

## Alexandria\_4.0.06 How-To

This document provides basic information on preparing and running the TRANSIMS programs using the Alexandria\_4.0.06 dataset. This dataset includes input data files and control and batch files to demonstrate GIS shapefile conversions, synthetic data generation for highway and transit networks, trip table conversions, Router-Microsimulator iterations, transit schedule adjustments, and subarea windowing. This How-To serves as a supplement to the TestNet\_4.0.06 How-To which provides more detail. A methodology similar to the one applied in the TestNet test case data is also applied in the Alexandria test case data. However, “real-world” data is now utilized from the City of Alexandria, Virginia to demonstrate how a TRANSIMS model simulation is developed.

### Revision History

4/26/2010      Created by RSG, Inc.

### Table of Contents

- 1.0      Assumptions and Prerequisites
  - 1.1      Download Test Data
  - 1.2      Network Data Source
- 2.0      TRANSIMS Overview
  - 2.1      RunSetup
- 3.0      Preparing Application Files
  - 3.1      Master Control Files
  - 3.2      RunSetup Control Files
  - 3.3      RunSetup Batch Files
  - 3.4      Multiple Iterations
- 4.0      Run the Program Tests

## 1.0 Assumptions and Prerequisites

This document assumes you have installed TRANSIMS Version 4.0.06 (December 2009) on a Windows or Linux computer system and that you understand the basic procedures and terminology for executing TRANSIMS programs. Files with \*.bat extensions are designed to work with Windows operating systems. Files with \*.sh extensions are designed to work with most Linux operating systems. The following discussion describes the Windows process. The Alexandria\_4.0.06 test case data has been packed with the v4.0.06 32-bit Windows compiled executables. 64-bit or Linux compiled executables can be downloaded from the SourceForge site.

Text files are used to store the input and output information. You need to be able to review and edit these files using a standard text editor (e.g., vi, Pico, WordPad) or other software that can manipulate tab-delimited files (e.g., Excel).

The process generated ArcView shapefiles to display information in map format. Familiarity with software that can read and display ArcView shapefiles is desirable, but not necessary. Several \*.mxd files for ArcGIS 9.2 are provided to show typical outputs.

## 1.1 Download Test Data

This How-To document uses information from the Alexandria\_4.0.06 dataset. To download the Alexandria data to your computer or local area network, select

<http://sourceforge.net/projects/transims/files/> → test data → 4.0.06 Test Cases → Alexandria\_4.0.06.zip

It assumes access to the TRANSIMS executables from a directory called “bin” at the same level as the Alexandria directory. For example, if the TRANSIMS executables are installed in the following location:

|                    |           |
|--------------------|-----------|
| c:\TRANSIMS\bin    | (Windows) |
| /home/TRANSIMS/bin | (Linux)   |

The Alexandria data should be copied to:

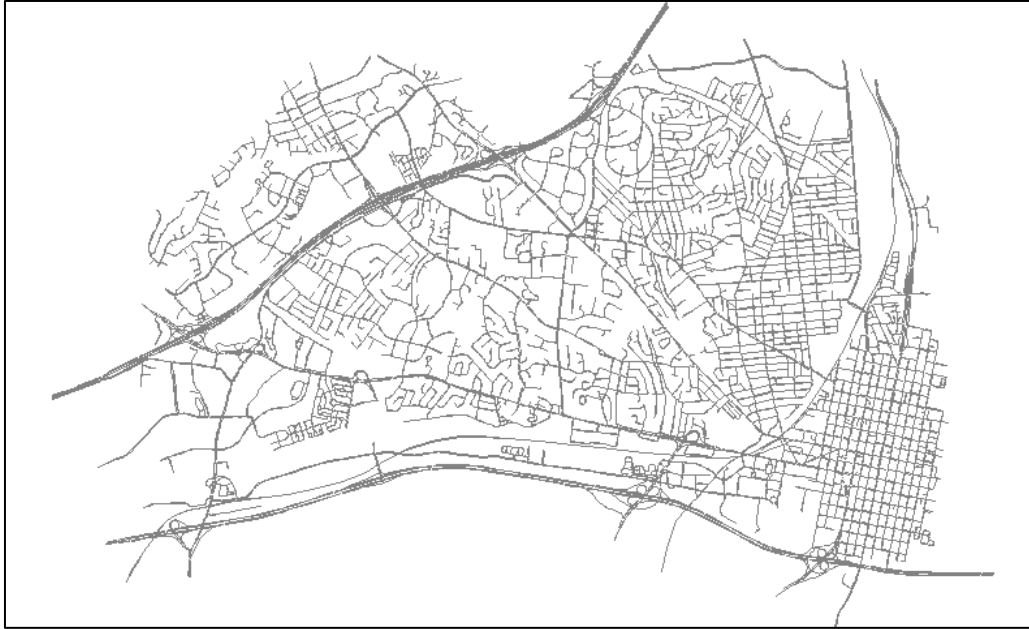
|                           |           |
|---------------------------|-----------|
| c:\TRANSIMS\Alexandria    | (Windows) |
| /home/TRANSIMS/Alexandria | (Linux)   |

