hot Emissions that the vehicle will make in one year. Knowing the pollution that will produce in one Km (in hot or cold conditions), will be only needed the information about how many Km will run in one year and how many of these Km will be in cold or hot conditions. Intuitively:

$$\begin{array}{ccccc}
 \text{Cold Pollutant} & & & & \text{Number of} \\
 \text{Emissions} & & & & \text{Km made by} \\
 \text{while the car is} & = & \text{Quantity of} & \times & \text{the vehicle} \\
 \text{in cold} & & \text{Pollutant per} & & \text{per year in} \\
 \text{conditions} & & \text{Km will be} & & \text{cold} \\
 & & \text{produced by the} & & \text{conditions} \\
 & & \text{vehicle in cold} & & \\
 & & \text{conditions} & & 
 \end{array}$$

Being the Hot Pollutant Emissions calculated, by analogy, in the same way.

Of course the number of Km made by a vehicle in cold or hot conditions is not obvious and is calculated by COPERT using many parameters.

To reach better refined results, COPERT calculates and uses the Mileage Degradation Factor and the Fuel Effect Factor. The first one takes in account that the age of the vehicle is important (older cars will produce more pollution). The last one is related to the effects that new fuel types can produce in older cars (improved fuels will make the cars produce less pollution even being elders).

This methodology is described more detailed in the chapter 2.

## 1.2 COPERT 4 prototype and development objectives

The prototype will be a complete and autonomous system capable of performing the calculations explained in section 2. The calculations are made following the guidelines of previous versions of COPERT[1] and its methodology [2] in some moments, but must be reprogrammed since new calculation methodologies are used.

The new prototype, focused also in the structure of the data stored, will have a new feature: keep in the same database the information calculated for several countries and several years which imply an extension of all the other functionalities to achieve this goal.

Summarizing, the prototype will re-implement the following functionalities:

- a. Management of Sectors, Subsectors, Technologies.
- b. Management of input data: Fleet Information, Fuel Information, Country information, etc...
- c. Calculation of Hot and Cold Emissions
- d. Calculation of Fuel Effects and Mileage Degradation Factors.

Also will implement with a new methodology the Calculation of Hot and Cold Emission Factors and will incorporate, as a new feature, the functionality of performing these calculations for several countries and several years' scenarios.

For more information about the functionalities offered by the prototype, in chapter 3 are given complete specifications of the requirements and the objectives.

As a last point, the way of coding the prototype will be focused on the extensibility, making easier future upgrades. One way to achieve this goal will be programming all the calculations as independent calculations. In previous versions of COPERT this procedure was not followed enough and made the modifications or upgrades difficult to perform. If it is true that this way of coding will be in detriment of the performance, the objective is to get enough good performance level as far as it is possible to keep this coding requirement.

## 1.3 Technical Solution for the prototype development

After evaluating all the possibilities, the chosen one, was to develop the prototype with Visual Basic .NET. The reasons are based on it is easy to use, economically is not a problem, it is easy to link with MSOffice tools and the Databases connection is good.

The easiness of use should not be something to take in account at all, but this prototype is in a special situation. Future developments will not be done by very experienced programmers or computer engineers. This could be a problem in the future since many engineers have large knowledge about their areas (like pollutants and vehicles) but not that much in software programming. Choosing an easy way since the beginning would improve a lot the view of the future.

Secondly, the way of linking with MS Office makes easy to import data from the Excel Datasheets that the users are very familiar with. When large quantity of data is going to be introduced in the system and frequently, this functionality can save a lot of time.

Finally, the connection to databases is also very important since all the data will be stored in databases. The chosen one at last is MS Access, what makes even stronger the decision of using both systems from the same company.

In the Database design, was decided to use MS Access because it does not need any server installation, is enough powerful to response to the SQL Queries needed and other reasons less but also important

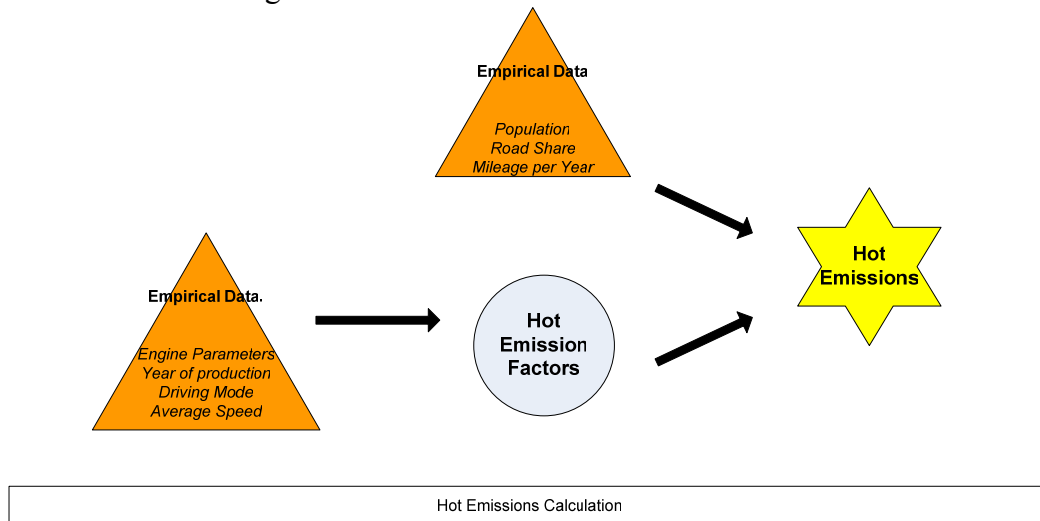
## 2. Methodology for calculation of COPERT

The objective of COPERT is the calculation of the pollution generated by the vehicle fleet of one country in one specific year. To reach this final value it is needed to follow some steps in sequence:

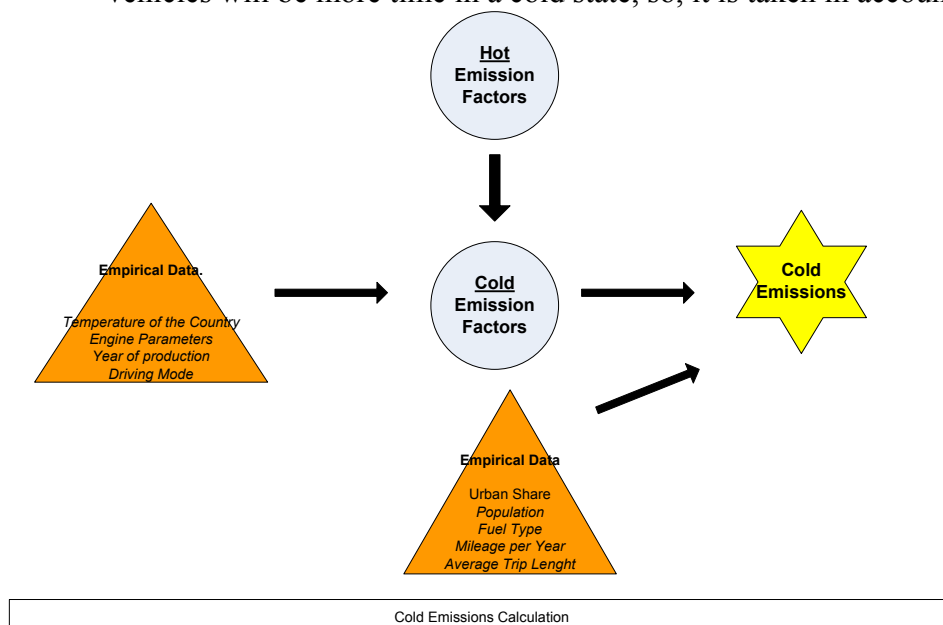
1. **Hot Emission Factors Calculation:** The Hot Emission factor is the numerical expression for the quantity of pollution that one vehicle generates per Km in normal conditions. This calculation takes many parameters:
  - a. Engine Parameters: Some coefficients depending on the engine type. These parameters are named with Greek letters (Alpha, Beta, Gamma, Delta, Epsilon, Zeta and Eta). Also it is important the type of petrol used by the vehicle.
  - b. Year of production: Depending on the year of production (resp. year of registration), vehicles must fulfill different legal requirements (emission standards) about emissions, etc... These values are also taken in account. Many times there is not enough information about the consumption of one engine, the pollutant quantity, etc.. but it is known that is the pollution of another vehicle type reduced with a coefficient (for example the new version of one car will reduce the pollutant quantity of its predecessor in a 30%). This coefficient (Reduction Factor) is also a parameter for the calculation of the Hot Emission Factors.
  - c. Driving Mode: The Driving mode is very important since the pollution produced will change depending on the type of road. We take in account 3 types of road, so 3 driving modes: Urban, Rural and Highway.
  - d. Average Speed: Depending on the average speed the pollutant production is different.
2. **Hot Emissions Calculation:** The Hot Emissions is the numerical expression for the quantity of pollution produced by all the vehicles of one country in normal conditions. Since we already have calculated the Hot Emission factors, now we only need to make a last calculation using other parameters:
  - a. Population: Number of vehicles of every type existing in the country,
  - b. Road Share: Percentage that explains the portion of Km that every vehicle runs in every type of road. (As an example, a truck will stay 80% in Highways and a small motorbike only a 10%. This will affect to the calculation of pollution)
  - c. Mileage per Year: Mileage made by every vehicle per year.
3. **Cold Emission Factor Calculation:** This emission factor represents the pollution produced per Km by the vehicles when they are not in a hot state. This happens in the beginning of any trip. This calculation is based in the Hot Emission Factors calculation. The other parameters used are:
  - a. Temperature of the country: It is taken in account the temperature of every month of the year in country since in colder seasons the engine will become hot later than in warm seasons.



- b. Engine Parameters: Depending on the engine, like in the Hot Emission Factors, the pollutant production differs.
- c. Year of Production: Like in the Hot Emission Factors
- d. Driving mode: For this calculation is only taken in account the Rural and Urban driving modes since it is supposed that a car will never be in a cold state in a Highway. Normally to reach the Highway will do some Km and get hot.

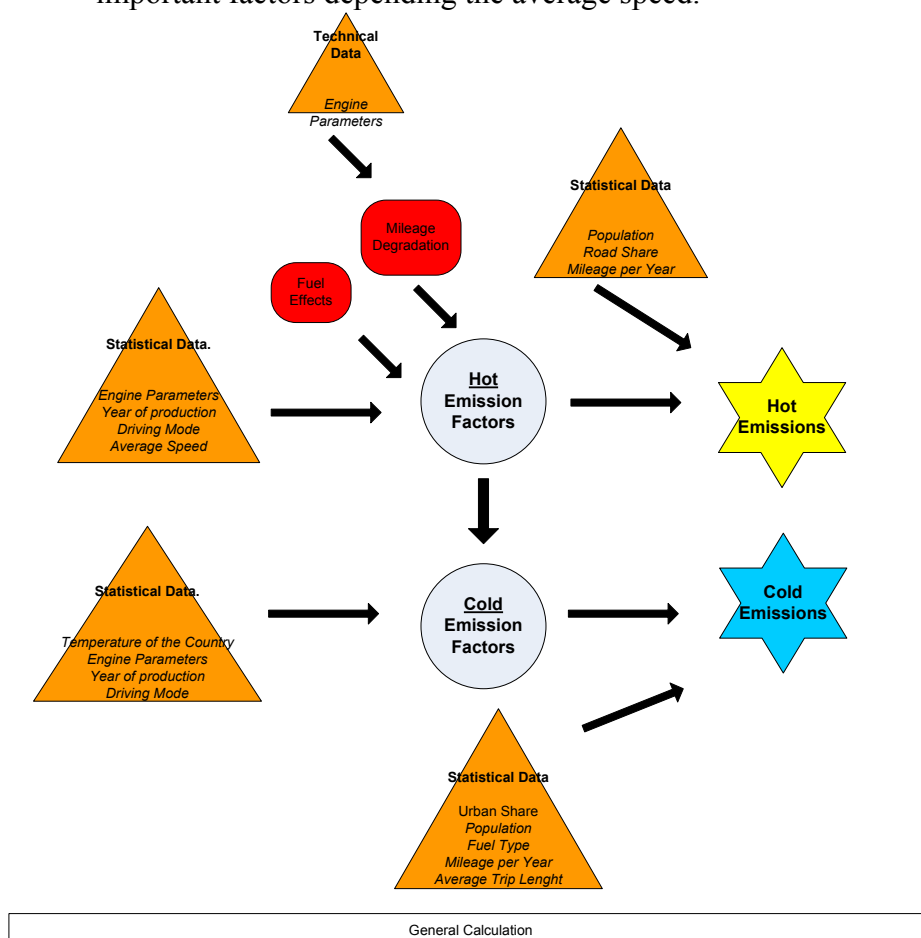


4. **Cold Emissions Calculation:** This is the number that represents the pollution produced in one year by the vehicles of one country when they are not in a hot state. For the calculation will be used:
- a. Urban Share: Percent of Km made in Urban roads
  - b. Mileage per Year: Km made in the simulation year.
  - c. Population: Number of vehicles of every type that are active in the country.
  - d. Fuel Type: Affects to the pollution produced.
  - e. Average Trip Length: 12.3 Km is taken as the standard average trip length. As more trips are made with fewer miles per year means that the vehicles will be more time in a cold state, so, it is taken in account.



