CEMDAP

User's Manual

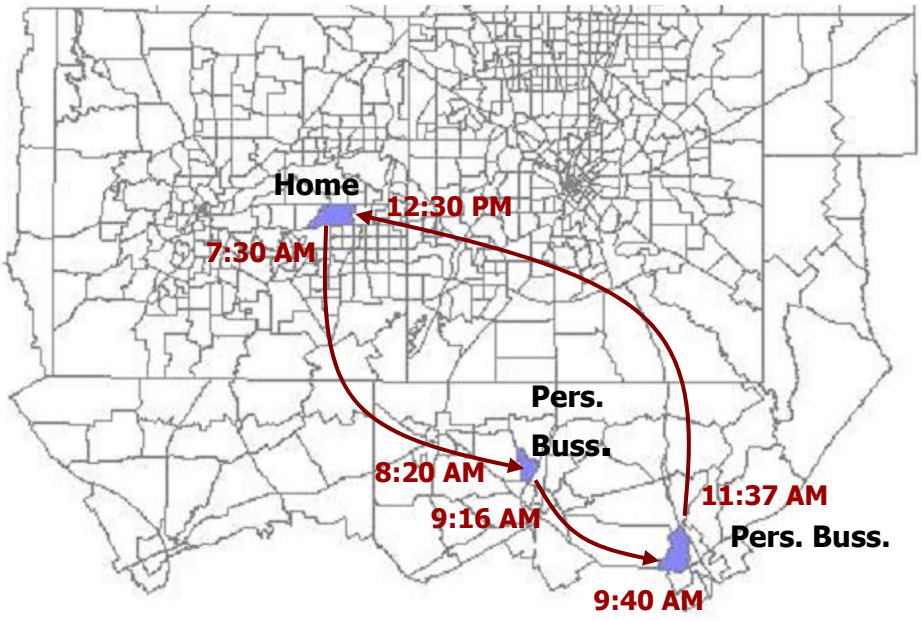

Table of Contents

List of Figures

List of Tables

Introduction

The "Comprehensive Econometric Micro-simulator for Daily Activity-Travel Patterns" (CEMDAP) is a software implementation of a system of econometric models that represent the decision-making behavior of individuals. It is one of the first systems to comprehensively simulate the activity-travel patterns of all household members (i.e., both adults and children, further distinguished based on employment/student status) in a continuous time domain, while also accommodating interactions among household members. Given various land-use, socio-demographic, activity system, and transportation level-of-service attributes as input, the system provides as output the complete daily activity-travel patterns for each individual in a population.

CEMDAP has been developed using an Object Oriented Design. It was implemented in Visual C++ using the Visual Studio .NET development platform. The software provides a user-friendly Windows environment to configure or modify the components of the modeling system, save or load model configurations, load the input data, and run the simulation. For an overview of the modeling system embedded in CEMDAP, the reader is referred to the *Guo et al. (2005) and Pinjari et al. (2006)*.

The rest of this manual is organized as follows. Chapter 1 describes the system requirements and installation instructions for the software. Chapter 2 provides an overview of the CEMDAP environment, including a description of the menus and commands available. Chapter 3 describes the prescribed input and output file specifications and formats. Chapter 4 guides the user through the basic operations such as loading inputs, and configuring model components.

Chapter 1. Installation Instructions

1.1 MINIMUM SYSTEM REQUIREMENTS

CEMDAP requires a Pentium IV-class processor with 1 Giga byte (or above) of Random Access Memory (RAM) and 2 Giga Hertz (or above) processing speed. The operating systems supported include Windows NT 4, Windows 2000, and Windows XP.

1.2 INSTALLATION INSTRUCTIONS

To install CEMDAP in your system, find the *Cemdap_5Threads.msi* or *Cemdap_10Threads.msi*[1] file in the installation CD and follow the steps outlined below.

1. Double click *Cemdap_5Threads.msi* or *Cemdap_10Threads.msi* to run the CEMDAP setup wizard.
2. In the *Welcome* screen, click *Next* to continue.

Figure 1.1: Cemdap Setup Wizard Welcome Screen

[1] Use *Cemdap_5Threads.msi* for machines with Random Access Memory less than 1 GB, and use *Cemdap_10Threads.msi* for machines with Random Access Memory greater than 1 GB.

3. This leads to the *Select Installation Folder* screen. Choose the destination location and folder name for the program installation (for example, C:\Program Files\University of Texas\Cemdap) and click *Next* to continue.

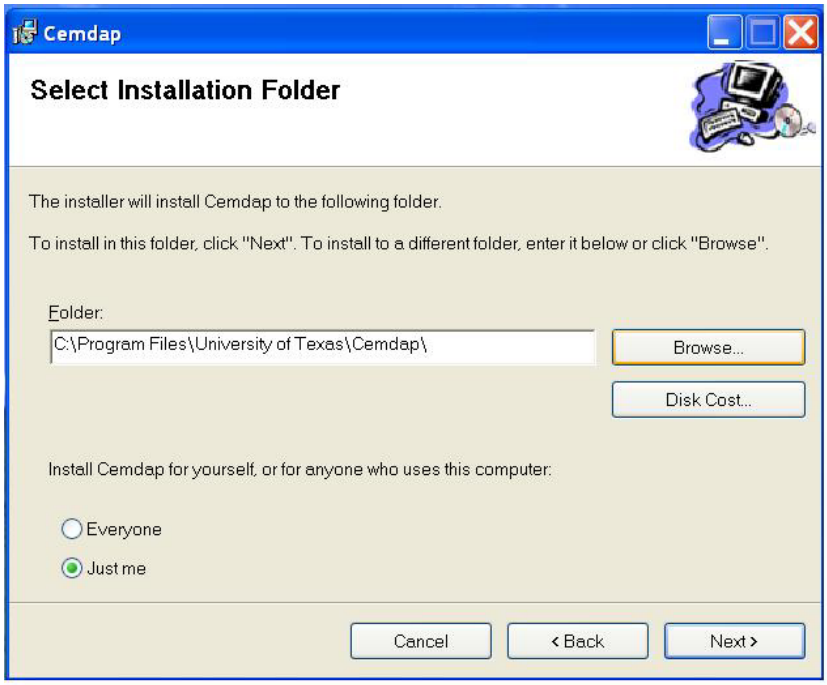

Figure 1.2: Select Installation Folder Screen

4. Click *Next* in the *Confirm Installation* screen.

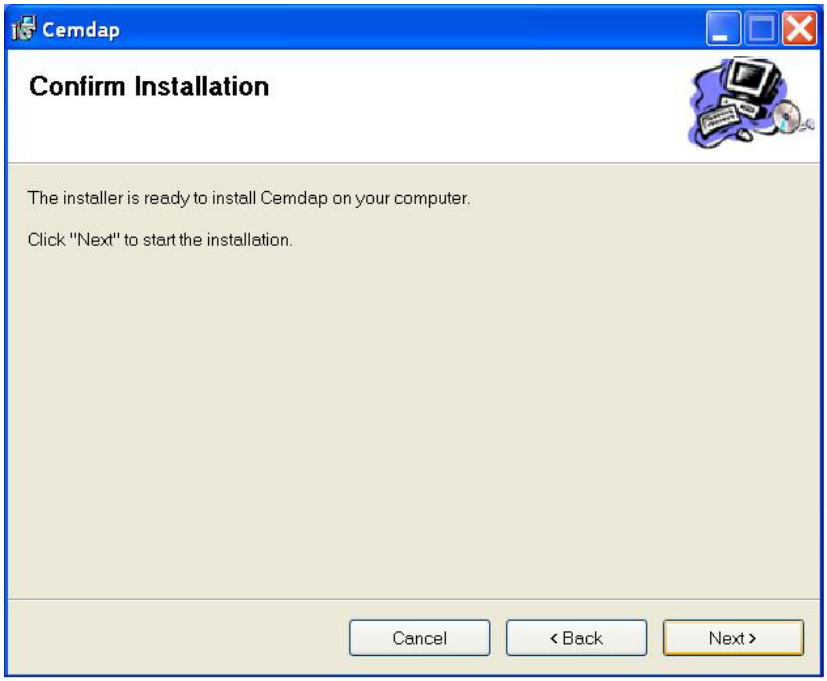

Figure 1.3: Confirm Installation Screen

5. Installation will complete in a few seconds.
6. Click Close to exit the wizard.

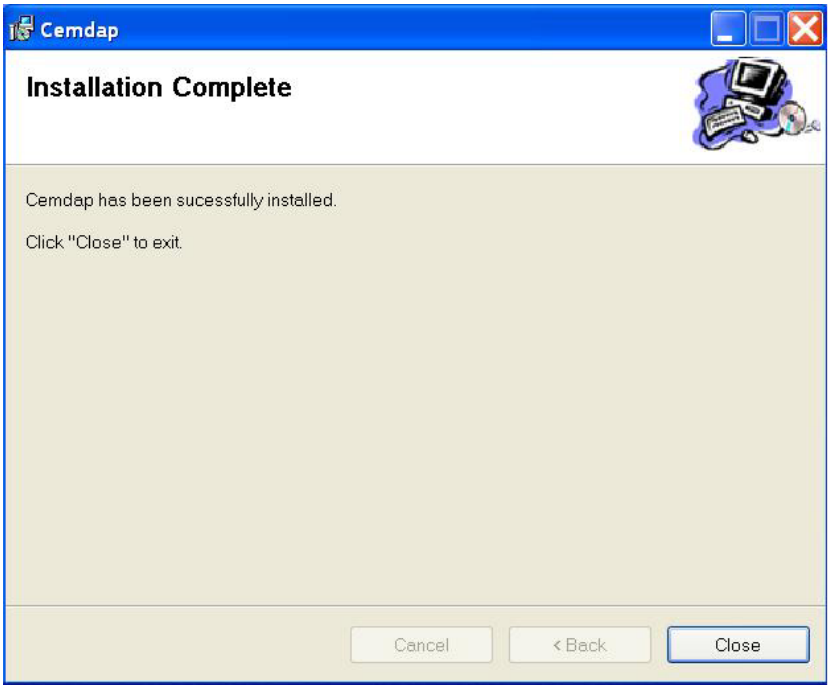

Figure 1.4: Installation Complete Screen

At the end of the installation process, CEMDAP will be added to the Program menu, and a shortcut to the program will be created on the Desktop. Double click the desktop shortcut to open the CEMDAP_5Threads/CEMDAP_10Threads folder. In this folder, double click on *cemdap.exe* to start CEMDAP.

The installation CD contains sample input data files (in the *TestData* folder), and the model specification (or the configuration) file for the Dallas-Fort-Worth area implementation (further instructions to run CEMDAP are provided in Chapter 4). The installation CD also contains an EXCEL files describing the CEMDAP inputs (input file description) and outputs (output file description). Prior to running CEMDAP, it is necessary to install PostgreSQL. The corresponding installation instructions are provided in Appendix B on page 60.

1.3 TO UNINSTALL

To uninstall CEMDAP use the *Cemdap_5Threads.msi* or *Cemdap_10Threads.msi* file in the installation CD and follow the instructions given below.

1. Double click *Cemdap_5Threads.msi* or *Cemdap_10Threads.msi* to run the CEMDAP setup wizard.
2. In the *Welcome* screen select the *Remove Cemdap* radial button and click *Finish*.

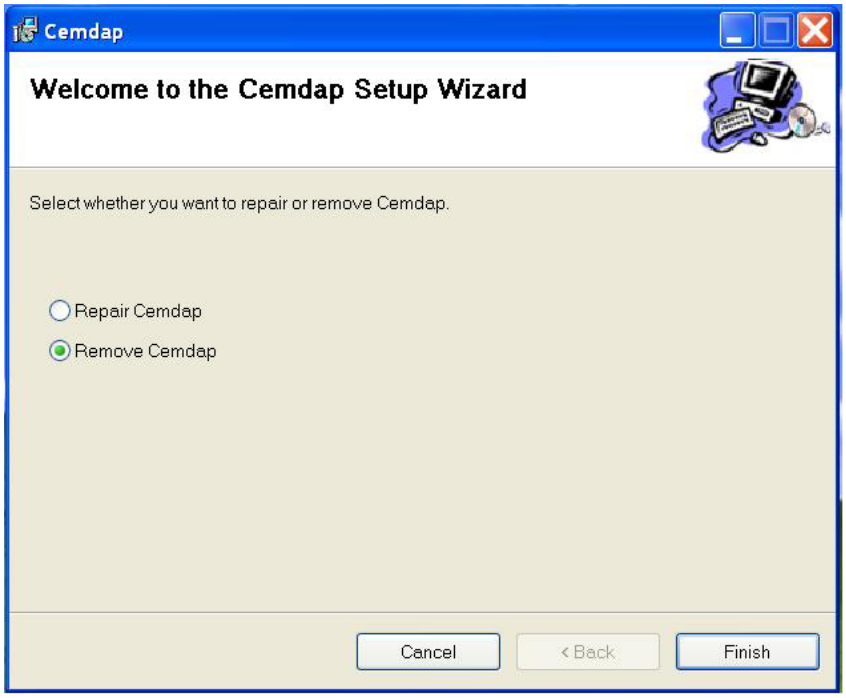

Figure 1.5: Cemdap Setup Wizard Welcome Screen once CEMDAP is Installed

3. Click *Close* to exit.

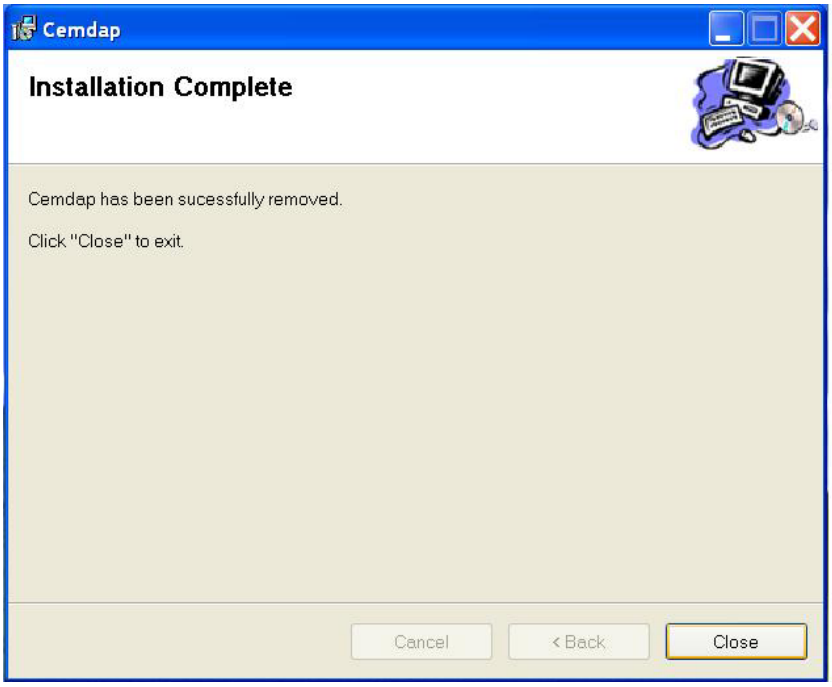

Figure 1.6: Installation is Complete Screen once CEMDAP is removed

Chapter 2. The CEMDAP Environment

CEMDAP offers a user-friendly environment to simulate the activity-travel patterns of a population by using standard Windows user interface features. In this chapter, we present the basic features of the CEMDAP environment.

To start CEMDAP, double-click the *shortcut to CEMDAP*_5Threads or CEMDAP_10Threads on your desktop, or find CEMDAP_5Threads or CEMDAP_10Threads in the *Start* menu under *All Programs* and double click on cemdap.exe. This opens up the following main window of CEMDAP.

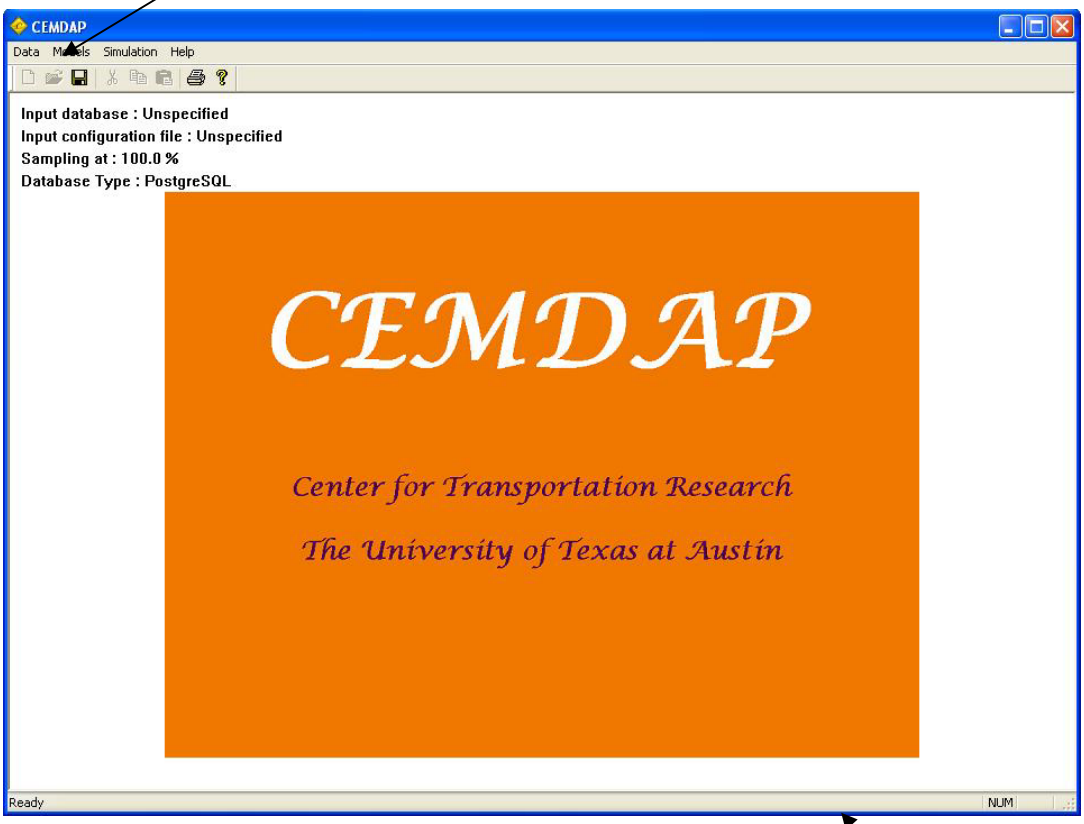

Figure 2.1: CEMDAP Main Window and Menus

The above main CEMDAP window remains open as long as the software is being used, and it hosts the important menu items and other dialog boxes. The primary functionality of CEMDAP lies within the menu commands. The commands below, available from the menu bar, and their functions, are tabulated below.

Table 2.1: Data Menu

Choose…	To…
Input	Load the input data. The input data must be 'registered' prior to loading it (refer to Chapter 3 for details).
Output	Specify the names and locations for the output files. The output file formats are described in detail in Chapter 3.
Exit	Quit from the CEMDAP environment.

Table 2.2: Models Menu

Choose…	To…
Load	Load the model configurations from file (refer to Chapter 4 on Basic Operations for instructions on configuring models).
Save	Save the model configurations to file (refer to Chapter 4 on Basic Operations for instructions on configuring models).
Generation Allocation	To access the model configuration dialog boxes for the suite of generation-allocation model components (refer to Chapter 4).
Scheduling-Children	To access the model configuration dialog boxes for the suite of Children-Scheduling model components (refer to Chapter 4).
Scheduling-Worker	To access the model configuration dialog boxes for the suite of Worker-Scheduling model components (refer to Chapter 4).
Scheduling-Non Worker	To access the model configuration dialog boxes for the suite of Non Worker-Scheduling model components (refer to Chapter 4).
Scheduling-Joint	To access the model configuration dialog boxes for the suite of Joint Activity-Scheduling model components (refer to Chapter 4).
Interactive UI	To open the interactive user interface that ties all model configuration dialog boxes together using the modeling system framework (refer to Chapter 4).

Table 2.3: Simulation Menu

Choose…	To…
Run	Initiate a simulation run. (Note that the input data must be loaded and the complete model system configured prior to using the *Run* command).

Table 2.4: Help Menu

Choose…	To…
About	Display the CEMDAP dialog box that shows copyright and version information

Chapter 3. Input and Output Files

3.1 INPUT DATA SPECIFICATION

The inputs required by CEMDAP can be broadly classified into two categories: a) **data inputs**, which include the population characteristics, zonal descriptives, and level-of-service data of the transportation network; and b) **model parameters** for all the components of the embedded model system.

3.1.1 Data Inputs

The data inputs to CEMDAP are provided in the form of tables. There are six main types of tables. The **Household** table contains the household-level characteristics (such as residential location and number of vehicles) of the population, and the **Persons** table contains the person-level characteristics (such as employment status and work location). The **Zones** table comprises data on the characteristics of each of the traffic analysis zones (such as population and employment). The **Zone2Zone** table includes data such as distance between each of the zonal pairs. CEMDAP allows the analyst to specify the inter-zonal level-of-service measures for any number of time periods. The **Losdir** table identifies the names of the LOS tables included, and the time-of-day period corresponding to each of these tables. Finally, there are as many **LOS** tables (with names as specified in the **Losdir** table), each containing several inter-zonal LOS measures such as travel times and costs for different modes.

Each table comprises several "required" data items or columns. These data include record identifiers (such as household ID and person ID), and other basic information required by the CEMDAP modeling system. In addition, the Households and Persons tables may also have additional columns corresponding to additional explanatory variables used in the underlying empirical models. Overall, each exogenous variable must have a corresponding column in the appropriate table. The structure of each of the input tables is presented below, identifying only the required data items and their ordering within the tables. Additional columns are simply appended to these tables.

Table 3.1: Households Table with Required Variables

Column	Variable	Variable Name
1	Household ID	HID
2	Number of adults	N_ADULTS
3	Total number of HH vehicles	N_AUTOS
4	Home TSZ location	ZONE_ID
5	Number of children	KIDS
6	Household structure	Structure

Table 3.2: Persons Table with Required Variables

Column	Variable	Variable Name
1	Household ID	HID
2	Person ID	PID
3	Employed	EMPLOYED
4	Student	STUDYING
5	Driver's license	LICENSE
6	Location of work	WORK_ZON
7	Location of school	STUD_ZON
8	Female	FEMALE
9	Parent	PARENT

Table 3.3: Zones Table with Required Variables

Column	Variable	Variable Name
1	Zone ID	ZID
2	Dallas CBD	DALCBD
3	Fort Worth CBD	FWCBD
4	Distance to the nearest shopping mall	SHOPDIST
5	Accessibility to retail employment	REMPACC
6	Accessibility to retail and service employment	RSEMPACC
7	Distance to the nearest shopping mall	SHOPDIST
8	Zonal service employment	SEMP
9	Zonal retail employment	REMP
10	Zonal population	NUMPERS

Table 3.4: Zone2Zone Table and Required Variables

Column	Variable	Variable Name	Comment
1	Origin Zone	ORIG_ZON	
2	Destination Zone	DEST_ZON	
3	Adjacent	ADJACENT	1 if origin and destination zones are adjacent
4	Distance	DISTANCE	Miles

Table 3.5: LOSdir Table with Required Variables

Column	Variable	Variable Name	Comment
1	Start time	STAT_T	Start time of the time period of the LOS table
2	End time	END_T	End time of the time period of the LOS table
3	Table name	TBNAME	Name (string) of the LOS table

Table 3.6: LOS Table with Required Variables

Column	Variable	Variable Name	Comment
1	Origin zone	ORIG_ZON	
2	Destination zone	DEST_ZON	
3	Same zone	SAME_ZON	
4	Adjacent	ADJACENT	
5	Distance	DISTANCE	miles
6	DA IVTT	DA_IVTT	minutes
7	DA OVTT	DA_OVTT	minutes
8	Transit availability	TR_AVAIL	
9	Transit IVTT	TR_IVTT	minutes
10	Transit OVTT	TR_OVTT	minutes
11	Transit cost	TR_COST	Dollars
12	DA cost	AU_COST	Dollars

Appendix A shows all the tables and variables, including the 'required variables' and some 'additional variables' used in the current implementation of CEMDAP for the DFW area. The Data Inputs are required to be in the postgreSQL database format. The input tables in the postgreSQL database must take titles as specified above and the variables must be of the '*double*' type. The variables names are not pre-specified; each variable is identified by the specific table (table number) it belongs to, and the position (column number) of the variable in the table. Appendix B presents the procedure to prepare the input database corresponding to the DFW implementation in the postgreSQL format.

3.1.2 Model Specifications

The model system embedded in CEMDAP is described in greater detail in Guo *et al.* (2005) and Pinjari *et al.* (2006). The reader is referred to Pinjari *et al.* (2006) for a list of all the model components. The variables and parameters of each of the model components that constitute this model system must be specified the first time CEMDAP is used. Thereafter, the model configurations can be saved into a file using the *Save* command in the *Models Menu* and reloaded as desired using the *Load* command in the *Models Menu*. The first time specification of the model configurations can be achieved in one of the following four ways (refer to Section 4.4 for further details).

1. Access each of the model configuration dialog boxes through the commands in the *Models Menu*. In particular, access the dialog boxes under the *Generation Allocation*, *Scheduling-Children, Scheduling-Worker*, *Scheduling-Non Worker*, and *Scheduling-Joint* entries in the *Models Menu*. (This method is recommended for changing the parameters of a few models).
2. Access the model configuration dialog boxes through the *Interactive UI*, which is also in the *Models Menu*. The interactive user interface ties all the dialog boxes together

using the embedded modeling framework. (This method is recommended for first time model parameter input).

3. The model parameters for all the model components can be coded into a text file (model configuration file) in a prescribed format, and then loaded into the system using the *Load* command in the *Models Menu*. A sample model configuration file can be found in */data* under the installation destination directory. Appendix C describes the prescribed format of the model configuration file (For beginners and /or initial users, it is recommended to specify the model parameters using the Graphical User Interface instead of coding them in the model configuration file).