Unzipping the Alexandria\_4.0.06.zip file will create a Alexandria directory with the following subdirectories:

|         |          |         |
|---------|----------|---------|
| batch   |          |         |
| census  |          |         |
| control |          |         |
| demand  |          |         |
| inputs  | /arcview |         |
| network | /arcview |         |
| results | /arcview |         |
| setup   | /arcview | /master |
| subnet  | /arcview |         |
| survey  |          |         |

The Alexandria\_4.0.06 test case data uses a “real-world” network with 3,600 links and roughly 420,000 trips. The intent of the TestNet test case data was to illustrate the use of the various tools and how they are applied in series to develop a complete TRANSIMS model simulation using a very simplified fictional network and data set. The Alexandria test case data however illustrates a more typical TRANSIMS model implementation using actual real-world data to represent the amount of travel demand utilizing the region’s transportation system.

The full model runs in about two hours on a typical desktop machine. The Alexandria \_4.0.06 network (links) is illustrated below.



## 2.0 TRANSIMS Overview

TRANSIMS is a console-based suite of programs that run in a command window on either Windows or Linux operating systems. The command syntax is

Program [-flag] [control\_file] [partition/parameter]

The control\_file is the file name of an ASCII file that contains the control strings expected by the program. If a file name is not provided on the command line, the program will prompt the user to enter the name of the control file. The flag parameters are optional. Any combination of the following flag parameters can be included on the command line:

|             |  |
|-------------|--|
| -Q[uiet]    | = execute without screen messages      |
| -H[elp]     | = show program syntax and control keys |
| -K[eyCheck] | = list unrecognized control file keys  |
| -P[ause]    | = pause before exiting                 |
| -N[oPause]  | = never pause before exiting           |
| -B[atch]    | = execute in batch processing mode     |

A number of programs also permit the user to enter a partition number or a parameter string. A partition number is used to run the program for a specific partition when data is distributed in a computer cluster.

A typical TRANSIMS application involves the execution of a number of programs in a sequential or iterative way. Numerous utility programs are also provided to assist with data processing or report and map generation. A TRANSIMS model is defined by the sequence of steps and the parameters used within each step that are calibrated for a given region or application. These steps are typically

executed using batch files or scripts to ensure consistency from one run to the next and reduce the dependency on the modeler to start each program one at a time within the model chain.

## **2.1 RunSetup**

RunSetup is a special purpose program included in the TRANSIMS suite to help automate the generation of control files and batch scripts that execute a series of programs. RunSetup accepts a control file like all other TRANSIMS programs. The control file specifies the sequence of programs to be executed in order using a specified batch file. A master control file is read for each program, customized for the given application, and written to the control directory for execution. Special control keys are provided within the RunSetup control file to replace special keywords in the master control file. Special keywords are identified using @ symbols. For example, every place in the master control file where @RUN@ is found, the program will replace the keyword with the text assigned to the RUN\_NAME key in the RunSetup control file.

The RunSetup approach enables the modeler to create master control files that define the calibrated parameters and default settings for a given program application. Keywords are then added to the master control file to identify the parameters the user needs to change or set for a particular application. This minimizes the likelihood that global search and replace commands will distort the model run. It also documents the program sequence and helps ensure that the modeler does not forget to execute critical steps in the modeling process.

RunSetup can also execute a sequence of RunSetup control files to create batch files for larger model runs. The SETUP\_CONTROL\_FILE\_# key is used to tell RunSetup to process the referenced file as a RunSetup control file and add the referenced commands to the master batch file for the model.