These basic calculations are good approximations to the real values, but, it is possible get better approximations taking account some other factors that have influence in the production of pollutants. In this prototype are two of these improvements implemented:

1. **Fuel Effect Calculation:** In order to calculate the Hot Emission Factors with more precision, the effect that the fuel makes in the emission is very important. That's why it is calculated and modifies the Hot Emission Factor values. In this way, the Fuel Effects are dependant of:
  - a. Fuel Type: Dependant on Diesel or Gasoline type
  - b. Post EURO I: The engines posterior to EURO I had better response to the fuel pollutants.
2. **Mileage Degradation Calculation:** Also the Hot Emission Factors are influenced by the state of the car. Intuitively we can imagine that a new brand car will produce less pollution than a car 25 years old. This appreciation is taken in account. To calculate the degradation are taken some parameters:
  - a. Mileage of the vehicle: The real parameter that matters is the mileage in front of the age of the car. A car very old can be in better condition that a newer if the new made more Km.
  - b. Speed in Different Roads: The speed in the different roads is important since many Km made in a Highway but slowly have a different influence than made at top speed for example.
  - c. Factors about speed ranges: Empirically are some values taken that are important factors depending the average speed.



### 3. Detailed prototype requirements analysis

The main objective of this project is to develop a first prototype for COPERT 4. The final COPERT 4 will be developed with successive refinements of this prototype. So, although these refinements are out of range of this project, the first developed prototype must be as much as possible a good basis for developing the next prototype.

#### 3.1 Functional Requirements

The prototype must perform these following calculations, in the way that are specified in the methodology [2].

1. Hot emission factors calculation.
2. Cold emission factors calculation.
3. Fuel Effect calculation.
4. Mileage Degradation calculation.
5. Hot emissions calculation.
6. Cold emissions calculation.
7. All emissions calculation.

All these calculations must be made for all vehicle technologies, sectors and subsectors selected by the user. This selection must be made one by one vehicle type.

The user must be able to see all the results from the calculation and all the intermediate data stored in the system used to calculate these values. Every calculation result will be in different screen.

Must exist an easy way to introduce all the data needed to perform these calculations. A way similar to old versions of COPERT is expected, graphical application using windows system.

The functionality of calculation for several countries and several years must be possible for the user. The user must be able to select different countries for different scenarios, using the data related to this country like temperature, fleet population, etc... happening the same with the year related data. The user will be able then to change the current country for calculations.

For real data, obtained by empirical methods, the user must have a way to introduce its own values for the calculation without override the real ones, having also the possibility to use some real data values mixed with some own values. These own values will be also stored with the other data in order to be used in posterior calculations saving the user of introduce them again.

Also the user must be able to Add/Delete vehicle technologies, sectors or subsectors to the system for more specific calculations. For existing vehicle technologies, sectors and subsectors, specified by the standard normative the user must be able to modify their characteristics.

All the data calculated must be stored in a permanent system like a Database or any other system with the same functionalities. The prototype must retrieve, on demand on the user, information stored in previous calculations. In the same way, the user must be able to store calculations performed by the prototype in a permanent system for later use.

The prototype must include help section for the user with topics related to the use of the software.

## 3.2 Non-functional requirements

### 3.2.1 General purpose requirements

The way that the prototype must save the data calculated must be in a one single file, making easier the interchange of data between users.

The prototype must contain the data needed to make the calculations for one scenario simulation consisting in one year in Greece.

The language used in the Graphical User Interface will be English.

### 3.2.2 Language and coding requirements

The prototype code must be as much as possible using Object Oriented Programming languages and technologies.

In order to make possible future refinements of the prototype, the code must be written following the coding style described in this document in the respective section.

All variables, functions, labels, comments, etc..., must be written in English.

The coding must give more priority to calculation efficiency in front of other qualities like upgradeability or connectivity until reach the appropriate level of low timing in the calculations.

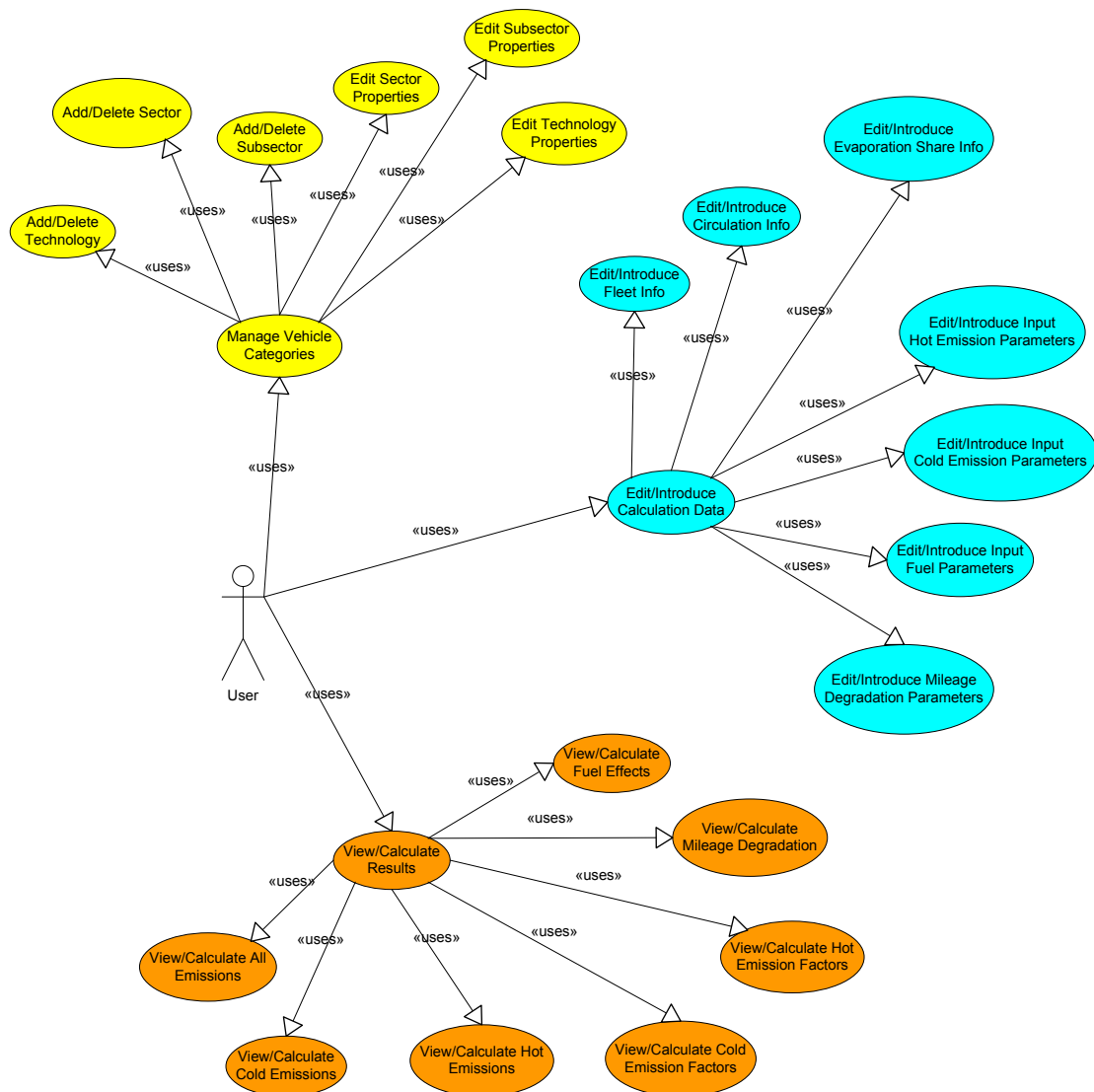
### 3.2.3 System requirements

The prototype must be able to work in good conditions in a personal computer with the following characteristics or better:

- Athlon AMD XP1800+, or quicker processor.
- 256 Mb RAM memory
- 300Mb Hard Drive free space for software. The space needs increase depending on the user data storage.
- CDROM x32
- 1024 x 768 screen
- Windows keyboard 88-/99 keys.
- Mouse or similar pointing system.
- Windows XP system installed

## 4. Diagram and Specification.

### 4.1 Diagram



## 4.2 Specification

<b>Use Case:</b> Add/Delete Technology
--

<b>Information in screen:</b>
-------------------------------

<b>Output:</b> List of current technologies
---

<b>Input:</b> Technology Name, Technology Order, Technology Euro Number
---

<b>Description:</b> Screen with the fields in blank to be filled by the user in order to add a new technology and its properties or the list of technologies on system. To delete one, the technology must be selected. On accept, the system will ask confirmation to the user.
--

<b>Use Case:</b> Add/Delete Sector
------------------------------------

<b>Information in screen:</b>
-------------------------------

<b>Output:</b> List of current sectors
--

<b>Input:</b> Sector Name, Sector Order
---

<b>Description:</b> Screen with the fields in blank to be filled by the user in order to add a new Sector and its properties or the list of technologies on system. To delete one, the Sector must be selected. On accept, the system will ask confirmation to the user.
--

<b>Use Case:</b> Add/Delete SubSector
---------------------------------------

<b>Information in screen:</b>
-------------------------------

<b>Output:</b> List of current subsectors
---

<b>Input:</b> SubSector Name, SubSector Order, SubSector Fuel Type
--

<b>Description:</b> Screen with the fields in blank to be filled by the user in order to add a new SubSector and its properties or the list of technologies on system. To delete one, the SubSector must be selected. On accept, the system will ask confirmation to the user.
--

<b>Use Case:</b> Edit Sector Properties
---

<b>Information in screen:</b>
-------------------------------

<b>Output:</b> List of current sectors
--

<b>Input:</b> Sector Name, Sector Order
---

<b>Description:</b> Screen with the list of sectors in the system. After selecting the sector, the user can edit the properties of the sector in the specified fields. After accepting the changes the system asks for confirmation.
--

**Use Case: Edit SubSector Properties****Information in screen:**

**Output:** List of current subsectors

**Input:** SubSector Name, SubSector Order, SubSector Fuel Type

**Description:** Screen with the list of subsectors in the system. After selecting the subsector, the user can edit the properties of the subsector in the specified fields. After accepting the changes the system asks for confirmation.

**Use Case: Edit Technology Properties****Information in screen:**

**Output:** List of current technologies

**Input:** Technology Name, Technology Order, Technology Euro Number

**Description:** Screen with the list of technologies in the system. After selecting the technology, the user can edit the properties of the technology in the specified fields. After accepting the changes the system asks for confirmation.

**Use Case: Edit/Introduce Fleet Info****Information in screen:**

**Input/Output:** List of vehicles(Sector, Subsector and Technology)

**Input:** Population, Annual Mileage, Fuel Injection and Evaporation Control.

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Fleet info, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.  
In any moment the user can create or delete new vehicles in the fleet, selecting the proper Sector, Subsector and Technology. On create will raise an error if exists another vehicle with the same Sector,Subsector and Technology.

**Use Case: Edit/Introduce Circulation Info****Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology)

**Input:** Urban Speed, Rural Speed, Highway Speed, Driving Share Rural, Driving Share Urban, Driving Share Highway

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Circulation info, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.

**Use Case: Edit/Introduce Evaporation Share Info****Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology)

**Input:** Urban Speed, Rural Speed, Highway Speed, Driving Share Rural, Driving Share Urban, Driving Share Highway

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Evaporation Share info, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.

These fields will contain the values for the percentage share in Urban, Rural and Highway roads.

On finishing the introduction or modification of data the system will check the integrity controlling sum of all the values for every vehicle are not bigger than 100%

**Use Case: Edit/Introduce Input Hot Emission Parameters****Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology)

**Input:** For each Vehicle Range Speed(Order, Alpha, Beta, Gamma, Delta, Epsilon, Zeta, Eta, Theta, Reduction Factor, Low Speed Limit, Top Speed Limit, Include Top Speed, Include Low Speed)

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Hot Emission Parameters, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.

These fields will contain the values for all the parameters needed to calculate the Hot Emission Factors. The values will be separated using the Driving Mode value, so, by Urban, Rural or Highway driving modes.

On accept the system will check the integrity of the values, avoiding having speed ranges with wrong values.

