Transit Network Conversion How-To

This document provides basic information on preparing and converting transit data to the format required by TRANSIMS programs. Data typically available from a travel demand forecasting model are used to synthetically generate the detailed information needed for microscopic simulation at a regional scale.

Revision History

9/28/2006	Created by AECOM Consult, Inc.
2/12/2007	Edited by AECOM Consult Inc.
4/15/2010	Edited by RSG, Inc.

Table of Contents

1.0 Assumptions and Prerequisites

This document assumes you have installed TRANSIMS Version 4.0.3 on a Windows or Linux computer system and understand the basic procedures and terminology for executing TRANSIMS programs.

The TRANSIMS software and documentation can be downloaded from http://sourceforge.net/projects/transims/files/ → software. Basic TRANSIMS procedures and terminology are addressed in the Installation and Testing How-To available in the documentation set.

You should also have a basic understanding of traditional transportation planning networks and transit network concepts, such as headways, dwell times, and schedules.

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

This document also describes how to generate ArcView shapefiles for displaying and editing network data in map format. Familiarity with software that can read and display ArcView shapefiles is desirable, but not necessary. A brief description about how this can be done using ArcGIS 9.1 is provided.

1.1 Download Network Data

This how-to document uses network information from Alexandria, Virginia to demonstrate procedures, discuss outcomes, and describe concepts

To download the Alexandria data to your computer or local area network, select http://sourceforge.net/projects/transims/files/ → test data → 4.0.3a Test Cases → Alexandria_4.0.3a.zip

You should create a directory with a name such as

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

and then unzip the downloaded file to this directory.

Prior to the execution of any programs described in this document you need to create the necessary network files by running the **TransimsNet** and **IntControl** batch files as described in the Highway Network Conversion How-to document. Note, the Alex.2005.Net.ConvertNet.bat runs TransimsNet, IntControl, TransitNet, and ArcNet as part of a full batched network conversion process.

2.0 TRANSIMS Transit Networks

The route paths and operational characteristics of the TRANSIMS transit network are defined by the following files:

- Transit Stop
- Transit Route
- Transit Fare
- Transit Driver
- Transit Schedule

The Transit Stop file specifies the physical locations, different from network nodes, along the network links where travelers can board or alight from a transit route. The Transit Route file specifies the list of stops and fare zones for a given route. The Transit Fare file includes the boarding and transfer fares by mode for a zone-based fare system. The Transit Driver file

specifies the list of links traversed by the transit vehicle to service each route. Finally, the Transit Schedule file specifies the scheduled departure times from each transit stop.

If the transit network includes guideways such as rail lines their links and nodes should exist in the network Node and Link files prior to starting the transit conversion process. In this case study, the last four links in the Link file (those having id's 10002, 10003 10004, 10011) represent the Alexandria portion of the D.C. MetroRail network. We included these links in the Input_Link.txt which served as input to the TransimsNet program. Access to those links is restricted to the rail mode by entering the keywords HEAVYRAIL and RAIL to the TYPE and USE fields respectively. Bus-only highway links can also be specified by restricting the type of vehicles operating on these links to BUS. You should make sure that all the links along a transit route are transit accessible. The following codes can be used to define transit accessibility:

```
BUS, LOCAL_BUS, EXPRESS, EXPRESS_BUS, TROLLEY, STREETCAR,
LIGHTRAIL, RAPIDRAIL, REGIONRAIL
```

The TRANSIMS transit network is a schedule-based representation of transit service. For this reason, the Transit Schedule file includes the departure time for each run at each stop on each route over the entire simulation time period. This requirement makes generating a TRANSIMS transit network considerably more complex and detailed than most transportation planners are accustomed to. It often requires planners to identify service characteristics for times of day they have not previously considered (e.g., pre-peak and late night service).

This document describes how the **TransitNet** program can be used to assist planners and others generate a reasonably accurate representation of a schedule-based transit system using data that is typically available from traditional travel demand forecasting models. This process is particularly helpful for studies that need transit schedules for future year forecasts. The following section describes the input files needed for this process.

