



SCENARGIE[®]

Scenargie[®]2.1 Multi-Agent Extension Module

モデルリファレンス

Space-Time Engineering, LLC

2016 年 9 月

目次

はじめに.....	1
1. マルチエージェントシミュレーション	2
1.1. 概要	2
1.2. システム	3
2. シナリオ作成例.....	4
2.1. スクラッチによる簡易シナリオ	4
2.2. 交通流シナリオ	7
2.3. 各種行動モデルと GIS オブジェクトを利用したシナリオ	12
2.4. 災害発生シナリオ	23
2.5. 実地図に基づく都市部シナリオ	26
2.6. 通信機を利用したシナリオ	30
3. シナリオ作成	33
3.1. Visual Lab の起動	33
3.2. 新規シナリオ作成	34
3.2.1. GIS 情報の作成.....	35
3.2.2. エージェントの配置.....	45
3.3. 既存シナリオの読み込み.....	50
3.3.1. ケースファイル.....	50
3.3.2. シミュレーション設定ファイル”.config”のインポート	51
3.4. シナリオのプロパティ設定	51
3.5. グローバル設定	52
3.5.1. シミュレーショングローバル設定	52
3.5.2. マルチエージェントグローバル設定	55
3.5.3. GIS グローバル設定	57
3.6. エージェント設定	59
3.6.1. プロファイル設定	60
3.6.2. 行動設定	62
3.7. エージェントタイムテーブル設定設定	64
3.8. GIS オブジェクト設定.....	68
3.8.1. 道路設定	68
3.8.2. 交差点設定	71
3.8.3. 信号設定	73
3.8.4. バス停設定	75
3.8.5. 建物設定	77
3.8.6. 公園設定	79

3.8.7.	線路設定	81
3.8.8.	駅設定	83
3.8.9.	入り口設定	85
3.8.10.	エリア設定	87
3.8.11.	POI 設定	89
4.	シミュレーション	91
4.1.	シミュレーションの実行	91
4.2.	統計値設定	93
4.3.	統計値表示	94
4.4.	編集したシナリオのコマンドラインからのシミュレーション実行	96
5.	プレイバック	97
5.1.	トレース表示	98
5.2.	ビデオクリップの作成	100
6.	機能構成	101
6.1.	マルチエージェントシミュレータ	103
6.1.1.	エージェントの生成/削除	103
6.1.2.	シミュレーションの実行	103
6.2.	エージェントモデル	105
6.2.1.	エージェント種別	105
6.2.2.	エージェントプロファイル	107
6.2.3.	エージェント行動	109
6.3.	行動モデル	110
6.3.1.	自由歩行	110
6.3.2.	道路歩行	111
6.3.3.	自転車	111
6.3.4.	自家用車ドライバ	111
6.3.5.	バスドライバ	112
6.3.6.	バスゲスト	112
6.3.7.	電車ドライバ	113
6.3.8.	電車ゲスト	113
6.3.9.	タクシードライバ	114
6.3.10.	タクシーゲスト	114
6.4.	マルチエージェント GIS 情報システム	115
6.4.1.	道路	116
6.4.2.	交差点	116
6.4.3.	信号	116

6.4.4.	バス停	116
6.4.5.	建物	116
6.4.6.	公園	116
6.4.7.	線路	117
6.4.8.	駅	117
6.4.9.	入り口	117
6.4.10.	エリア	117
6.4.11.	POI	117
6.5.	経路検索システム	118
6.5.1.	公共交通機関経路(全検索)	118
6.5.2.	公共交通機関経路(制限付き)	118
6.5.3.	道路歩行経路	118
6.5.4.	自転車経路	118
6.5.5.	自家用車経路	119
6.5.6.	タクシー経路	119
6.6.	公共交通機関情報システム	120
6.7.	通信機インターフェース	121
7.	シナリオ構成	122
7.1.	構成ファイル	122
7.2.	コンフィグレーションファイル	124
7.2.1.	プロパティ	124
7.2.2.	統計値設定	126
7.2.3.	トレース設定	126
7.3.	エージェントプロファイル定義ファイル	130
7.3.1.	プロファイルタイプ定義	130
7.3.2.	パラメータ設定	130
7.3.3.	計算式の設定	135
7.4.	エージェント行動定義ファイル	144
7.4.1.	場所のグループ定義	144
7.4.2.	行動タイプ定義	144
7.4.3.	行動リスト設定	145
7.4.4.	パラメータの変更	154
7.5.	エージェントタイムテーブル設定ファイル	160
7.5.1.	車両定義	160
7.5.2.	運行ラインの定義	160
7.5.3.	運賃設定	161