**Use Case: Edit/Introduce Input Cold Emission Parameters****Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology)

**Input:** For each Vehicle Range Speed and Month(Order,Month, A, B, C, Low Speed Limit, Top Speed Limit, Include Top Speed Limit, Low Temperature Limit, Top Temperature Limit, Include Low Temperature Limit)

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Cold Emission Parameters, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.

These fields will contain the values for all the parameters needed to calculate the Cold Emission Factors. The values will be separated using the Driving Mode value, so, by Urban, Rural or Highway driving modes.

On accept the system will check the integrity of the values, avoiding having speed



ranges or temperature ranges with wrong values.

**Use Case: Edit Input Fuel Parameters**

**Information in screen:**

**Input/Output:** Gasoline E100, Gasoline E150, Gasoline Aromatics, Gasoline Sulphur, Gasoline Oxygenates, Gasoline Olefins, Gasoline Benzane, Diesel Density, Diesel PCA, Diesel CN, Diesel T95, Diesel Sulphur

**Description:** Screen with all the values related to the different Fuel Types, depending on the year.

These fields will be able to be changed and after accepting the changes a correctness check will be performed.

**Use Case: Edit/Introduce Mileage Degradation Parameters**

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology) and Pollutants

**Input:** For each Vehicle and Pollutant (Am 19Km, Am 63Km, Bm 19Km, Bm 63Km)

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Mileage Degradation Parameters, the fields with the already introduced data for these vehicles or blank in case of nothing introduced yet.

These fields will contain the values for all the parameters needed to calculate the Mileage Degradation Factors.

On accept the system will check the integrity of the values, avoiding having erroneous values.

**Use Case: Calculate/View Fuel Effects**

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Fuel Effect Values

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Fuel Effects, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition, the Fuel Effects will be calculated and this screen will be updated.

**Use Case: View/Calculate Mileage Degradation**

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Urban Value, Rural Value, Highway value

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Mileage Degradation Calculation, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition, the Mileage Degradation will be calculated and this screen will be updated.

**Use Case:** View/Calculate Hot Emission Factors

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Urban Values, Rural Values, Highway values for Base Type and Calculated Type

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Hot Emission Factors Calculation, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition, Hot Emission Factors will be calculated and this screen will be updated.

**Use Case:** View/Calculate Cold Emission Factors

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants, Months and Urban Value, Rural Value, Highway value.

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Cold Emission Factors Calculation, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition Cold Emission Factors will be calculated and this screen will be updated.

**Use Case:** View/Calculate Hot Emissions

**Information in screen:**

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Urban Value, Rural Value, Highway value.

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Hot Emissions Calculation, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition Hot Emissions will be calculated and this screen will be updated.

**Use Case:** View/Calculate Cold Emissions

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Urban Value, Rural Value.

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Cold Emissions Calculation, the fields with the already calculated data for these vehicles or blank in

case of nothing calculated yet.

On response to user petition Cold Emissions will be calculated and this screen will be updated.

**Use Case:** View/Calculate All Emissions

**Output:** List of vehicles(Sector, Subsector and Technology),Pollutants and Urban Value, Rural Value, Highway Value.

**Description:** Screen with the list of vehicles in the system, showing the Sector, Subsector and Technology. For each vehicle, related to the Cold Emissions Calculation and Hot Emissions Calculation and the Sum of them, the fields with the already calculated data for these vehicles or blank in case of nothing calculated yet.

On response to user petition Cold Emissions will be calculated and this screen will be updated.

## 5. Kernel and Database programming

### 5.1 Database Technologies

For this stage prototype would be possible to think on using a normal file to store all the data that is going to be used. In order to fulfill all the functionalities expected, the quantity of data stored would not be very large, so, in this case affordable by a normal binary file. If this solution was revoked was because of three main reasons.

- a. The commodity of the SQL Query system: Using a simple file to store the data, we had to have all the data in memory, stored probably in matrix, and since all the calculations are made in a sequential way, would produce a chaotic code. The calculations are enough complicated to made them more complicated because of the code.
- b. The easy portability for future uses of the data: A possible future step for this data will be a Web Programmed interface, making accessible the data over the world. Obviously having this future situation in mind, the normal file way is not a good choice since from all the Web technologies the access to Databases is very easy, contrary to file access.
- c. Import system from other applications: If a file is used, the idea of exporting data from other applications like, for instance, MS Excel, is not a straight job. Having a Database, with other applications, if the Database has ODBC driver makes easier the way to introduce data.
- d. Future enlargement of the data: In the future is possible to have a system with stored data for many years calculation, many countries at the same time, the file would be a limitation.

So, after these reasons, the way of storing the data seems to have a clear manner in shape of a Database. The first look was to MS SQL Server or Oracle. These two Databases, with very good performance would be very useful and really good. Unfortunately we find the first important problem, the economical. If COPERT would use a pay based Database, all the users whose would use it must have the proper licence. This idea is not good when COPERT is supposed to be for all type of budgets.

Discarding the pay based Databases we can focus on others systems like MySQL or PostgreSQL. These two systems, at first sight were the correct choice, but lately we discovered that to force the user to install any of these Databases could be problematic. That is how we found finally the solution for our necessities.

Finally MS Access was chosen in order to store all the persistent data for COPERT, for several reasons:

- a. SQL Queries: Is true that is not the best SQL based system, but enough to perform the queries that COPERT needs.
- b. ODBC Compliant: Offers a very easy way to introduce data. Using ODBC, from other MS Office programs, etc...
- c. Data Portability: In case of need another Database, the portability of the data is very easy and at any case can be performed using MS Access programming code to transfer to XML, a binary file or any other way.

- d. Licence: If it is true that to program in MS Access requires a licence, to distribute a MS Access file does not require anything. And, if the user would like to open the database by itself and change something, is expectable that the user will have MS Office licence.
- e. No server needed: The database is working by itself, without the need of installing new servers or additional systems. Also for information exchange between users is very useful since only is needed to send one file.

Another possibility that was thought was to have a SQL Server or Oracle system running in a central server and give access to the users that wanted to use COPERT. This solution was also revoked because many institutions or countries have no access to Internet all the time and have connection problems. MS Access being offline or with basic online features for intranets seems to be enough.

## 5.2 Kernel Technologies

The decision about which technology to use by the kernel of the prototype was since the beginning an easier question. The first idea was to program it on Java, it is a free technology, easy for the programmer and with good connectivity with Databases.

But, since this is only a prototype, and future versions will be made taking this prototype as basis, it was a very important point to think who was going to continue with the development. This made to decide finally to program in Visual Basic. The future programmers of COPERT were going to be mechanical engineers, all of them very used to work with Visual Basic and MsOffice, but unfortunately not so used with Java technologies.

So, in this way we decided the Visual Basic programming, also good on connectivity with databases (better even with Microsoft Databases which was the case).

Currently, the developing in Visual Basic 96 is getting down because of the increasing programmer community working with its new version Visual Basic .NET. Also, this new version had some important features that made it the best choice:

- a. New Middleware for connection with databases, ADO.NET
- b. Possible use of other languages, since in .NET framework is possible to compile software using several programming languages together. This could be useful if in future versions a new language can be used by any expert.
- c. Future web application: The .NET technology makes easier to transfer an application not internet focused to a whole web based application. The future of COPERT could be focused on this direction and could be useful in the future.
- d. Possible link with MS Office: The users of COPERT are used to work with MS Excel for example. To make an import procedure from Excel is easier with Visual Basic .NET

At last, following all these reasons made the Visual Basic .NET the chosen option. Unfortunately, the ADO.NET at last was not used since had very bad performance results accessing to MS Access databases, something really unexpected. A completely reprogramming of the access to the database had to be done to make the access quicker.

At last was used DAO technology, which archaic technology but very efficient. The newer versions of DAO (ADO or ADO.NET) incorporated new interesting features to increase the functionalities but where not useful at all for the COPERT calculations.

Visual Studio .NET is not a free technology. A license is required, but fortunately the problem of the license was inexistent in this case because the University had already the license of Visual Studio .NET.

The last decision to take was where were going to be placed all the queries. It was possible in two ways:

- a. Queries in the kernel code: This way would separate completely the data layer from the calculation layer. Very useful for future upgrades of the system. Unfortunately the performance was not the desired.
- b. Queries in the database: This option, the chosen finally, was taken to improve the performance. The fact of having the query precompiled in the database made it much quicker.

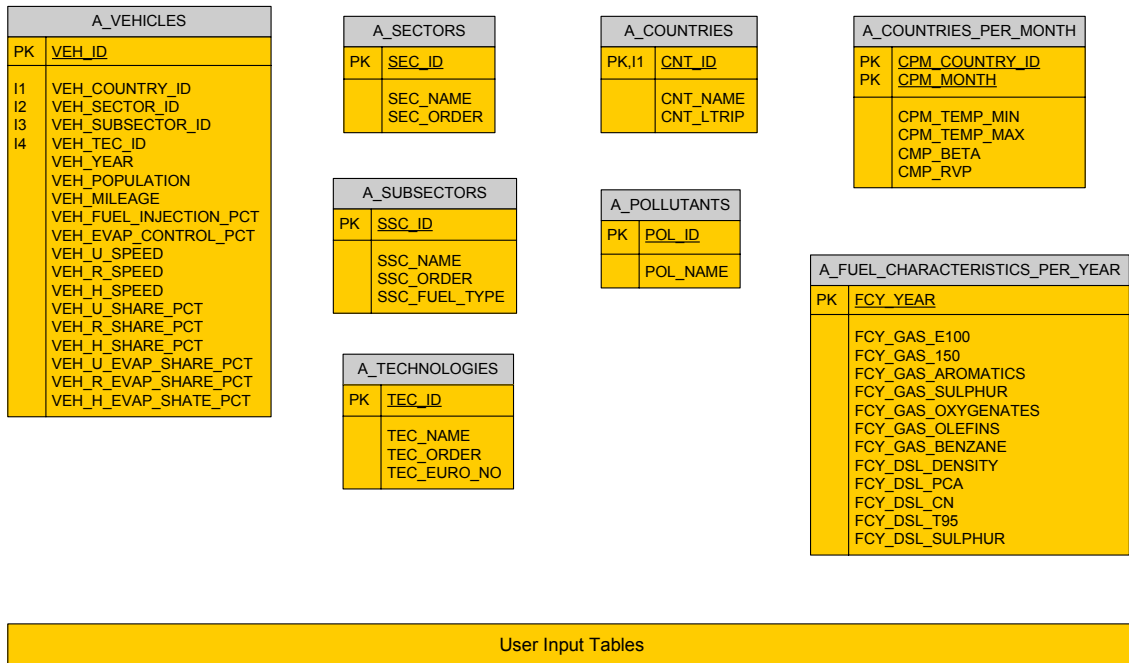
The bad point of the query in the database was that anybody could open the database by its own and change any query used by the system. This could cause system crashes not really desirable and probably, with malicious minds, destroy some important data. To avoid this problem the queries used by the system are under administrator rights making impossible to modify by the user.

## 6. Database design and construction

The design of the Database is according to the implementation. Unfortunately, the MS Access is a very small system with not many features. In some cases, in order to implement correctly the data system would have been useful some foreign key system or some triggering system. With these restrictions these functionalities had to be implemented in the kernel. Anyway in this section are explained the major constrains for the data even is not implemented properly in the database.

In the following diagrams can be appreciated that the database is divided in four sectors. The prefix (A\_, B\_, C\_ or D\_) also shows de distinction.

- a. User Input Tables: Tables used to store the classical variable data that the user can introduce. Does not mean that the user has no access to insert or update in the other tables, but these tables are the ones that suffer this action more commonly.



- b. COPERT Empirical Parameters Tables: Tables used to store all the parameters and constants that COPERT has as default values. All the values contained in this tables are values obtained in real environments, the user can change them although is not very recommendable.
- c. Intermediate Calculation Tables: Tables with the Emission Factors calculated. These values do not change significantly if the data stored in the b. section is not changed.
- d. Final Calculation Tables: Tables with the results of Emissions.