3.2 OUTPUT DATA FILES

CEMDAP produces as output the complete activity-travel patterns for a day for every individual in the population of interest. There are seven output files, corresponding to characteristics of (1) Adults, (2) Children, (3) Workers, (4) Students, (5) Non-workers, (6) Tours, and (7) Stops. Each of these files is partitioned into smaller subsets based on the number of threads. The files Adults.out and Children.out contain the decisions to undertake activities of different types for adults and children, respectively. The files Workers.out, Nonworkers.out, and Childstu.out contain the pattern-level attributes of the workers' (including adult students), non-workers', and child students' patterns, respectively. The files Tours.out and Stops.out contain the tour-level and stop-level attributes, respectively, for all the individuals that undertake travel. These output files can easily be imported into any spreadsheet of your choice. The formats and content of these output files are given below. Detailed description of the output variables can be found in an EXCEL file (output file specification) in the */data* under the installation destination directory.

Table 3.7: Adults.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Adult goes to work on the day
4	Adult undertakes work-related activity
5	Adult drops-off children at school
6	Adult picks-up children from school
7	Adult undertakes joint discretionary activities with children
8	Adult undertakes shopping activity
9	Adult undertakes HH/personal business activity
10	Adult undertakes social/recreational activity
11	Adult undertakes eat-out activity
12	Adult undertakes other serve passenger activity

Table 3.8: Children.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Child goes to school on the day
4	Child undertakes joint discretionary activities with parent
5	Child undertakes independent discretionary activities

Table 3.9: Workers.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Work/school start time
4	Work/school end time
5	Number of before-work tours
6	Number of work-based tours
7	Number of after-work tours

Table 3.10: Childstu.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	School start time
4	School end time
5	Child gets dropped off at school by parent
6	Child gets picked up from school by parent

Table 3.11: Nonworkers.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Total number of tours made

Table 3.12: Tours.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Tour identification number
4	Home/work stay duration before tour
5	Tour mode
6	Tour duration
7	Number of stops in tour

Table 3.13: Stops.Out Format and Content

Column No.	Description
1	Household identification number
2	Person identification number
3	Tour identification number
4	Stop identification number
5	Activity type
6	Start time of travel to the stop
7	Travel time to stop
8	Stop duration
9	Stop location (zone) ID
10	Origin zone ID
11	Trip distance (zone to zone)
12	Activity type at the previous stop

Chapter 4. Basic Operations

4.1 REGISTERING THE INPUT DATABASE

The input database, which is in the postgreSQL database format, must be registered before it can be loaded in CEMDAP. The database registration should be done each time a new database is used. **PostgreSQL should be installed in the computer prior to running CEMDAP**. Further, data needs to be populated into the PostgreSQL database (the reader is referred to Appendix B for installation and database loading instructions). After installation of PostgreSQL and populating the database, follow the instructions below to register an input file.

1. Click on the *Input* command in the *Data Menu* of CEMDAP.

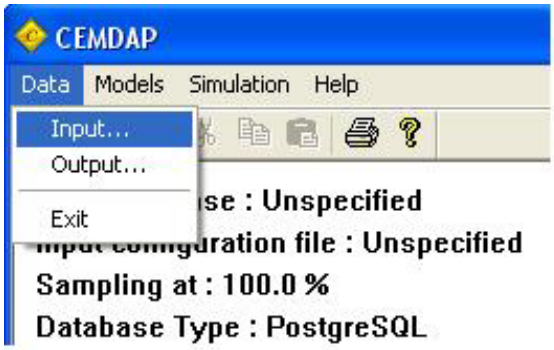

Figure 4.1: Input command in the Data Menu

2. In the *Select Data Source* dialog that opens up, select the *Machine Data Source* tab and click on the *New* button.

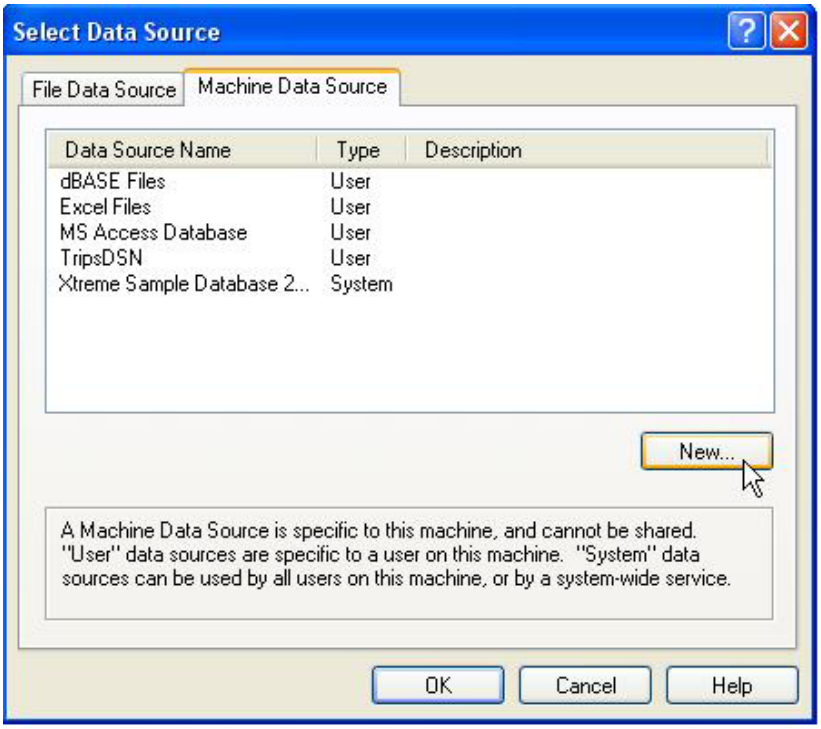

Figure 4.2: Machine Data Source Tab within the Select Data Source Screen

3. In the *Create New Data Source* dialog that opens up, select the *System Data Source* button and click on the *Next* button.

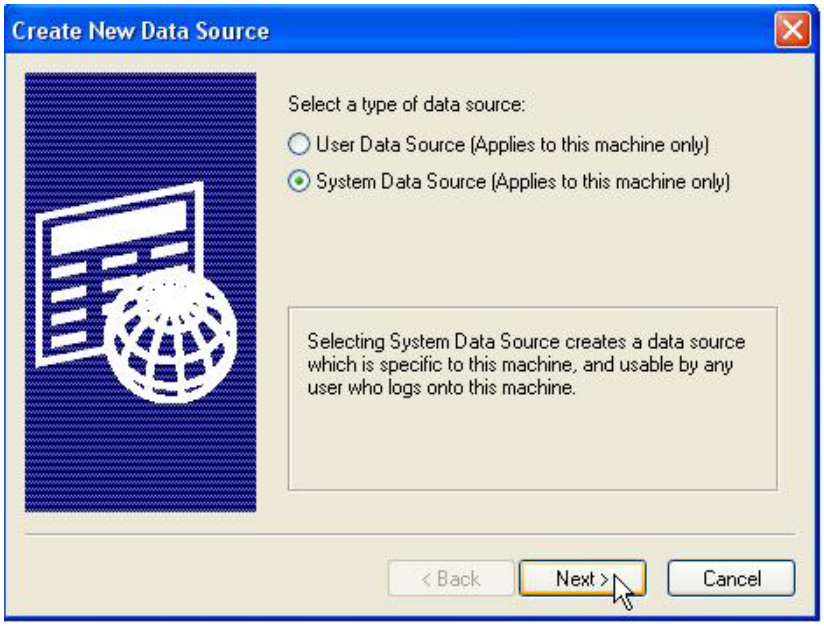

Figure 4.3: Create New Data Source Dialog Box

4. Scroll down and select the *PostgreSQL ANSI* driver from the list of drivers and click *Next* as displayed below.

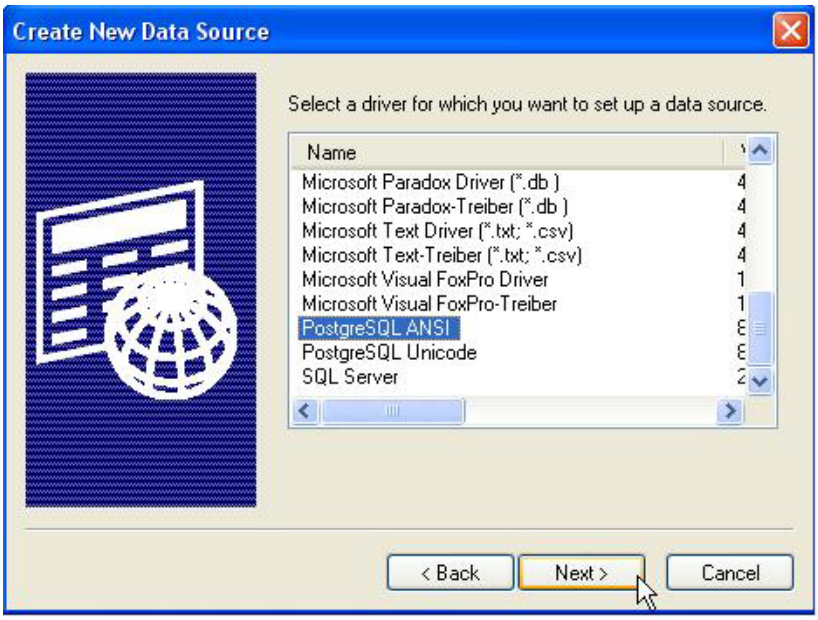

Figure 4.4: Select driver Screen within Create New Data Source Dialog Box

5. Once you click *Next*, the following box will appear. Then, click *Finish* to open up the *PostgreSQL ANSI ODBC* dialog box.

Figure 4.5: Create New Data Source Finish Screen

6. In the *PostgreSQL ANSI ODBC* dialog box (shown below), enter the name of the *Data Source* (*Note* The Data Source can be given any name, therefore you can leave the default name), name of the *Database (ex: CEMDAP_Test_Data. Note* this name should be the same as the database name that was created in data loading process described in Appendix B*), localhost* as the name of the server, the appropriate username and password of the postgreSQL data base installed in the machine (see Appendix B), *5432* as the *Port*, *prefer* as the *SSL Mode*, and provide a description (ex: *Test Data*) of the database. After entering all the above mentioned items, click *Save.*

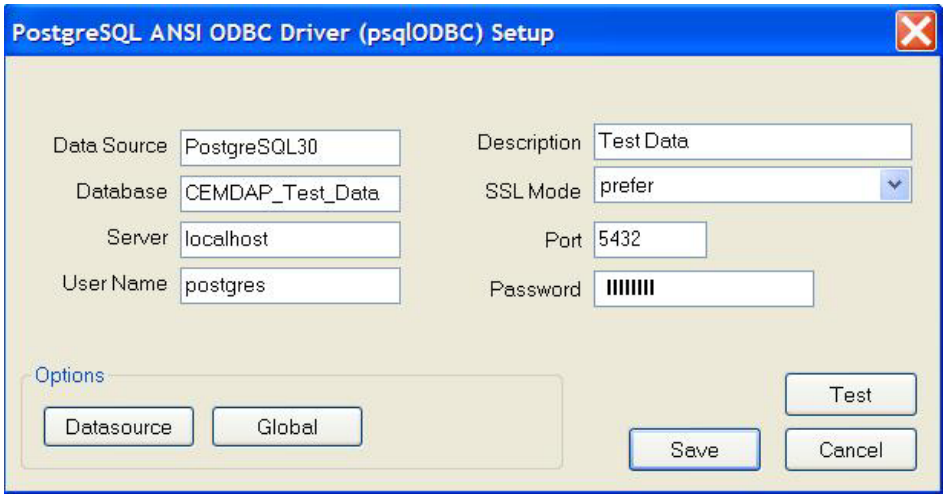

Figure 4.6: PostgreSQL ANSI ODBC Dialog Box

7. Finalize the database registration process by clicking *OK* to close the dialog boxes. CEMDAP is now set up to access the database file you registered.

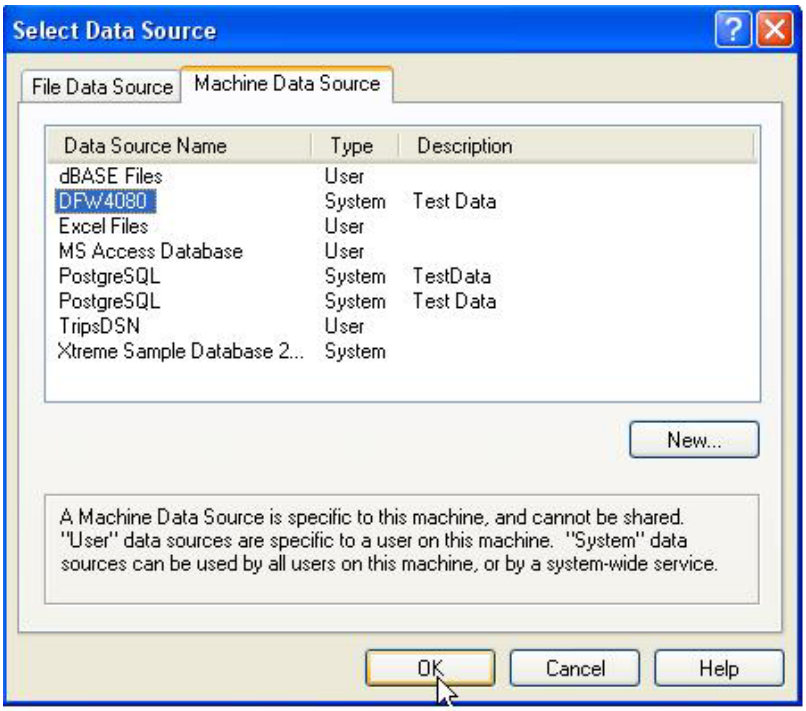

Figure 4.7: Final Select Data Source Screen

4.2 LOADING THE INPUT DATA FILE

Any PostgreSQL database that satisfies the prescribed format (refer to Chapter 3), and that has been registered as explained above, can be loaded in CEMDAP as input. The following procedure should be followed to load the input data.

Note: **If the database has just been registered, this step (i.e., loading the input data file step) must be skipped.** These steps are essential for loading an already registered database.

1. Click on the *Input* command in the *Data Menu* of CEMDAP.

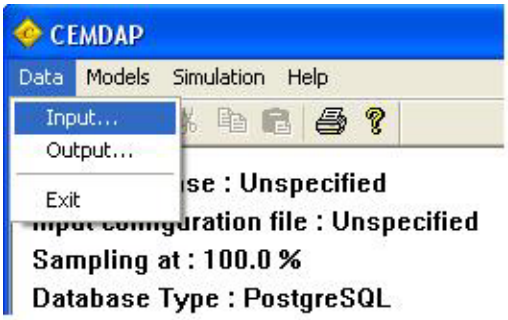

Figure 4.8: Input command in the Data menu (2)

2. In the *Select Data Source* dialog that opens up, select the *Machine Data Source* tab (first figure on this page), and choose the registered data source to be loaded (for example, PostgreSQL30).
3. Click *OK* and the input data is loaded.

4.3 SPECIFYING THE OUTPUT FILES

During a simulation run, CEMDAP creates seven output files (refer to Chapter 3 for file formats). The following steps are used to specify the location of these seven files.

1. Click on the *Output* command in the *Data* menu of CEMDAP.

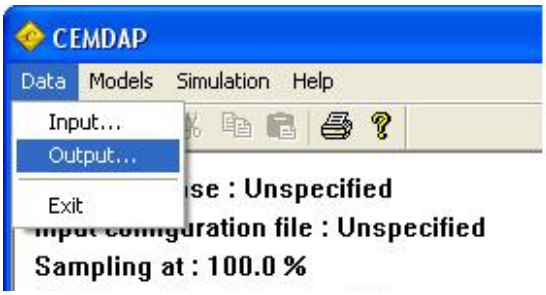

Figure 4.9: Output command in the Data menu

2. In the *Save Simulation Results As…* dialog box that opens, click on *Browse* to select the folder location for all the output files.

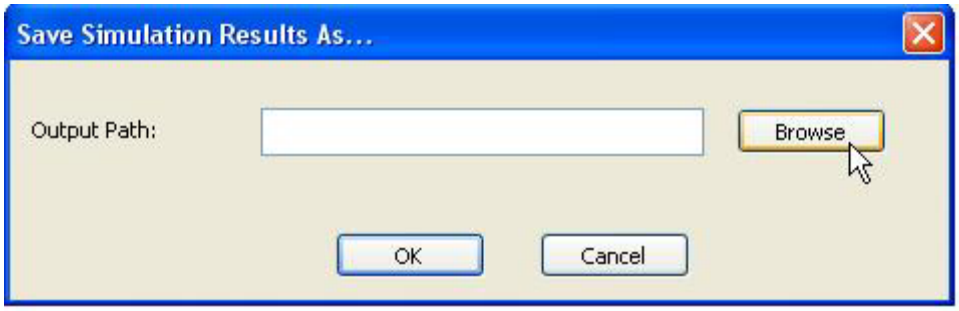

Figure 4.10: Save Simulation Results As… dialog box

3. Select the location (or enter the path name) for all the output files, and then click *OK*.

4.4 CONFIGURING THE MODEL SYSTEM

All the components of the model system embedded in CEMDAP must be configured into a model configuration file (also referred to as a model specification file) and input to the system before starting a simulation run. Model configuration involves selecting the relevant variables and inputting their parameters. The model system can be completely configured either by accessing all the corresponding dialog boxes to specify the models or by manually preparing a model configuration file (as described in Appendix C). Sections 4.4.1 and 4.4.2 describe two methods of accessing all the model dialog boxes to configure the entire model system. Section 4.4.3 explains how the complete model configuration entered using the dialog boxes can be saved in a file. Section 4.4.4 explains how to load a model configuration file into CEMDAP. All the model components fall into one of seven categories: Linear Regression, Hazard Duration, Multinomial Logit, Binary Logit, Location Choice, Ordered Probit or Work Start/End Time. Section 4.4.5 takes an example of each of these types to explain how they must be configured.

Note: For Testing purposes, the reader is recommended to skip to Section 4.4.4 to load the ready-made model configuration file provided in the CEMDAP CD (labeled as "CEMDAP_DFW_Model_Specification_file.cml").

4.4.1 Accessing model module dialog boxes through the Models Menu

The model configuration dialog boxes for each of the model components can be accessed through the *Models Menu*. Each entry in the *Models Menu* allows access to a specific suite of model components. Clicking on any of the models in each suite opens up the corresponding dialog box to configure the model (see the following figures). For example, the figure below shows that the models (such as "Children's decision to go to school") can all be accessed by clicking on "Work and School Activities" under

"Generation Allocation". The subsequent figures (in the following pages) show how each of the remaining CEMDAP model components can be directly accessed from the menu items.

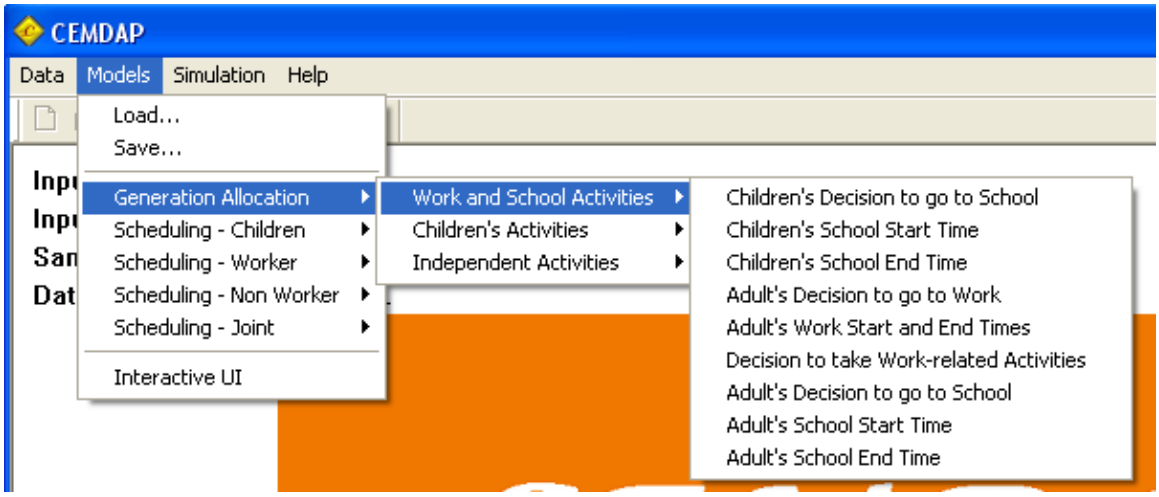

Figure 4.11: List of Work and School Activities Models within Generation Allocation in the Models menu

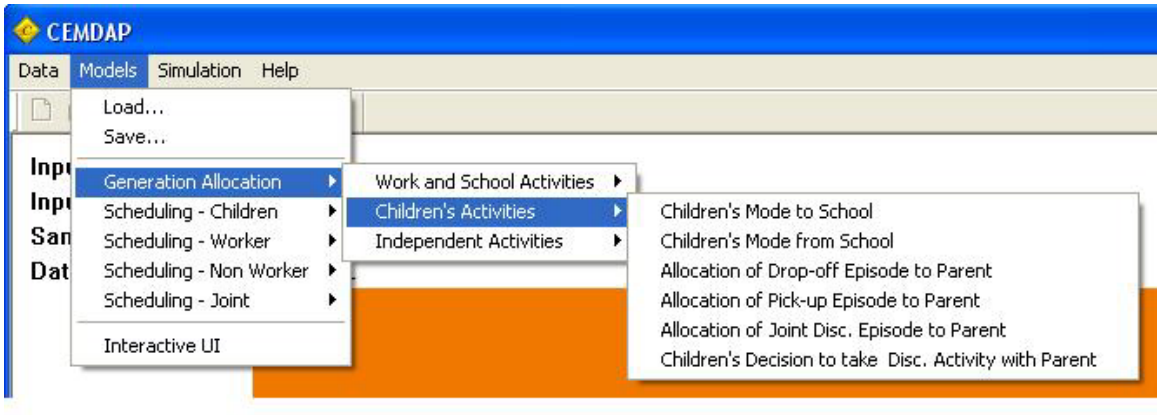

Figure 4.12: List of Children's Activities Models within Generation Allocation in the Models menu

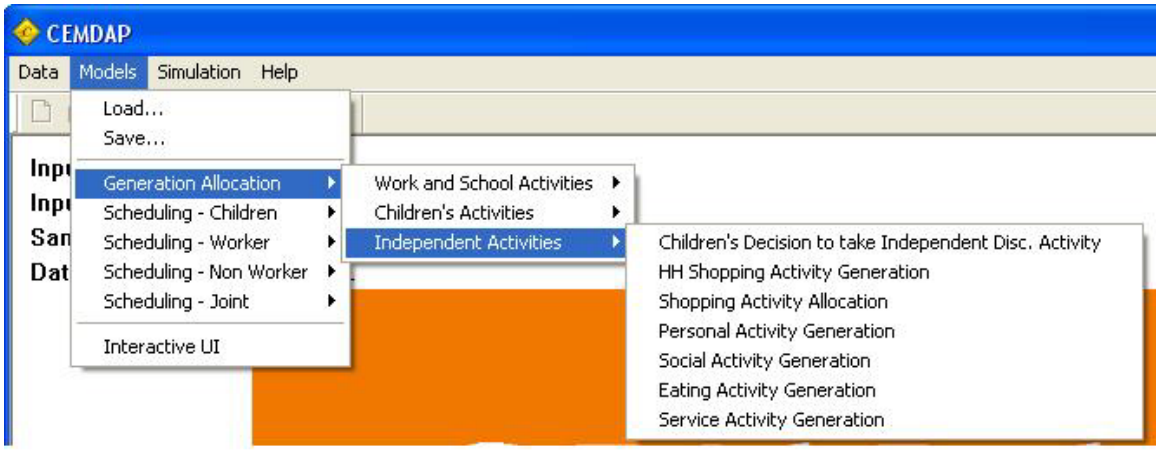

Figure 4.13: List of Independent Activities Models within Generation Allocation in the Models menu