## **3.0 Prepare Application Files**

The setup directory within the Alexandria dataset includes control files and batch files for executing the RunSetup program. Each batch file can be executed one at a time or the TripModel.bat(.sh) file can be executed to generate all of the scripts at one time. The TripModel.bat(.sh) file merges the batch commands into a master batch file called TripModel.bat(.sh). It also copies the ArcGIS 9.2 \*.mxd files from the arcview subdirectory to the batch directory. Once the TripModel batch file is executed the user should be able to run the full model and visualize the results from within the batch directory.

### **3.1 Master Control Files**

The master control files for the RunSetup applications are stored in the setup/master directory. This directory includes master control files for the various different TRANSIMS programs. Note that here are two master control files for ArcNet processing and two master control files for PlanPrep processing. For these programs, it was more convenient to define different master control files than attempt to account for major application differences using RunSetup replacement keys.

The two master control files for PlanPrep are shown below as an example. The first file is called PlanPrep\_Sort.ctl which is used to convert a traveler sorted plan file generated by the Router into the

time sorted plan file needed by the Microsimulator. The second file is called PlanPrep\_Merge.ctl which is used to merge plans from an incremental Router application into the full plan file.

### PlanPrep\_Sort.ctl

|                     |  |
|---------------------|--|
| TITLE               | Sort the Plan Files for Simulation @RUN@.@ALT@ |
| DEFAULT_FILE_FORMAT | @FORMAT@                                       |
| PROJECT_DIRECTORY   | @PROJECT@                                      |
| INPUT_PLAN_FILE     | demand/@RUN@.@ALT@.@YEAR@.@MODEL@.@PARAM@Plans |
| OUTPUT_PLAN_FILE    | demand/@RUN@.@ALT@.@YEAR@.@MODEL@.TimePlans    |
| PLAN_SORT_OPTION    | TIME   |

### PlanPrep\_Merge.ctl

|                     |  |
|---------------------|--|
| TITLE               | Merge the Re-Routed Travel Plans for @RUN@.@ALT@   |
| DEFAULT_FILE_FORMAT | @FORMAT@   |
| PROJECT_DIRECTORY   | @PROJECT@  |
| INPUT_PLAN_FILE     | demand/@RUN@.@ALT@.@YEAR@.@MODEL@.Plans            |
| MERGE_PLAN_FILE     | demand/@PREVIOUS@.@ALT@.@YEAR@.@MODEL@.TravelPlans |
| OUTPUT_PLAN_FILE    | demand/@RUN@.@ALT@.@YEAR@.@MODEL@.TravelPlans      |

The two master control files show how the keywords are used to generate unique filenames for a given run. The filename include the run number, alternative name, analysis year, and model name. The previous run number is used to define the filename from the previous model iterations. Within a given iteration, the sorting step followings the merging step. In this case the output plan file from the merge application becomes the input plan file for the sort application.

## 3.2 RunSetup Control Files

The RunSetup control file for a Router-Microsimulator iteration called MsimRuns.ctl and is listed below.

|                       |                                   |   |
|-----------------------|-----------------------------------|---|
| TITLE                 | Run Setup Controls                |   |
| PROJECT_DIRECTORY     | ../                               |   |
| PROGRAM_DIRECTORY     | ../../bin                         |   |
| NETWORK_DIRECTORY     | network                           |   |
| CONTROL_DIRECTORY     | control                           |   |
| BATCH_DIRECTORY       | batch                             |   |
| BATCH_NAME            | MsimRuns                          |   |
| MODEL_NAME            | Trip                              |   |
| ANALYSIS_YEAR         | 2005                              |   |
| ALTERNATIVE_NAME      | Alex                              |   |
| RUN_NAME              | 2                                 |   |
| PREVIOUS_RUN_NAME     | 1                                 |   |
| PARAMETER1            | _2                                | //---- parking / process link number ---- |
| PARAMETER2            | _3                                | //---- activity location number ----      |
| MASTER_CONTROL_FILE_1 | master/PlanSelect.ctl             |   |
| DESCRIPTION_1         | ---- Msim Feedback Run @RUN@ ---- |   |
| PROGRAMS_1            | PlanSelect                        |   |
| PARAMETER_1           | 2.0                               | //---- select time ratios ----            |
| COMMENT_FLAG_1        | YES                               | //---- NO = select by time ratios ----    |

```

PARAMETER2_1          10.0          //---- percent time difference
COMMENT2_FLAG_1       NO           //---- NO = select by time difference ----

MASTER_CONTROL_FILE_2 master/Router.ctl
PROGRAMS_2            Router
COMMENT_FLAG_2        TRUE          //---- FALSE = update Plan File ----
COMMENT1_FLAG_2       TRUE          //---- FALSE = process Activity File ----
COMMENT2_FLAG_2       FALSE         //---- FALSE = process Trip File ---

MASTER_CONTROL_FILE_3 master/PlanPrep_Merge.ctl
PROGRAMS_3            PlanPrep
CONTROL_NAME_3        PlanMerge

MASTER_CONTROL_FILE_4 master/PlanPrep_Sort.ctl
PROGRAMS_4            PlanPrep
CONTROL_NAME_4        PlanSort
PARAMETER_4           Travel

MASTER_CONTROL_FILE_5 master/Microsimulator.ctl
PROGRAMS_5            Microsimulator
CONTROL_NAME_5        Msim

MASTER_CONTROL_FILE_6 master/LinkDelay.ctl
PROGRAMS_6            LinkDelay
PARAMETER_6           1.           //---- Previous Weight Base ----
PARAMETER1_6          3           //---- Smoothing Group Size ----

