TOP Objects

Definitions and features

The purpose of this document is to present a brief background on the project of the Transportation Object Platform (TOP), and to define certain key objects, their inherent features and the relations between them.

1 Background

For the past 5 years the GIS-group of ScanRail Consult has been engaged in several national and international projects in the field of modeling volume, distribution and socio-economic and environmental impact of transport.

These projects have all shown that the traditional way of handling data in transport models suffers from a number of fundamental problems. These problems are, among others:

- A non-intelligent data model makes it difficult to edit and enforce integrity in the data
- The lack of display tools for complex topologies makes it difficult to visualize data
- The lack of proper tools for scenario management makes it difficult to handle different scenarios
- Different data formats from different vendors lead to numerous one-time translation and aggregation tasks

These problems lead to a very complex and time demanding workflow.

With the introduction of ArcInfo 8 an elegant way of solving all of the mentioned problems is now possible. The solution is to create an intelligent, open and extensible object model for transportation as an extension to the Geodatabase. This is exactly what the Transportation Object Platform is. The scope of TOP however, is broader than the advanced transport models described above. TOP has a variety of useful tools for transportation analysis and management of transportation and infrastructure data. Examples are:

- Tools to optimize and evaluate timetable data
- Environmental impact models

TOP is a fully internally funded software development project at ScanRail Consult, and as such independent. However, the platform is intended to be adapted to comply with the findings of the UNETRANS consortium.

TOP is developed with the intention of optimizing internal procedures in ScanRail Consult and with the intention of sale of the final software.

TOP is developed using CASE tools for designing the Geodatabase Objects, Visual C++ to implement the custom Geodatabase COM objects. Visual Basic is used to develop various ArcMap extensions and Stand-alone applications.

The core functionalities of TOP are implemented as database-customizations. Functionalities such as special purpose editors for transportation data are implemented as extensions to ArcMap. Advanced models for prediction and analysis of transportation and environmental impact are implemented as stand-alone applications using ArcObjects to access the data in the Geodatabase.

ScanRail Consult is an ESRI International Business Partner.

2 A description of the basic objects in the Transportation Object Platform

This section presents a brief introduction to the main objects in the Transportation Object Platform. Each object is described in the terms of its purpose, its relations to other objects and how it behaves in an editing situation. A few examples of real world use are given as well.

2.1 The Physical Network

The Physical network represents static, physical infrastructure, such as streets and their intersections and railroad tracks.

2.1.1 TransportJunction

A TransportJunction is the basic junction element in the physical network. It can represent street intersections, railroad switches, airports, etc. A *TransportJunction*'s primary relations are to its connected *TransportEdges* and *Connectors* and to its related *Turns*. These are updated when a *TransportJunction* is edited. The updates depend on the type of edit being performed. Possible edits are: Create, Delete, Move and Change Attribute.

2.1.2 TransportEdge

A TransportEdge is the basic edge-element in the physical network. Examples of *TransportEdges* are: Streets, railroad links and inland waterways. A *TransportEdge* primarily relates to: *TransportJunction*, *RouteSegment*, *Stop*, *Turn*. Just as with *TransportJunction*, these are updated whenever a *TransportEdge* is edited. Possible edits are: Create, Delete, Move, Reshape, Split, Merge and Change Attribute.

2.1.3 Turn

Turns describe possible movements at *TransportJunctions* between specific *TransportEdges*. *Turns* are used to model turn restrictions in street networks and for detailed street intersection delay modeling. One *Turn* relates to two *TransportEdges* and one *TransportJunction*. However, editing a turn does not necessitate updates to any of these.

2.2 Interchanges

Interchanges is a category of objects, which are used to model passengers changing at stations or bus terminals.

2.2.1 Stop

A *Stop* is a physical point location for embarking and disembarking route-based transportation, such as busses and trains. Passengers are able to change between busses at bus stops and between trains at railroad stations. A *Stop* relates to the physical network via a number of methods. Chief among these are:

- Related to a *TransportEdge* via linear referencing
- Directly related to a *TransportJunction*

Stops are also related to *RouteSegments*. This relation is described in detail in section 2.3.1. When an edit is being performed on a *Stop*, all its related *RouteSegments* will need to be updated. Possible edits are: Create, Delete, Move and Change Attribute.