B_MILEAGE_DEGRADATION_FACTORS	
PK	MDF_VEHICLE_ID
PK,I1	MDF_POLLUTANT_ID
	MDF_AM_19KM MDF_BM_19KM MDF_AM_63KM MDF_BM_63KM MDF_U_VALUE MDF_R_VALUE MDF_H_VALUE

B_REDUCTION_PARAMETERS	
PK,I1	REP_TECH_ID
PK,I2	REP_POLLUTANT_ID
	REP_VALUE

B_FUEL_EFFECTS	
PK,I2	FEF_VEHICLE_ID
PK,I1	FEF_POLLUTANT_ID
	FEF_VALUE

B_HOT_EMISS_PARAMETERS	
PK	HEP_VEH_ID
PK	HEP_DRIVING_MODE
PK	HEP_POLLUTANT_ID
PK	HEP_SPEED_RANGE_ORDER
	HEP_ALPHA HEP_BETA HEP_GAMMA HEP_DELTA HEP_EPSILON HEP_ZITA HEP_ITA HEP_THITA HEP_RF HEP_LOW_SPEED_LIMIT HEP_TOP_SPEED_LIMIT HEP_INCLUDE_TOP_SPEED_LIMIT HEP_INCLUDE_LOW_SPEED_LIMIT HEP_VALUES_K

B_COLD_EMISS_PARAMETERS	
PK,I2	CEP_VEH_ID
PK,I1	CEP_POLLUTANT_ID
PK	CEP_SPEED_RANGE_ORDER
PK	CEP_MONTH
	CEP_A CEP_B CEP_C CEP_LOW_SPEED_LIMIT CEP_TOP_SPEED_LIMIT CEP_INCLUDE_TOP_SPEED_LIMIT CEP_LOW_TEMP_LIMIT CEP_TOP_TEMP_LIMIT CEP_INCLUDE_LOW_TEMP_LIMIT CEP_INCLUDE_TOP_TEMP_LIMIT CEP_VALUES_K

COPERT Empirical parameters.

C_HOT_EMISS_FACTORS	
PK	HEF_VEHICLE_ID
PK	HEF_POLLUTANT_ID
	HEF_U_VALUE_CALC HEF_R_VALUE_CALC HEF_H_VALUE_CALC HEF_U_VALUE_COR HEF_R_VALUE_COR HEF_H_VALUE_COR HEF_U_VALUE_USR HEF_R_VALUE_USR HEF_H_VALUE_USR HEF_U_VALUE_CALC_BASE HEF_R_VALUE_CALC_BASE HEF_H_VALUE_CALC_BASE HEF_U_VALUE_K HEF_R_VALUE_K HEF_H_VALUE_K

C_COLD_EMISS_FACTORS_MONTHLY	
PK	CEF_VEHICLE_ID
PK	CEF_POLLUTANT_ID
PK	CEF_MONTH
	CEF_U_VALUE_CALC CEF_U_VALUE_COR CEF_U_VALUE_USR CEF_U_VALUE_K

Intermediate Calculations

D_HOT_EMISSIONS	
PK,I2	HEM_VEHICLE_ID
PK,I1	HEM_POLLUTANT_ID
	HEM_U_EMISS HEM_R_EMISS HEM_H_EMISS

D_COLD_EMISSIONS	
PK,I2	CEM_VEHICLE_ID
PK,I1	CEM_POLLUTANT_ID
	CEM_U_EMISS CEM_R_EMISS

Final Calculations



## 6.1 Table Explanation

### 6.1.1. A\_COUNTRIES

This table contains the information for every country inside the system. The average Length trip is in this table because every country can have its own value although generally is used the agreed one 12.3 Km

Column Name	Data Type	Description
<u>CNT_ID</u>	Long	Id for the country
CNT_NAME	Text	Name of the country
CNT_LTRIP	Double	Average Length trip for the country

### 6.1.2. A\_COUNTRIES\_PER\_MONTH

This table contains the attributes monthly dependant for every country. Although a good choice had been to store the monthly data all in one row, for simplification of the code was created a new row for every month.

The Beta and RVP parameters are used to calculate the Cold Emission Factors.

Table Restrictions: CPM\_TEMP\_MIN cannot be greater than CPM\_TEMP\_MAX.

Column Name	Data Type	Description
<u>CPM_COUNTRY_ID</u>	Long	Id for the country
<u>CPM_MONTH</u>	Integer	Month number
CPM_TEMP_MIN	Double	Minimum temperature for this month
CPM_TEMP_MAX	Double	Maximum temperature for this month
CMP_BETA	Double	Beta parameter for this month
CMP_RVP	Double	RVP parameter for this month

### 6.1.3. A\_FUEL\_CHARACTERISTICS\_PER\_YEAR

In the system needs to be stored some fuel characteristics for three special year: 1996, 2000, 2005. This fuel characteristics are used to calculate the fuel effects in the Hot Emission Factors.

Column Name	Data Type	Description
<u>FCY_YEAR</u>	Integer	Year (1996,2000 or 2005)
FCY_GAS_E100	Double	Gasoline E100 value
FCY_GAS_150	Double	Gasoline E150 value

FCY_GAS_AROMATICS	Double	Gasoline Aromatics value
FCY_GAS_SULPHUR	Double	Gasoline Sulphur value
FCY_GAS_OXYGENATES	Double	Gasoline Oxygenates value
FCY_GAS_OLEFINS	Double	Gasoline Olefins value
FCY_GAS_BENZANE	Double	Gasoline Benzane value
FCY_DSL_DENSITY	Double	Diesel Density value
FCY_DSL_PCA	Double	Diesel PCA value
FCY_DSL_CN	Double	Diesel CN value
FCY_DSL_T95	Double	Diesel T95 value
FCY_DSL_SULPHUR	Double	Diesel Sulphur value

#### 6.1.4. A\_POLLUTANTS

This table stores the basic information for the pollutants.

Column Name	Data Type	Description
<u>POL_ID</u>	Long	Identifier for the pollutant
POL_NAME	Text	Technical name of the pollutant. ('CO', 'NOx', etc...)

#### 6.1.5. A\_SECTORS

The sectors are the general categories to classify the vehicles. Are the most general, including descriptions as 'Buses' or 'Motorcycles'.

This table stores this sectors information. Every sector has a popular name and a sequence order. This order is used because all the sectors are show always to the user in the same order.

Column Name	Data Type	Description
<u>SEC_ID</u>	Long	Identifier for the Sector
SEC_NAME	Text	Popular name of the sector ('Passanger Car', 'Light Duty Vehicle', etc...)
SEC_ORDER	Integer	Order to be shown to the user. Always positive.

#### 6.1.6. A\_SUBSECTORS

The subsectors are more specific classification for the vehicles. Are part of this level categories like 'Gasoline 1,4 - 2,0 l' or '2-Stroke'.

This table stores this subsectors information. Every subsector has a popular name and a sequence order. This order is used because all the subsectors are show always to the user in the same order. The information about the Fuel type used by these subsectors is stored in this table as well.

Column Name	Data Type	Description
<u>SSC_ID</u>	Long	Identifier for the Subsector
SSC_NAME	Text	Popular name of the subsector ('Coaches', 'Gasoline 1,4 - 2,0 l', etc...)
SSC_ORDER	Integer	Order to be shown to the user. Always positive.
SSC_FUEL_TYPE	Text	'GAS' for Gasoline, 'DSL' for Diesel, 'LPG' for Liquid Petroleum Gas.

#### 6.1.7. A\_TECHNOLOGIES

The technologies are the third level of categorization and the most specific. These categories are related to the technology of the engine and are based on standards. Some of these categories are '91/441/EEC' or 'Open Loop'.

This table stores the technologies information. Every technology has the label or popular name, the order and the technology which represents relative to the EURO I. For many calculations is important to know if a vehicle uses a technology posterior to the technology EURO I.

Column Name	Data Type	Description
<u>TEC_ID</u>	Long	Identifier for the technology
TEC_NAME	Text	Name for the technology
TEC_ORDER	Integer	Order to follow when is shown the list of technologies to the user.
TEC_EURO_NO	Integer	0 if is a Pre or EURO I Technology x for Technologies EURO x

#### 6.1.8. A\_VEHICLES

This table contains the information about all the vehicles types of all countries. The VEH\_ID is considered Primary Key, as an alternative key exists: (VEH\_COUNTRY\_ID, VEH\_SECTOR\_ID, VEH\_SUBSECTOR\_ID, VEH\_TEC\_ID, VEH\_YEAR)

Also with every vehicle all the data related to it in the country like the population, average speed in urban ways, etc...

Column Name	Data Type	Description
<u>VEH_ID</u>	Long	Identifier for the vehicle
VEH_COUNTRY_ID	Long	Identifier of the country which it belongs

VEH_SECTOR_ID	Long	Identifier of the sector of the vehicle
VEH_SUBSECTOR_ID	Long	Identifier of the subsector of the vehicle
VEH_TEC_ID	Long	Identifier of the technology of the vehicle
VEH_YEAR	Integer	Year of the vehicle data
VEH_POPULATION	Double	Population of this type of vehicle in the given year
VEH_MILEAGE	Double	Mileage of this type of vehicle in the given year
VEH_FUEL_INJECTION_PCT	Double	Fuel Injection Percentage of this type of vehicle in the given year
VEH_EVAP_CONTROL_PCT	Double	Evaporation Control Percentage of this type of vehicle in the given year
VEH_U_SPEED	Double	Average Urban Speed of this type of vehicle in the given year
VEH_R_SPEED	Double	Average Rural Speed of this type of vehicle in the given year
VEH_H_SPEED	Double	Average Highway Speed of this type of vehicle in the given year
VEH_U_SHARE_PCT	Double	Urban Share Percentage of this type of vehicle in the given year
VEH_R_SHARE_PCT	Double	Rural Share Percentage of this type of vehicle in the given year
VEH_H_SHARE_PCT	Double	Highway Share Percentage of this type of vehicle in the given year
VEH_U_EVAP_SHARE_PCT	Double	Urban Evaporation Share Percentage of this type of vehicle in the given year
VEH_R_EVAP_SHARE_PCT	Double	Rural Evaporation Share Percentage of this type of vehicle in the given year
VEH_H_EVAP_SHATE_PCT	Double	Highway Evaporation Share Percentage of this type of vehicle in the given year

#### **6.1.9. B\_COLD\_EMISS\_PARAMETERS**

This table contains the parameters needed to calculate the Cold Emission Factors for every vehicle. Every factor parameter has a speed range, for instance, if the vehicle runs faster than 30 Km/h will have different parameters for the calculation than if the vehicle runs faster than 100 Km/h. In this way every parameter has several ranges of calculation. Every parameter is also related to a month since the parameters change

depending the month (ambient temperature). Also the temperature range is used to chose which parameters are used for the calculation.

Vehicle restriction: No speed range with greater order can specify a speed range of lower speeds than another speed range with lower order.

Column Name	Data Type	Description
<u>CEP_VEH_ID</u>	Long	Identification of the vehicle
<u>CEP_POLLUTANT_ID</u>	Single	Identification of the pollutant
<u>CEP_SPEED_RANGE_ORDER</u>	Integer	Order of the speed range. Always bigger than 0.
<u>CEP_MONTH</u>	Integer	Month which can be applied the parameter.
CEP_A	Double	A parameter
CEP_B	Double	B Parameter
CEP_C	Double	C Parameter
CEP_LOW_SPEED_LIMIT	Double	Low Bound of the speed range
CEP_TOP_SPEED_LIMIT	Double	Top Bound of the speed range
CEP_INCLUDE_TOP_SPEED_LIMIT	Boolean	True if the upper bound is included in the range
CEP_LOW_TEMP_LIMIT	Double	Low Bound of the temperature range.
CEP_TOP_TEMP_LIMIT	Double	Top Bound of the temperature range
CEP_INCLUDE_LOW_TEMP_LIMIT	Boolean	True if the lower bound of the temperature is included in the range
CEP_INCLUDE_TOP_TEMP_LIMIT	Boolean	True if the upper bound of the temperature is included in the range
CEP_VALUES_K	Boolean	Not used in this version

#### 6.1.10. B\_FUEL\_EFFECTS

This table contains all the fuel effects calculated values for all the pollutants and vehicles

Column Name	Data Type	Description
<u>FEF_VEHICLE_ID</u>	Long	Identifier of the vehicle
<u>FEF_POLLUTANT_ID</u>	Long	Identifier of the pollutant
FEF_VALUE	Double	Fuel Effect Value

### 6.1.11. B\_HOT\_EMISS\_PARAMETERS

This table contains the parameters needed to calculate the Hot Emission Factors for every vehicle. Every factor parameter has a speed range, for instance, if the vehicle runs faster than 30 Km/h will have different parameters for the calculation than if the vehicle runs faster than 100 Km/h. In this way every parameter has several ranges of calculation.

Vehicle restriction: No speed range with greater order can specify a speed range of lower speeds than another speed range with lower order.

Column Name	Data Type	Description
<u>HEP_VEH_ID</u>	Long	Identifier of the vehicle
<u>HEP_DRIVING_MODE</u>	Text	U for Urban Mode, R for Rural Mode, H for Highway Mode.
<u>HEP_POLLUTANT_ID</u>	Long	Identifier of the pollutant
<u>HEP_SPEED_RANGE_ORDER</u>	Integer	Order of the speed range. Always greater than 0.
HEP_ALPHA	Double	Alpha parameter
HEP_BETA	Double	Beta parameter
HEP_GAMMA	Double	Gamma parameter
HEP_DELTA	Double	Delta parameter
HEP_EPSILON	Double	Epsilon parameter
HEP_ZITA	Double	Zita parameter
HEP_ITA	Double	Ita parameter
HEP_THITA	Double	Thita parameter
HEP_RF	Double	Reduction Factor relative to the EURO I vehicle.
HEP_LOW_SPEED_LIMIT	Double	Low Bound for the speed range
HEP_TOP_SPEED_LIMIT	Double	Top Bound for the speed range
HEP_INCLUDE_TOP_SPEED_LIMIT	Boolean	True if the upper bound is included in the range
HEP_INCLUDE_LOW_SPEED_LIMIT	Boolean	True if the lower bound is included in the range
HEP_VALUES_K	Boolean	Not used in this version

#### 6.1.12. B\_MILEAGE\_DEGRADATION\_FACTORS

This table contains the parameters to calculate the Mileage Degradation Factors. Also the factors will be stored in this table once calculated.