7.5.4. 駅/バス停設定	161
7.5.5. 経由交差点設定	162
7.5.6. 運行スケジュール設定	162
7.6. 信号パターン定義ファイル	169
7.6.1. 間隔指定	169
7.6.2. スケジュール指定	169
7.7. シェープファイル	171
7.7.1. 道路ファイル	171
7.7.2. バス停ファイル	171
7.7.3. 建物ファイル	172
7.7.4. 公園ファイル	172
7.7.5. 線路ファイル	172
7.7.6. 駅ファイル	172
7.7.7. 線路ファイル	172
7.7.8. エリアファイル	173
7.7.9. POIファイル	173
7.7.10. シェープの作成について	173
7.8. OSM(OpenStreetMap)ファイル	174
7.8.1. ファイルの入手	174
7.8.2. ファイルの読み込み	175
8. 行動アルゴリズム	179
8.1. 状態遷移	179
8.2. 自由歩行	184
8.3. 道路歩行	184
8.4. 自転車	185
8.5. 自家用車ドライバ	185
8.6. バスドライバ	187
8.7. バスゲスト	187
8.8. 電車ドライバ	188
8.9. 電車ゲスト	188
8.10. タクシードライバ	188
8.11. タクシーゲスト	189
9. 経路決定	190
9.1. 目的地の決定	190
9.2. 電車・バスを利用した経路の取得(制限付き)	191
9.2.1. 経路候補の作成	191

9.2.2.	終点からの経路検索	192
9.2.3.	始点からの経路検索	195
9.2.4.	料金計算	198
9.2.5.	経路コストの計算	199
9.3.	歩行経路	200
9.3.1.	歩行者経路の計算方法	200
9.3.2.	経路コストの計算	200
9.4.	自転車経路	202
9.4.1.	自転車経路の計算	202
9.4.2.	経路コストの計算	202
9.5.	タクシー経路	203
9.5.1.	タクシー経路の計算	203
9.5.2.	経路コストの計算	203
9.6.	自家用車経路	204
9.6.1.	自家用車経路の計算	204
9.6.2.	経路コストの計算	204
9.6.3.	経路コストの計算	205
9.7.	電車・バスを利用した経路の取得(全検索)	206
9.8.	経路候補の選択	206
9.9.	経路選択アルゴリズム	206
10.	経路再計算と目的地の変更	207
10.1.	経路の再計算	207
10.2.	目的地の変更	207
10.3.	経路変更のタイミング	207
10.4.	経路変更後の移動手段	208
11.	シミュレーション実行時に初期化される変数の確認	209
11.1.	シミュレーション実行時に自動生成される GIS オブジェクト	209
11.2.	自動生成される GIS オブジェクトの確認	210
11.3.	エージェント初期位置の確認	212
11.4.	プロファイル初期値の確認	212
11.5.	プロファイル初期値の再利用	214
12.	API 一覧	215
12.1.	マルチエージェントシミュレータ	215
12.2.	エージェントモデル	216
12.3.	行動モデル	219
12.4.	マルチエージェント GIS 情報システム	222

12.5.	経路検索システム.....	223
12.6.	公共交通機関情報システム.....	224
12.7.	通信機用インターフェース	227
13.	通信とユーザ行動の連携	228
13.1.	通信結果に応じたユーザ行動の指定	229
13.2.	通信機の着脱	237
13.3.	通信アプリケーションでユーザ情報を取得.....	240
13.4.	道路ネットワークに依存する伝搬モデルの利用	241
13.5.	通信と車両の状態の連携	242
14.	その他	243
14.1.	乱数シード.....	243
14.2.	マルチエージェントデバッグ情報	245
14.3.	GIS デバッグ情報	246
14.4.	エージェントの挙動に関して注意点.....	247
15.	参考文献	248

はじめに

本書は、離散事象シミュレータ Scenargie2.0 Multi-Agent Extension Module のモデルリファレンスを示す。

関連ドキュメント

インストレーションガイド
プログラマーズガイド
Visual Lab ユーザガイド
Base Simulator ユーザガイド
Multi-Agent Extension Module ユーザガイド