2.2.2 StopGroup

A *StopGroup* is a logical collection of *Stops*, usually located within close proximity of each other, such as bus stops immediately around a street intersection. Another example is a collection of platforms within a train station. The *StopGroup* object has a number of purposes:

- To represent administrative groupings of *Stops*
- To facilitate easy visualization of *Stops* on an aggregated level
- To enable easy and intuitive editing of *Transfers* and *ChangeEdges*

A *StopGroup* is always represented by a point to which *Transfers* and *ChangeEdges* can connect. The *StopGroup* may also have a polygon associated with it for visualization purposes.

A StopGroup relates to *Stops*, *Terminals*, *Transfers* and *ChangeEdges*. Edits (Create, Delete, Move, Reshape, Split, Merge and Change Attribute) performed on a StopGroup may influence related *Stops*, *Transfers* and *ChangeEdges*.

2.2.3 Terminal

A *terminal* is very similar to a *StopGroup*. The primary difference is that a *Terminal* can contain *StopGroups* as well as *Stops*.

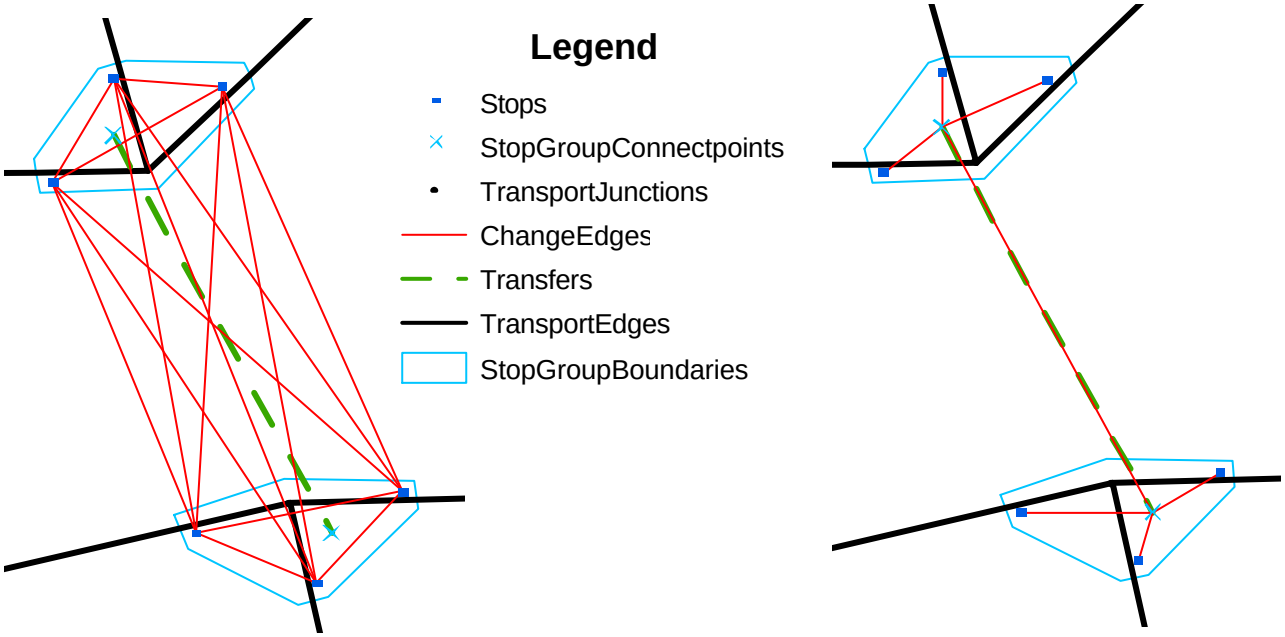
The purpose of a *Terminal* is to enable aggregated editing of all the features in the contained Stops and StopGroups.

2.2.4 Transfers and ChangeEdges

Transfers and *ChangeEdges* are used to specify between which stops passengers can change. *ChangeEdges* are the basic network-elements, which are used by solvers, while *Transfers* are objects, used for aggregated editing of *ChangeEdges*.