3.0 Transit Network Input Files

The following two files are necessary for the transit network generation process performed by **TransitNet**:

- Route Header
- Route Nodes

The route service levels expressed in the form of headways are stored in the Route Header file, while the route path is stored in the Route Nodes file. These two files contain information (route headways and node lists) typically used by traditional regional travel demand forecasting models. Software packages such as TP+, TransCAD, and EMME/2 code transit networks in formats that are relatively easy to convert to the TRANSIMS input files. Two optional input files to **TransitNet** are the Park-and-Ride and Fare Zone files containing information about park-and-ride locations and fares between transit zones.

3.1 Route Header Data

The Route Header file contains information about the route number, transit mode, and headways throughout the simulation time period. The sample file provided in the inputs directory is called "Route_Header.txt". This file contains the following fields: ROUTE, NAME, MODE, TTIME, HEADWAY_x, and OFFSET_x. The ROUTE field contains a unique positive integer that identifies each route. The MODE field contains one of the TRANSIMS mode labels listed in Section 2.0

The HEADWAY_x and OFFSET_x fields define the service levels for each time period. The "_x" denotes the time period (e.g., HEADWAY_2 is the headway for time-period 2). Time periods are defined in the TransitNet control file. OFFSET_x is the number of minutes between the beginning of the time period "x" and the first run of the route in that time period. If offset fields are not provided or if the value of the field is negative, the offset will be set as a random percentage of the headway. You can define the optional field TTIME to enter the route travel time in minutes from start to end. If provided, link travel times will be adjusted so that the total running and dwell times sum up to the value entered.

3.2 Route Nodes Data

The Route Nodes file contains information about the path of each transit route, the travel time between nodes, and the dwell time at stop locations. The sample file provided in the inputs directory is called "Route_Nodes.txt". This file contains the following fields: ROUTE, NODE, DWELL, TTIME, and SPEED. The ROUTE field contains the route number listed in the Route Header file described in Section 3.1. Records in this file with the same route ID represent the sequence of network nodes on the route path. The DWELL field defines the amount of time in seconds that each transit vehicle stops at the node defined by the NODE field. A value of zero indicates that the transit vehicle does not stop at or near the node. The TTIME field defines the total travel time from the previous node to this node. If the TTIME field is zero, the program calculates the travel time based on the link length, travel speed and the dwell time. If the SPEED field is not provided, the link free flow speed is used.

3.3 Park-and-Ride Data

A Park-and-Ride file is optional. If provided, it contains the network nodes where park-and-ride lots should be added. The sample file provided in the inputs directory is called "Park_Ride.txt". This file contains the following fields: NODE, CAPACITY, and NOTES. The NODE field identifies the closest highway network node to the park-and-ride lot, while the CAPACITY field defines the number of parking spaces in the lot. If this file is not provided, the Parking file should be manually edited to designate existing parking lots as park-and-ride options. If this is not done, the TRANSIMS Router will not be able to construct travel plans for park-and-ride trips.

3.4 Fare Zone Data

A Fare Zone file is optional. If provided, it equates traffic analysis zones defined in the Activity Location file with fare zones within the transit system. The sample file provided in the inputs directory is called "Fare_Zone.txt". In this file each fare zone is defined using two lines. The first line contains the fare zone ID and a description of the fare zone. The second line contains the range of traffic analysis zones included in the fare zone. A fare zone ID will be assigned to each transit stop based on the traffic analysis zone assigned to near-by activity locations. To have the TRANSIMS Router take into account transit fares when building transit paths, you need to manually create the Transit Fare file containing zone-to-zone fare data. Such a file is not provided in this how-to. For more information about the format of this file, see the Router documentation (Router v4.0.pdf).