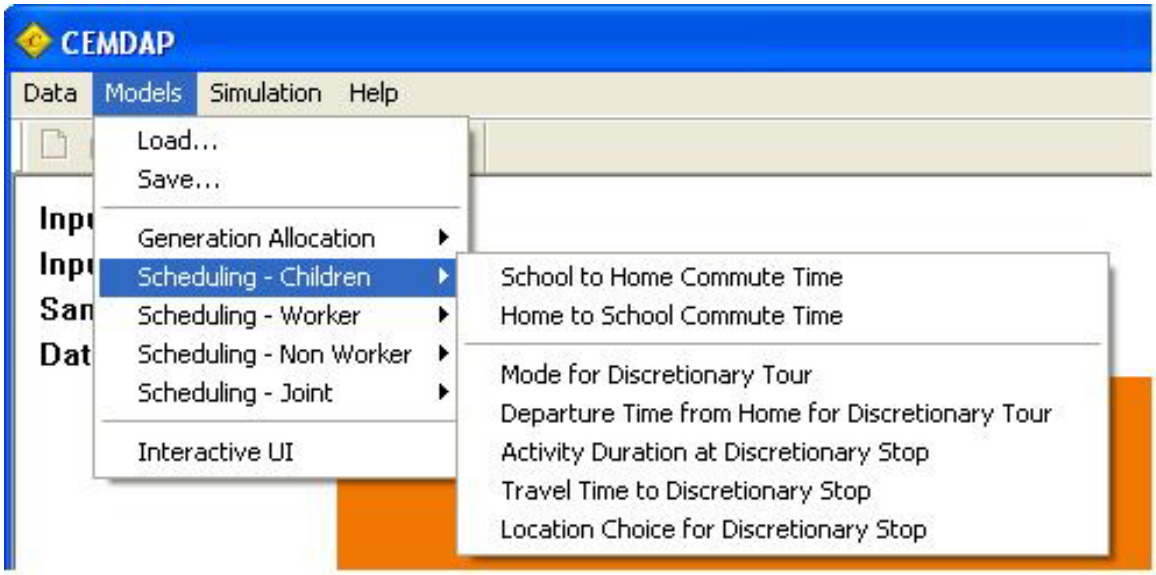

Figure 4.14: List of Scheduling – Children Models within the Models menu

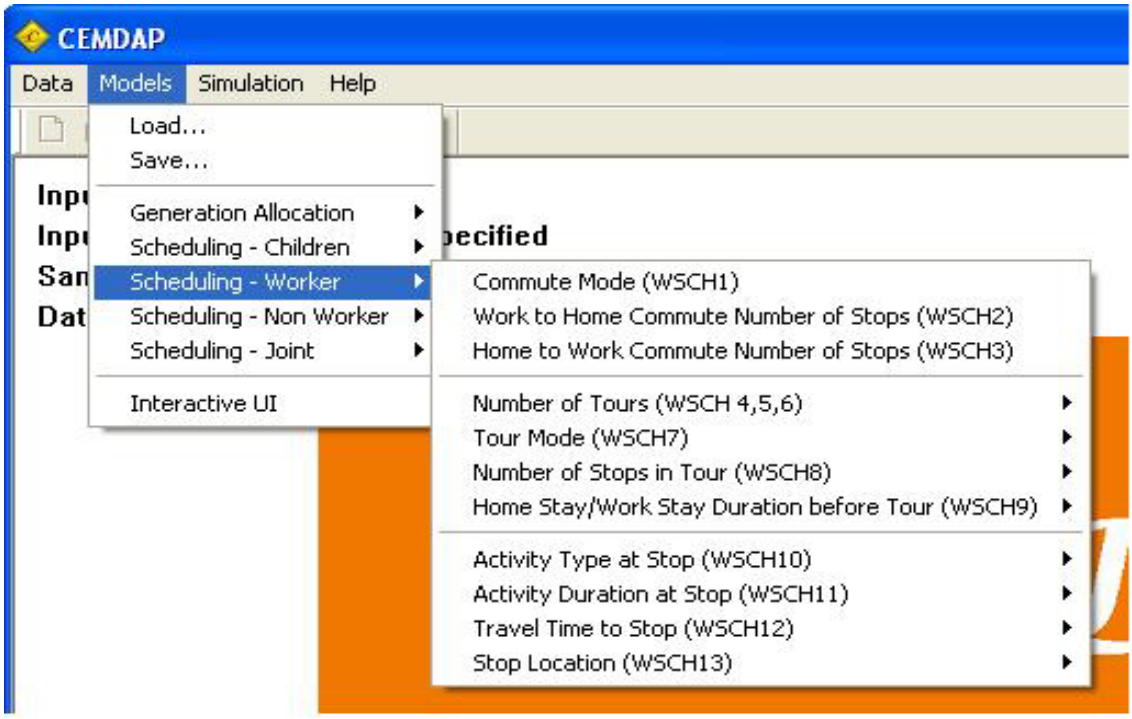

Figure 4.15: List of Scheduling – Worker Models within the Models menu

Figure 4.16: List of Number of Tours Models within Scheduling-Worker in the Models menu

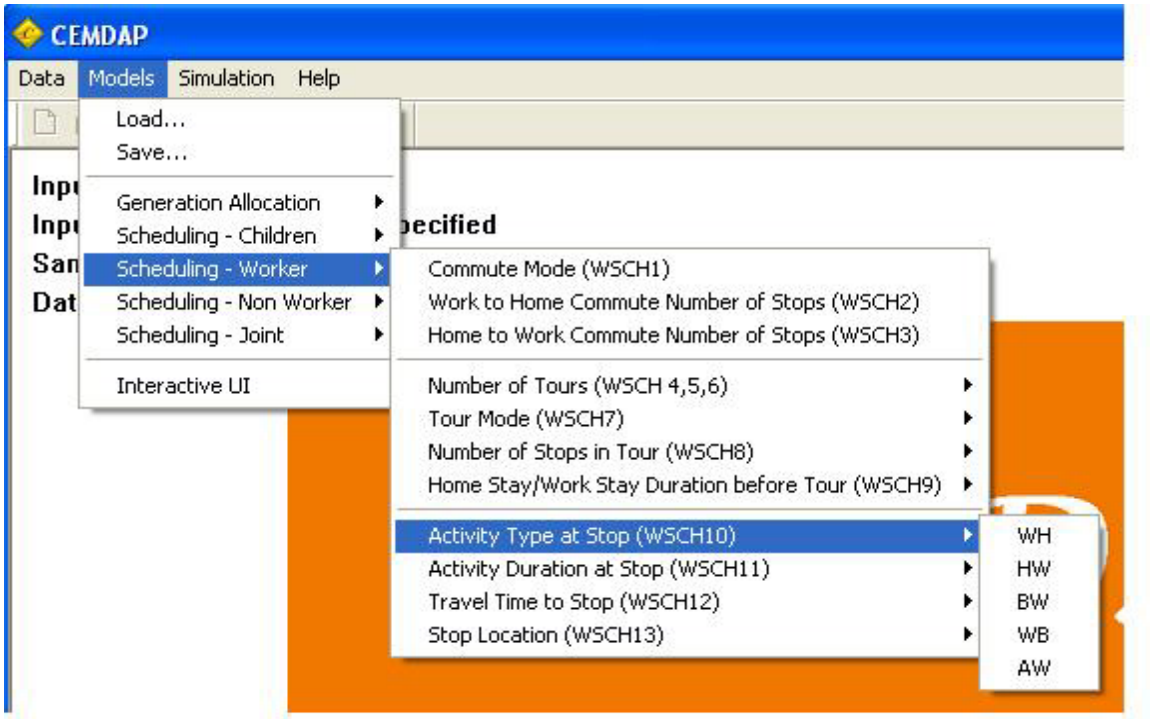

Figure 4.17: List of Activity Type at Stop Models within Scheduling-Worker in the Models menu

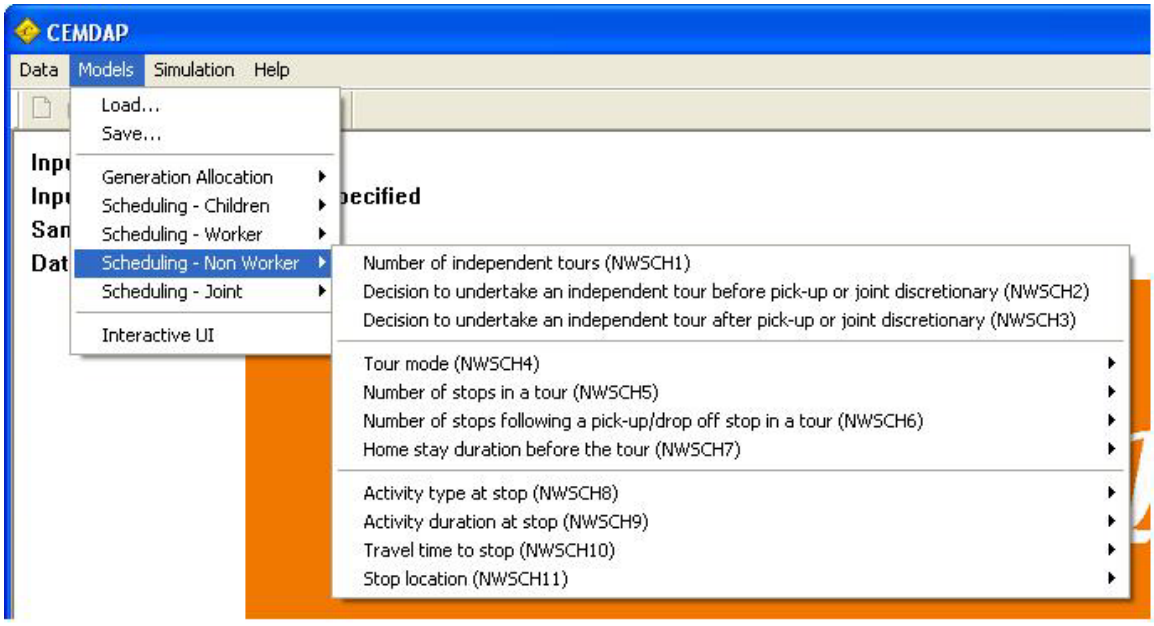

Figure 4.18: List of Scheduling – Non Worker Models within the Models menu

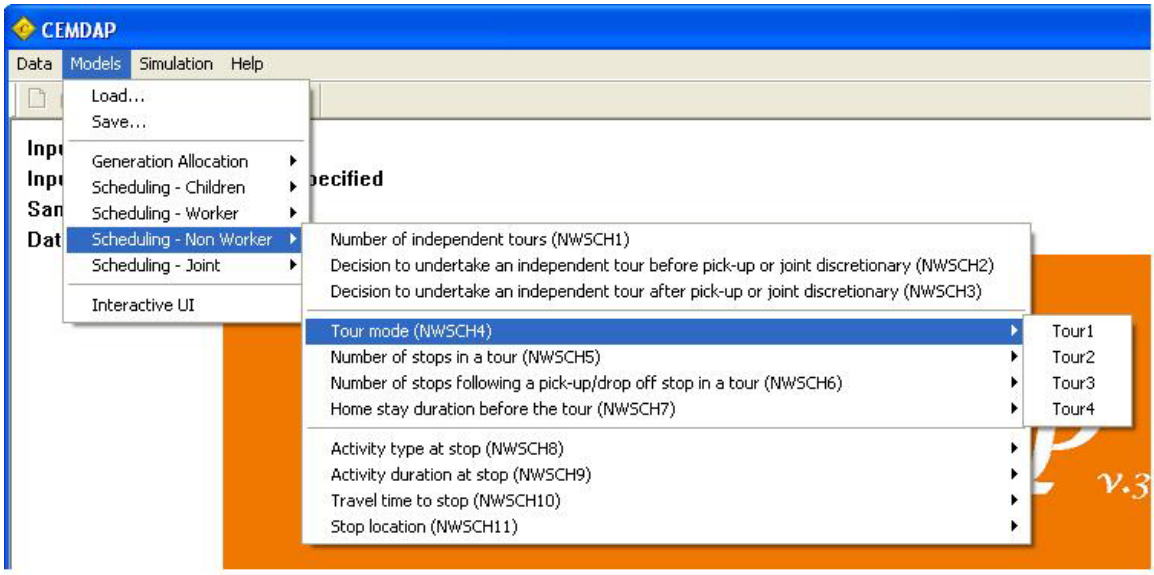

Figure 4.19: List of Tour Mode Models within Scheduling – Non Worker in the Models menu

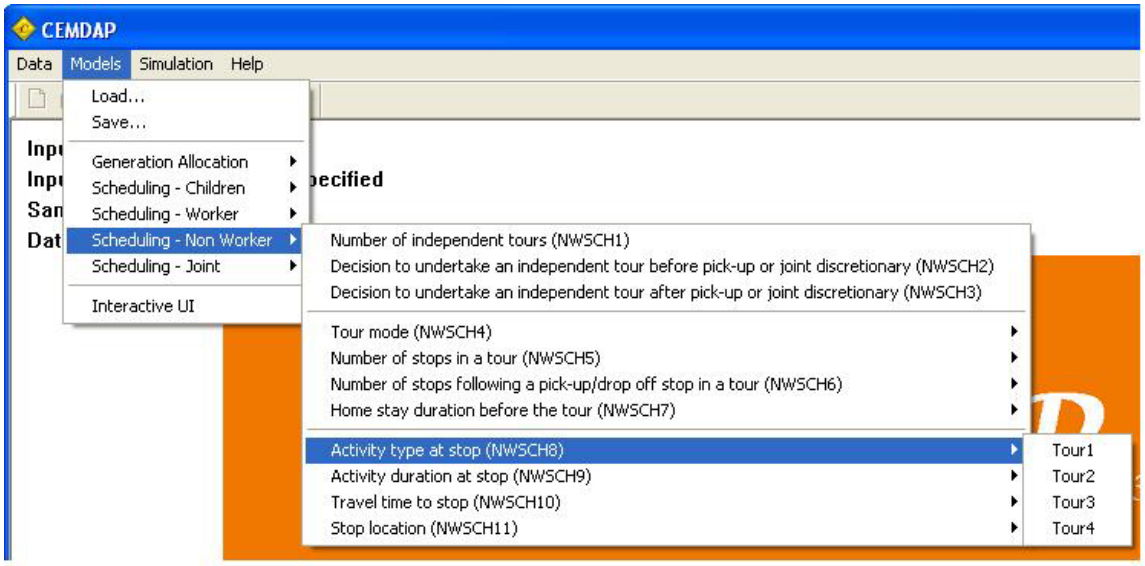

Figure 4.20: List of Activity Type at Stop Models within Scheduling – Non Worker in the Models menu

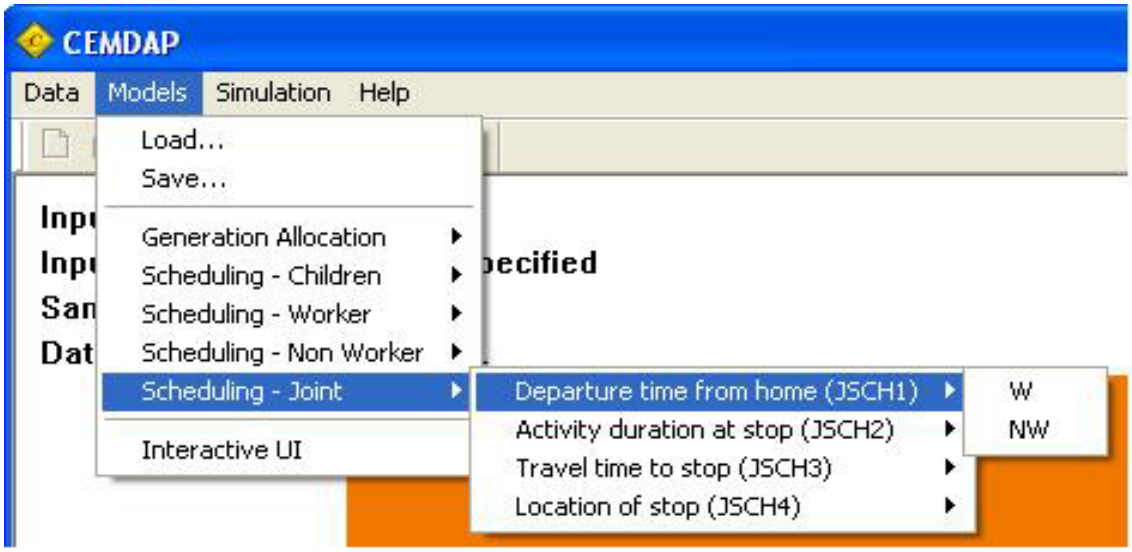

Figure 4.21: List of Departure Time from Home Models within Scheduling – Joint in the Models menu

4.4.2 Accessing model module dialog boxes through the Interactive User Interface

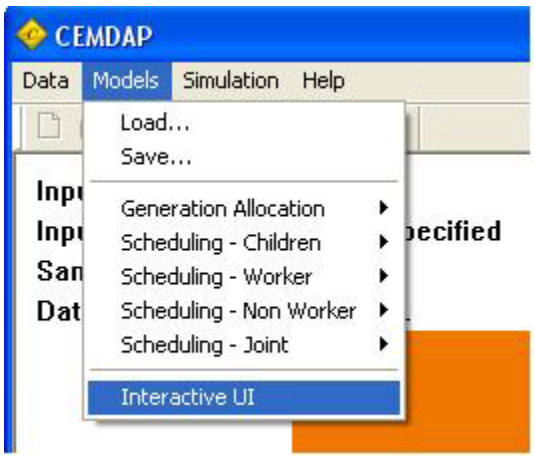

Figure 4.22: Interactive UI in the Models menu

The *Interactive UI* command in the *Models Menu* opens up the interactive user interface that ties together the model configuration dialog boxes for all the model components that constitute the embedded model system. This interactive UI is also reflective of the underlying modeling framework embedded within CEMDAP. Specifically, the analyst can understand the process used by CEMDAP to generate the activity-travel patterns by following flow represented by the Interactive UI windows. The main dialog box of the interactive user interface looks as shown below.

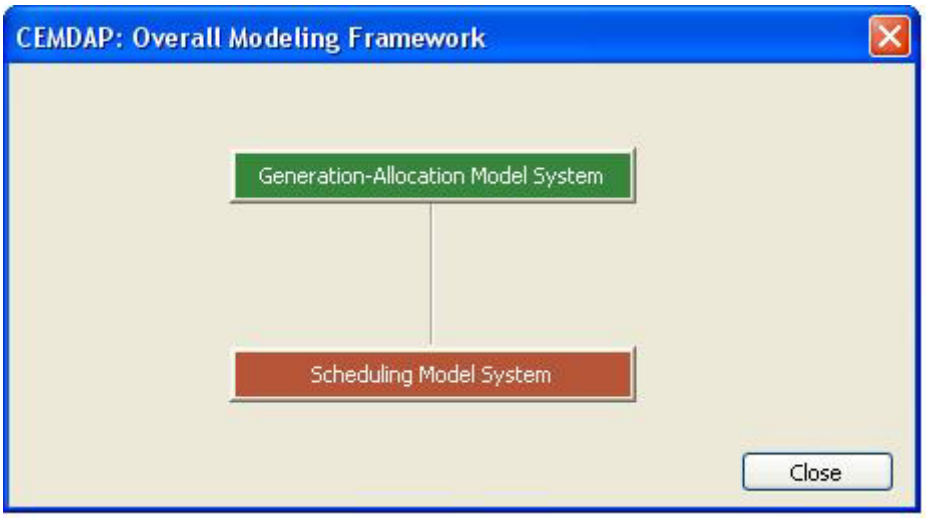

Figure 4.23: Main Dialog Box of the Interactive User Interface

The *Generation-Allocation Model System* button opens up a dialog box within the generation-allocation modeling framework as shown below with buttons for (1) *Generation and Timing of Work and School Activities of Adults and Children*, (2) *Generation of Children's Travel Needs and Allocation of Responsibilities to Parents*, and (3) *Generation of Independent Activities by Adults and Children for Personal and Household Serving Needs*.

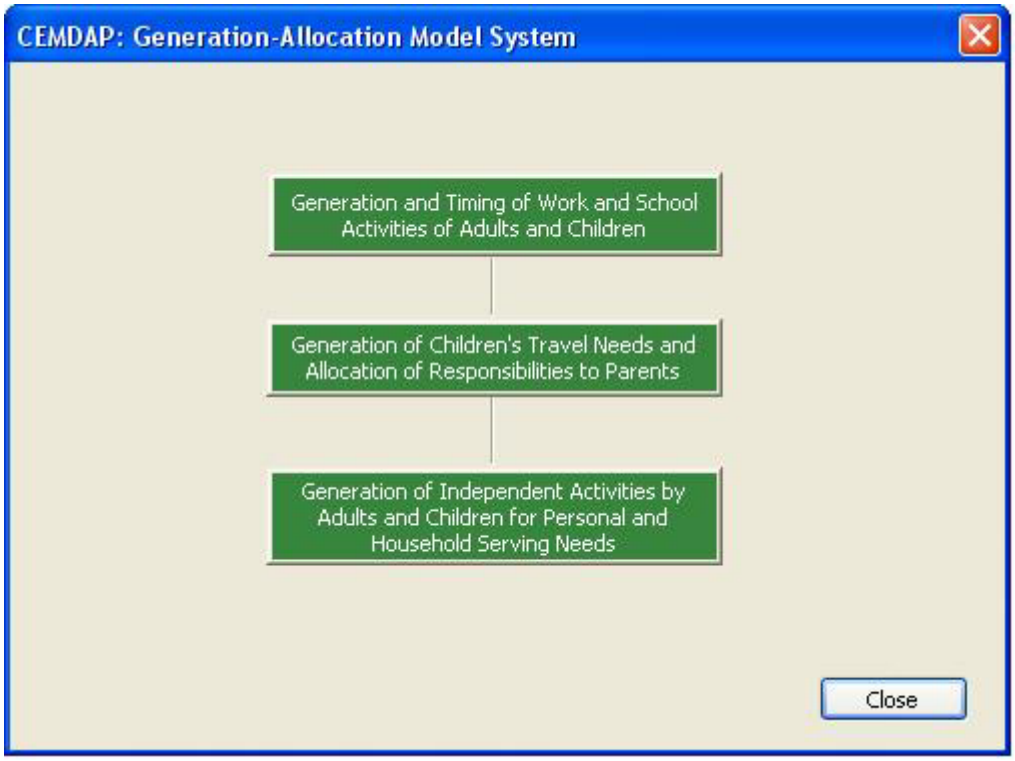

Figure 4.24: Generation-Allocation Model System Interactive User Interface

Clicking on each of the buttons opens up the corresponding suite of individual model components (shown in the following pages).

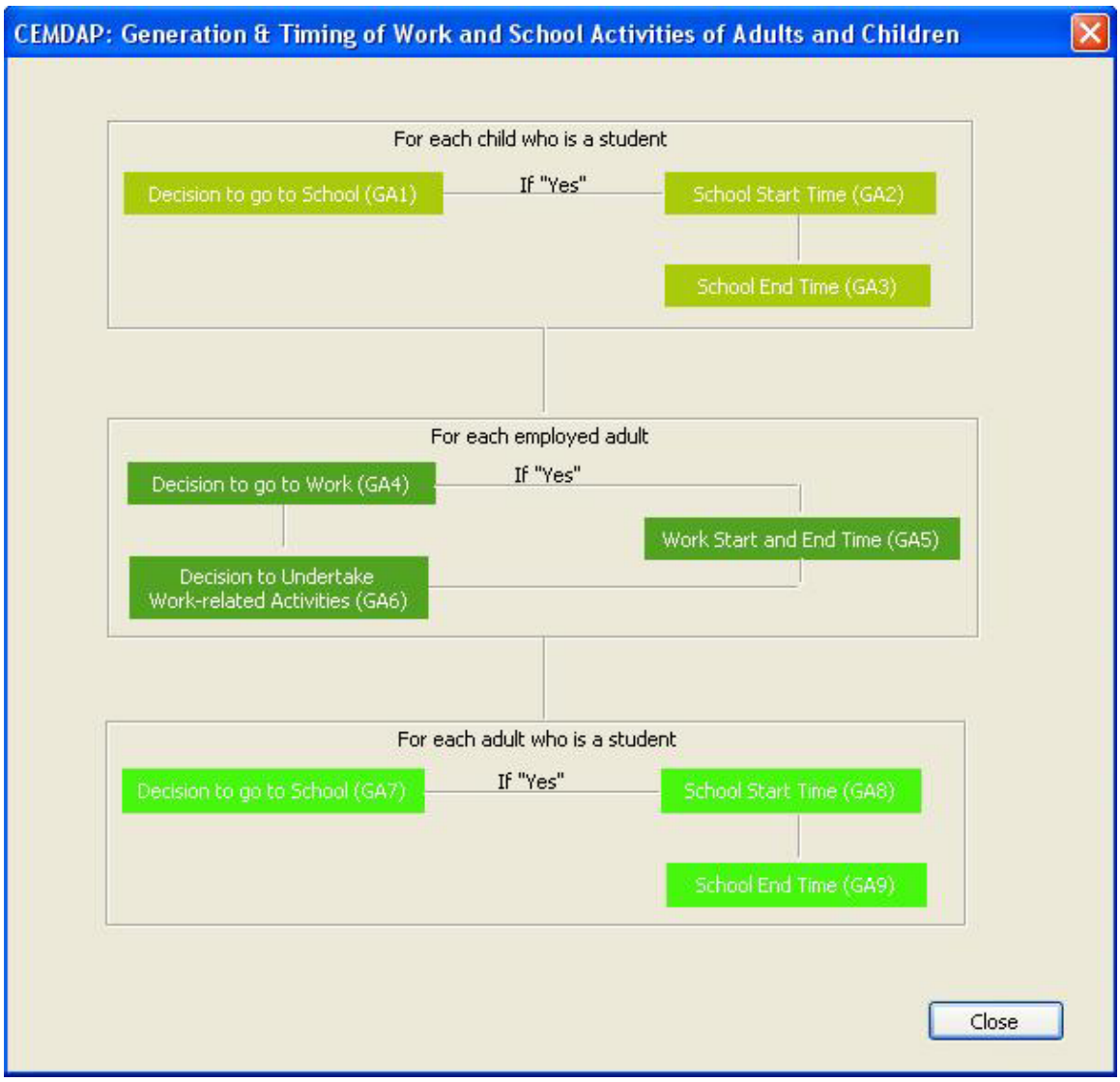

Figure 4.25: Generation & Timing of Work and School Activities of Adults Interactive User Interface