This editing is performed by creating *Transfers*, which connect *StopGroups* or *Terminals* with *Stops*, *StopGroups* or *Terminals*. This again, results in the creation of a number of *ChangeEdges* in the underlying network, according to whatever editing policy is in effect. The *ChangeEdges* are related to the *Transfer*, which resulted in their creation.

Two scenarios with different policies for *ChangeEdge*-creation are illustrated below.



Editing *Transfers* results in updates to related *ChangeEdges* and vice versa.

2.3 Route Network

This Category of objects describes the route-based transportation services and its timetables.

2.3.1 RouteSegment

RouteSegments describe the precise way in which a *Route* (such as a bus route) uses the Physical Network between two *stops* on the *route.* As such *RouteSegments* can be considered to be the building blocks for routes. A *RouteSegment* relates to the following other TOP-objects:

- *Stop*: A *RouteSegment* starts and ends at a *Stop*.
- *TransportEdge*: A *RouteSegment* is related via linear referencing to one or a sequence of *TransportEdges*.
- *Route*: A *RouteSegment* is used by 0 to many *Routes.*

A *RouteSegment* need not be related to the physical network. However, if this is the case, the *RouteSegment* will not be considered as valid by TOP.

Edits performed on a *RouteSegment* will only affect related *Routes*.

2.3.2 Route

A *Route* describes the geographical path of e.g. a bus route (or run). It is defined as a sequence of *RouteSegments.* A Route relates to *RouteSegment, RouteGroup, StopPattern* and *Timetable:*

- *RouteSegment*: A *Route* is defined as a sequence of *RouteSegments*
- *RouteGroup*: A *Route* can be a member of a *RouteGroup*
- *StopPattern*: A number of *StopPatterns* can follow a *Route.*
- *TimeTable*: A number of *TimeTables* can use a *Route.*

Edits performed on a *Route* may cause updates to its related *RouteSegments, StopPatterns* and *TimeTables*.

2.3.3 RouteGroup

A *RouteGroup* is an administrative grouping of *Routes*. Its primary purpose is to enable the association to each other of different geographical variations of one administrative route. Each geographical variation will be modeled as a distinct *Route.*

A *RouteGroup* relates to its associated *Routes*. Edits to a *RouteGroup* may cause updates to related *Routes*.

2.3.4 StopPattern

A *StopPattern* defines a collection of *Stops* along a *Route.* The purpose of *StopPatterns* is to be able to model variations in stopping patterns of one distinct geographical route. A *StopPattern* is related to one *Route* and a number of *TimePatterns*. *StopPatterns* do not relate directly to *Stops*. They simply refer to an index in the sequence of *Stops,* which is visited by a *Route.* Edits performed on a *StopPattern* may cause updates to *Routes* and *TimePatterns*.

2.3.5 TimePattern

A *TimePattern* defines driving times between *Stops* in a *StopPattern*. A typical use for *TimePatterns* is to model varying driving times during the day for busses in a specific *StopPattern* along a bus route. A typical cause of these variations is rush hour congestion. A *TimePattern* relates to one *StopPattern* and a number of Runs. Edits performed on a *TimePattern* may cause updates to *StopPatterns* and *Runs*.

2.3.6 Run
A *Run* is a single instance of a *TimePattern*. It is one departure defined by the absolute time of departure from the first *Stop* in a *TimePattern*. A *Run* is related to one *TimePattern.*

2.3.7 TimeTable
TimeTables provide a simpler, disaggregated and more space-consuming method of modeling timetable information, than the method involving *StopPatterns*, *TimePatterns* and *Runs*. A *TimeTable* contains explicit stopping and driving-time information for one specific instance of a *Route.* As such it corresponds to a *Run.* It is possible to convert a *Run* – by using its related *TimePattern* and *StopPattern* – to a *TimeTable*. A *TimeTable* is related to a *Route,* and edits to the *TimeTable* may cause updates to the *Route.*

2.4 Demand
The demand category of objects is primarily used to represent trip-patterns on a zone-to-zone basis. The category also deals with the connection between the zones and the physical network.