Column Name	Data Type	Description
<u>MDF_VEHICLE_ID</u>	Long	Identifier of the vehicle
<u>MDF_POLLUTANT_ID</u>	Long	Identifier of the pollutant
MDF_AM_19KM	Double	Parameter Am for speed lower than 19 Km/h
MDF_BM_19KM	Double	Parameter Bm for speed lower than 19 Km/h
MDF_AM_63KM	Double	Parameter Am for speed greater than 63 Km/h
MDF_BM_63KM	Double	Parameter Bm for speed greater than 63 Km/h
MDF_U_VALUE	Double	Mileage Degradation value for Urban Roads
MDF_R_VALUE	Double	Mileage Degradation value for Rural Roads
MDF_H_VALUE	Double	Mileage Degradation value for Highway Roads

#### 6.1.13. B\_REDUCTION\_PARAMETERS

This table stores some constant reduction parameters used to calculate the Cold Emission Factors.

Column Name	Data Type	Description
<u>REP_TECH_ID</u>	Long	Identifier of the technology
<u>REP_POLLUTANT_ID</u>	Long	Identifier of the pollutant
REP_VALUE	Double	Value for the reduction parameter

#### 6.1.14. C\_COLD\_EMISS\_FACTORS\_MONTHLY

This is the table that stores the calculated cold emission factors for all the vehicles. Every cold emission factor is depending as well of the month. Also exist a corrected value, this is the calculated value with some modifications for other calculations. The user can set its own value in the User field. This value is taken in account if the Value\_K field is set to true.

Only the values for Urban Road are stored since it is supposed that the cold emissions are made always on Urban Roads.

Column Name	Data Type	Description
<u>CEF_VEHICLE_ID</u>	Long	Identifier of the vehicle
<u>CEF_POLLUTANT_ID</u>	Long	Identifier of the vehicle

<u>CEF_MONTH</u>	Integer	Month number
CEF_U_VALUE_CALC	Double	Urban Calculated Value
CEF_U_VALUE_COR	Double	Urban Corrected Value
CEF_U_VALUE_USR	Double	Urban User Given Value
CEF_U_VALUE_K	Boolean	True if user value is used.

#### 6.1.15. C\_HOT\_EMISS\_FACTORS

This table stores all the calculated hot emission factors data. Every vehicle has a collection of values associated also to every pollutant. This values are stored in this table as it follows.

Column Name	Data Type	Description
<u>HEF_VEHICLE_ID</u>	Long	Identifier of the vehicle
<u>HEF_POLLUTANT_ID</u>	Long	Identifier of the pollutant
HEF_U_VALUE_CALC	Double	Urban calculated value
HEF_R_VALUE_CALC	Double	Rural calculated value
HEF_H_VALUE_CALC	Double	Highway calculated value
HEF_U_VALUE_COR	Double	Urban corrected value
HEF_R_VALUE_COR	Double	Rural corrected value
HEF_H_VALUE_COR	Double	Highway corrected value
HEF_U_VALUE_USR	Double	Urban User value
HEF_R_VALUE_USR	Double	Rural User value
HEF_H_VALUE_USR	Double	Highway User value
HEF_U_VALUE_CALC_BASE	Double	Urban Calculated Base value (without reduction factors)
HEF_R_VALUE_CALC_BASE	Double	Rural Calculated Base value (without reduction factors)
HEF_H_VALUE_CALC_BASE	Double	Highway Calculated Base value (without reduction factors)
HEF_U_VALUE_K	Boolean	True if User value is used for Urban Value
HEF_R_VALUE_K	Boolean	True if User value is used for Rural Value
HEF_H_VALUE_K	Boolean	True if User value is used for Highway Value

#### 6.1.16. D\_COLD\_EMISSIONS

This store keeps the data of cold emissions. For each vehicle and pollutant keeps the Urban and the Rural Emissions. It is possible to have Rural Emissions if the quantity of



emissions exceeds the Urban Share of the vehicle. In this case, the remaining emissions are placed as Rural Emissions.

Column Name	Data Type	Description
<u>CEM_VEHICLE_ID</u>	Long	Identifier of the vehicle
<u>CEM_POLLUTANT_ID</u>	Long	Identifier of the pollutant
CEM_U_EMISS	Double	Urban Cold Emissions value
CEM_R_EMISS	Double	Rural Cold Emissions value

#### 6.1.17. D\_HOT\_EMISSIONS

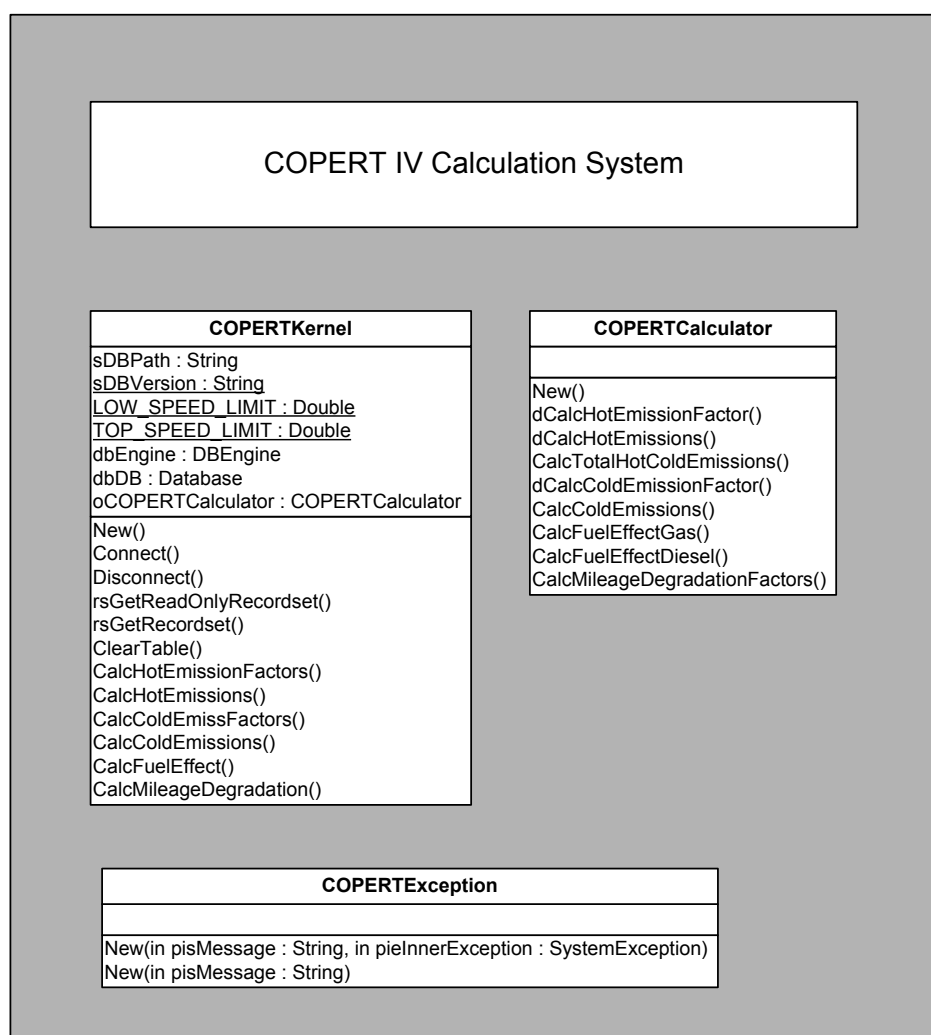
This table stores the Hot Emissions Data. For every vehicle and pollutant the Emissions for Urban, Rural and Highway are kept as it shows.

Column Name	Data Type	Description
<u>HEM_VEHICLE_ID</u>	Long	Identifier for the vehicle
<u>HEM_POLLUTANT_ID</u>	Long	Identifier for the pollutant
HEM_U_EMISS	Double	Urban Hot Emissions value
HEM_R_EMISS	Double	Rural Hot Emissions value
HEM_H_EMISS	Double	Highway Hot Emissions value

## 7. Calculation system design

### 7.1 Class Diagram

This is the main structure of the COPERT 4 prototype calculation system. As it shows there are two main parts: COPERTKernel and COPERTCalculator.



COPERTCalculator encapsulates all the calculations made by prototype. All formulas, constants, etc... are introduced in this module. In this way, from other parts of the code, it is not important to know what really does. Just passing the arguments, encapsulating every function, reduces the spread of responsibilities focusing the calculations in this member.

In the other hand, COPERTKernel is the manager of the data. Is the responsible to connect with the Database and take all the important data to pass it to COPERTCalculator. Later, on every result, it will update the Database.

Another feature important in the COPERTKernel is the way that the operations are divided. The programming is made focusing on the independence of every operation. The user for example, on changing the population of one vehicle type, does not need to calculate again all the steps. Only with the last step, the emissions, can appreciate the changes. This was not that clear in previous versions of COPERT what makes this version more useful avoiding long waiting times.

At the bottom is shown the COPERTException. This class is responsible of the error management. On every error from the database for example, this class overrides the standard error. It is very useful for the error handling from the front end of the application, giving specific error messages to the user.

It is very special that in a project of this size only exist three classes. The truth is that although an Object Oriented Programming language is used, the background is not a Object Oriented Application. If it is true that the Object Orientation is very useful, in this project is not that much since the performance is a big deal. If the system should have an instance for every vehicle that is being processed and every pollutant, etc... not only would have a enormous waste of memory, also would have a very big lost of performance since the use of every instance would be very small in time speaking.

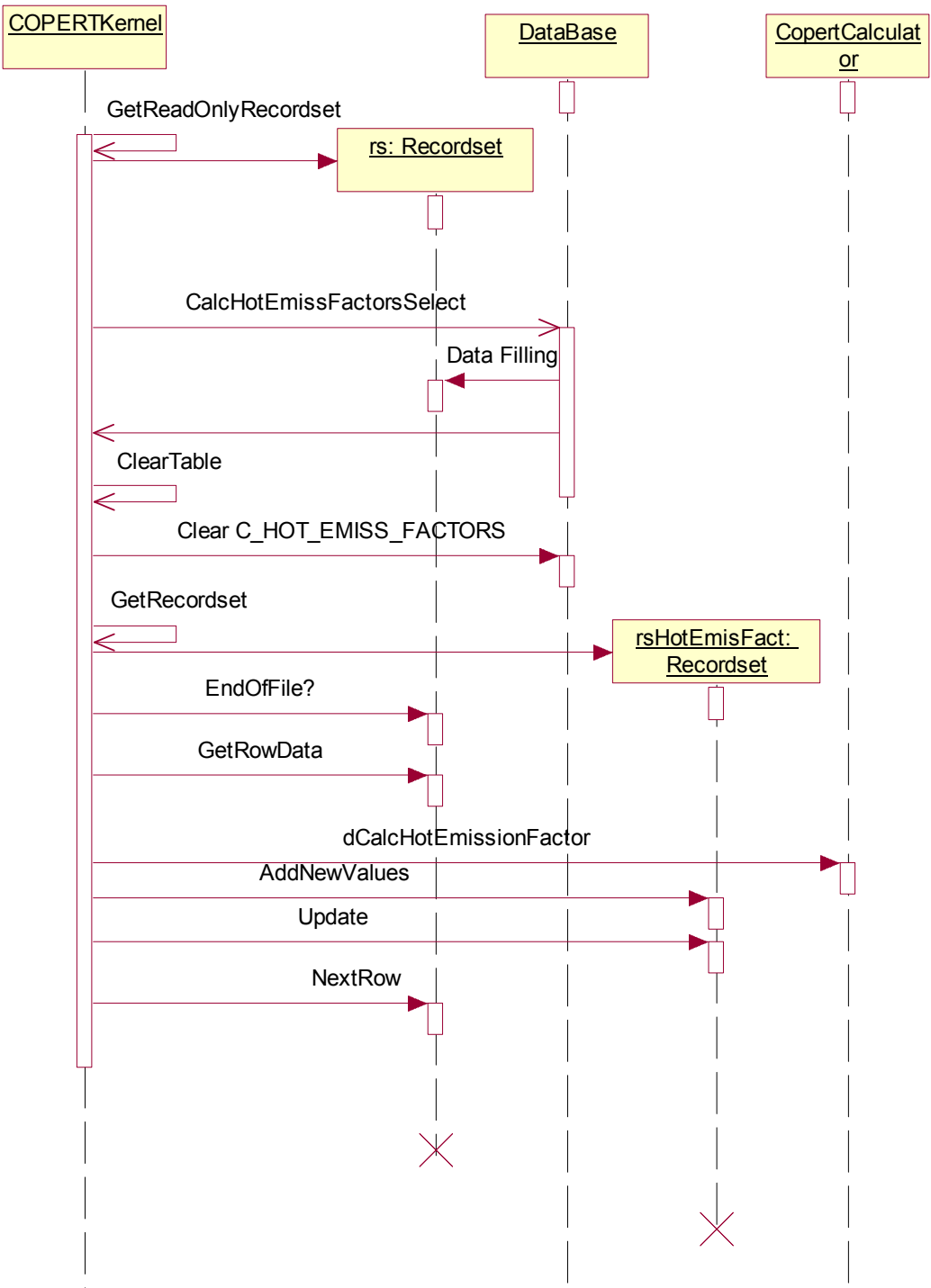
Is in this way that the prototype is seen more as a batch processing software, using the data as it comes from the data base, processing it and bringing back to the database as a result. In this way, not many classes are needed, as it is the case.