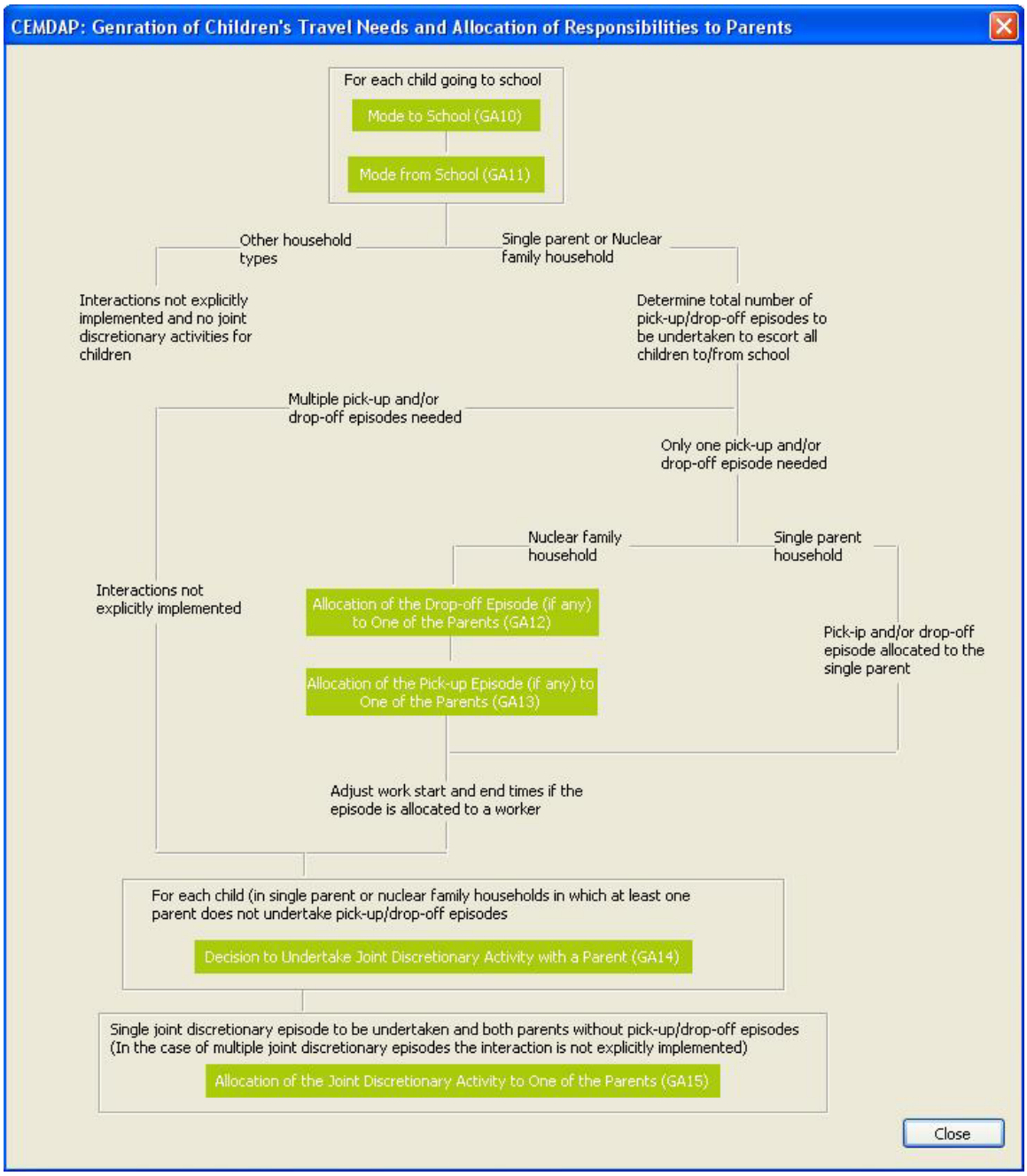

Figure 4.26: Generation of Children's Travel Needs and Allocation of Responsibilities to Parents Interactive User Interface

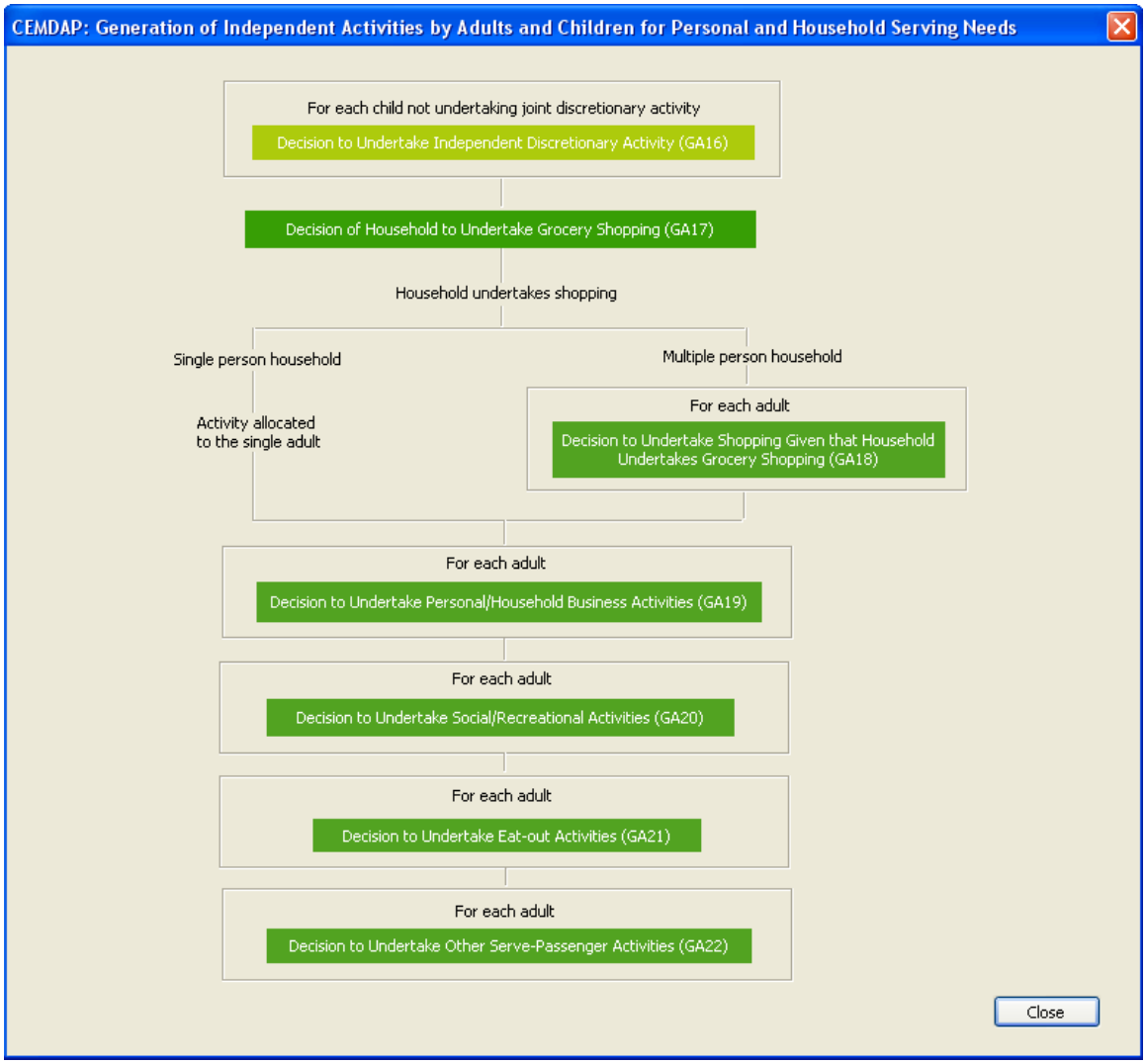

Figure 4.27: Generation of Independent Activities by Adults and Children for Personal and Household Serving Needs Interactive User Interface

When all the components of the generation-allocation model system have been configured, click *Close* to return to the main dialog box. The scheduling system user interface (shown on the next page) can be obtained by clicking on the *Scheduling Model System* button in the *Overall Modeling Framework* box. The *Scheduling Model System* box has buttons for *Work to Home Commute, Home to Work Commute, Non Worker Tour Containing the Drop-off at School Activity, Non Worker Tour Containing the Pick-up at School Activity, School-Home and Home-School Commutes, Worker's Joint Discretionary Tour with Children, Non Worker's Joint Discretionary Tour with Children, Independent Home-based and Work-based Tours for workers, Independent Home-based Tours for non workers,* and *Independent Home-based Tours Undertaken by Children.*

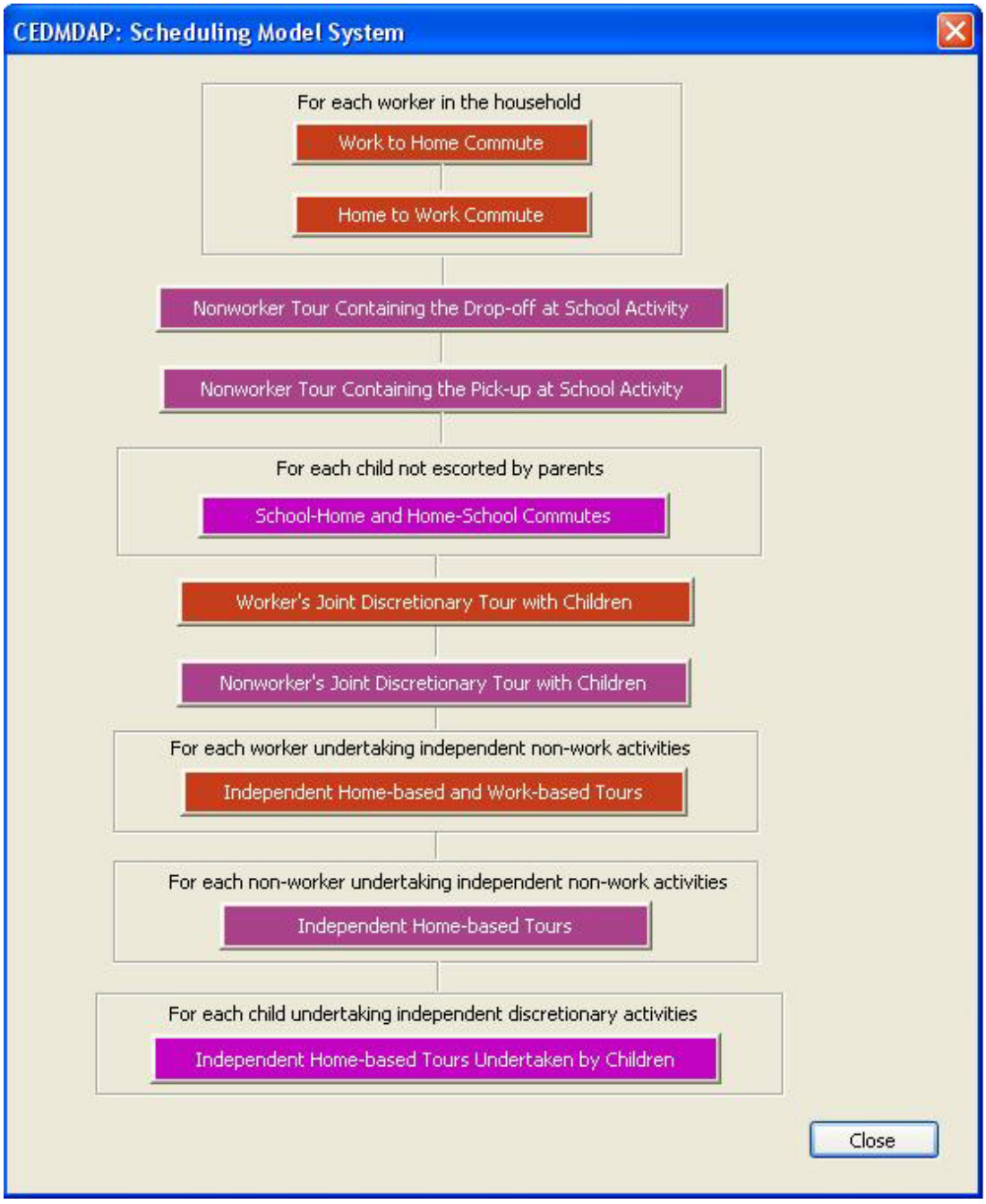

Figure 4.28: Scheduling Model System Interactive User Interface

Clicking on each of the buttons opens up the corresponding suite of individual model components (shown on the following pages). After configuring the individual models of each suite, click the *Close* button to return to the *Scheduling Model System* interface, in order to access the next suite of models.

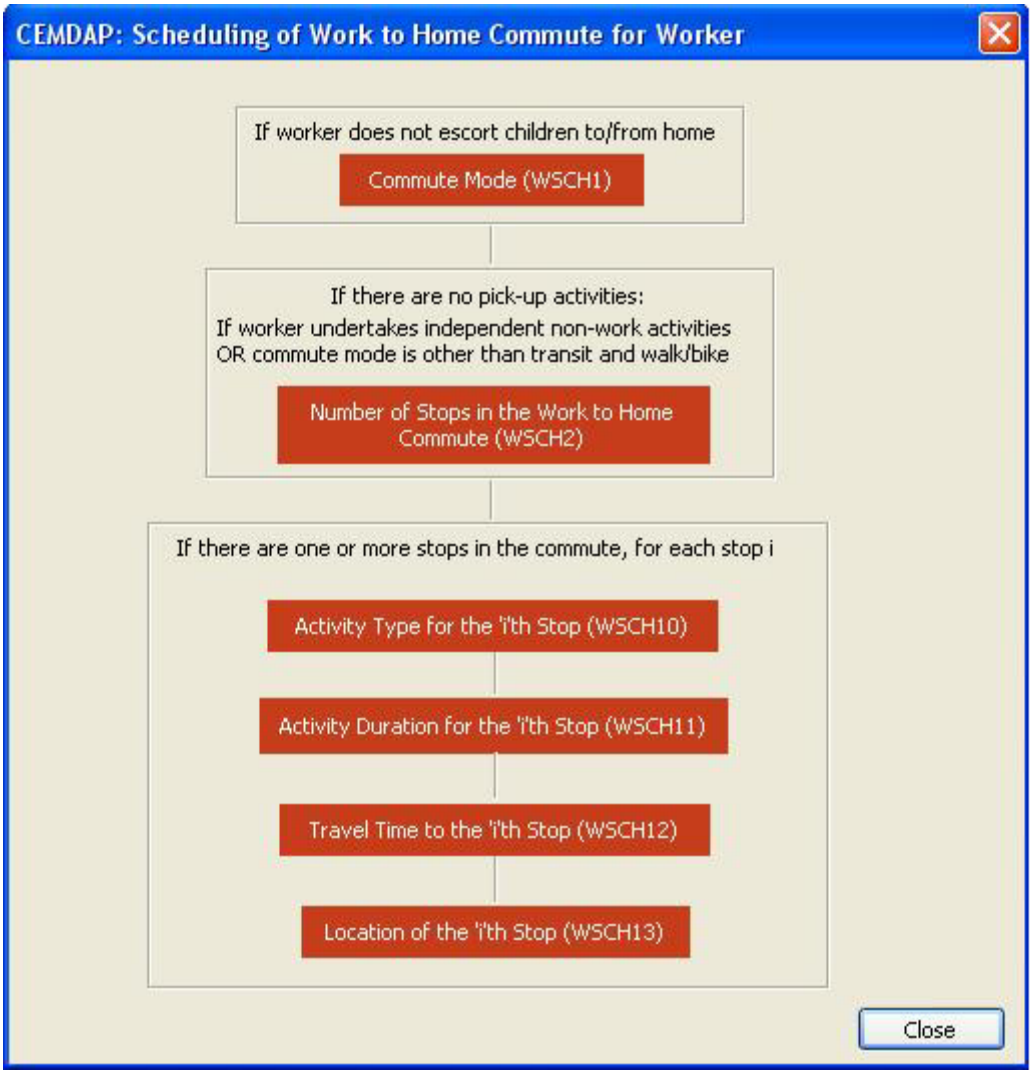

Figure 4.29: Scheduling of Work to Home Commute for Worker Interactive User Interface

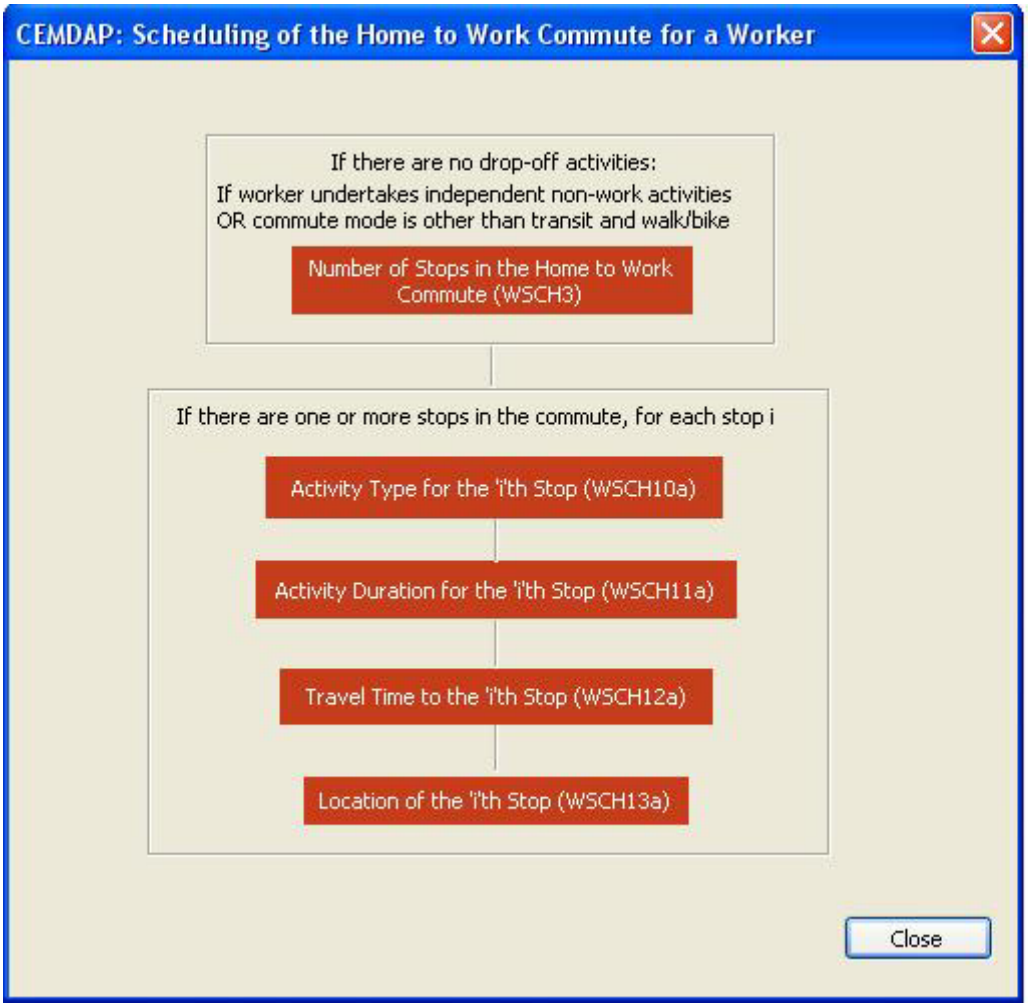

Figure 4.30: Scheduling of the Home to Work Commute for a Worker Interactive User Interface

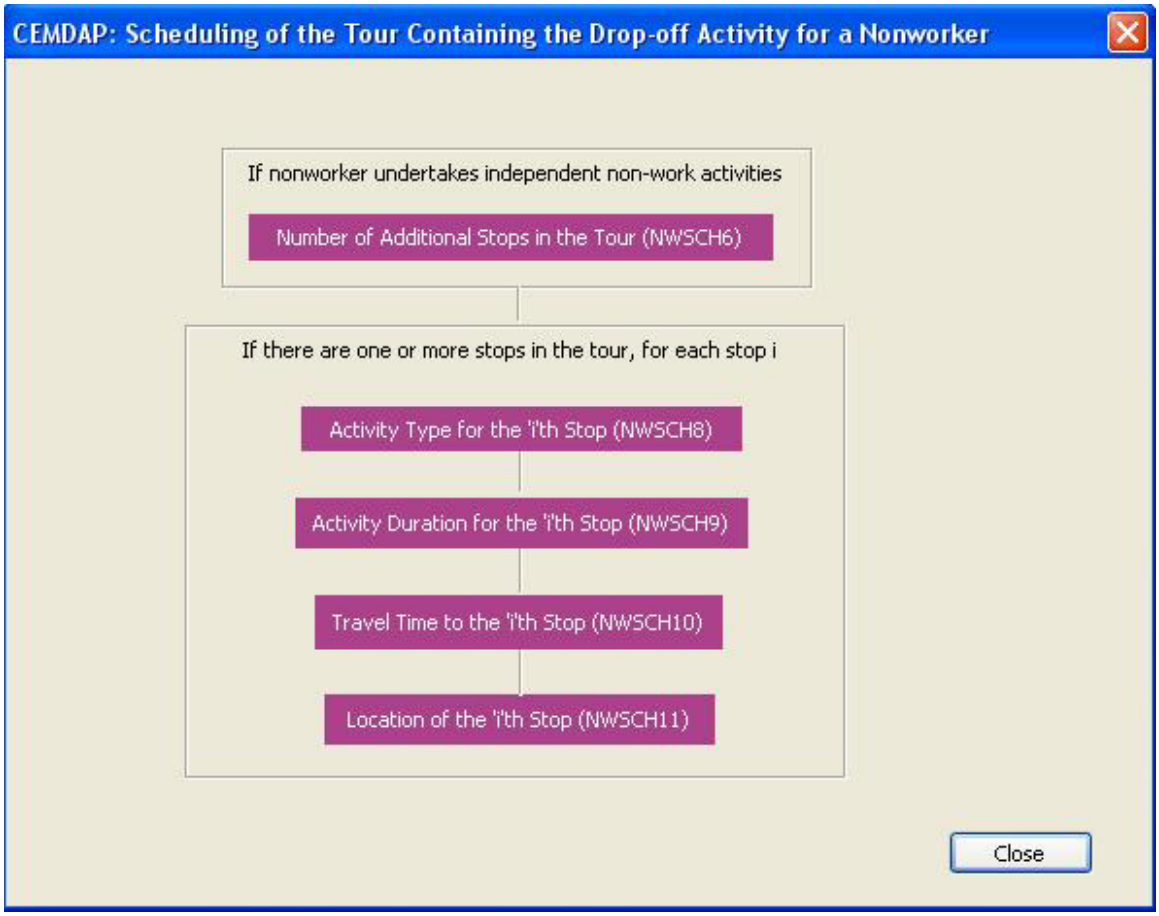

Figure 4.31: Scheduling of the Tour Containing the Drop-off Activity for a Nonworker Interactive User Interface

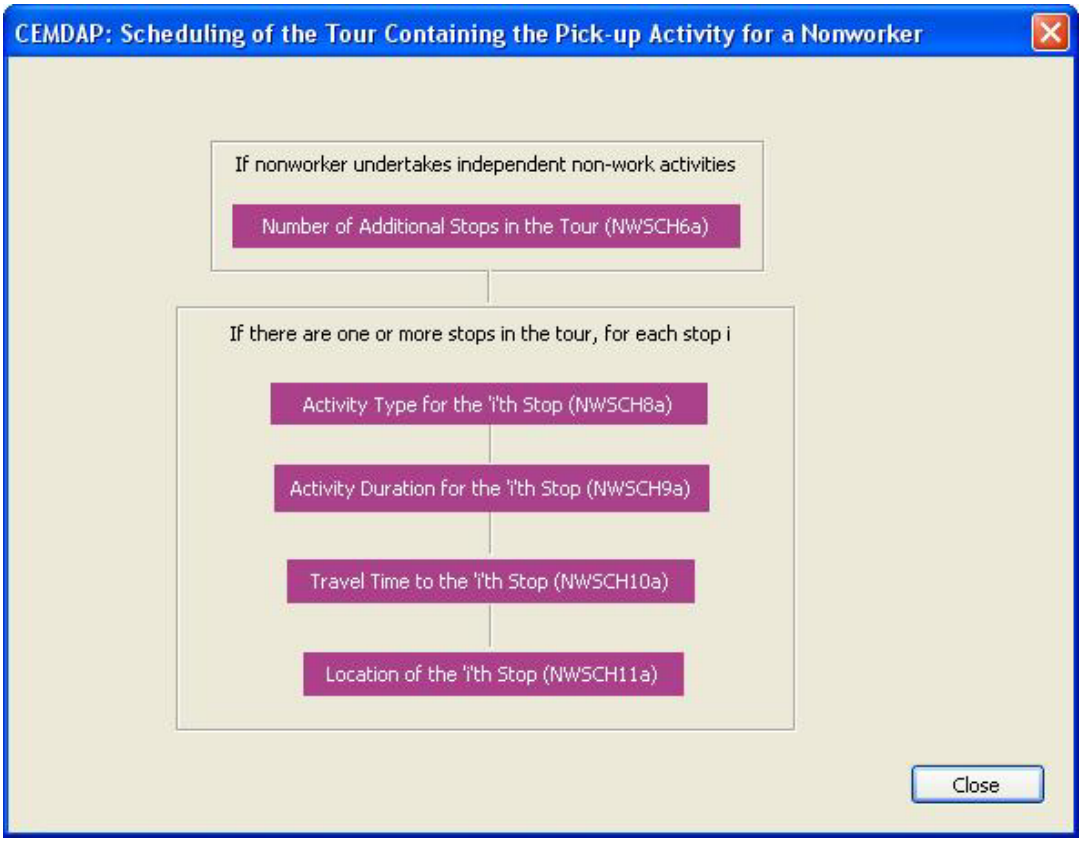

Figure 4.32: Scheduling of the Tour Containing the Pick-up Activity for a Nonworker Interactive User Interface

Figure 4.33: Scheduling of the Tour Containing the Pick-up Activity for a Nonworker Interactive User Interface

Figure 4.34: Scheduling of the Joint Discretionary Tour Undertaken by a Worker with Children Interactive User Interface

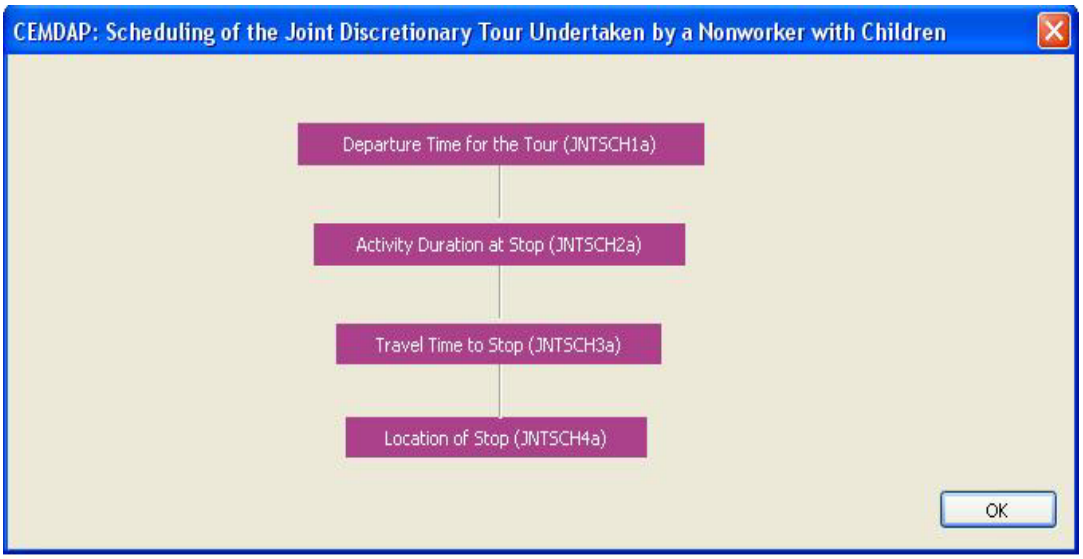

Figure 4.35: Scheduling of the Joint Discretionary Tour Undertaken by a Nonworker with Children Interactive User Interface

