TRANSIMS Version 5

Development Plans and Design Concepts

David Roden, AECOM

User Feedback

- User interface concerns
  - Network files are too cumbersome for efficient editing
    - Version 3/4 field names, multi-file/record dependencies,…
  - It is too easy to introduce errors in control files
    - Inconsistent key names, units of measure, key groups,…
  - Plan file processing and sorting problems
    - Node/link, traveler scaling, multi-leg trips, time/traveler sort,…
  - Partitioning difficulties
    - File extensions vs. command lines, aggregate statistic reports,…
  - How to link tools into modeling algorithms
    - Router/Microsimulator stabilization, user-equilibrium convergence
  - GUI tools for editing, running and visualizing?

Desired Improvements

- Functionality and performance needs
  - A higher fidelity and scalable Microsimulator is needed
    - Cell-based speeds, lost vehicles, signal coordination, …
    - Single processor limitations – simulation size and processing time
  - Better coordination between Router and Microsimulator
    - Plan leg scheduling issues, transit options, on-the-fly re-routing
  - Path attributes to support other models/software
    - Forward and backward path building (time control points)
    - One-to-many skims without creating plan files
    - Linkable routing service class/subroutine
  - The custom data classes are too complicated for new programmers to quickly build upon
    - Needs to be easier to learn/use with fewer/no variations

Guiding Principles – User Help

- Simplify Editing
  - Simplify the network coding requirements
  - Reduce the number of coded dependencies between files
  - Use data nesting to avoid sorting problems and record inconsistencies
- Reduce User Errors
  - Provide more program-based help information
  - Standardize control keys and key definitions
  - Interpret user-provided unit specifications

Guiding Principles – Performance

- Enhance Performance
  - Multi-threading and multi-processor options
  - Streamline the Router → Plan Processing → Microsimulator interaction
  - Enhance the Router and Microsimulator functionality and fidelity
- More Programmer Friendly
  - Standard Template Library
    - strings, streams, vectors, maps, etc.
  - Centralize codes, standardize and automate processing
  - Create DLL services for linkages to other software

User Interface

- English and metric units
  - Control key or global configuration file
    - Defaults to metric for backward compatibility
  - Consistent internal units
    - Tenths of feet (meters), feet/second (meters/second), or seconds
  - Automatically converts units from one system to the other
- Global time format
  - Control key supports minutes and hour clocks (e.g., 27:00)
  - Individual control keys can include time units (e.g., 15 minutes)
  - Model start and end time
    - Multiple days and start times other than midnight (e.g., 3:00)

Control Keys

- New control key data services
  - Nesting levels, optional/required, data type, help messages
    - Written to the screen with –h command
  - Help messages provide default values and units
  - Values read using default units or user overrides
  - Standard methods for data type conversion, range checking and error messaging
  - Key and range values are converted to English/metric units when printed to the report/screen

XML and *.def files

- New XML flag (-x)
  - Creates an XML file containing the control key structure and key values
  - May also contain processing results/values
  - XML2CTL creates a control file from an XML file
    - Creates a full key template for the program
- Enhanced *.def file
  - Specifies software version
  - Includes units descriptions for all fields
    - Enables English/metric conversions
    - Automates text string/time conversion
  - Binary files use codes rather than strings to reduce file size and improve performance

Data Files

- Standard file key conventions
  - Input file keys = *_FILE and *_FORMAT
  - Output file keys = NEW_*_FILE and NEW_*_FORMAT
  - TAB_DELIMITED is the new default file type
    - Several Version 3 file formats are no longer supported.
- Network files
  - Key names changed to standard and "refined"
  - Network directory key dropped
    - All file use the project directory or current working directory
  - Data errors are replaced by warning messages
  - Record IDs are converted to internal indices
    - Relational keys are enforced

Network Redesign

- Primary changes
  - Multi-node signals and reusable timing and phasing plans
  - Transit schedules restructured around run numbers
  - Parking cost and access/egress time by time and use type
  - Tolls added to lane use file
    - Tolls by lane and fixed/variable processing rates
      - Simulate HOT lanes, toll plazas and ramp metering
  - Process links mostly eliminated → access link file
    - Only needed for special connections
  - No pocket lanes in link file – only main lanes
    - Pocket lanes use left or right side numbering
  - Lane range codes use "L" and "R" codes (e.g., 2..L)
    - Reduce lane renumbering problems

Demand Files

- Primary changes
  - Trip and activity files consolidated into trip file
    - OD location/time + Activity duration
  - New plan file
    - Full trip in a single nested record
    - Includes input trip fields and path skim data in header
    - Travel time, distance, cost and impedance on each link
  - Household and person files combined
    - Vehicles numbered using household ID
  - Household List → Selection File
    - Household, person, tour, and trip selection options
  - Version 3.x link-delay and vehicle type files dropped

Router

- Split Router into multiple programs and services
  - Router (trip file) and PathSkim (one-to-many)
  - Router Service added as an executable platform
    - Process control keys and prepare data
  - Path Builder class in SysLib
    - Supports multiple threads and DLL integration
- New features
  - Forward and backward paths based on time constraints
  - Builds paths with or without access links
  - Uses impedance sorting to minimize transit transfer problems
  - Models parking time and cost by time of day
  - Lane use rather than link use restrictions
  - Uses consistent mode codes for all TRANSIMS modules
  - Outputs link-based plans for complete trips
    - No traveler scaling, link vs. node files, walk-leg-only trip problems

Microsimulator

- New design
  - Cell-based → actual vehicle locations and speeds
    - Lane-based car following with intersection control-based sorting
  - Multi-threading and multi-processor (MPI) versions
- New features
  - Multi-node signal coordination
  - Integrated multi-modal trip plans
    - Critical for transit trip schedule coordination
  - No link/vehicle length/speed restrictions
    - Length, maximum speed, and acceleration-deceleration rates
      - Tenths of feet (meters), feet/second (mps), and seconds
    - Output MOVES speed bins (5 mph)
- Performance research
  - Time sorting vs. plan indexing
  - On-the-fly re-routing of lost vehicles

Software Changes

- Standard Template Library
  - All data structures use vectors, maps and strings
    - Easier for C++ programmers to read and write
    - Tighter data management – creation, use, and clean-up
- Consolidated and streamlined program services
  - Centralized and automated code/text conversions
  - Standard processing methods for network and demand files
    - Makes code sharing between programs feasible
  - Screen and report outputs routed through STL streams
  - Expanded functionality for standard strings
    - Trimming, cleaning and parsing, case insensitive comparisons, printf methods
  - Time processing encapsulated in Dtime class
- SysLib can be dynamically linked in Windows and Linux
  - The Boost library is used for multi-threading
  - The copyright notice is now "TRANSIMS Open-Source"