4.0 Synthesize a TRANSIMS Transit Network

The synthetic transit network generation process is executed using the utility program called **TransitNet**. This section describes how to set up and run the **TransitNet** program.

4.1 Overview of the TransitNet Program

The TransitNet program reformats and expands the following transit network input files:

- Route_Header.txt
- Route_Nodes.txt
- Park_Ride.txt
- Fare_Zone.txt

described in the previous section into the following TRANSIMS network files:

- Transit_Stop
- Transit_Route
- Transit_Driver
- Transit_Schedule.

One of the major differences between these files is the way transit stops are identified. In most travel demand forecasting networks, transit stops are coded at network intersections (i.e., nodes). TRANSIMS does not permit transit routes to stop in the middle of network intersections. They could be located at the near-side or far-side of the intersection, and there might be more than one stop on a given link. The **TransitNet** program synthesizes transit stops for each link and then re-codes the route nodes to utilize these stops. An activity location and process links are then attached to each stop to connect the transit stop to the walk network. These are used by the TRANSIMS Router to make transfers and provide ways to walk from the origin and destination activity locations to the transit stops.

4.2 The TransitNet Control File

A sample control file for the **TransitNet** program is provided in the control directory. The file "Alex.2005.Net.TransitNet.ctl" is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                                   Convert Transit Network
DEFAULT_FILE_FORMAT                     TAB_DELIMITED
PROJECT_DIRECTORY                       ../

#---- Input Files ----

NET_DIRECTORY
NET_NODE_TABLE                          ../network/Node
NET_ZONE_TABLE                          ../network/Zone
NET_LINK_TABLE                          ../network/Link
NET_PARKING_TABLE                       ../network/Parking
NET_ACTIVITY_LOCATION_TABLE             ../network/Activity_Location
NET_PROCESS_LINK_TABLE                  ../network/Process_Link
NET_LANE_CONNECTIVITY_TABLE             ../network/Lane_Connectivity

ROUTE_HEADER_FILE                       ../inputs/Route_Header.txt
ROUTE_NODES_FILE                        ../inputs/Route_Nodes.txt

PARK_AND_RIDE_FILE                      inputs/Park_Ride.txt
ZONE_EQUIVALENCE_FILE                   inputs/Fare_Zone.txt

#---- Output Files ----

NEW_DIRECTORY                           ../network
NEW_PARKING_TABLE                       Parking_2
NEW_ACTIVITY_LOCATION_TABLE             Activity_Location_2
NEW_PROCESS_LINK_TABLE                  Process_Link_2
NEW_TRANSIT_STOP_TABLE                  Transit_Stop
NEW_TRANSIT_ROUTE_TABLE                 Transit_Route
NEW_TRANSIT_SCHEDULE_TABLE              Transit_Schedule
NEW_TRANSIT_DRIVER_TABLE                Transit_Driver

CREATE_NOTES_AND_NAME_FIELDS            YES

#---- Parameters ----

STOP_SPACING_BY_AREATYPE        100, 150, 200, 250, 300, 350, 400, 450    //---- meters ----
TRANSIT_TIME_PERIODS            5:00, 7:00, 9:00, 15:00, 17:00, 20:00
TRANSIT_TRAVEL_TIME_FACTOR      1.0, 1.1, 1.22, 1.1, 1.25, 1.1, 1.0
MINIMUM_DWELL_TIME              5           //---- seconds ----
INTERSECTION_STOP_TYPE          FARSIDE