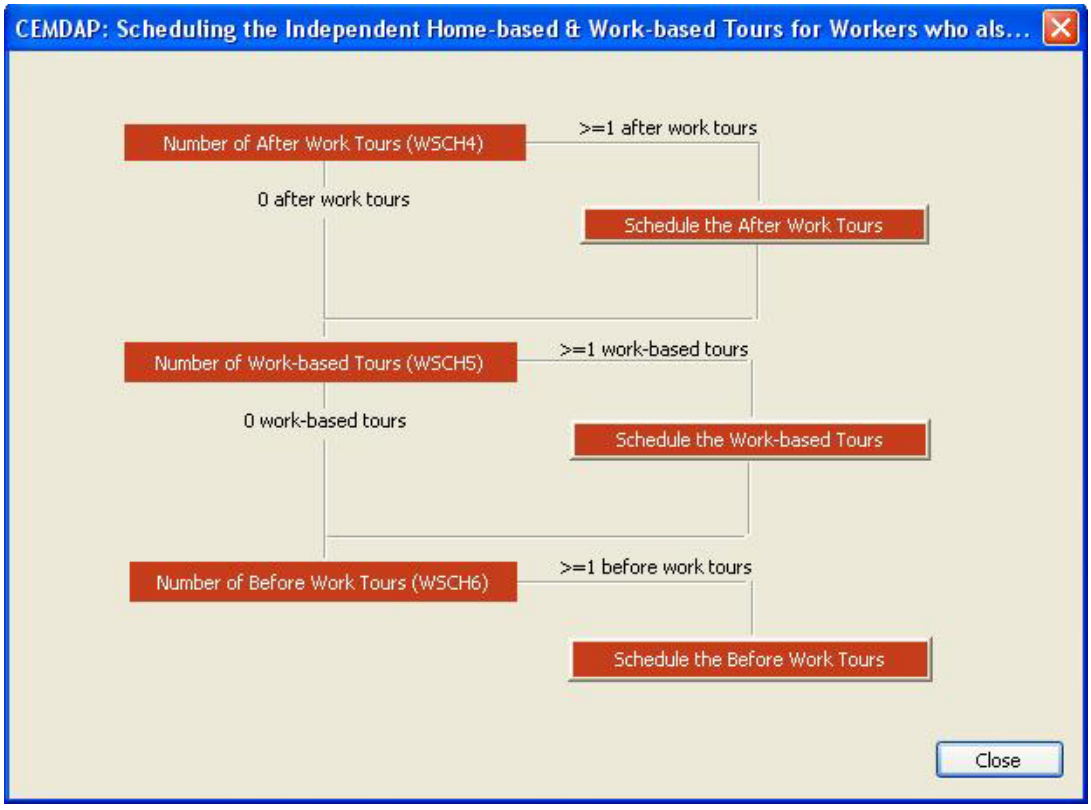

Figure 4.36: Scheduling of the Independent Home-based & Work-based Tours for Workers… Interactive User Interface

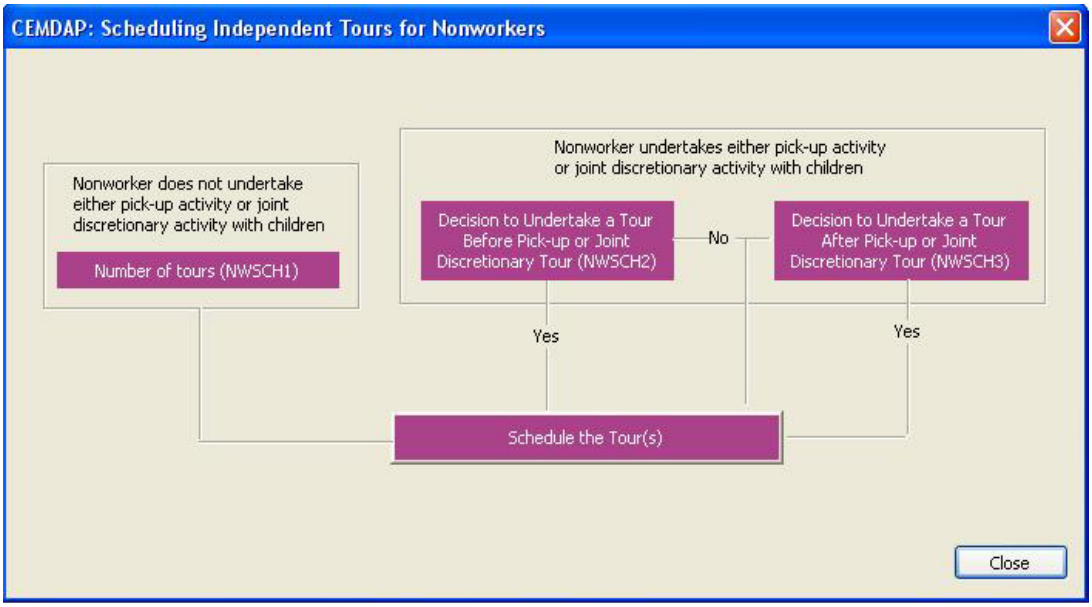

Figure 4.37: Scheduling Independent Tours for Nonworkers Interactive User Interface

Figure 4.38: Scheduling the Independent Discretionary Tour Undertaken by a Child Interactive User Interface

When the entire scheduling model system for workers has been configured, return to the main dialog box by clicking *Close* till the main dialog box appears.

4.4.3 Saving model configuration file

The model configurations entered using the dialog boxes can be saved into a model configuration file for future use. This is achieved by clicking on the *Save* command in the *Models Menu.*

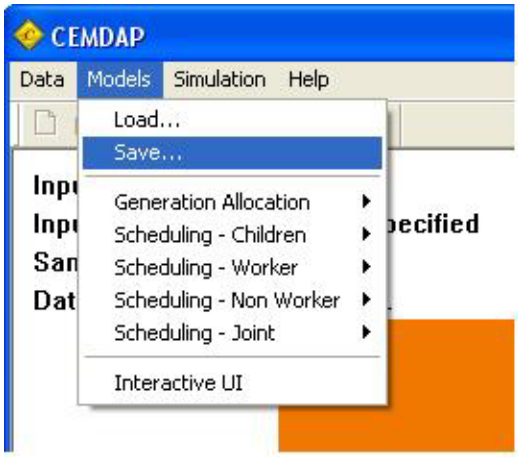

Figure 4.39: Save command in the Models Menu

The *Save* command opens the *Save As* dialog box. Choose the desired location and type in the desired model configuration filename in this dialog box, and click *Save* to finish.

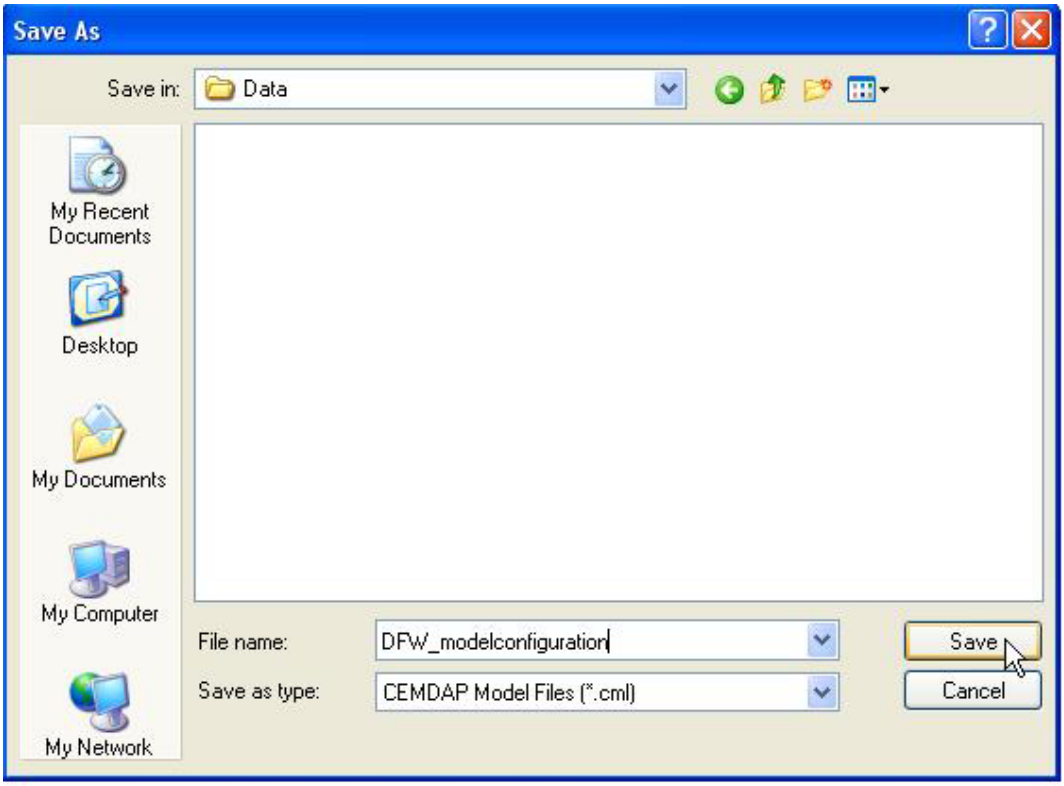

Figure 4.40: Save As Dialog Box from the Save Command

4.4.4 Loading model configuration file

In the current section instructions to load the prepared model configuration file are provided. The model configuration file provided in the installation CD for testing purposes (or saved using the *Save* command in the *Models Menu*, or created manually by entering the model configurations into a text file), can be loaded into CEMDAP using the *Load* command in the *Models Menu*.

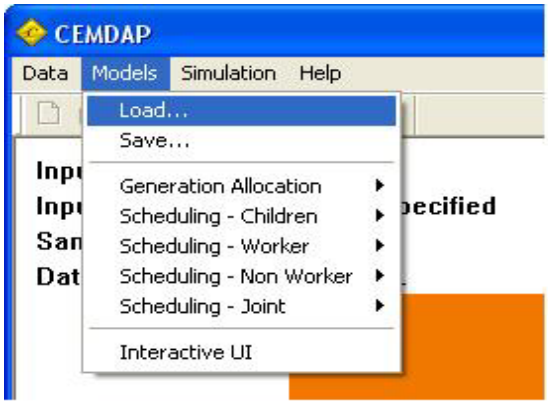

Figure 4.41: Load Command in the Models Menu

The *Load* command opens the *Save As* dialog box (note that the model configuration file must have already been saved as indicated in the previous section). Navigate to the location of the model configuration file and select the file. Click *Save* to finish loading the configuration file.

Note: For Testing the CEMDAP software, the reader is recommended to skip to Section 4.5

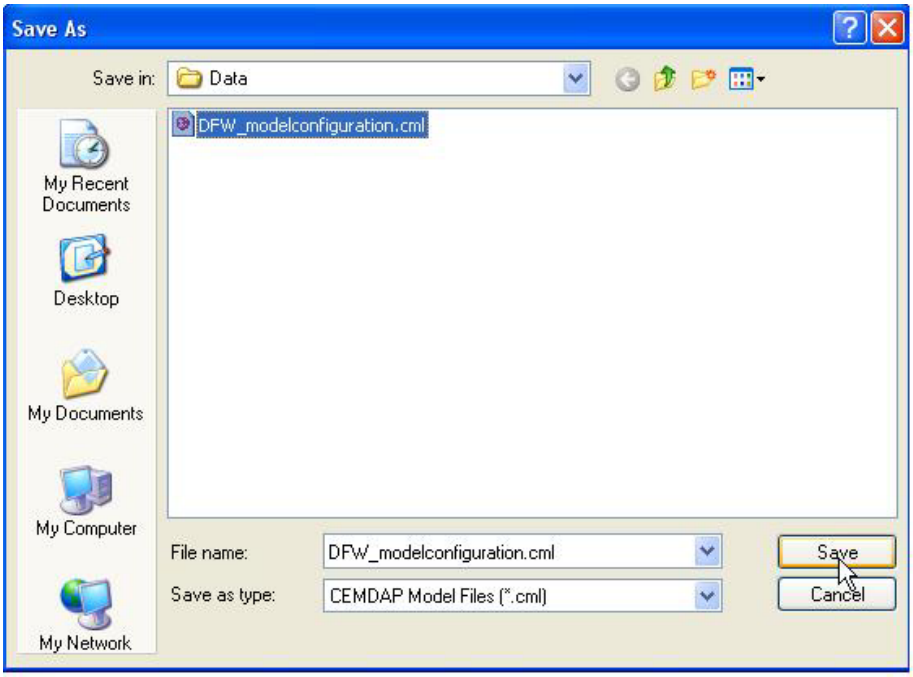

Figure 4.42: Save As Dialog Box from the Load Command

4.4.5 Configuring specific model types

All the model components in the embedded model system fall into one of seven categories based on their econometric structures: Linear Regression, Hazard Duration, Binary Logit, Multinomial Logit, Location Choice, Ordered Probit, or Work Start/End Time (please see Guo et al. 2005 for a discussion on the structure of work start/end time model, and Pinjari et al. 2006 for all the model components and their prescribed econometric structures in CEMDAP). In this section, we take an example of each of these model types and illustrate their methods of configuration. (Note: Clicking *OK* in any dialog box only saves the information temporarily. In order to save a model configuration, choose the *Save* command in the Models Menu).

Linear Regression

An example of a linear regression model is the Adult's School Start Time. (Note that the dependent variable in the regression models for duration is assumed to be the log of the duration. This is done to ensure that we always predict a positive value).

The inputs required for this model are the variance parameter, the variable parameters, and variables in the linear regression equation. First, enter the variance in the box titled '*Variance*' in the top left corner of the dialog box. Then enter the linear regression equation by picking the relevant variable from the *Variable* list and entering its parameter in the space designated '*Coefficient*'. Click the arrow button (=>) to add the term to the linear regression expression. The '*From*' option lists the sets of variables available. '*Households'* contains all the household-related variables from the input

household table. '*Persons'* contains the person-related variables from the input person table and *'Zones'* contains the zonal variables from the input zones table. *'Other'* (which is highlighted in the figure below) is a list of other variables including the constant, any interaction terms and variables that may be generated during the simulation process by models higher up in the modeling framework.

In the figure below, we enter the constant by selecting '*Other*' in the '*From*' list and '*Constant (One)*' from the corresponding *Variable* list. Enter the value of the constant in the *Coefficient* box and click the arrow button (=>) to enter the constant.

The reader will note that the number that appears at the beginning of variables in the *'Other'* list is the ID of the variable. It is the variable ID that gets stored in the model specification file and it is the variable ID that is recognized by CEMDAP. (These IDs are shown only for endogenous variables).

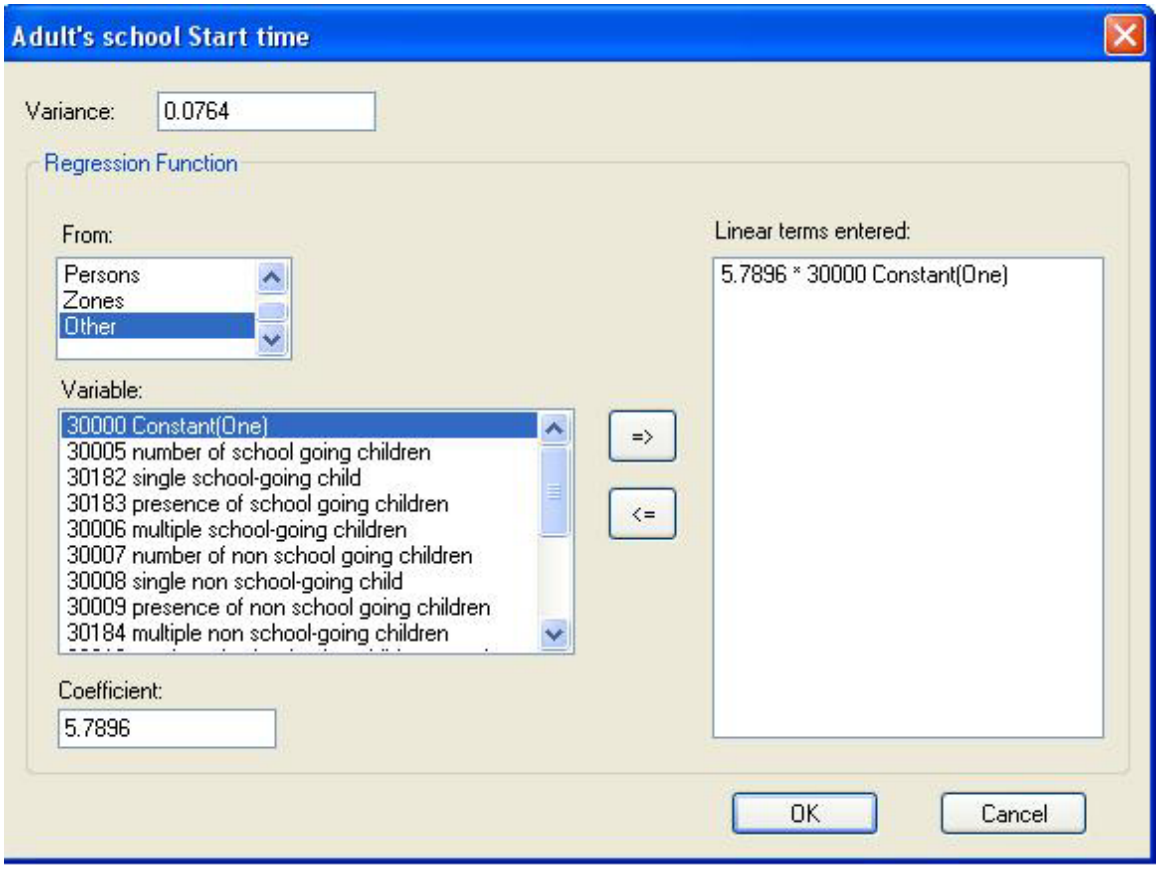

Figure 4.43: Linear Regression Model Interactive User Interface

When the linear regression equation has been completely configured, the dialog box looks as shown below. The arrow buttons (=> and <=) can be used to add, remove, or edit terms.

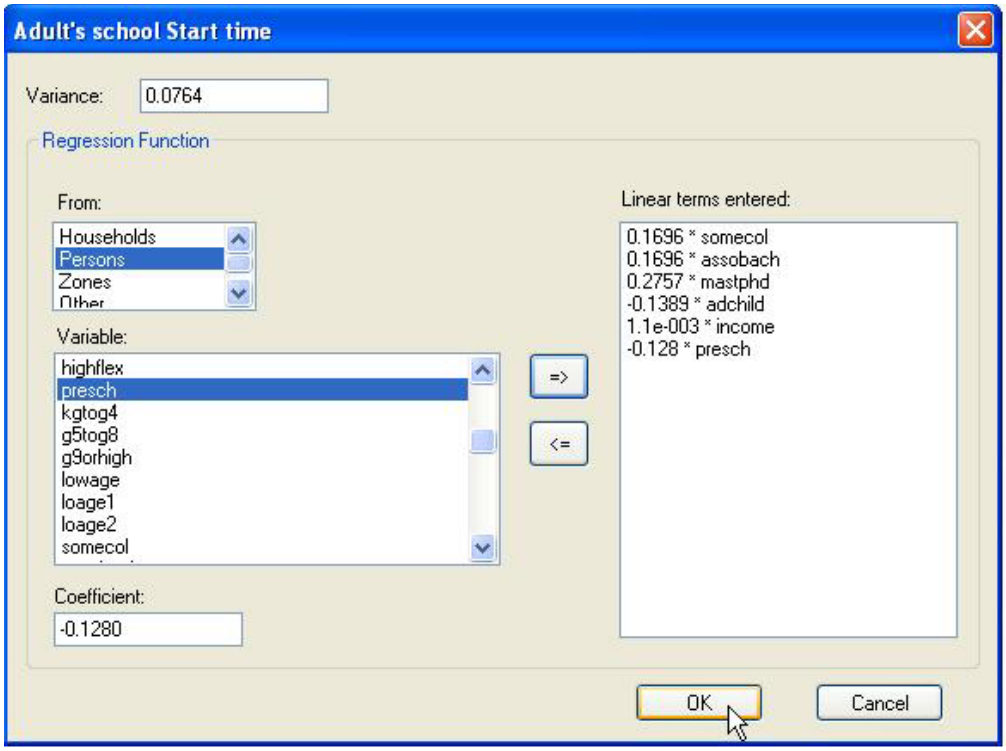

Figure 4.44: Example of Fully Configured Linear Regression Model

Once the configuration of the individual models is finished, click *OK* to close the model dialog box, and then click the *Close* button to return to the corresponding model system interface, and then access the next suite of models. When the entire model system has been configured, return to the main dialog box by clicking *Close* till the main dialog box appears. This process is similar for all the individual models embedded in the model system.

Hazard Duration

An example of a hazard duration model is the Children's School Start Time model within the generation-allocation model system. The model type supported is a hazard-based duration model with a non-parametric baseline and gamma heterogeneity. The inputs required for this model can be configured in three steps. Step One: enter the parameter estimated for the gamma heterogeneity term in the box titled '*Variance*' in the top left corner of the dialog box. Step Two: configure the parameters on the covariates. This procedure is identical to entering the linear regression parameters. Select the relevant input table from the '*From*' list, and pick the variables from the *Variable* list. Specify the coefficient for each variable and use the arrow button (=>) to add the terms.

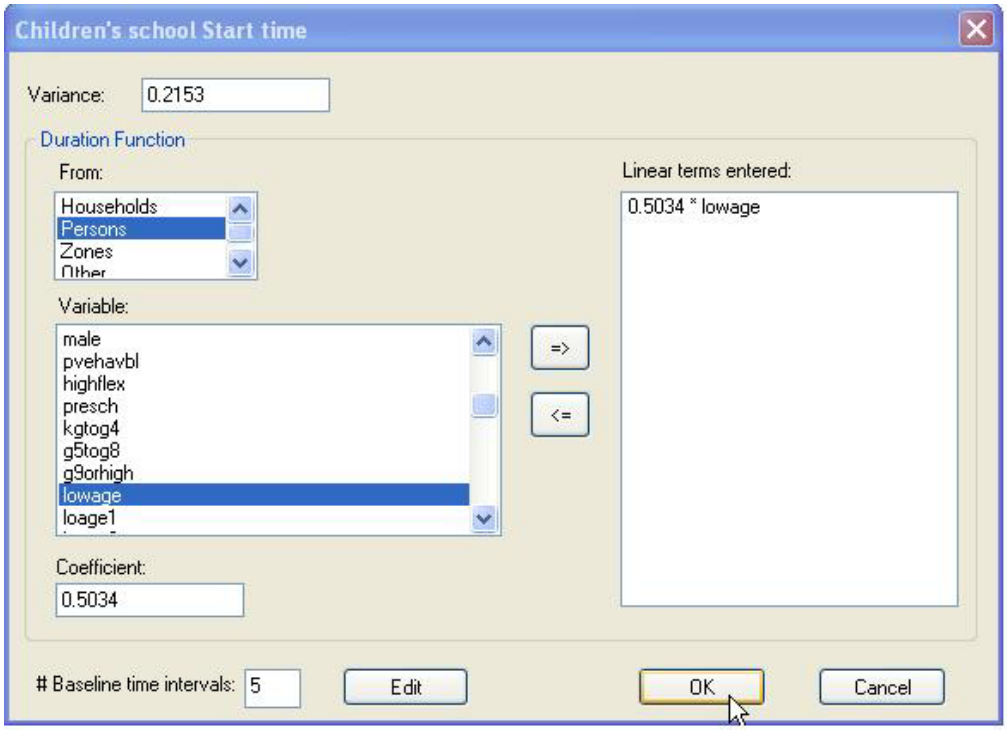

Figure 4.45: Hazard Duration Model Interactive User Interface

Step Three: after all the covariates have been configured (refer to the figure on the next page), enter the threshold parameters. First, in the box titled '*# Baseline time intervals*' enter the number of discrete time periods (N) into which the duration is divided (Note: Please do not enter zero because there cannot be zero number of discrete time periods). Then, to enter the threshold parameters (number of threshold parameters = N-1), click *Edit*. Additional boxes open up to enter the start times of each of the discrete periods and the corresponding parameter estimates (sequentially from start time of the second discrete period to the start time of the last discrete period, hence there are N-1 parameters). (Note: The end time of any discrete duration period is the start time of the next period. The start time of the first discrete period is to be specified as zero, while the end time of the last discrete period, which is not specified, is implicitly taken as infinity.). Click on the arrow buttons (=> and <=) to scroll. When the hazard model is fully configured, the dialog box looks as shown below. Note: If the user needs to change the number of base line time intervals after clicking *Edit*, it is advised that the model configuration be cancelled and reconfigured.