```

The control file defines a number of global parameters such as the project, program, network, control and batch directories. By default, these directories are all relative to the program directory. If a given key value includes directory symbols (\ or /), the key is interpreted as a full path string and the project directory is not added to the beginning of the key.

The output batch file for this application is a file called MsimRuns.bat(.sh) in the ../batch directory. The first line of the batch file sets the path to the executable programs. For Windows the command is:

```
path=%PATH%;../bin
```

Other default or global parameters include alternative name, analysis year, run\_name, parameter, parameter1 and parameter2. These key values will replace keywords @ALT@, @YEAR@, @RUN@, @PARAM@, @PARAM1@, and @PARAM2@ in each of the master control files.

The MsimRuns batch executes five programs as defined by the five master control file groups. The first file group reads the master control file called PlanSelect.ctl in the setup/master directory. Before the command is added to the batch file, DESCRIPTION\_1 is added using the Windows remark command “rem ---- Feedback Run 2 ----“. The PlanSelect executable (as defined by the PROGRAMS\_1 key) is then than called with a control file named 2.Alex.2005.PlanSelect.ctl stored in the control directory. This name is constructed using the run number, the alternative name, and the program name.

The second program is the Router using the Router.ctl master control file and outputting the control file 2.Alex.2005.Trip.Router.ctl in the control directory.

The third program is PlanPrep using the PlanPrep\_Merge.ctl shown above. The output control file name will be 2.Alex.2005.Trip.PlanMerge.ctl as specified by the CONTROL\_NAME\_3 key. Given the keyword codes shown in the master control file above, the resulting control file will have the following keys.

|                     |   |
|---------------------|---|
| TITLE               | Merge the Re-Routed Travel Plans for 2.Alex |
| DEFAULT_FILE_FORMAT | TAB_DELIMITED                               |
| PROJECT_DIRECTORY   | ../   |
| INPUT_PLAN_FILE     | demand/2.Alex.2005.Trip.Plans               |
| MERGE_PLAN_FILE     | demand/1.Alex.2005.Trip.TravelPlans         |
| OUTPUT_PLAN_FILE    | demand/2.Alex.2005.Trip.TravelPlans         |

The fourth program is PlanPrep using the PlanPrep\_Sort.ctl file and generating a control file called 2.Alex.2005.Trip.PlanSort.ctl in the control directory.

The last program is the Microsimulator with an output control file name of 2.Alex.2005.Trip.Msim.ctl.

### 3.3 RunSetup Batch Files

As mentioned above, executing RunSetup with the MsimRuns.ctl control file creates a batch file called MsimRuns.ctl in the control directory. Given the keys built into the RunSetup program, this file will have the following commands:

```
path=%PATH%;../../bin

rem ---- Router Feedback Run 2 ----

PlanSelect.exe ../control/2.Alex.2005.Trip.PlanSelect.ctl
if %ERRORLEVEL% == 1 exit 1

Router.exe ../control/2.Alex.2005.Trip.Router.ctl
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/2.Alex.2005.Trip.PlanMerge.ctl
if %ERRORLEVEL% == 1 exit 1

PlanSum.exe ../control/2.Alex.2005.Trip.PlanSum.ctl
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe ../control/2.Alex.2005.Trip.LinkDelay.ctl
if %ERRORLEVEL% == 1 exit 1
```

It starts with the path defined in the PROGRAM\_DIRECTORY key and is followed by the comment defined by the DESCRIPTION\_1 key. This is followed by the PROGRAMS\_1 value and the control file name. The control file name is composed of the PROJECT\_DIRECTORY and CONTROL\_DIRECTORY values, the RUN\_NAME and ALTERNATIVE\_NAME (if not NULL) values separated by periods, and the program name followed by “.ctl”. Since the EXIT\_CHECK key is not specified, the default exit command for Windows is added to the batch file. The process is repeated with the keys from groups 2 thru 5.