TRANSITNET_REPORT_1             FARE_ZONE_EQUIVALENCE
```

The program begins by copying all of the activity locations, parking lots, and process links from the input files to the output files. It then identifies a link with auto access and transit service that is connected to each node listed in the park-and-ride file. A new parking lot, an activity location, and process links are added to the output files for each park-and-ride lot.

The program then converts the route information from the Router Header and Route Nodes files into TRANSIMS Stop, Route, Schedule, and Driver files. The list of nodes is checked against links in the link file. If a given node was deleted during the highway network conversion process, the **TransitNet** program skips this node and adds the travel time to the next node. If a link is not found between each node pair in the collapsed network, an error message is generated and the program is terminated. The list of network links is stored in the transit driver file.

Transit stops are added to links along the routes based on the stop spacing criteria. The `STOP_SPACING_BY_AREATYPE` key defines the number of meters between transit stops for links in different area types. The area type of a link is based on the zone numbers assigned to the activity locations located on the link. The zone file defines the area type for each zone. The dwell time field in the Route Nodes file determines if a given route stops at the transit stops assigned to a link.

Given the route and the travel times between stops, the program generates the route schedule. The program assumes that the first time period starts at midnight (i.e., zero) and the last time period ends at midnight (i.e., 24:00). The values listed in the `TRANSIT_TIME_PERIODS` key are the breakpoints between time periods. In this example, time period #1 goes from 0:00 to 5:00, time period #2 goes from 5:00 to 7:00, etc., until time period #7 from 20:00 to 24:00. This means that the Route Header file should include seven headway fields (e.g., HEADWAY_1 = 0:00 to 5:00).

The program then uses the headway and offset values for each time period to schedule runs within the time period. If the headway is zero, the period is assigned zero runs. The start time of the first run in each period is based on the start time of the period plus the offset. Each run after that starts “headway” minutes later until the end of the time period.

The transit schedule file includes the estimated departure time from each stop of each run in each time period. The travel time between stops is used to calculate the departure time from each stop. This travel time can be adjusted by time of day using the `TRANSIT_TRAVEL_TIME_FACTOR` key, which includes a travel time adjustment factor for each time period. This example includes a travel time adjustment of 1.22 for the 7:00 to 9:00 time period. As the time of day on a route progresses from one time period to another, the travel time on each link along the path is adjusted accordingly.

4.3 Reviewing the Results

The **TransitNet** program can be executed using the batch file included in the batch directory:

Alex.2005.Net.ConvertNet.bat (Windows)

Note, the Alex.2005.Net.ConvertNet.bat runs TransimsNet, IntControl, TransitNet, and ArcNet as part of a full batched network conversion process.

The printout file "Alex.2005.Net.TransitNet.prn" will be created by the process in the control directory, as well as new data files stored in the network directory. If a transit route has been coded incorrectly because there is no lane connectivity between the links defined by two subsequent nodes in the Route Nodes file, the **TransitNet** program will display an error message and terminate its execution. The printout file may also include warning messages that should be reviewed before continuing further. For example, the program will print a warning for every node identified as a park-and-ride lot that does not have auto access. In such a case, the user should review all the links adjacent to this node to verify that cars are allowed on them. If such a warning message remains unresolved, the TRANSIMS Router program will not be able to build paths using this park-and-ride lot.