## 7.2 Operations Specification

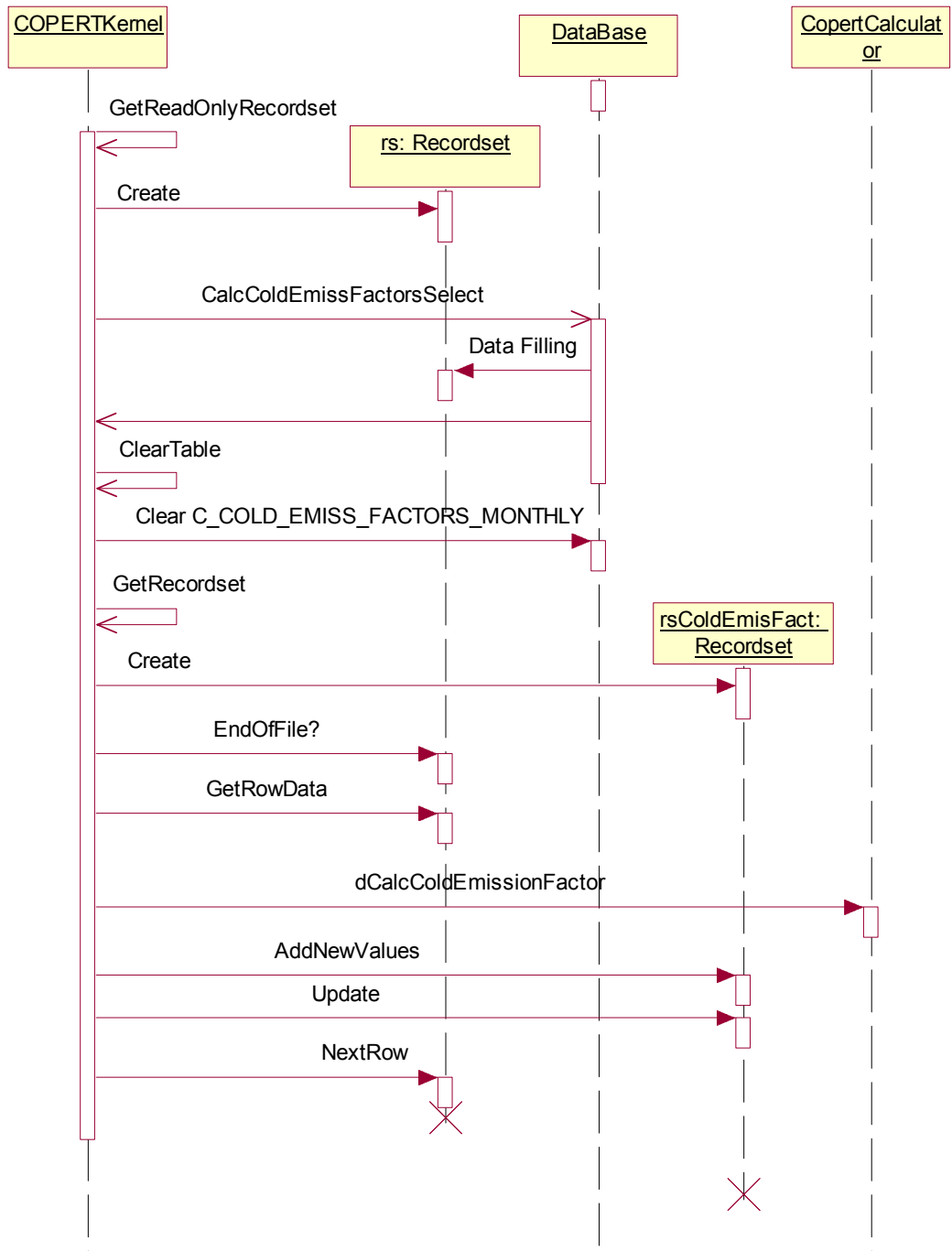
<b>Context:</b> COPERTKernel
<b>Method:</b> CalcHotEmissionFactors(Optional ByVal pisSQLQuery as String)
<b>Precondition:</b> pisSQLQuery must be a well formed SQL “WHERE” clause or NULL
<b>Postcondition:</b>  Clear the table C_HOT_EMISS_FACTORS  For all vehicles obtained from the CalcHotEmissFactorsSelect using if not null pisSQL as WHERE clause in the query:  1. Call COPERTCalculator.dCalcHotEmissionFactor with the values obtained.  2. Insert the results obtained in the table C_HOT_EMISS_FACTORS(HEF_x_VALUE_CALC,HEF_x_VALUE_CALC_BASE) where x is the driving mode of the current row.  3. Set value 0 to fields: HEF_U_VALUE_COR,HEF_R_VALUE_COR,HEF_H_VALUE_COR, HEF_U_VALUE_USR,HEF_R_VALUE_USR,HEF_H_VALUE_USR  4. Set value false to fields HEF_U_VALUE_K, HEF_R_VALUE_K,HEF_H_VALUE_K

<b>Context:</b> COPERTKernel
<b>Method:</b> CalcColdEmissionFactors(Optional ByVal pisSQLQuery as String)
<b>Precondition:</b> pisSQLQuery must be a well formed SQL “WHERE” clause or NULL
<b>Postcondition:</b>  Clear the table C_COLD_EMISS_FACTORS_MONTHLY  For all vehicles obtained from the CalcColdEmissFactorsSelect using if not null pisSQL as WHERE clause in the query:  1. Call COPERTCalculator.dCalcColdEmissionFactor with the values obtained.  2. Insert in the table C_HOT_EMISS_FACTORS(CEF_U_VALUE_CALC) the value obtained if it is not a vehicle post EURO I, if it is and the value is less than 1, insert value 1, else insert the value.  3. Set value 0 to fields: CEF_U_VALUE_COR, CEF_U_VALUE_USR  4. Set value false to field CEF_U_VALUE_K

# CalcHotEmissionFactors



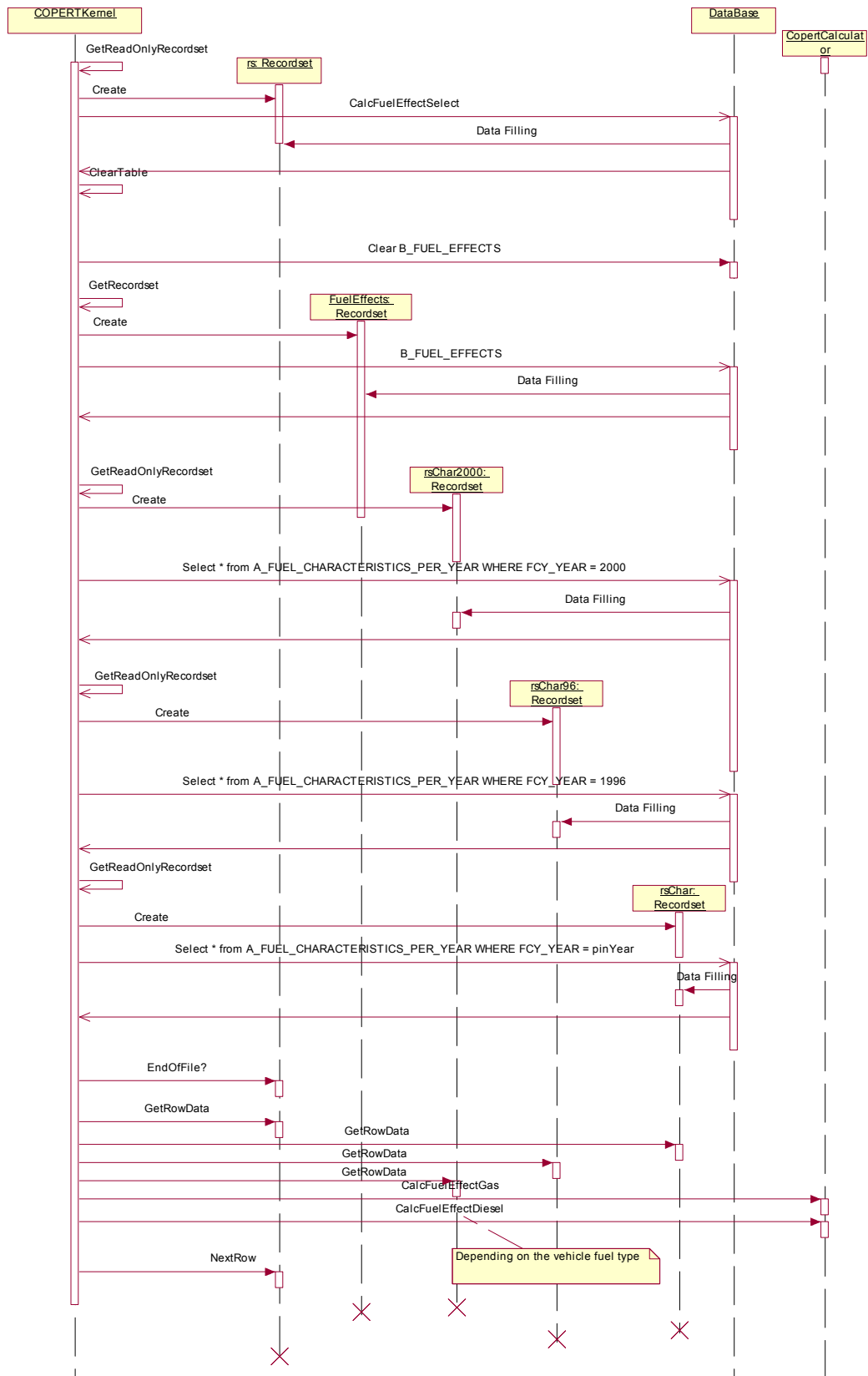
# CalcColdEmissFactors



<b>Context:</b> COPERTKernel
<b>Method:</b> CalcFuelEffect (ByVal pinYear As Integer)
<b>Precondition:</b> pinYear must be a valid year posterior to 1970
<b>Postcondition:</b>  Clear the table B_FUEL_EFFECTS  Obtain the Fuel characteristic for years 1996,2000,2005  For all vehicles obtained from the CalcFuelEffectSelect:  1. Call COPERTCalculator.CalcFuelEffectGas or call COPERTCalculator.CalcFuelEffectDiesel with the values obtained, depending on the fuel type of the vehicle. Call these procedures for the year equal to pinYear and for the base year that will be 2000 if the vehicle is posterior EURO II and pinYear is 2005, otherwise 1996  2. Insert in the table B_FUEL_EFFECTS (FEF_POLLUTANT_ID, FEF_VALUE) for pollutants CO,NOx,PM and VOC the values divided by the base year value. For the rest pollutants insert the value 1