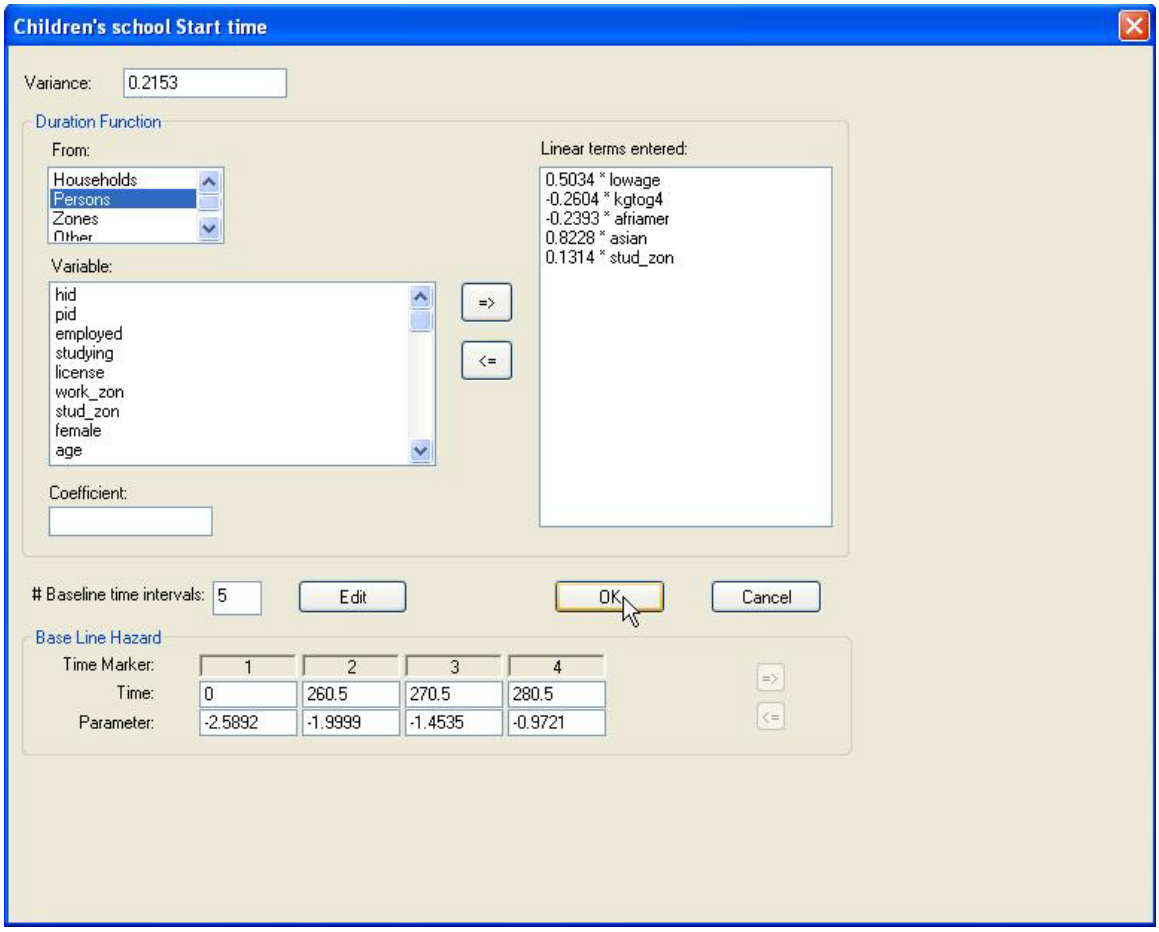

Figure 4.46: Example of Fully Configured Hazard Duration Model

Multinomial Logit

An example of a multinomial logit model is the Commute Mode Choice Model within the Worker's Activity-Travel Scheduling model system. The discrete choices for this model are drive alone, share auto ride as a passenger, share auto ride as a driver, transit, and walk/bike. The model configuration dialog box for this looks as follows.

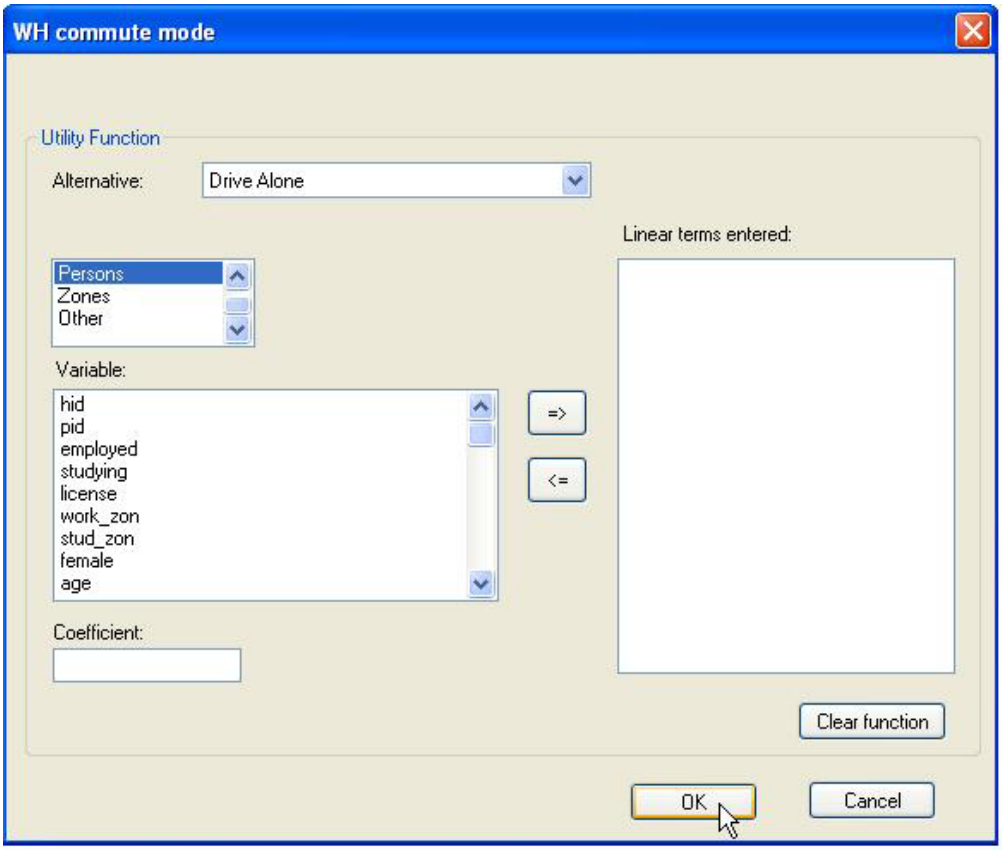

Figure 4.47: Multinomial Logit Model Interactive User Interface

To configure the multinomial logit model, the utility functions of each of the alternatives must be specified. Select the alternatives one at a time from the drop down menu titled '*Alternative*' to configure the corresponding utility function. In the example shown above, we have selected the *Drive alone* alternative.

To configure the utility function for the drive alone alternative, select the relevant variables one at a time and enter the coefficient in the designated space. Then click the arrow button (=>) to add the term to the utility function for that alternative. This step is similar to configuring the linear regression expression. In the example shown below we have selected the constant and entered the corresponding coefficient for the utility function of the drive alone alternative.

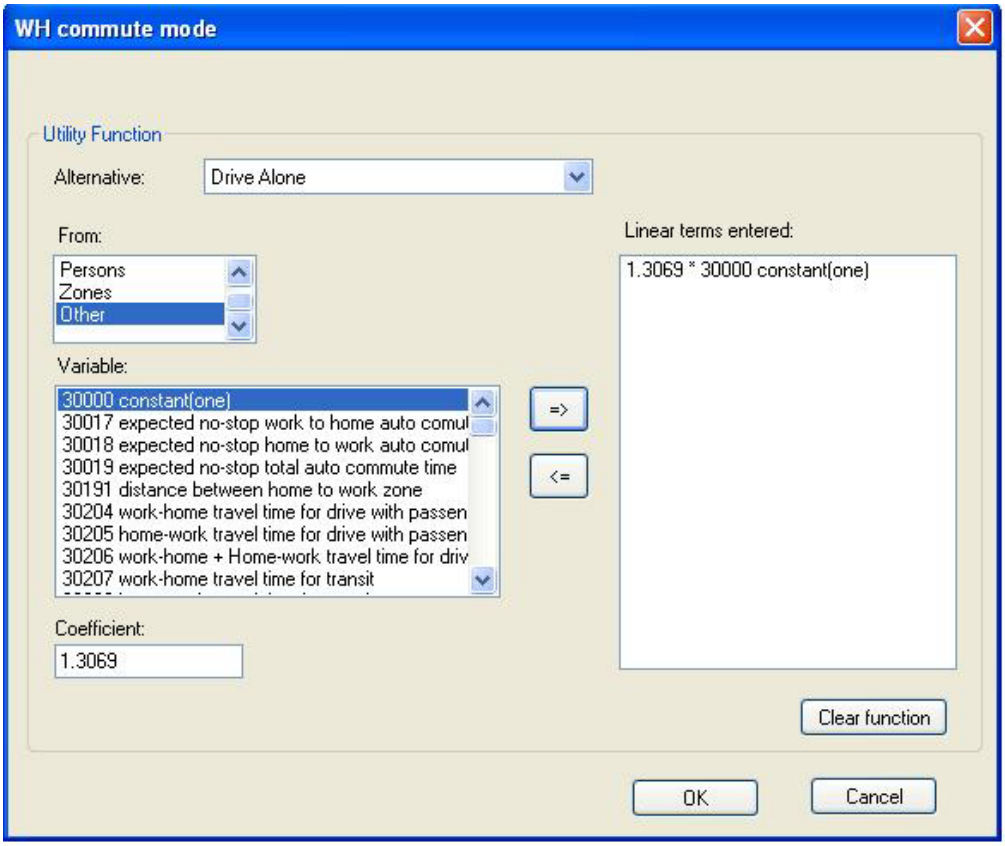

Figure 4.48: Example of Constant and Corresponding Coefficient for the Utility Function of one of the Multinomial Logit Model Alternatives

When the utility function for the alternative has been fully configured, the dialog box looks as shown on the next page. The utility functions for all other alternatives can be similarly configured. The utility of a base alternative with zero utility can be specified by providing a value of zero for the constant of that alternative. After the utility functions for all alternatives are configured, click on *OK* to complete the model configuration specification. The reader should note that clicking on the *Clear Function* button clears all the coefficients and variables so that the utility function can be reconfigured.

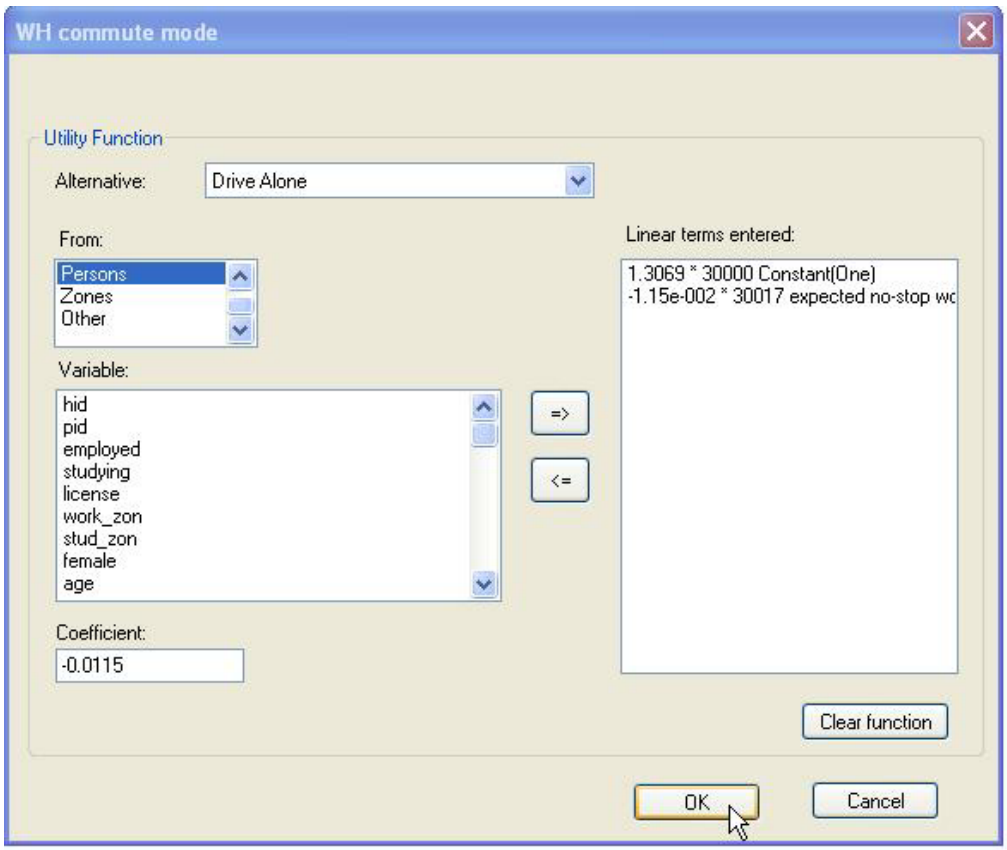

Figure 4.49: Example of Fully Configured Multinomial Logit Model

Binary Logit

The binary logit model is a special case of the multinomial logit model, with only two alternatives. It follows the same method of configuration as the multinomial logit. The binary logit model, however, is simpler since it has only one utility function (corresponding to the *Yes* alternative) to be configured. The *No* alternative is always taken as the base with zero utility. For example, consider the Decision to go to Work Model within the generation-allocation model system. The alternatives are either that the person decides to go to work on the given day or not. Hence, there is only a single utility function to be configured with the other alternative (*No*) serving as the base. The completely configured dialog box looks as shown below. Note: Clicking on the *Clear Function* button clears all the coefficients and variables so that the utility function and can be reconfigured.

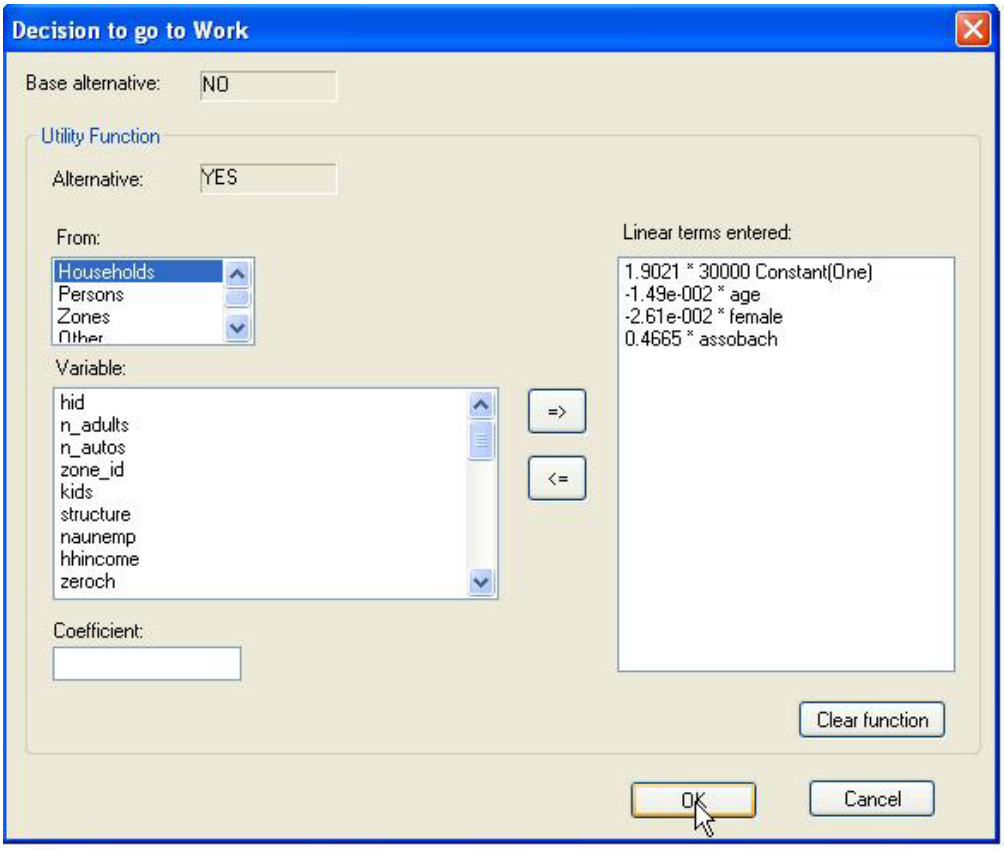

Figure 4.50: Binary Logit Model Interactive User Interface

Location Choice

An example of a location choice model is the Work-Home Stop Location Model. Since there is only one utility function to be configured, configuration of the location choice model is similar to that of a linear regression model. First, enter the variance determined for the travel time model corresponding to the tour under consideration in the space designated '*Variance'* (in this case enter the variance obtained for the travel time model for stops in the Work-Home commute). Then configure the utility function in the same way that the linear regression equation is configured. That is, select the relevant input table from the '*From*' list, and pick the variables from the '*Variables*' box. Specify the coefficient for each variable and use the arrow button (=>) to add the term. The completely configured dialog box looks as shown below.

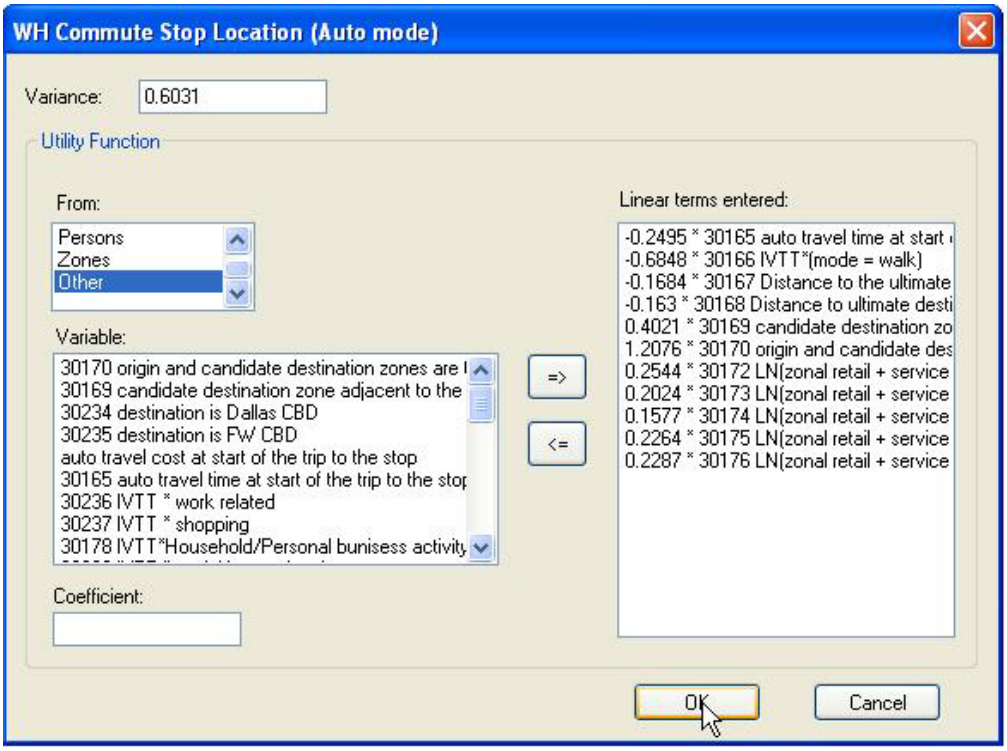

Figure 4.51: Example of Fully Configured Binary Logit Model

Note that the list of variables supported by the location choice model is pre-determined and available only under the '*Other*' variables list. The list of variables includes the inter-zonal LOS characteristics, and interactions between household or person characteristics, and destination zone or inter-zonal LOS characteristics. The user must use only these variables in the calibration and enter the parameters appropriately.

Ordered Probit

An example of an ordered probit model is the Work-Home Commute Number of Stops Model. The inputs required for this model are the propensity function and threshold parameters. Configuring the propensity function is identical to configuring the linear regression equation. That is, select the relevant variables from each list (*Households, Persons, Zones* and *Other*). Enter the coefficient corresponding to the variable and click on the arrow button (=>) to add the term. The maximum number of alternatives to be input for any model is pre-determined and based on the representation frameworks supported for workers and non-workers. However, CEMDAP supports any number of alternatives less than the maximum number. For example, the software supports a maximum of four tours for non-workers in the D-FW model system. However, if the maximum number of tours in a particular context is only 3, the number of threshold parameters for the ordered probit model for # tours for non-workers is 2. This number appears in the '*# Threshold Values*' box in the bottom left corner of the dialog box. In the Work-Home Commute Number of Stops Model, the software supports three stop options

(0, 1, and 2 stops) and hence the number of threshold parameter is 2. Click on *Edit* to specify the threshold parameters. The completely configured dialog box looks as shown on the next page. Note: If the user needs to change the number of base line time intervals after clicking *Edit*, it is advised that the model configuration be cancelled and reconfigured.

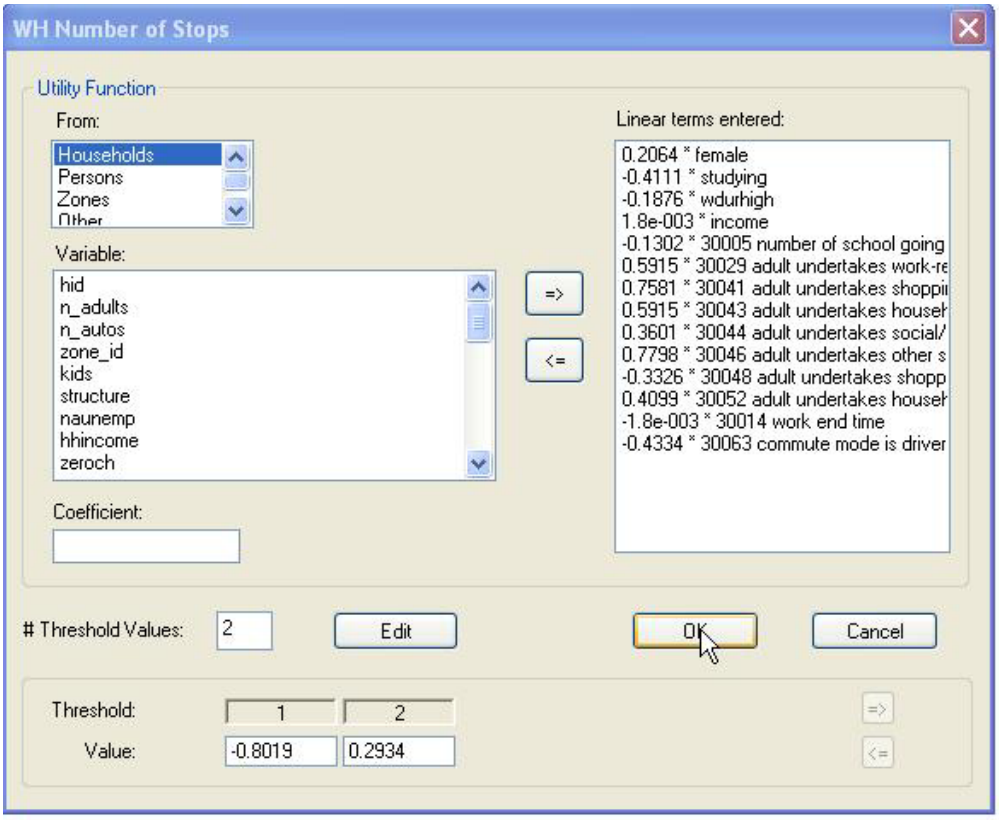

Figure 4.52: Example of Fully Configured Ordered Probit Model

Work Start and End Time Model

The work start and end time model has a different model structure and configuration than all other model structures discussed above (see Guo et al. 2005 for the details on its model structure). The inputs required for this model (parameters and variables) are entered just like the parameters and variables of the linear regression model. Note that the list of variables supported by the work start and end time model is pre-determined and available only under the '*Other*' variables list. The user must use only these variables in the calibration and enter the parameters appropriately.

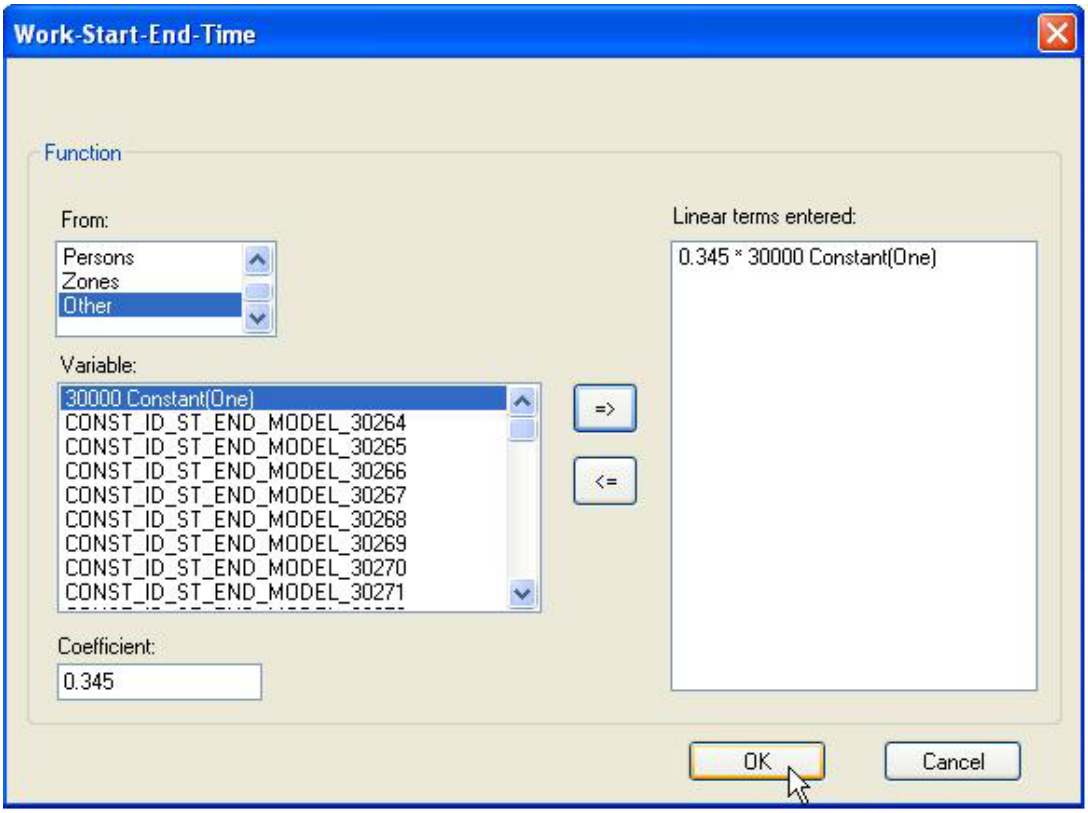

Figure 4.53: Work-Start-End-Time Model Interactive User Interface

4.5 INITIATING A SIMULATION RUN

Prior to starting a simulation, ensure that the input database is registered and loaded, the output files are specified, and the model system is completely configured. Then select the *Run* command in the *Simulation Menu*.