5.0 How to Create ArcView Shapefiles from a Transit Network

In most cases, the user will want to review the synthetic transit routes to confirm the stop locations and correct site-specific issues. The best way to do this review is typically through a network map. The **ArcNet** utility enables the user to convert the TRANSIMS highway and transit network to a series of ArcView shapefiles that can be displayed and edited in ArcGIS or other mapping software. This section describes how to set up and run the **ArcNet** program to display the highway and the transit network.

5.1 ArcNet Program Options

Section 6 of the Highway Network Conversion How-To provides an overview of the **ArcNet** program and presents the control file needed to visualize the highway network. This section describes the additional controls that can make transit review and editing more convenient.

The **ArcNet** program includes a number of "offset" keys that enable you to control the way the shapefiles are constructed. Most TRANSIMS data files identify the location of a given object based on its offset along the link. In this case, the term "offset" refers to the distance along the link where the object is located. **ArcNet** uses this information to determine where the object starts and ends along the link's shape. The **ArcNet** side offset keys are used to control the offset of the object to the right of the centerline of the link. These keys are used to spread the objects out so they all are not drawn on top of one another. Separate offset keys are provided for controlling each direction of two-way roadways, each direction of a transit route, and the location of stops, parking lots, and activity locations.

Another option for drawing transit routes is controlled by the combination of data files that are provided to the ArcNet program. Basic route paths can be drawn using only the Route Header and Route Nodes files. Alternatively, the user can provide only the Transit Driver file. If the Transit Route and Stop files are provided without a Transit Driver file, the program will draw the route as straight-line links between the stop locations. If all three files are provided, the route is drawn along the roadways with pull-outs shown for each stop. If the Transit Schedule file is also provided, the program will summarize the number of runs and average headway on the route in any user-defined set of time periods.

5.2 The ArcNet Control File

A sample control file for the **ArcNet** program is provided in the control directory. The file "Alex.2005.Net.ArcNet_TransitNet.ctl" is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                                Convert the Network to Shapefiles
DEFAULT_FILE_FORMAT                  TAB_DELIMITED
PROJECT_DIRECTORY                    ../

#---- Input Files ----

NET_DIRECTORY                         ../network
NET_NODE_TABLE                       Node
NET_LINK_TABLE                       Link
NET_SHAPE_TABLE                      Shape
NET_ACTIVITY_LOCATION_TABLE          Activity_Location_2
NET_PARKING_TABLE                    Parking_2
NET_PROCESS_LINK_TABLE               Process_Link_2
NET_POCKET_LANE_TABLE                Pocket_Lane
NET_LANE_CONNECTIVITY_TABLE          Lane_Connectivity

NET_TRANSIT_STOP_TABLE               Transit_Stop
NET_TRANSIT_ROUTE_TABLE              Transit_Route
NET_TRANSIT_SCHEDULE_TABLE           Transit_Schedule
NET_TRANSIT_DRIVER_TABLE             Transit_Driver

#---- Output Files ----

ARCVIEW_DIRECTORY                     ../network/arcview

#---- Parameters -----

DRAW_NETWORK_LANES                   YES
LANE_WIDTH                           3.5          //---- meters ----
LINK_DIRECTION_OFFSET                0.0          //---- meters ----
ACTIVITY_LOCATION_SIDE_OFFSET        15           //---- meters ----
PARKING_SIDE_OFFSET                  5           //---- meters ----
TRANSIT_STOP_SIDE_OFFSET             8           //---- meters ----
TRANSIT_DIRECTION_OFFSET             4           //---- meters ----
TRANSIT_OVERLAP_FLAG                 NO

OUTPUT_COORDINATE_SYSTEM              UTM, 18N, METERS
```