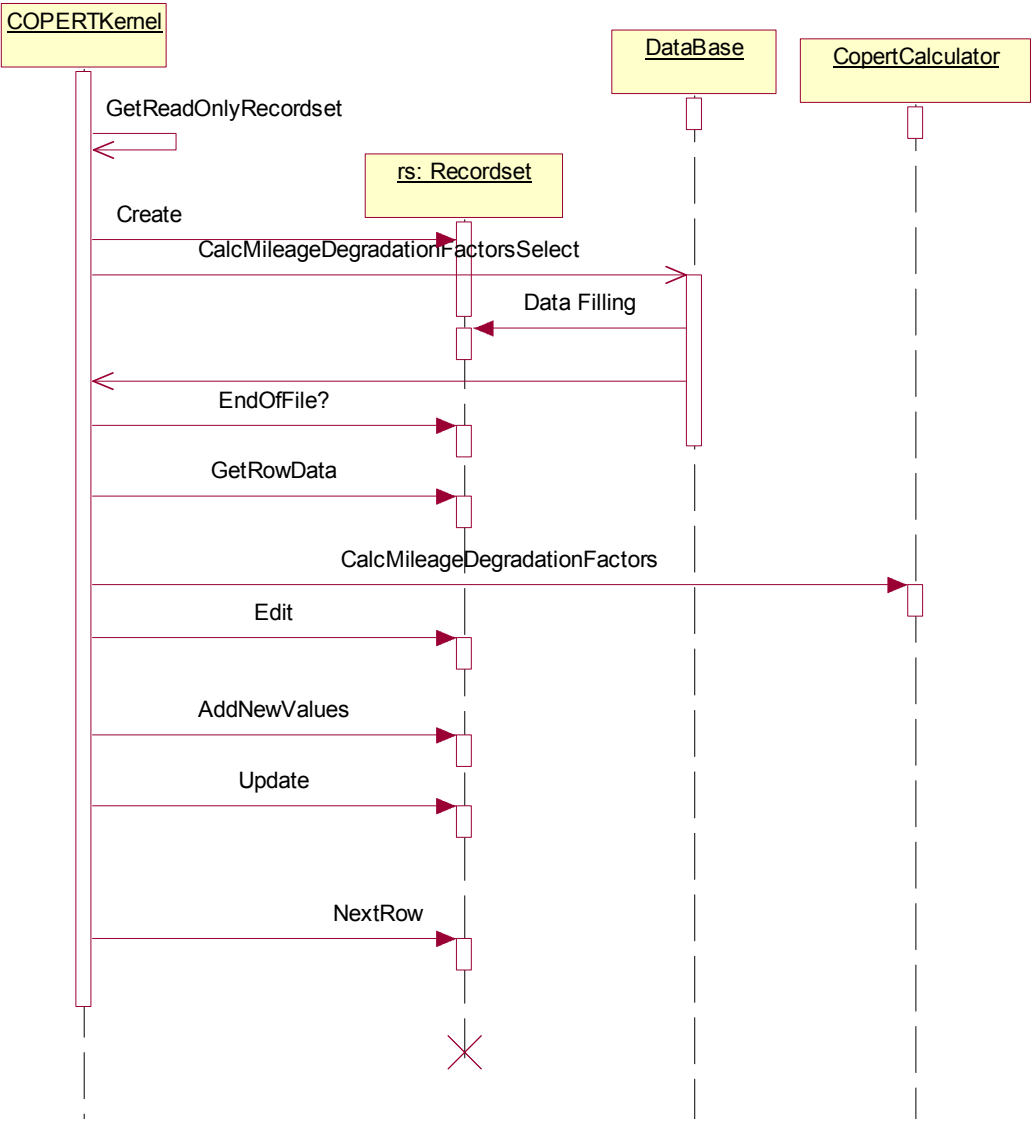
<b>Context:</b> COPERTKernel
<b>Method:</b> CalcMileageDegradation ()
<b>Precondition:</b> None
<b>Postcondition:</b>  For all vehicles obtained from the CalcMileageDegradationFactorsSelect:  1. Call COPERTCalculator.CalcMileageDegradationFactors with the values obtained  2. Insert in the table B_MILEAGE_DEGRADATION_FACTORS(MDF_U_VALUE, MDF_R_VALUE, MDF_H_VALUE)

## CalcFuelEffects





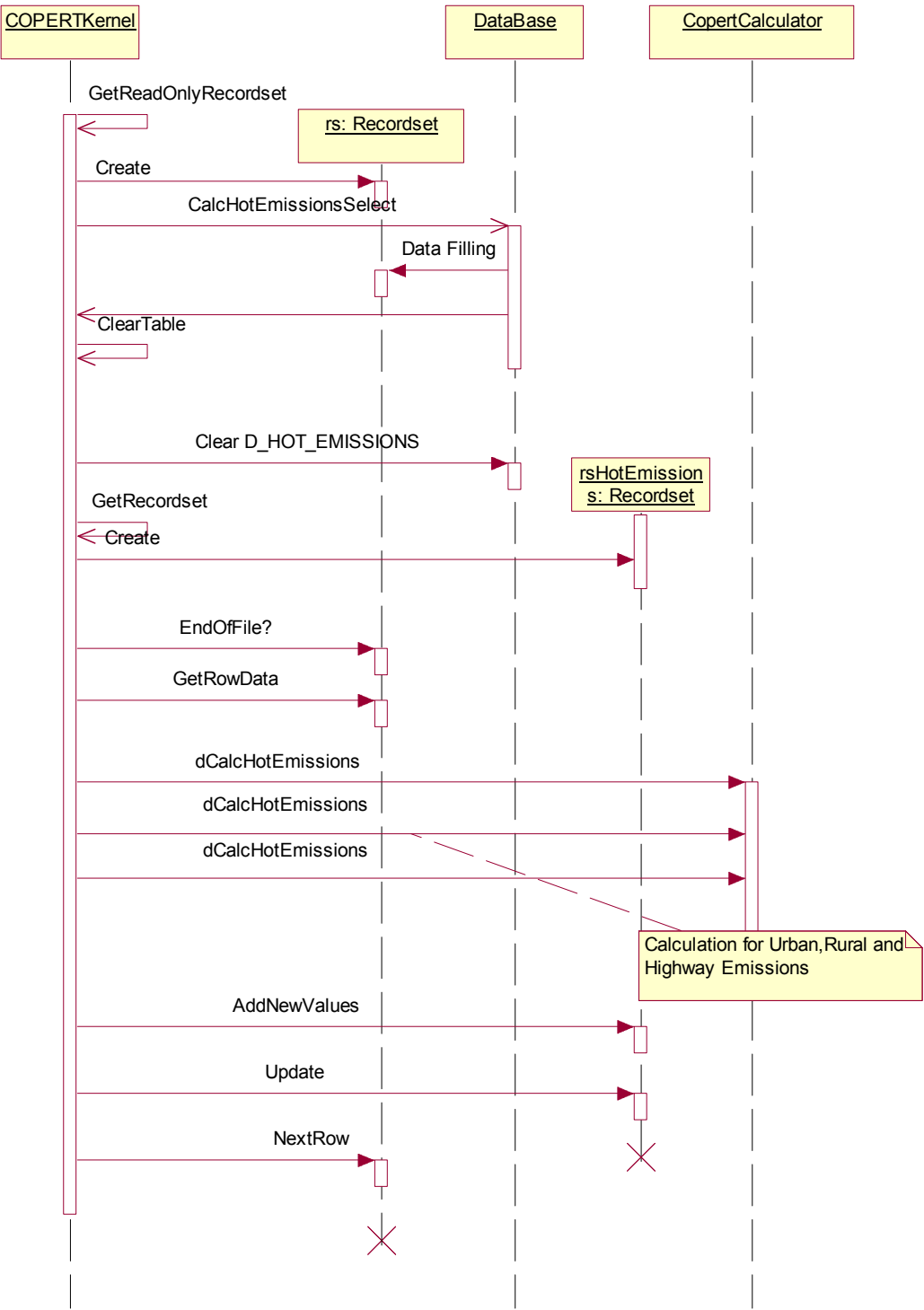
**CalcMileageDegradation**



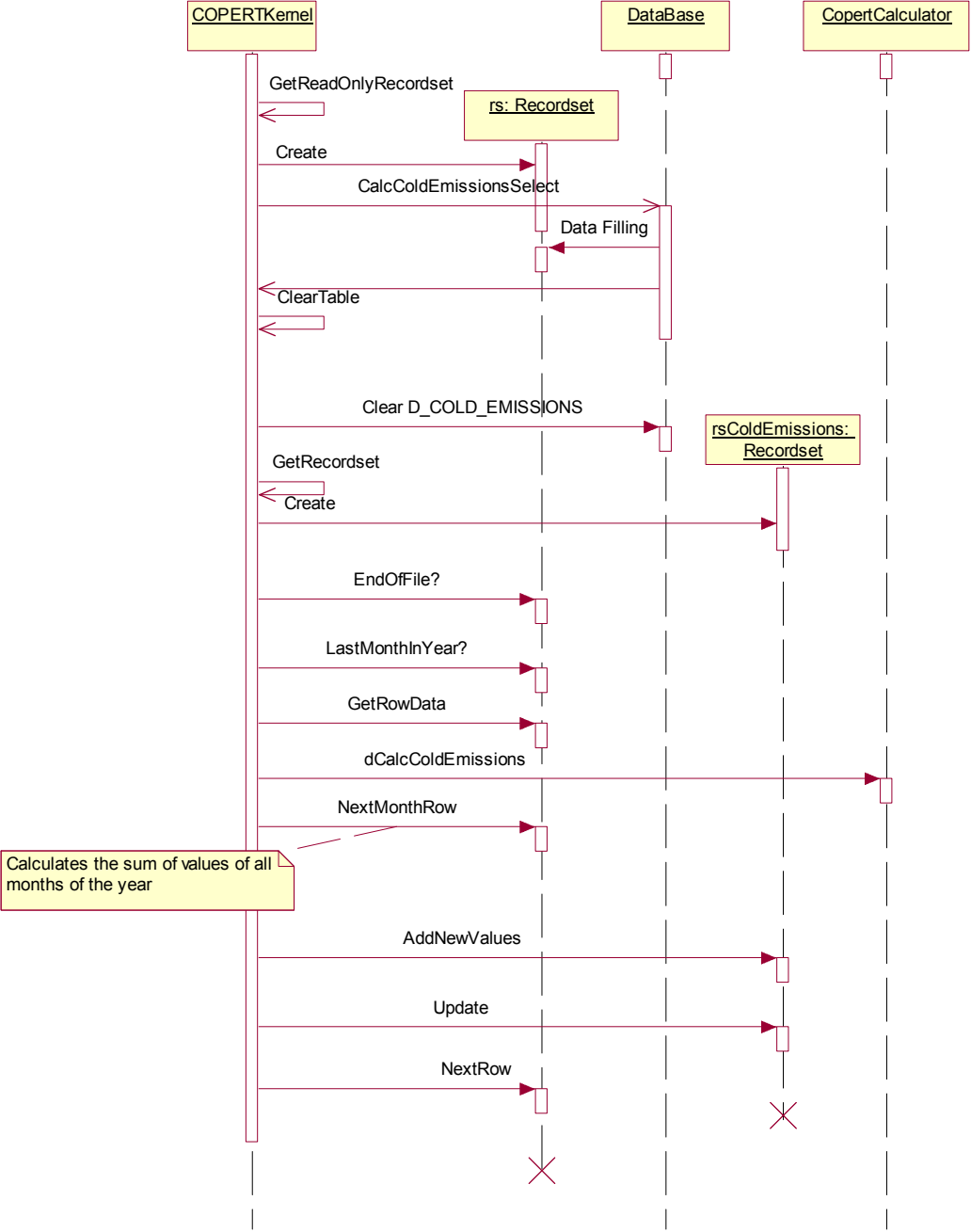
<b>Context:</b> COPERTKernel
<b>Method:</b> CalcHotEmissions ()
<b>Precondition:</b> None
<b>Postcondition:</b>  Clear the table D_HOT_EMISSIONS  For all vehicles obtained from the CalcHotEmissionsSelect:  1. Call COPERTCalculator.dCalcHotEmissions with the values obtained, getting the values for all driving modes for the current vehicle.  2. Insert in the table D_HOT_EMISSIONS (HEM_U_EMISS, HEM_R_EMISS, HEM_H_EMISS) the values obtained.

<b>Context:</b> COPERTKernel
<b>Method:</b> CalcColdEmissions ()
<b>Precondition:</b> None
<b>Postcondition:</b>  Clear the table D_COLD_EMISSIONS  For all vehicles obtained from the CalcColdEmissionsSelect:  1. Call COPERTCalculator.dCalcColdEmissionswith the values obtained, getting the values for all driving modes for the current vehicle.  2. Sum all the monthly cold emissions for each vehicle.  3. Insert in the table D_COLD_EMISSIONS (CEM_U_EMISS, CEM_R_EMISS) the sum of all months for every vehicle.