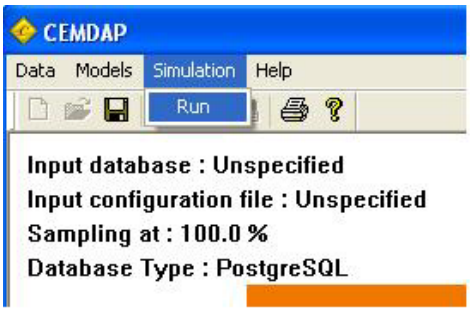

Figure 4.54: Run command in the Simulation Menu

The progress bar is launched whenever a simulation run is initiated, and serves to keep track of the progress of the simulation. In particular, the progress bar displays the number of households processed, time elapsed, estimated simulation time left, number of threads running, and the number of threads completed.

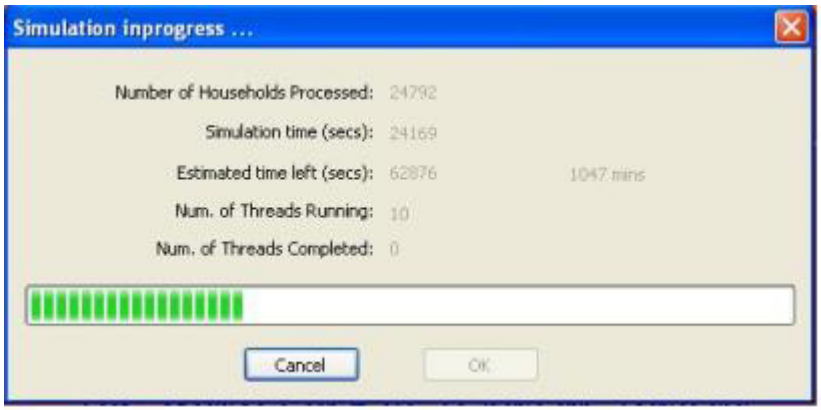

Figure 4.55: Simulation in Progress Dialog Box

The title of the progress bar reads '*Simulation in Progress...*' when the simulation is in progress and reads '*Simulation Completed Successfully*' when it is completed.

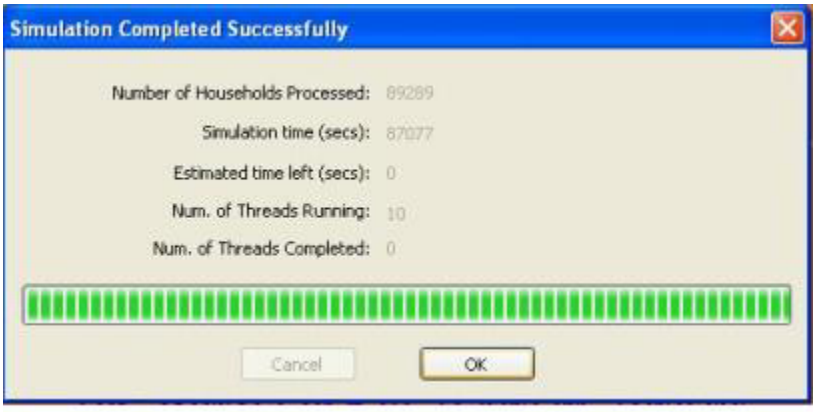

Figure 4.56: Simulation Completed Successfully Dialog Box

Click *OK* and close the progress bar and ensure that the output files have been created. The output files can then be imported into any spreadsheet and the simulated activity-travel patterns of the population can be analyzed.

REFERENCES

Guo, J.Y., S. Srinivasan, N. Eluru, A. Pinjari, R. Copperman, and C.R. Bhat, "Activity-Based Travel-Demand Analysis for Metropolitan Areas in Texas: CEMSELTS Model Estimations and Prediction Procedures, 4874 Zone System CEMDAP Model Estimations and Procedures, and the SPG Software Details," Report 4080-7, prepared for the Texas Department of Transportation, October 2005.

A. Pinjari, N. Eluru, R. Copperman, I.N. Sener, J.Y. Guo, S. Srinivasan, and C.R. Bhat, "Activity-Based Travel-Demand Analysis for Metropolitan Areas in Texas: CEMDAP Models, Framework, Software Architecture and Application Results, " Report 4080-8, prepared for the Texas Department of Transportation, October 2006

Appendix A. D-FW Input Data

The input data requirements for running a simulation of the Dallas-Fort Worth (D-FW) metropolitan area are dependent on the model configurations estimated for D-FW. This input data therefore includes not only the 'required variables' specified in Chapter 3 but also additional variables that are specific to the estimated D-FW models. The folder test_data, included in the /data directory under the installation destination directory, contains the data required for a small sample of the D-FW population for test purposes. The following tables contain the specifications for the input datasets.

Table A.1 Households Table Specification for D-FW Area

Column #	Variable Description	Variable Name
1	Household ID	HHID
2	Number of adults (age >=18)	NADULT
3	Total number of HH vehicles, including motorcycles and RVs	NVEH
4	Home TSZ location	HOMETSZ
5	Number of children	NCHILD
6	Household structure	HHSTRUCT
7	Number of unemployed adults	NAUNEMP
8	Household income in 1000s of dollars	HHINCOME
9	Household with no children	ZEROCH
10	Number of persons in household	NPERS
11	Multiple number of adults (1 if nadult > 1, 0 otherwise)	MULTADU
12	Vehicles per licensed driver	VEHBYLIC
13	Single person household	SPERSON
14	Single parent household	SPARENT
15	Male-female couple household, no children	COUPLE
16	Male-female couple household, with children	NUCLEAR
17	Other household type	OHHTYPE
18	Number of licensed drivers	NUMLIC
19	Household with one child	ONECH
20	Household with two or more children	TWOCH
21	Number of employed adults	NAEMP
22	Household with no employed adult	ZEROEMP
23	Household with one employed adult	ONEEMP
24	Household with two or more employed adults	TWOEMP
25	Number of adult (age >= 18) students	NASTU
26	Number of children (age < 18) who are not students	NCNOTSTU
27	Number of children who are students	NCSTU
28	Own residential unit	OWNHOME
29	Single family detached housing unit	SFDUNIT
30	Single family attached housing unit	SFAUNIT
31	Apartment type housing unit	APTUNIT
32	Other type of housing unit	OTHUNIT

Table A.2 Persons Table Specification for D-FW Area

Column #	Variable Description	Variable Name
1	Household ID	HHID
2	Person ID	PerID
3	Adult is employed	Aemp
4	Adult or child is a student	Stu
5	Is the person licensed to drive	License
6	Work TSZ	WorkTSZ
7	School TSZ	SchTSZ
8	Person is female	Female
9	Age of the person in 1996	Age
10	Adult is a parent	Parent
11	Caucasian	Cauc
12	African American	Afamer
13	Asian or pacific islander	Asian
14	Gender of the person	Male
15	Personal vehicle availability (# vehicles/#Licensed Drivers)	pvehavbl
16	High work flexibility	Highflex
17	Pre school completed, child	Presch
18	Kindergarten to grade 4 completed, child	kgtog4
19	Grade 5 to grade 8 completed, child	G5tog8
20	Grade 9 or higher completed, child	G9orhigh
21	Age of the person <= 5 years	Lowage
22	Lowage * one employed adult in household	loage1
23	Lowage * two employed adults in household	loage2
24	Some college completed, adult	Somecol
25	Associate or bachelors degree completed, adult	Assobach
26	Masters or PhD degree completed	Mastphd
27	Income in 1000s of dollars	Income
28	Income / HH income	Incomef
29	Work duration between 0 and 20 hours a week	Wdurlow
30	Work duration between 20 and 40 hours a week	Wdurmed
31	Employment type: construction and manufacturing	Emptype1
32	Employment type: wholesale and transportation	Emptype2
33	Employment type: personal, professional and financial services	Emptype3
34	Employment type: public and military	Emptype4
35	Employment type: retail and repair	Emptype5
36	Female parent in a single parent or nuclear family household	Mother
37	Male parent in a single parent or nuclear family household	Father
38	No school completed, child	Nosch
39	Child not a student	Cnotstu
40	Adult is unemployed	Aunemp
41	Adult son or daughter in a single parent or nuclear family household	Adchild
42	Person is 16 years of age or older	Adult
43	Hispanic	Hisp
44	Other race	Othrace
45	Grade 8 or lower completed, adult	G8orlow

46	Grade 9 to grade 12 completed, adult	G9tog12
47	High school completed, adult	Highsch
48	Total weekly work duration (excluding weekend)	Workhrs
49	Work duration greater than 40 hours a week	Wdurhigh
50	Medium work flexibility	Medflex
51	Low or no work flexibility	Lowflex
52	Employment type: other industries	Emptype6

Table A.3 Zones Table Specification for DF-W Area

Column #	Variable Description	Variable Name
1	TSZ zone id	ZONEID
2	Distance to the nearest major shopping zone	SHOPDIST
3	Accessibility with cost – retail employment	REMPACC
4	Accessibility to retail+ service employment	RSEMPACC
5	Accessibility to total employment	TEMPACC
6	Accessibility to population	POPACC
7	Dallas CBD dummy variable	DALCBD
8	Fort Worth CBD dummy variable	FWCBD
9	Median income of the zone (1000s of Dollars)	MEDINC
10	Number of households in the zone	NUMHH
11	Number of persons in the zone	NUMPERS
12	Basic employment in the zone	BEMP
13	Retail employment in the zone	REMP
14	Service employment in the zone	SEMP
15	Total employment in the zone	TOTEMP
16	Parking cost	PARKCOST
17	County	COUNTY
18	Special land use	SPLLUSE
19	Internal zone dummy variable	INTERNAL

Note: The procedure to calculate the accessibility measures is provided at the end of this appendix.

Table A.4 Zone2Zone Table Specification for DF-W Area

Column #	Variable Description	Variable Name
1	TSZ zone id – origin	Origin_zone
2	TSZ zone id – destination	Dest_zone
3	Dummy variable indicating origin and destination are adjacent zones	Adjacent
4	Distance between zones	Distance

Table A.5 LOS Tables (AMPeak, PMPeak, and OFFPeak) Specifications for DF-W Area

Column #	Variable Description	Variable Name
1	TSZ zone id – origin	Origin
2	TSZ zone id – destination	Dest
3	Dummy variable indicating origin and destination are the same zone	samezone
4	Dummy variable indicating origin and destination are adjacent zones	Adjacent
5	Distance between zones	Distance
6	Auto In-vehicle travel time between origin and Destination zones	autoIVTT
7	Auto Out-of-vehicle travel time between origin and Destination zones	autoOVTT
8	Dummy variable indicating if transit is available between zones	Travail
9	Transit In-vehicle travel time between origin and Destination zones	TrIVTT
10	Transit Out-of-vehicle travel time between origin and Destination zones	TrOVTT
11	Transit cost	Trcost
12	Auto cost	COST

Table A.6 LOSDir Tables (AMPeak, PMPeak, and OFFPeak) Specifications for DF-W Area

Column #	Variable Description	Variable Name
1	Start Time of the time period	start_t
2	End Time of the time period	End_t
3	Name of the LOS table	tbname

Table A.7 LOSdir Table Currently Present in CEMDAP

Row#	start_t	End_t	tbname
1	0	210	Losoffpk
2	360	780	Losoffpk
3	930	1440	Losoffpk
4	210	360	lospeakam
5	780	930	lospeakpm

Accessibility Measure Calculations

Accessibility to Retail Employment

Have to be re-evaluated when

1) retail employment changes, and/or

2) impedance measures change

Accessibility to *retail employment* measure for a time period *t*:

$$Acc_{i,t} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{\left(\text{retail\_emp}_j\right)^{\gamma}}{\left(\text{Impedence}_{ij,t}\right)^{\beta}}\right)$$

$$\text{Impedence}_{ij,t} = IVTT_{ij,t} + \delta \cdot Cost_{ij,t}$$

$\gamma = 0.2868$

$\delta = 0.0992$

$\beta = 3.0779$

t : index for time period

i: index for origin zone

j: index for destination zone

$IVTT$ in minutes

$Cost$ in cents

N is the number of zones

The overall daily accessibility to *retail employment* measure is then computed as

$$Acc_i = \frac{1}{1440}\sum_t Dur_t \cdot Acc_{i,t}$$

Dur_t = Duration of time period t in minutes

Use the following three TOD (time-of-day) periods: AM peak, off peak, PM peak

IF (TOD <= 209 | (TOD >=360 & TOD <= 719) | (TOD >= 930)) IVTT = off peak
IF (TOD >=210 & TOD <= 359) IVTT = AM peak.
IF (TOD >=720 & TOD <= 929) IVTT = PM peak.
TOD in minutes from 3 AM

Other accessibility measures

- General formula = 1/N*SUM(size/impedance)

- Impedance = peak auto IVTT
- Create 3 different accessibility measures using the following as the size measures:

  (1) retail + service employment, (2) total employment, and (3) population
- Sum over all internal zones, excluding the airport zones (if we don't have the internal identifier, sum over all zones except the airport zones). The airport zones are identified in the special land use variable.

There measures have to be re-evaluated when the corresponding size measure changes.

Appendix B. Input Database Preparation in PostgreSQL Format

This document provides the instructions for setting up PostgreSQL and the CEMDAP dataset on your computer.

1) Installing PostgreSQL

1. Go to the PostgreSQL on-line download page at: http://www.postgresql.org/ftp/binary, and click on the latest folder of PostgreSQL (i.e. the click on the folder that is farthest down the page that is not a beta version. *Note* these instructions assume a version 8.2.4 is downloaded.). Next, locate the win32 folder and click on it.

2. Next, locate and download the file postgresql-8.2.4-1.zip from one of the mirror sites onto a temporary directory (e.g. C:\Temp) on your machine (either http or ftp). (*Note* that the filename of the file changes with the version number).

3. Locate and extract the zip file.

4. Run the Windows Installer postgresql-8.2.msi (the file name varies with the version number).

5. Follow the installation instructions listed on: http://pginstaller.projects.postgresql.org/
   Tips:
   a) In the Service configuration page
      1. Install as a service
      2. Allow PostgreSQL to auto-generate the password. Hit NEXT by leaving the Account password and Verify password section blank (the password is not required for running PostgreSQL).

   b) In the Initialize database cluster page
      1. Leave the *Initialize database cluster* check box checked.
      2. Choose the Superuser name (We recommend using *postgres* as the user name)
      3. Enter the password (we recommend using *postgres* as the password)
      4. Please make sure you specify and write down your account name (say, postgres) – this will be used as the User name – and the password. This information will be required for accessing your database later.

   c) In the Enable procedural languages page
      1. Please accept the default settings and Hit Next

   d) In the Enable contrib modules

1. Please accept the default settings and Hit Next

e) In the PostGIS page
   1. Please accept the default settings and Hit Next

2) Setting up the CEMDAP database

Once PostgreSQL is properly installed on your machine, follow Start/Programs/PostgreSQL/pgAdmin III to run PostgreSQL. This opens up a database maintenance window. You should be able to see in the database explorer (left-hand side panel) the name of the database server (e.g. PostgreSQL Database Server 8.2) that you created. To access the database, double click on the database server name. This will prompt a dialog box for the password. Enter the user password you set up during the installation process (*Tip:* have the password stored to avoid having to type it in every time you want to access the database). After you enter the password, the Databases icon will be visible under the database server name. To create a new database, right-click on the Databases icon, and choose *New Database*. This will prompt a new dialog box. In this dialog box, enter the name of the database and note down the given name. *Note*: The name can be whatever the user desires (Ex: CEMDAP_Test_Data). Select Encoding as "SQL_ASCII", and leave the other fields empty. The procedure of loading data into this empty database is described in the subsequent sections.

2.1. Create Tables

To access the empty database, click on the "+" symbol to the left of the database name (*Note* you may have to double click on the database name the first time you access the database. Press OK if a new window pops up). Then, you should be able to see items such as Casts, Languages, Schemas, and Replication listed under the database name in the Explorer. Expand Schemas and expand Public. You will see that the database currently contains no tables (as indicated by Tables (0)). Therefore, the next step is to create the skeleton of CEMDAP input tables. To do so:

1. Select Tools->Query Tool from the menu.

2. Paste the query statement provided below into the Query window and press F5 (or select Query-Execute from the menu) to run the query. A message will appear "Query returned successfully with no result…".
3. Exit out of the Query window. Click NO to save changes.

The query creates empty tables (as can be seen under Tables) with the appropriate column properties. These columns are empty, and so our next step is to populate the columns.

Query Statement for Table Creation

```
CREATE TABLE households
(
  hid float8,
  n_adults float8,
  n_autos float8,
  zone_id float8,
  kids float8,
  structure float8 DEFAULT 0,
  naunemp float8,
  hhincome float8,
  zeroch float8,
  npers float8,
  multadu float8,
  vehbylic float8,
  sperson float8,
  sparent float8,
  couple float8,
  nuclear float8,
  ohhtype float8,
  numlic float8,
  onech float8,
  twoch float8,
  naemp float8,
  zeroemp float8,
  oneemp float8,
  twoemp float8,
  nastu float8,
  ncnotstu float8,
  ncstu float8,
  ownhome float8,
  sfdunit float8,
  sfaunit float8,
  aptunit float8,
  othunit float8
)
WITH OIDS;
ALTER TABLE households OWNER TO postgres;

CREATE TABLE persons
(
  hid float8,
  pid float8,
  employed float8,
```

```
studying float8,
license float8,
work_zon float8,
stud_zon float8,
female float8,
age float8,
parent float8 DEFAULT 0,
caucasia float8,
afriamer float8,
asian float8,
male float8,
pvehavbl float8,
highflex float8,
presch float8,
kgtog4 float8,
g5tog8 float8,
g9orhigh float8,
lowage float8,
loage1 float8,
loage2 float8,
somecol float8,
assobach float8,
mastphd float8,
income float8,
incomef float8,
wdurlow float8,
wdurmed float8,
emptype1 float8,
emptype2 float8,
emptype3 float8,
emptype4 float8,
emptype5 float8,
mother float8,
father float8,
nosch float8,
cnotstu float8,
aunemp float8,
adchild float8,
adult float8,
hisp float8,
othrace float8,
g8orlow float8,
g9tog12 float8,
highsch float8,
workhrs float8,
wdurhigh float8,
```

```
  medflex float8,
  lowflex float8,
  emptype6 float8
)
WITH OIDS;
ALTER TABLE persons OWNER TO postgres;

CREATE TABLE losdir
(
  start_t float8,
  end_t float8,
  tbname varchar(50)
)
WITH OIDS;
ALTER TABLE losdir OWNER TO postgres;

-- Table: losoffpk
-- DROP TABLE losoffpk;

CREATE TABLE losoffpk
(
  orig_zon float8,
  dest_zon float8,
  same_zon float8,
  adjacent float8,
  distance float8,
  da_ivtt float8,
  da_ovtt float8,
  tr_avail float8,
  tr_ivtt float8,
  tr_ovtt float8,
  tr_cost float8,
 au_cost float8
)
WITH OIDS;

-- Table: lospeakam
-- DROP TABLE lospeakam;

CREATE TABLE lospeakam
(
  orig_zon float8,
  dest_zon float8,
  same_zon float8,
  adjacent float8,
  distance float8,
```

```
  da_ivtt float8,
  da_ovtt float8,
  tr_avail float8,
  tr_ivtt float8,
  tr_ovtt float8,
  tr_cost float8,
au_cost float8
)
WITH OIDS;
ALTER TABLE lospeakam OWNER TO postgres;

ALTER TABLE losoffpk OWNER TO postgres;

-- Table: lospeakpm
-- DROP TABLE lospeakpm;

CREATE TABLE lospeakpm
(
  orig_zon float8,
  dest_zon float8,
  same_zon float8,
  adjacent float8,
  distance float8,
  da_ivtt float8,
  da_ovtt float8,
  tr_avail float8,
  tr_ivtt float8,
  tr_ovtt float8,
  tr_cost float8,
 au_cost float8
)
WITH OIDS;
ALTER TABLE lospeakpm OWNER TO postgres;

-- Table: zone2zone
-- DROP TABLE zone2zone;

CREATE TABLE zone2zone
(
  orig_zon float8,
  dest_zon float8,
  adjacent float8,
  distance float8
)
WITH OIDS;
ALTER TABLE zone2zone OWNER TO postgres;
```

```
-- Table: zones
-- DROP TABLE zones;

CREATE TABLE zones
(
  zid float8,
  shopdist float8,
  rempacc float8,
  rsempacc float8,
  tempacc float8,
  popacc float8,
  dalcbd float8,
  fwcbd float8,
  medinc float8,
  numhh float8,
  numpers float8,
  bemp float8,
  remp float8,
  semp float8,
  totemp float8,
  parkcost float8,
  county float8,
  splluse float8,
  internal float8
)
WITH OIDS;
ALTER TABLE zones OWNER TO postgres;
```

2.2. Load Tables

The process of loading data files into the tables is done in DOS environment under command prompt. To do so:

1. Open the Command Prompt window. To do this,
    a. Click on Start/Run in windows

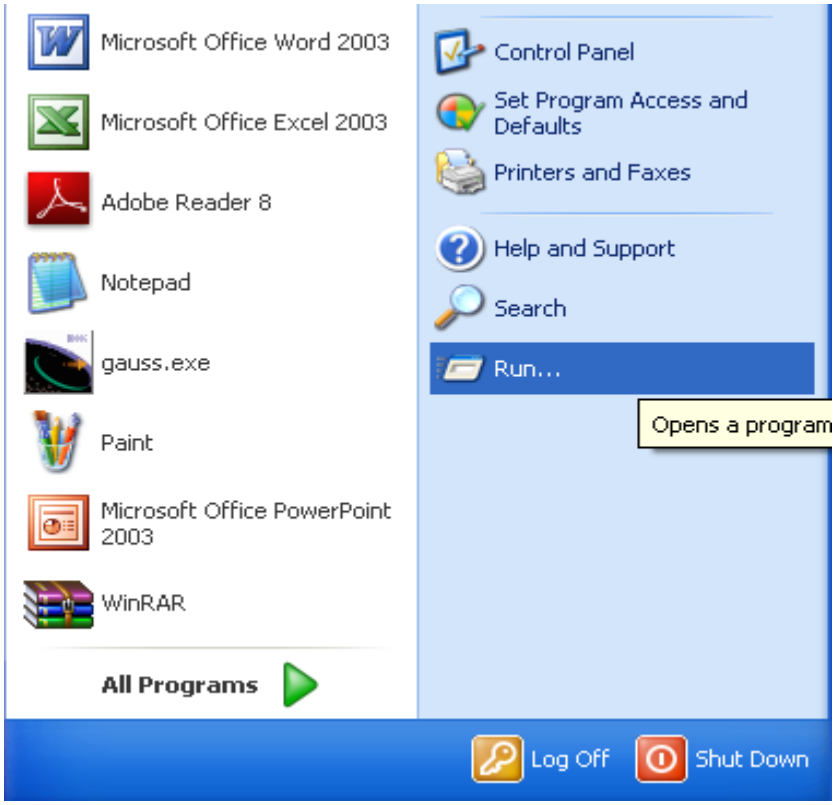

Figure B-1 Run Command in Start menu of Windows

b. Type cmd and click OK

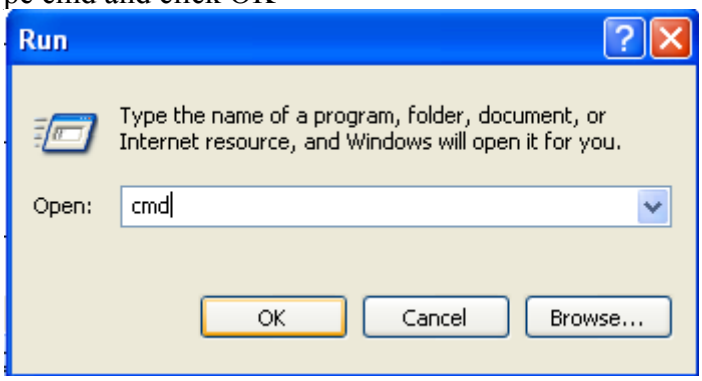

Figure B-2 Run dialog Box

2. At the command prompt, go to the \bin subdirectory under the Postgresql program directory. To do this please follow the steps below:

    a. At the command prompt type *cd* \ and press "Enter".