This application will create shapefiles for the new Activity Location, Parking, and Process Link files. These files will display the connections to the transit stops and the park-and-ride lots. It also creates two shapefiles for the transit service: one for transit stops and one for the transit routes. Information from the Transit Route, Schedule, and Driver Plan files is combined into a single shapefile. The shapefile will be called "Transit_Route.shp". The routes in each direction will be offset from the roadway centerline by 4.0 meters, the stops will be offset by 8.0 meters, the parking lots by 5.0 meters and the activity locations will be offset by 15.0 meters.

A similar process occurs if the user provides Route Header and Nodes files rather than the transit network files generated by **TransitNet**.

```
ROUTE_HEADER_FILE              Route_Header.txt
ROUTE_NODES_FILE               Route_Nodes.txt
```

In this case a shapefile called "Route_Nodes.txt.shp" is generated where the polyline is constructed from the coordinates of the nodes in the Route Nodes file and the data file includes the fields in the Route Header file.

5.3 Visualizing the Results

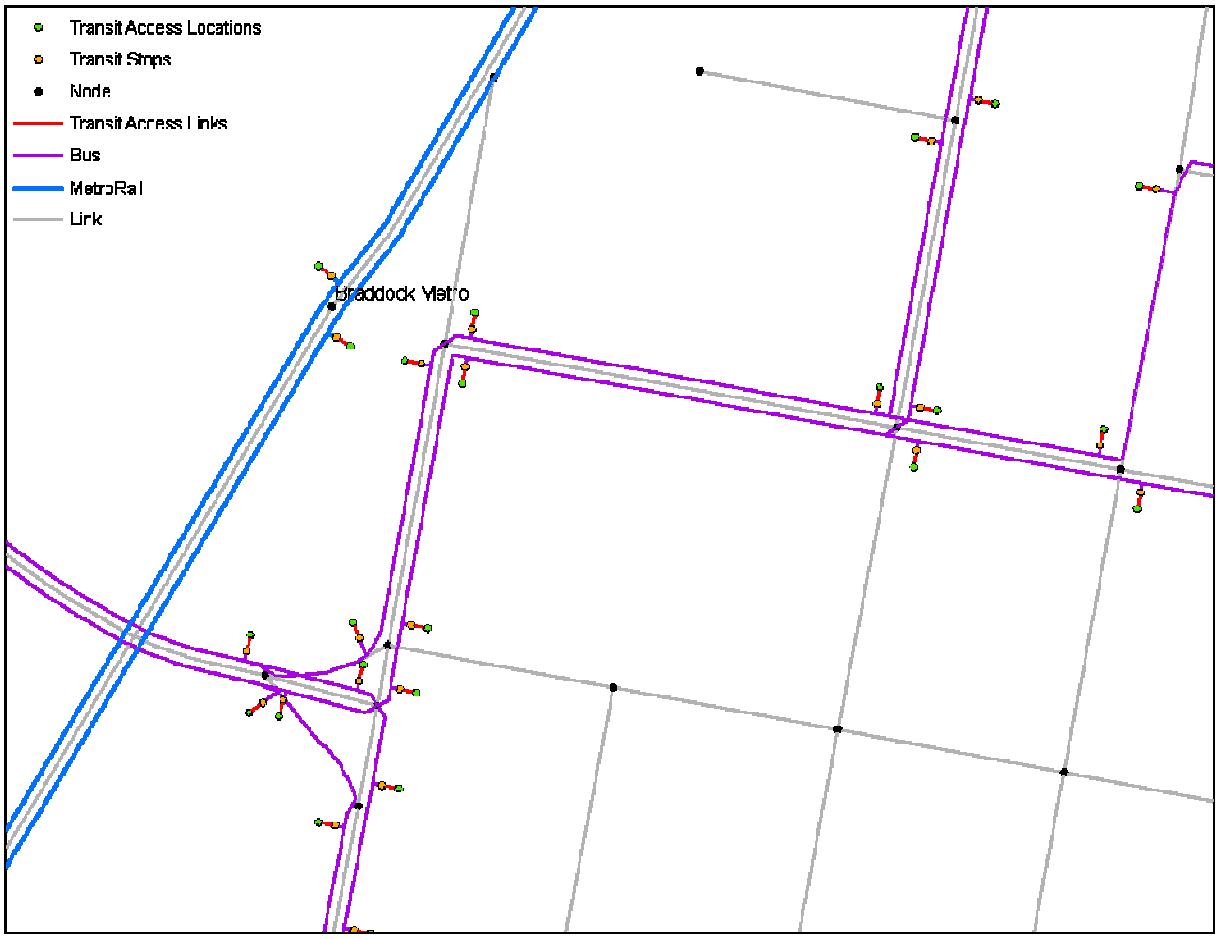
The **ArcNet** program can be executed using the batch file included in the batch directory:

Alex.2005.Net.ConvertNet.bat (Windows)

Note, the Alex.2005.Net.ConvertNet.bat runs TransimsNet, IntControl, TransitNet, and ArcNet as part of a full batched network conversion process.