2.4.1 Zone
A *Zone* represents an entity, which generates or attracts traffic. As such it is the origin and the destination of traffic. It is represented by a feature of any shape, though – as the name hints, polygons are the norm. A *Zone* is related to one *Terminator*, which provides an ending and starting point for traffic to and from a *Zone* in network analyses. Editing a *Zone* causes updates to its *Terminator*.

2.4.2 Terminator
A *Terminator* is a point feature, which serves as the ending and starting point for traffic to and from a *Zone*. *Terminator* relates to *Zone*, *Matrix* and *Connector*. A *Zone* produces traffic for a *Terminator*. A *Matrix* describes travel patterns between *Terminators*. A C*onnector* links a *Terminator* to a *TransportJunction* or a *Stop*. Editing a *Terminator* causes updates to *Connectors*, *Matrices* and *Zones*.

2.4.3 Connector
A *Connector* is a special network element, which connects *Terminators* with *TransportJunctions* and *Stops*. It often designated 'fictive link' in transportation modeling. A *Connector* relates to one *Terminator* in one end and one *TransportJunction* or one *Stop* in the other end.

2.4.4 Matrix
A *Matrix* describes travel patterns between *Terminators*. It relates only to Terminators. A full *Matrix* contains a matrix element for each combination of from- and to-*Terminators.*

3 Invalid objects
In order to facilitate a flexible editing environment, TOP takes a multi-level approach to the enforcement of integrity constraints. This aspect of the Transportation Object Platform is best explained by an example: A *RouteSegment*, which does not relate to any *TransportEdges* is not a valid object; it can however be stored in a TOP-database. This behavior will be useful in a number of situations. One example is: When making updates to a street network, which is used by a number of bus-routes, it might be necessary to first delete some *TransportEdges* and then add some new TransportEdges, which represent the same actual streets. After this has been done, all the *RouteSegments*, which used the old *TransportEdges* can simply be "associated" to the new *TransportEdges*. This is possible because the *RouteSegments* are not automatically deleted when they become invalid, which happens, when they are no longer associated with any *TransportEdges*.

However, the automatic deletion of invalid objects can be useful in other context. For this reason, TOP provides a multi-level approach to enforcing constraints.

Applications, which are clients of TOP, can query for invalid objects and ask TOP to show the result of this query to users – either geographically or in a table or both.

As the following section will demonstrate, the ability of TOP to store invalid objects is not synonymous with the lack of tools for automatically updating related objects when editing – it is simply a necessary feature for the user to be able to configure TOP to use the precise level of automated updating of related objects in an editing situation, which he or she desires.

4 Editing Policies

Edits performed on most objects mentioned in Section 2 will necessitate updates to their related objects, in order for the entire dataset to stay in a valid state. TOP provides functionalities to automate this.

Users can configure the level of automation with which these updates are performed, according to the list below.

1. Fully automated: All updates will be performed in the background without any notification of the user.
2. Fully automated with user conformation: The user will be presented with a single dialog, asking for confirmation of all edits to related objects or alternatively undoing the original edit.
3. Semi-automated with user confirmation: The user will be asked to confirm updates to individual related objects.
4. No automation: The user must perform all updates manually.

The user has the option of configuring the level of automation to be used (1-4) individually for each possible combination of:

- Source object – The class (*TransportJunction*, *RouteSegment*, *StopPattern*, etc.) of the object, which is being edited.
- Destination object – The class (*TransportJunction*, *RouteSegment*, *StopPattern*, etc.) of the object, which is being updated.
- Source Operation – The type of Edit operation (Create, Delete, Move, Change attribute, Split, etc.) being performed on the object being edited.
- Destination Operation – The type of Edit operation (Create, Delete, Move, Change attribute, Split, etc.) being performed on the object being updated.

All these configurations can be applied at different levels: For the entire database; for a collection of objects (a dataset); or for an individual object.

Scanrail Consult
GIS and Software Development
Pilestraede 58
DK-1112 Copenhagen, Denmark
Phone +45 82 34 21 50
Facsimile: +45 33 12 31 09
E-mail: TOP-info@rdg.bane.dk