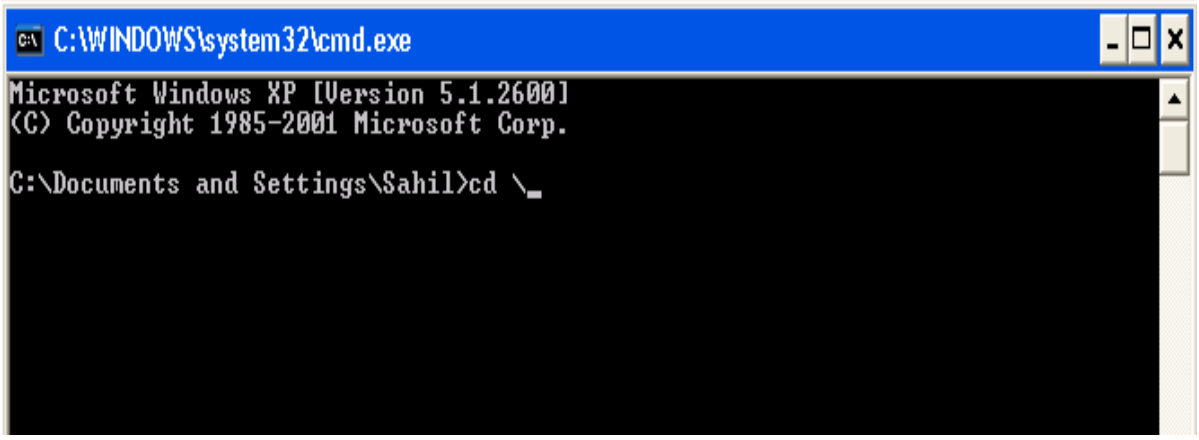

Figure B-3 Command Prompt dialog box: cd

    b. Type *cd Program files\postgresql\8.2\bin* and press "Enter". (Please note that the version number 8.2 depends on the PostgreSQL version)

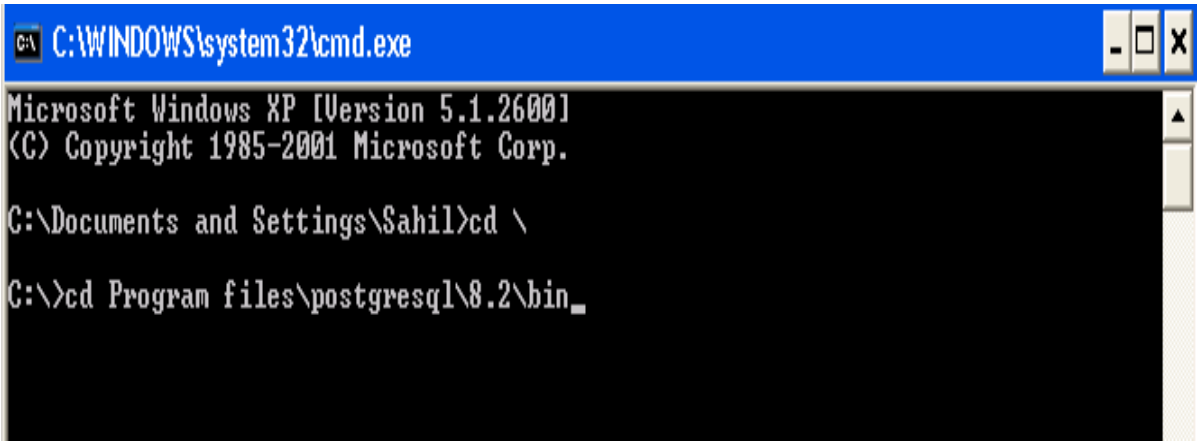

Figure B-4 Command Prompt dialog box: cd Program files\postgresql\8.2\bin

3. Type in the following command, and press 'Enter':

   psql –d *database name* –U *user name*

   where *database name* is the name of the data base for which you are populating data and *user name* is the user name you set up for accessing the database. For example, if the *database name* is CEMDAP_Test_Data and *user name* is postgres, the command should be:

   psql –d CEMDAP_Test_Data –U postgres

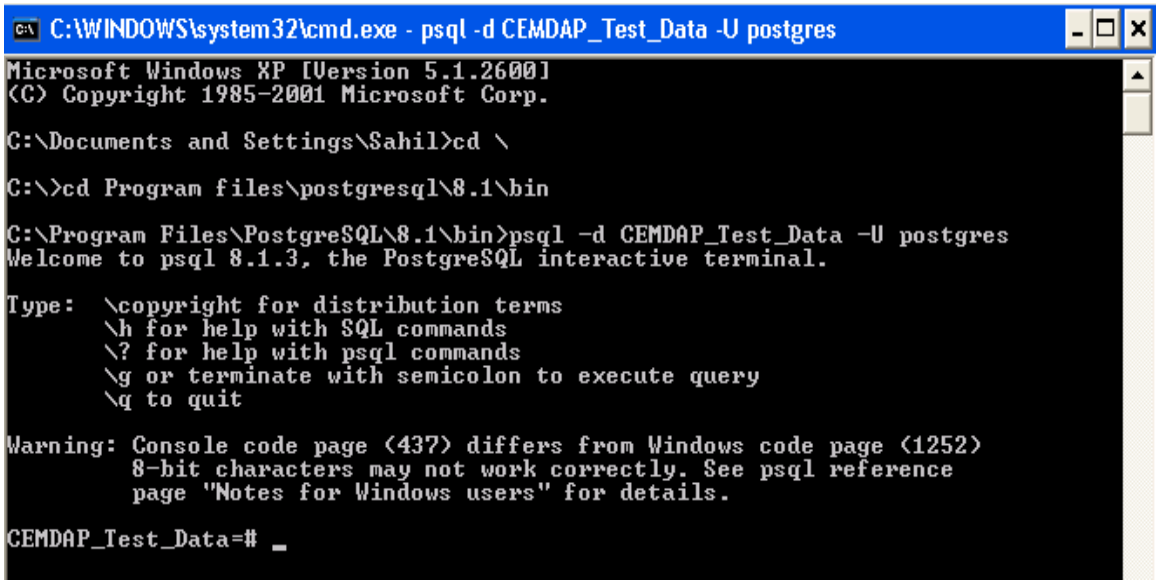

Figure B-5 Command Prompt dialog box: psql –d database name –U user name

This command will direct you to the corresponding database with the database name as defined above (the prompt is updated as the database name, in the case of our example it becomes CEMDAP_Test_Data= #)

4. Once at the new prompt, type in the following command, and press 'Enter':

   \copy *name of the table* from *filename*

   For example, if the *name of the table* is households and the *filename* is households.dat, the command should be as in the following:

\copy households from 'C:\\Program Files\\PostgreSQL\\8.2\\bin\\TestData\\households.dat'

For testing CEMDAP, please copy the TestData folder from the installation CD and place it in the C:\Program Files\PostgreSQL\8.2\bin. *Tip: C:\\Program Files\\PostgreSQL\\8.2\\bin\\TestData\\households.dat* shows the location where the data is loaded. In the example above, your data is located in the TestData folder under C:\\Program Files\\PostgreSQL\\8.2\\bin. It is best to write out the command line instead of copying and pasting.

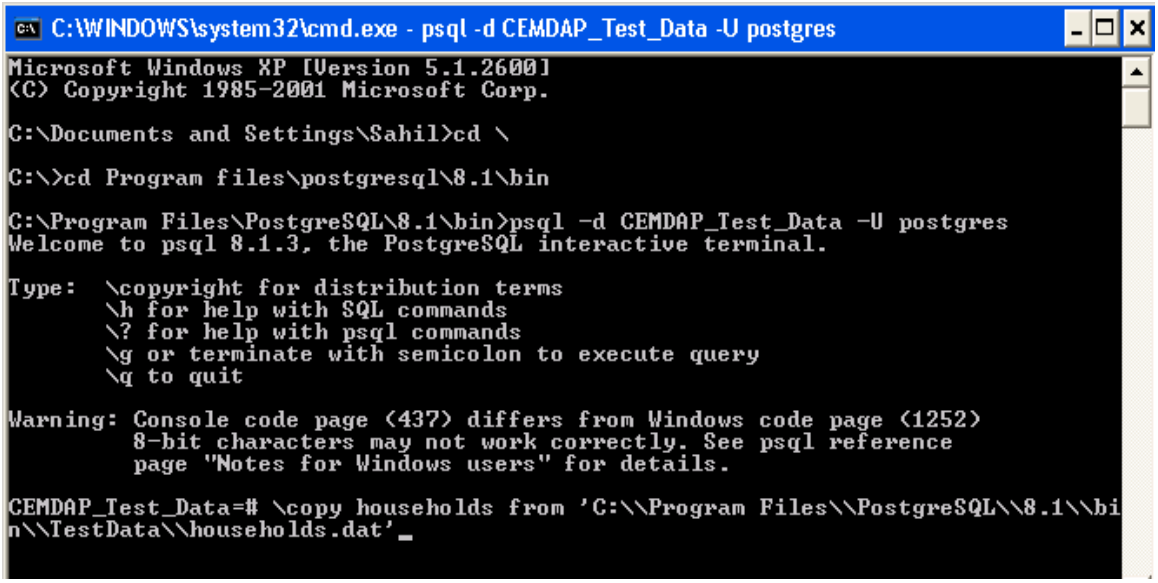

Figure B-6 Command Prompt dialog box: C:\\Program Files\\PostgreSQL\\8.2\\bin\\TestData\\households.dat

Tip: Note that *name of the table* corresponds to an empty table created in pgAdmin III under the Table icon, and *filename* is the complete path name of the file from which data is to be loaded.

5. Step four must be repeated for each table that needs to be populated with data (i.e. losdir, losoffpk, lospeakam, lospeakpm, persons, zone2zone, and zones).

6. Exit out of the Command Prompt window.

2.3. Create Index

Once all files are loaded into the database, you will need to create indices for all tables (households, losoffpk, etc.). To do so, you can follow two ways.

One of the ways of creating indices can be performed by directly using a sql statement to create the indexes for all the tables that you have. To do so:

1. In the pgADMIN window, Select Tools->Query Tool from the menu.

2. Paste the query statement provided below into the Query window and press F5 (or select Query-Execute from the menu) to run the query. Once you run the query (a message will appear "Query returned successfully with no result…"), exit the Query window (Click NO to saving the text), right click on the 'Tables' icon, and then click "Refresh' to see the indices updated.

```
CREATE UNIQUE INDEX households_1x
  ON households
  USING btree
  (hid);

CREATE UNIQUE INDEX losdir_1x
  ON losdir
  USING btree
  (start_t, end_t);

CREATE UNIQUE INDEX losoffpk_1x
  ON losoffpk
  USING btree
  (orig_zon, dest_zon);

CREATE UNIQUE INDEX lospeakam_1x
  ON lospeakam
  USING btree
  (orig_zon, dest_zon);

CREATE UNIQUE INDEX lospeakpm_1x
  ON lospeakpm
  USING btree
  (orig_zon, dest_zon);

CREATE UNIQUE INDEX persons_1x
  ON persons
  USING btree
  (pid, hid);

CREATE UNIQUE INDEX zone2zone_1x
  ON zone2zone
  USING btree
  (orig_zon, dest_zon);
```

```
CREATE UNIQUE INDEX zones_1x
  ON zones
  USING btree
       (zid);
```

The second way of creating indexes is as follows:

1. Go back to the Database Explorer in pgAdmin III
2. Go to Schemas->Public->Tables
3. For each table listed under Tables:
    a. Right click on 'Indexes(0)'.
    b. Select 'New Index' to open the New Index dialog box.
    c. On the Properties tab page, enter the name of the index as "[name of the file]_1x" (for example, enter "households_1x" for the households table). Check Unique.
    d. Next, go to the Columns page and add the column name(s) as shown in the table below:

Table B.1 Column names of Postgres Tables

Table Name	Column(s) used for indexing
households	hid
Losdir	start_t, end_t
Losoffpk	orig_zon, dest_zon
Lospeakam	orig_zon, dest_zon
Lospeakpm	orig_zon, dest_zon
Persons	hid, pid
zone2zone	orig_zon,dest_zon
Zones	zid

If an index is set up correctly, you will see that Index (0) has become Index (1).

Appendix C. Structure of the Model Specification File

All the components of the model system embedded in CEMDAP must be configured into a model configuration file (also referred to as a model specification file) and input to the system before starting a simulation run. The preparation of model configuration file involves providing the relevant variables and their parameters in a specific format as explained below. Alternatively, a complete model specification file, which can be used for testing purposes, is provided in the installation CD.

The specification file contains a complete description of each of the models included in the CEMDAP modeling framework. The specification file starts with an opening "<Configuration>" command and ends with a closing "</Configuration>" command. Then, the individual models that are embedded in CEMDAP are coded according to their specifications.

The first line of each model starts with the name of the model. Then, the class of the model that identifies the modeling template from which the particular choice model is derived (*classCBLogitMM* for Binary Logit, *classCMNLogitMM* for Multinomial Logit, *classCRegressMM* for Linear Regression, *classCOrdProbitMM* for Ordered Probit, *classCHazardMM* for Hazard Duration, *classCBLogitMM* for Location Choice, and *Work-Start-End-Time* for Work Start and End Time Model), is recorded. The subsequent lines then describe the IDs of each variable in the models and the parameter values specified to those variables (a complete list of endogenous variables - variable IDs, and explanations- used in the models can be found in the corresponding models in the GUI). To close the model, each opening statement should be closed (The reader will note that each statement should start with an opening command and end with a closing command. That is, the name and the class of the models are captured between the opening "< ModelConfig>" command and the closing "</ModelConfig>" command. All the variables in the model are captured between the opening "<Variables>" command and the closing "</Variables>" command. If variance should be given in the model, the value of variance is captured between the opening <Variance>" command and the closing "</Variance>" command. Similarly, the alternatives in the Multinomial Logit model are captured between the opening "<Alternatives>" command and closing "</Alternatives>" command. The threshold values in the Ordered Probit model are captured between the opening "<Thresholds>" command and closing "</Thresholds>" command. The interval parameters for the Hazard Duration model are captured between the opening "<Intervals>" command and closing "</Intervals>" command).

The following template represents the main structure of the specification file; however, the reader will note that this template can be modified according to the different specifications of each model embedded in the software. The number of variables can be different for each model or some of the models can have different configurations.

```
<Configuration>
<ModelConfig id="enter name of the model" optional-type="enter type of the model
*">
<Variables>
<Variable ID="enter the variable ID" value="enter the value of the corresponding
variable" optional-description="enter the description of the variable" />
    </ Variables >
    </ModelConfig>
< ModelConfig id ="enter name of the model" optional-type="enter type of the
model *">
< Variance>enter the value of the variance of the model</Variance>
<Variables>
<Variable ID="enter the variable ID" value="enter the value of the corresponding
variable" optional-description="enter the description of the variable" />
<Variable ID="enter the variable ID" value="enter the value of the corresponding
variable" optional-description="enter the description of the variable" />
    </Variables>
    </ModelConfig>
    </Configuration>
```

In CEMDAP, there exist seven different choice models: 1) Binary Logit, 2) Multinomial Logit, 3) Linear Regression, 4) Ordered Probit, 5) Hazard Duration, 6) Location Choice, and 7) Work Start and End Time. The format of the entry in the model specification file, for each of the seven models is slightly different. Below, the specification file format for each of the seven models is given in detail.

1) Binary Logit Model

The first line in the Binary Logit Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCBLogitMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables, the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables embedded in the model are defined, the model is closed (The reader will note that each row under the "<Variables>" command represents a single variable.) The following configuration presents the framework for one of the Binary Logit models in CEMDAP: Children's decision to go to school. The first variable for the "Children's decision to go to school" model is *constant* that has a variable ID of *30000* and the corresponding value of the parameter is *-0.5765*.

```
<ModelConfig id="Children's decision to go to school" optional-type="class
CBLogitMM *">
<Variables>
<Variable ID="30000" value="-0.5765" optional-description="unavailable" />
<Variable ID="10016" value="0.9046" optional-description="unavailable" />
<Variable ID="10017" value="1.9349" optional-description="unavailable" />
<Variable ID="10018" value="1.8628" optional-description="unavailable" />
<Variable ID="10019" value="1.6204" optional-description="unavailable" />
<Variable ID="7" value="0.0056" optional-description="unavailable" />
    </Variables>
    </ModelConfig>
```

2) Multinomial Logit Model

The first line in the Multionomial Logit Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCMNLogitMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax is the ID of the base alternative (which should be always zero). The subsequent lines then describe the various parameters associated with Multinomial Logit model for each alternative. In order to describe the variables in each alternative define the ID of the alternative in the *Alternative ID* part. Then, the variables are described in the *Variables* part under the corresponding alternative. The Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables for each alternative in the model are defined, the model is closed (The reader will note that each row under the "<Variables>" command represents a single variable for the corresponding alternative.) The following configuration presents the framework for one of the Multinomial Logit models in CEMDAP: Allocation of drop off episode to parent. The model has 3 alternatives (Alternative 0 (base), Alternative 1, and Alternative 2). The first variable in Alternative 1 is *constant* that has a variable ID of *30000* and the corresponding value of the parameter is *-0.5807*

```
<ModelConfig id="Allocation of drop off episode to parent" optional-type="class
CMNLogitMM *">
<Alternatives baseID="0">
<Alternative ID="0">
<Variables />
    </Alternative>
<Alternative ID="1">
<Variables>
<Variable ID="30000" value="-0.5807" optional-description="unavailable" />
<Variable ID="30013" value="0.0041" optional-description="unavailable" />
<Variable ID="30015" value="-0.0047" optional-description="unavailable" />
<Variable ID="10032" value="0.9955" optional-description="unavailable" />
    </Variables>
```

```
    </Alternative>
<Alternative ID="2">
<Variables>
<Variable ID="30013" value="0.0041" optional-description="unavailable" />
<Variable ID="30015" value="-0.0047" optional-description="unavailable" />
<Variable ID="10032" value="0.9955" optional-description="unavailable" />
    </Variables>
    </Alternative>
    </Alternatives>
    </ModelConfig>
```

3) Linear Regression Model

The first line in the Linear Regression Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCRegressMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax specifies the variance value of the model (if the value of the variance is zero, then the user should enter a very small value for variance such as 0.00001). The third line indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables, the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables embedded in the model are defined, the model is closed (The reader will note that each row under the "<Variables>" command represents a single variable.) The following configuration presents the framework for one of the Linear Regression models in CEMDAP: Adult's school start time (time from 3 AM). The variance of the model has a value of *0.0764*. The first variable of the model is *constant* that has a variable ID of *30000* and the corresponding value of the parameter is *5.7896*.

```
<ModelConfig id="Adult's school start time (time from 3 AM)" optional-
type="class CRegressMM *">
<Variance>0.0764</Variance>
<Variables>
<Variable ID="30000" value="5.7896" optional-description="temp" />
<Variable ID="10023" value="0.1696" optional-description="temp" />
<Variable ID="10024" value="0.1696" optional-description="temp" />
<Variable ID="10025" value="0.2757" optional-description="temp" />
<Variable ID="10040" value="-0.1389" optional-description="temp" />
<Variable ID="10026" value="0.0011" optional-description="temp" />
<Variable ID="16" value="-0.1280" optional-description="temp" />
    </Variables>
    </ModelConfig>
```

4) Ordered Probit Model

The first line in the Ordered Probit Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCOrdProbitMM*) is also

identified in the first line of the configuration in the *optional-type* part. The second line indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables embedded in the model are defined, variables part is closed (The reader will note that each row under the "<Variables>" command represents a single variable.). The following line indicates an opening command for the threshold values of the model. The values of each threshold are entered in the *Threshold value* part one by one. Once the entire threshold values are entered, the model is closed. The following configuration presents the framework for one of the Ordered Probit models in CEMDAP: Number of stops in a tour (before-work tour). The first variable of the model has a variable ID of *30085* and the corresponding value of the parameter is *–0.728*. There are four threshold parameters in this model; and their values are *3.1768, 4.0896, 4.7494* and *5.194*.

```
<ModelConfig id="Number of Stops in a tour (before-work tour)" optional-
type="class COrdProbitMM *">
<Variables>
<Variable ID="30085" value="-0.728" optional-description="unavailable" />
<Variable ID="30104" value="0.402" optional-description="unavailable" />
<Variable ID="30092" value="0.0024" optional-description="unavailable" />
<Variable ID="30093" value="0.763" optional-description="unavailable" />
<Variable ID="30094" value="0.7391" optional-description="unavailable" />
<Variable ID="30096" value="0.5848" optional-description="unavailable" />
<Variable ID="30070" value="-0.2946" optional-description="unavailable" />
<Variable ID="30072" value="-0.3462" optional-description="unavailable" />
<Variable ID="30029" value="0.8791" optional-description="unavailable" />
<Variable ID="30041" value="0.6869" optional-description="unavailable" />
<Variable ID="30043" value="1.0051" optional-description="unavailable" />
<Variable ID="30044" value="0.6032" optional-description="unavailable" />
<Variable ID="30045" value="0.5741" optional-description="unavailable" />
<Variable ID="30046" value="0.8444" optional-description="unavailable" />
<Variable ID="10002" value="-0.2135" optional-description="unavailable" />
<Variable ID="12" value="0.2483" optional-description="unavailable" />
<Variable ID="30022" value="0.1465" optional-description="unavailable" />
    </Variables>
<Thresholds>
<Threshold value="3.1768" />
<Threshold value="4.0896" />
<Threshold value="4.7494" />
<Threshold value="5.194" />
    </Thresholds>
     </ModelConfig>
```

5) Hazard Duration Model

The first line in the Hazard Duration Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCHazardMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax specifies the variance value of the model. The third line indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables embedded in the model are defined, variables section is closed (The reader will note that each row under the "<Variables>" command represents a single variable). The following line indicates an opening command for the intervals that are specified in the model. The interval start time and corresponding parameter is entered for each of the interval in the model. Once all the interval characteristics are defined, the model is closed. The following configuration presents the framework for one of the Hazard Duration models in CEMDAP: Children's school start time (time from 3 AM). The variance of the model has a value of *0.0310*. The first variable of the model has a variable ID of *10020* and the corresponding value of the parameter is *0.5034*. The first interval time for the model is *0*, and the corresponding parameter is *-2.5892*.

```
<ModelConfig id="Children's school start time (time from 3 AM)" optional-
type="class CHazardMM *">
<Variance>0.0310</Variance>
<Variables>
<Variable ID="10020" value="0.5034" optional-description="unavailable" />
<Variable ID="10017" value="-0.2604" optional-description="unavailable" />
<Variable ID="10011" value="-0.2393" optional-description="unavailable" />
<Variable ID="10012" value="0.8228" optional-description="unavailable" />
<Variable ID="6" value="0.1314" optional-description="unavailable" />
    </Variables>
<Intervals>
<Interval start="0" param="-2.5892" />
<Interval start="260.5" param="-1.9999" />
<Interval start="270.5" param="-1.4535" />
<Interval start="280.5" param="-0.9721" />
<Interval start="285.5" param="-0.6452" />
<Interval start="290.5" param="-0.4148" />
<Interval start="295.5" param="-0.0264" />
<Interval start="300.5" param="0.2779" />
<Interval start="310.5" param="0.5515" />
<Interval start="320.5" param="0.7849" />
<Interval start="330.5" param="1.0679" />
<Interval start="350.5" param="1.3303" />
    </Intervals>
    </ModelConfig>
```

6) Location Choice Model

The first line in the Location Choice Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*classCLocationMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire variables embedded in the model are defined, the model is closed (The reader will note that each row under the "<Variables>" command represents a single variable). The following configuration presents the framework for one of the Location Choice models in CEMDAP: Location choice: stops in a workers tour/commute. The first variable of the model has a variable ID of *30165* and the corresponding value of the parameter is *-0.2495*.

```
<ModelConfig id="location choice: stops in a workers tour/commute" optional-
type="classCLocationMM *">
<Variables>
<Variable ID="30165" value="-0.2495" optional-description="unavailable" />
<Variable ID="30166" value="-0.6848" optional-description="unavailable" />
<Variable ID="30167" value="-0.1684" optional-description="unavailable" />
<Variable ID="30168" value="-0.163" optional-description="unavailable" />
<Variable ID="30169" value="0.4021" optional-description="unavailable" />
<Variable ID="30170" value="1.2076" optional-description="unavailable" />
<Variable ID="30171" value="-1.259" optional-description="unavailable" />
<Variable ID="30172" value="0.2544" optional-description="unavailable" />
<Variable ID="30173" value="0.2024" optional-description="unavailable" />
<Variable ID="30174" value="0.1577" optional-description="unavailable" />
<Variable ID="30175" value="0.2264" optional-description="unavailable" />
<Variable ID="30176" value="0.2287" optional-description="unavailable" />
   </Variables>
   </ModelConfig>
```

7) Work Start and End Time Model

The first line in the Work Start and End Time Model starts with the name of the model that is coded in the *ModelConfig id* part. The class of the model (*class CStartAndEndTimeMM*) is also identified in the first line of the configuration in the *optional-type* part. The second line in the syntax indicates an opening command for the variables of the model. The subsequent lines then describe the various parameters associated with that model. In order to describe the variables the Variable ID of each variable is first entered in the *Variable ID* part, and then the corresponding parameter of the variable is coded in the *value* part. It is also possible to give a description for the variable in the *optional-description* part. Once the entire set of variables embedded in the

model are defined, the model is closed (The reader will note that each row under the "<Variables>" command represents a single variable.) The following configuration presents the framework for the Work Start and End Time model in CEMDAP. The first variable of the model has a variable ID of *30264* and the corresponding value of the parameter is *-1.8751*.

```
<ModelConfig id="Work-Start-End-Time" optional-type="class
CStartAndEndTimeMM *">
<Variables>
<Variable ID="30264" value="-1.8751" optional-description="unavailable" />
<Variable ID="30265" value="2.3648" optional-description="unavailable" />
<Variable ID="30266" value="1.0647" optional-description="unavailable" />
<Variable ID="30267" value="-7.9309" optional-description="unavailable" />
<Variable ID="30268" value="-4.499" optional-description="unavailable" />
<Variable ID="30269" value="-1.4412" optional-description="unavailable" />
<Variable ID="30270" value="6.9739" optional-description="unavailable" />
<Variable ID="30271" value="3.6043" optional-description="unavailable" />
<Variable ID="30272" value="0.7273" optional-description="unavailable" />
<Variable ID="30273" value="-4.3365" optional-description="unavailable" />
<Variable ID="30274" value="-0.7668" optional-description="unavailable" />
<Variable ID="30275" value="0.1056" optional-description="unavailable" />
<Variable ID="30015" value="82.4853" optional-description="unavailable" />
<Variable ID="30018" value="-0.0295" optional-description="unavailable" />
   </Variables>
   </ModelConfig>
```