The printout file "Alex.2005.Net.ArcNet_TransitNet.prn" should be created by the process in the control directory along with the new ArcView shapefiles in the arcview subdirectory of the network directory. The parameters selected above will create a map similar to the one shown below. In this example, the transit lines, stops, and activity locations are shown offset to the right side of the centerline of the roadway. Note that all stops in TRANSIMS are also placed 5 meters before or after the intersection node based on the stop type (NEARSIDE or FARSIDE) specified in the **TransitNet** control file.

Notice also that the Braddock Metro Station is not directly connected to the nearby transit lines. Any network link that permits walk access can be used in a transit path to transfer between stops at an intersection or rail station. Since you can not walk on the MetroRail links, the coding needs to be changed to connect the station to the nearby walk network. This can be done by reassigning the activity location attached to the rail station to a nearby roadway or by adding a process link to attach the station to a nearby bus stop.



6.0 Troubleshooting

Transit Node Problems

When nodes are removed from the highway network by the **TransimsNet** program, the **TransitNet** program may identify connectivity or endpoint problems on a transit route. Such a case arises when a node serves no purpose in the highway network, but is critical to a transit route. If a critical mid-block transit station was deleted by the **TransimsNet** program because it was not needed for highway access, the best course of action may be to rerun the **TransimsNet** program with a `KEEP_NODE_LIST` key that points to a file containing the list of nodes not to be removed from the network. If you add the critical transit stop locations to this file, **TransimsNet** will not delete these nodes from the highway network and thereby make the nodes available for transit stops. Alternatively, the end point of the transit route could be modified to start or end at a node that is preserved in the highway network.

Lane Connectivity Problems

In order for the Router and Microsimulator to build paths on a transit network, there must be lane connectivity between each link included in the route. If the user includes the Lane Connectivity

file (the output of **TransimsNet**) in the **TransitNet** application, the program will check these conditions. If a warning is included in the printout file about lane connectivity, the user should review and address each location. In most cases, a connection can be added to the lane connectivity file to resolve the problem. This frequently involves adding a U-turn connection to a link where the bus turns around. In some situations, however, it may be better to add network links to more accurately replicate the turn around maneuver than to simply add a U-turn. In the TRANSIMS Microsimulator, U-turns can cause undesirable backups at busy intersections.

Stops on Transit-Only Links

Access to rail stations can be a major problem that users often overlook. In order to access a transit station or stop, the TRANSIMS Router requires walk access to the stop. Stops are located on links, and links are assigned access restrictions. If the access restrictions on the link do not permit walking, the traveler will not be able to walk directly to the stop. A process link connected to an activity location that is attached to a link that does permit walking is thus required to access the stop. This is a particular problem for rail stations because rail links typically do not permit walking. This tends to be a better solution than adding walk access to the rail link since this can have undesirable consequences if walking trips start using the rail link to travel from one place to another.

7.0 Frequently Asked Questions

Are there utility programs to convert files from other software packages to a TRANSIMS input network?

> Utilities for EMME/2 and TP+ have been written to translate data from these packages to TRANSIMS route header and nodes files.

When would I want to code start-time offsets for a given time period?

> The start-time offset concept is a way to coordinate transit routes that share a common segment or are scheduled to facilitate timed transfers. In order for the TRANSIMS Router and Microsimulator to realistically replicate the wait times experienced by transit users, it needs to know when a bus arrives at a given stop. Traditional models assume that routes on a common segment will be coordinated to provide a uniform headway when all of the routes are combined. This same concept is implemented in TRANSIMS by coordinating the route schedules. The offset parameter is one way this can be implemented.

Why do I need to estimate headways for time periods that are not included in my travel demand forecasting model?

> If TRANSIMS needs to route and simulate daily transit trips, it needs to know the buses and trains that are running at any given time of day in any given direction. Traditional models often use an AM peak transit network to load production-attraction transit trips to the network. This assumes that the PM peak includes an exact mirror image of the transit routes and service levels that are found in the AM peak network. All trips included in

TRANSIMS represent the time of day and direction of travel in an origin-destination format. The PM peak transit routes and service levels must thus be explicitly coded along with the service available during other times of day.