# CalcHotEmissions



CalcColdEmissions



<b>Context:</b> COPERTCalculator
<b>Method:</b> dCalcHotEmissionFactor(ByVal pidEuroNo As Integer, ByVal pisFuelType As String, ByVal pidSpeed As Double, ByVal pidAlpha As Double, ByVal pidBeta As Double, ByVal pidGamma As Double, ByVal pidDelta As Double, ByVal pidEpsilon As Double, ByVal pidZita As Double, ByVal pidIta As Double, ByVal pidThita As Double, ByVal pidRF As Double, ByRef podCalcEmissFactorBase As Double,ByRef podCalcEmissFactor As Double, Optional ByVal pidFuelEffect As Double = 1, Optional ByVal pidMileageDegradation As Double = 1)
<b>Precondition:</b> All not optional parameters are not null.
<b>Postcondition:</b>  <pre> podCalcEmissFactor = ((pidAlpha * pidSpeed ^ 2) + (pidBeta * pidSpeed) + pidGamma + (pidDelta * Log(pidSpeed)) + (pidEpsilon * Exp(pidZita * pidSpeed)) + (pidIta * (pidSpeed ^ pidThita))) * (1 - pidRF) * pidFuelEffect  If pidEuroNo &gt; 1 And pisFuelType = "GAS" Then     podCalcEmissFactorBase = podCalcEmissFactor / (1 - pidRF) Else     podCalcEmissFactorBase = podCalcEmissFactor End If  podCalcEmissFactor = podCalcEmissFactor * pidMileageDegradation </pre>

<b>Context:</b> COPERTCalculator
<b>Function:</b> CalcColdEmissionFactor(ByVal pidSpeed As Double, ByVal pidTemperature As Double, ByVal pidParamA As Double, ByVal pidParamB As Double, ByVal pidParamC As Double)
<b>Returns:</b> Double
<b>Precondition:</b> All parameters are not null
<b>Postcondition:</b> <b>Return</b> <pre> (pidSpeed * pidParamA) + (pidTemperature * pidParamB) + pidParamC </pre>

<b>Context:</b> COPERTCalculator
<b>Method:</b> CalcMileageDegradationFactors(ByVal pidMileage As Double, ByVal pidUrbanSpeed As Double, ByVal pidRuralSpeed As Double, ByVal pidHighwaySpeed As Double, ByVal pidAm19Km As Double, ByVal pidBm19Km As Double, ByVal pidAm63Km As Double, ByVal pidBm63Km As Double, ByRef podUrbanValue As Double, ByRef podRuralValue As Double, ByRef podHighwayValue As Double)
<b>Precondition:</b> All parameters are not null
<b>Postcondition:</b>  <pre> Dim dMileageCorrected As Double Dim dMileageCorrectedUrban As Double Dim dMileageCorrectedRural As Double Dim dMileageCorrectedHighway As Double  dMileageCorrected = IIf(pidMileage &gt; 120000, 120000, pidMileage)  If pidUrbanSpeed &lt;= 19 Then     podUrbanValue = pidAml9Km * dMileageCorrected + pidBml9Km ElseIf pidUrbanSpeed &gt;= 63 Then     podUrbanValue = pidAm63Km * dMileageCorrected + pidBm63Km Else     podUrbanValue = (pidAml9Km * dMileageCorrected + pidBml9Km) + ((pidUrbanSpeed - 19) * ((pidAm63Km * dMileageCorrected + pidBm63Km) - (pidAml9Km * dMileageCorrected + pidBml9Km)) / 44) End If  If pidRuralSpeed &lt;= 19 Then     podRuralValue = pidAml9Km * dMileageCorrected + pidBml9Km ElseIf pidRuralSpeed &gt;= 63 Then     podRuralValue = pidAm63Km * dMileageCorrected + pidBm63Km Else     podRuralValue = (pidAml9Km * dMileageCorrected + pidBml9Km) + ((pidRuralSpeed - 19) * ((pidAm63Km * dMileageCorrected + pidBm63Km) - (pidAml9Km * dMileageCorrected + pidBml9Km)) / 44) End If </pre>

```

If pidHighwaySpeed <= 19 Then
    podHighwayValue = pidAml9Km * dMileageCorrected + pidBml9Km
ElseIf pidHighwaySpeed >= 63 Then
    podHighwayValue = pidAm63Km * dMileageCorrected + pidBm63Km
Else
    podHighwayValue = (pidAml9Km * dMileageCorrected + pidBml9Km) + ((pidHighwaySpeed - 19) * ((pidAm63Km *
dMileageCorrected + pidBm63Km) - (pidAml9Km * dMileageCorrected + pidBml9Km)) / 44)
End If

podUrbanValue = IIf(podUrbanValue = 0, 1, podUrbanValue)
podRuralValue = IIf(podRuralValue = 0, 1, podRuralValue)
podHighwayValue = IIf(podHighwayValue = 0, 1, podHighwayValue)

```

#### Context: COPERTCalculator

**Function:** CalcFuelEffectGas(ByVal pidGasE100 As Double,  
ByVal pidGasE150 As Double, ByVal pidGasAromatics As Double,  
ByVal pidGasSulphur As Double, ByVal pidGasOxygenates As Double,  
ByVal pidGasOlefins As Double, ByVal pidGasBenzane As Double,  
ByRef podCO As Double, ByRef podVOC As Double,  
ByRef podNOx As Double, ByRef podPM As Double)

**Precondition:** All parameters are not null

#### Postcondition:

```

podCO = (2.459 - 0.05513 * pidGasE100 + 0.0005343 * pidGasE100 * pidGasE100 + 0.009226 * pidGasAromatics -
0.0003101 * (97 - pidGasSulphur)) * (1 - 0.037 * (pidGasOxygenates - 1.75)) * (1 - 0.008 * (pidGasE150 - 90.2))

podNOx = (0.1884 + 0.001438 * pidGasAromatics + 0.00001959 * pidGasAromatics * pidGasE100 - 0.00005302 * (97 -
pidGasSulphur)) * (1 + 0.004 * (pidGasOlefins - 4.97)) * (1 + 0.001 * (pidGasOxygenates - 1.75)) * (1 + 0.008 *
(pidGasE150 - 90.2))

podVOC = (0.1347 + 0.0005489 * pidGasAromatics + 25.7 * pidGasAromatics * Exp((-0.2642 * pidGasE100)) -
0.0000406 * (97 - pidGasSulphur)) * (1 - 0.004 * (pidGasOlefins - 4.97)) * (1 - 0.022 * (pidGasOxygenates -
1.75)) * (1 - 0.01 * (pidGasE150 - 90.2))

podPM = 1

```

<b>Context:</b> COPERTCalculator
<b>Method:</b> CalcFuelEffectDiesel(ByVal pidDensity As Double, ByVal pidPCA As Double, ByVal pidCN As Double, ByVal pidT95 As Double, ByVal pidSulphur As Double, ByVal pidSectorId As Integer, ByRef podCO As Double, ByRef podVOC As Double, ByRef podNOx As Double, ByRef podPM As Double)
<b>Precondition:</b> All parameters are not null
<b>Postcondition:</b>  If pidSectorId = 1 or pidSectorId = 2  podCO = -1.3250726 + 0.003037 * pidDensity - 0.0025643 * pidPCA - 0.015856 * pidCN + 0.0001706 * pidT95 podNOx = 1.0039726 - 0.0003113 * pidDensity + 0.0027263 * pidPCA - 0.0000883 * pidCN - 0.0005805 * pidT95 podVOC = -0.293192 + 0.0006759 * pidDensity - 0.0007306 * pidPCA - 0.0032733 * pidCN - 0.000038 * pidT95 podPM = (-0.3879873 + 0.0004677 * pidDensity + 0.0004488 * pidPCA + 0.0004098 * pidCN + 0.0000788 * pidT95) * (1 - 0.015 * (450 - pidSulphur) / 100)  else  podCO = 2.24407 - 0.0011 * pidDensity + 0.00007 * pidPCA - 0.00768 * pidCN - 0.00087 * pidT95 podNOx = -1.75444 + 0.00906 * pidDensity - 0.0163 * pidPCA + 0.00493 * pidCN + 0.00266 * pidT95 podVOC = 1.61466 - 0.00123 * pidDensity + 0.00133 * pidPCA - 0.00181 * pidCN - 0.00068 * pidT95 podPM = (0.06959 + 0.00006 * pidDensity + 0.00065 * pidPCA - 0 * pidCN) * (1 - 0.0086 * (450 - pidSulphur) / 100)  End if



<b>Context:</b> COPERTCalculator
<b>Function:</b> dCalcHotEmissions(ByVal pidVEH_POPULATION As Double, ByVal pidShare As Double, ByVal pidVEH_MILEAGE As Double, ByVal pidCalcEmissFactor As Double, ByVal pidCorrEmissFactor As Double, ByVal pidUsrEmissFactor As Double, ByVal pibKeepValue As Boolean) As Double
<b>Precondition:</b> All parameters are not null
<b>Postcondition:</b>  <pre> Dim dEmission As Double Dim sEmissFactor As Double      If pibKeepValue Then         sEmissFactor = pidUsrEmissFactor     Else         sEmissFactor = IIf(pidCorrEmissFactor = Nothing, pidCalcEmissFactor, pidCorrEmissFactor)     End If     dEmission = pidVEH_POPULATION * (pidShare / 100 * pidVEH_MILEAGE) * (sEmissFactor / 1000000)  Return dEmission </pre>

<b>Context:</b> COPERTCalculator
<b>Method:</b> CalcColdEmissions(ByVal pidBeta As Double, ByVal pidCirculationShare As Double, _ ByVal pinPopulation As Integer, ByVal pinMileageKm As Integer, ByVal pidColdOverHotRatio As Double, ByVal pidHotEmissFactor As Double, ByRef podColdEmissionUrban As Double, ByRef podColdEmissionRural As Double, _ Optional ByVal dCorrFactorIfGasolineAndPostEuroI As Double = 1)
<b>Precondition:</b> All parameters are not null
<b>Postcondition:</b>  <pre> Dim dBetaMinusUrbanShare As Double Dim dBetaMinusUrbanShareCorrected As Double Dim dBetaCorrected As Double  dBetaMinusUrbanShare = dCorrFactorIfGasolineAndPostEuroI * pidBeta - pidCirculationShare / 100 dBetaMinusUrbanShareCorrected = IIf(dBetaMinusUrbanShare &gt; 0, dBetaMinusUrbanShare, 0) </pre>

```
    dBetaCorrected = IIf(dBetaMinusUrbanShare < 0, pidBeta, pidBeta - dBetaMinusUrbanShare)

    podColdEmissionUrban = (dBetaCorrected * pinPopulation * (pinMileageKm / 12) * pidHotEmissFactor *
(pidColdOverHotRatio - 1)) / 1000000 * dCorrFactorIfGasolineAndPostEuroI

    podColdEmissionRural = (dBetaMinusUrbanShareCorrected * pinPopulation * (pinMileageKm / 12) *
pidHotEmissFactor * (pidColdOverHotRatio - 1)) / 1000000 * dCorrFactorIfGasolineAndPostEuroI
```

## 8. Summary & Status of the software

This report presents the software characteristics of the new beta version of Copert 4.

From the calculations point of view all the calculation functionalities of Copert III have been introduced successfully in this beta version: Hot and Cold Emission Factors Calculation, Hot and Cold Emissions Calculation, Evaporation losses, Mileage Degradation Factor Calculation and Fuel Effect Factor Calculation.

All calculations have been tested at LAT with real past data and the behaviour of the prototype is as expected. This remains to be further confirmed by national experts and other users, before the official introduction of Copert 4.

The beta version has been coded focusing on efficiency and structure. The current code is open to be extended in future refinements of the software. This has been achieved taking care of the extensive coding features in the Visual Basic .NET code and placing the SQL queries in the Database for easy management and better efficiency. Also, the independency of every calculation inside the code makes easier the manipulation or addition of code in future versions and the performance of the calculations is adequate. The required time for the calculations is also satisfactory on a common personal computer. This is also expected to be confirmed by users.

A similarity to Copert III was required from the point of view of User Interface. This has been fulfilled although some screens have been modified slightly from the previous versions of COPERT due the Visual Basic .NET limitations. The previous versions, being programmed with MS Access had better solutions for showing data in screen than the Visual Basic .NET libraries. Anyway the user can perform the expected functionalities in a very similar way and any changes do not require any extra training.

The coding of the User Interface has been also performed focusing on the upgradeability. The insertion or modification of screens could be made easily. Additionally, error manipulation algorithms have been introduced and proper error messages are given when the software identifies errors, without these crushing the system.

In this bet version, a slightly slow response has been identified in some cases during data manipulation. This is produced by the connection between Visual Basic .NET and MS Access through ADO.NET. At this point, nothing can be done to fix this, unless Microsoft introduces a patch. This is because the use ADO.NET is compulsory when Visual Studio.NET Graphical libraries are used.

## 9. References

[1]

Web page of COPERT III

Aristotle University of Thessaloniki - Lab of Applied Thermodynamics

<http://vergina.eng.auth.gr/mech/lat/copert/copert.htm>

[2]

Leonidas Ntziachristos and Zissis Samaras

COPERT III Computer Programme to calculate emissions from road transport.

Methodology and emission factors (version 2.1)

November 2000

[http://reports.eea.eu.int/Technical\\_report\\_No\\_49/en/tech49.pdf](http://reports.eea.eu.int/Technical_report_No_49/en/tech49.pdf)