### 3.4 Multiple Iterations

In addition to the sequential types for applications outlined above, RunSetup can be used to implement multiple iterations of a standard feedback process. This is done by adding a parameter string to the end of the command line as is demonstrated in the batch file called MsimRuns\_6-10.bat. This file contains the following commands:

```
path=%PATH%;../../bin

RunSetup.exe -h -k -p MsimRuns.ctl 6..10
```

The parameter “6..10” specifies the sequence of run numbers that are used for each iteration of the MsimRuns.ctl command sequence described above. In the first iteration, the RUN\_NAME key value is overridden by the value “6” and the PREVIOUS\_RUN\_NAME key is overridden by the value “5”. The sequence is then repeated with RUN\_NAME equal to “7” and PREVIOUS\_RUN\_NAME equal to “6”. The process is repeated for runs 8 thru 10 as well.

The output batch file includes the parameter value convert to 6-10.Alex.2005.Trip.MsimRuns.bat in the batch file name and the program calls for all five iterations. Since the description key in the MsimRuns.ctl file includes the @RUN@ keyword, the comment field for each iteration is also updated in the batch file. The resulting batch file is shown below.

```
path=%PATH%;../../bin

rem ---- Msim Feedback Run 6 ----

PlanSelect.exe ../control/6.Alex.2005.Trip.PlanSelect.ctl
if %ERRORLEVEL% == 1 exit 1

Router.exe ../control/6.Alex.2005.Trip.Router.ctl
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/6.Alex.2005.Trip.PlanMerge.ctl
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/6.Alex.2005.Trip.PlanSort.ctl
if %ERRORLEVEL% == 1 exit 1

Microsimulator.exe ../control/6.Alex.2005.Trip.Msim.ctl
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe ../control/6.Alex.2005.Trip.LinkDelay.ctl
if %ERRORLEVEL% == 1 exit 1

rem ---- Msim Feedback Run 7 ----

PlanSelect.exe ../control/7.Alex.2005.Trip.PlanSelect.ctl
if %ERRORLEVEL% == 1 exit 1

Router.exe ../control/7.Alex.2005.Trip.Router.ctl
if %ERRORLEVEL% == 1 exit 1
```



```

PlanPrep.exe ../control/7.Alex.2005.Trip.PlanMerge.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/7.Alex.2005.Trip.PlanSort.ct1
if %ERRORLEVEL% == 1 exit 1

Microsimulator.exe ../control/7.Alex.2005.Trip.Msim.ct1
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe ../control/7.Alex.2005.Trip.LinkDelay.ct1
if %ERRORLEVEL% == 1 exit 1

rem ---- Msim Feedback Run 8 ----

PlanSelect.exe ../control/8.Alex.2005.Trip.PlanSelect.ct1
if %ERRORLEVEL% == 1 exit 1

Router.exe ../control/8.Alex.2005.Trip.Router.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/8.Alex.2005.Trip.PlanMerge.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/8.Alex.2005.Trip.PlanSort.ct1
if %ERRORLEVEL% == 1 exit 1

Microsimulator.exe ../control/8.Alex.2005.Trip.Msim.ct1
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe ../control/8.Alex.2005.Trip.LinkDelay.ct1
if %ERRORLEVEL% == 1 exit 1

rem ---- Msim Feedback Run 9 ----

PlanSelect.exe ../control/9.Alex.2005.Trip.PlanSelect.ct1
if %ERRORLEVEL% == 1 exit 1

Router.exe ../control/9.Alex.2005.Trip.Router.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/9.Alex.2005.Trip.PlanMerge.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe ../control/9.Alex.2005.Trip.PlanSort.ct1
if %ERRORLEVEL% == 1 exit 1

Microsimulator.exe ../control/9.Alex.2005.Trip.Msim.ct1
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe ../control/9.Alex.2005.Trip.LinkDelay.ct1
if %ERRORLEVEL% == 1 exit 1

rem ---- Msim Feedback Run 10 ----

PlanSelect.exe ../control/10.Alex.2005.Trip.PlanSelect.ct1
if %ERRORLEVEL% == 1 exit 1

```

```

Router.exe    ../control/10.Alex.2005.Trip.Router.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe  ../control/10.Alex.2005.Trip.PlanMerge.ct1
if %ERRORLEVEL% == 1 exit 1

PlanPrep.exe  ../control/10.Alex.2005.Trip.PlanSort.ct1
if %ERRORLEVEL% == 1 exit 1

Microsimulator.exe  ../control/10.Alex.2005.Trip.Msim.ct1
if %ERRORLEVEL% == 1 exit 1

LinkDelay.exe  ../control/10.Alex.2005.Trip.LinkDelay.ct1
if %ERRORLEVEL% == 1 exit 1

```

## 4.0 Run the Program Tests

To initialize the run, double-click the TripModel.bat in the \setup\ directory. This batch file will create the control files and batch files needed to execute the trip-based model. Once this execution has terminated successfully, navigate to the \batch\ directory and double-click the TripModel.bat. This will run the entire Alexandria\_4.0.06 TRANSIMS model simulation.

The program tests are run from the batch directory. The TripModel.bat(.sh) file will run each step in the testing process in order. This batch file was created by a special application of the RunSetup program using SETUP\_CONTROL\_FILE\_# keys. The TripModel batch file included in the setup directory has the following control keys. These commands instruct the RunSetup program to process each of the RunSetup control file in order and generate a master batch file called TripModel.bat(.sh) in the batch directory.

|                      |                              |
|----------------------|------------------------------|
| TITLE                | Batch Control for Trip Model |
| PROJECT_DIRECTORY    | ../                          |
| PROGRAM_DIRECTORY    | ../../bin                    |
| BATCH_DIRECTORY      | batch                        |
| BATCH_NAME           | TripModel                    |
| SETUP_CONTROL_FILE_1 | ConvertNet.ct1               |
| SETUP_CONTROL_FILE_2 | ConvertTrips.ct1             |
| SETUP_CONTROL_FILE_3 | RouteAll.ct1                 |
| SETUP_CONTROL_FILE_4 | RouterRuns.ct1               |
| SETUP_PARAMETER_4    | 2..5                         |
| SETUP_CONTROL_FILE_5 | MsimRuns.ct1                 |
| SETUP_PARAMETER_5    | 6..10                        |
| SETUP_CONTROL_FILE_6 | OutputSum.ct1                |
| SETUP_PARAMETER_6    | 10                           |

The printout files (\*.prn) for each step in each process will be written to the control directory.

Several ArcGIS 9.2 control files are copied to the batch directory for displaying ArcView shapefile output. The GIS.mxd and 3D.mxd files relate to the output of the GISTests.bat procedure. Inputs.mxd, Network.mxd, and Transit.mxd relate to the output of the ConvertNet.bat process. Nework3.mxd displays the output of the ConvertTrips.bat process. Performance.mxd and Ridership.mxd display the results of the first simulation generated by the 6.OutputSum.bat. These setups can also be used to show the output of the 10.OutputSum.bat process by modifying the source filenames. SubNet.mxd shows the result of the Subarea.bat set.

The \setup\ directory also includes a batch script called ActivityModel.bat for creating the batch files and program controls for a full run of the Alexandria activity-based case study as opposed to the TripModel.bat which runs the trip based case study. This application focuses on the Population Synthesizer and Activity Generator tools. To initialize the run, double-click the ActivityModel.bat in the \setup\ directory. This batch file will create the control files and batch files needed to execute the activity-based model. Once this execution has terminated successfully, navigate to the \batch\ directory and double-click the ActivityModel.bat. Please refer to the Population Synthesizer and Activity Generator How-To documentation for additional details on these particular TRANSIMS utilities. The ActivityModel batch takes about 10 minutes to run to completion on a typical desktop machine.

Finally, a third batch script is included in the \setup\ directory for adding a subarea simulation to the output of the trip-based model. The SubareaModel.bat file creates the batch files and program controls for running a subarea extraction and simulation. To initialize the run, double-click the SubareaModel.bat in the \setup\ directory. This batch file will create the control files and batch files needed to execute the subarea model. Once this execution has terminated successfully, navigate to the \batch\ directory and double-click the SubareaModel.bat. The SubareaModel batch also takes about 10 minutes to run to completion on a typical desktop machine.