1. マルチエージェントシミュレーション

マルチエージェントシミュレーションについて解説する。

1.1. 概要

マルチエージェントシミュレーションは意思決定を行う要素単位をエージェントとして定義し、複数のエージェント間での相互作用をシミュレーションする技術である。Scenargie MultiAgentExtensionModule では人(ユーザ)をエージェントとして定義し、人の意思決定と人同士の行動の相互作用を計算機上で仮想的に再現し、実社会を模した人の動きをシミュレーション評価することが可能である。

ユーザはそれぞれが時系列に沿ったの行動予定を保持し、これに従って最適な移動経路・移動手段を選択して地理的な移動を繰り返す。実情報に基づいてユーザのプロファイルと行動の設定を行うことで、現実に則した形でユーザ自身が考え・判断して行動を行う。

各ユーザは自身の行動を独立して計算するが、ユーザの行動は互いに影響を与え合い、流動的に変化する環境に応じてユーザは自身の行動を変更する。ユーザ行動には、歩行、自転車、自動車、バス、電車、タクシーを利用がある。これによりマルチエージェントシミュレーションでは、ユーザ行動の変化による地域の人口密度の変化の評価、交通機関に対する満足度の評価、ユーザに経路情報を提供する与える経路検索システムの評価、災害時の避難行動シミュレーション、または交通渋滞が頻繁に発生する地域での渋滞緩和のためのソリューションの提言といったことを目的としてシミュレーションを行うことが可能である。



図 1-1 ユーザの行動例

1.2. システム

マルチエージェントシミュレーションのシステム概要を示す。マルチエージェントのシミュレーション機能は Scenargie Simulator の Multi Agent Extension Module 上に実装される。

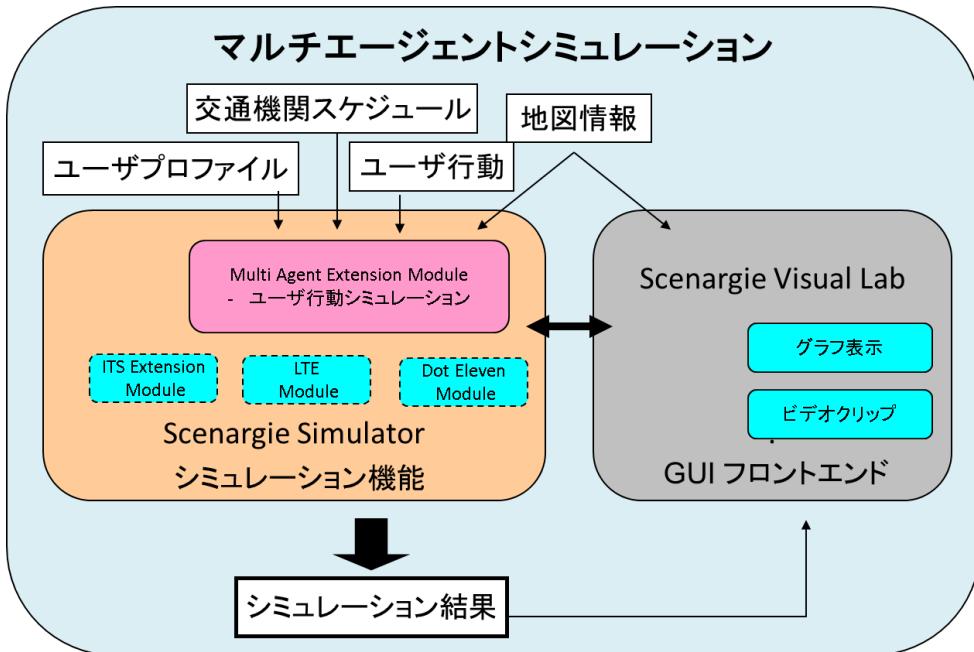


図 1-2 システム概要

ユーザプロファイル:ユーザプロファイルの定義

ユーザ行動:ユーザ行動の定義

エージェントタイムテーブル設定:公共交通機関のスケジュール

地図情報:実環境の地図情報

Scenargie Simulator:シミュレータ

Multi Agent Extension Module:マルチエージェントシミュレーション拡張

※ITS Extension Module:ITS の通信に関する拡張

※LTE Module:LTE 通信拡張

※Dot Eleven Module:dot11 通信拡張

Scenargie Visual Lab:GUI からの表示・編集機能

シミュレーション結果:シミュレーションの統計情報

グラフ表示機能:統計情報の可視化

※移動に関するユーザ行動シミュレーションのみの場合、ITS Extension Module、LTE Module、Dot Eleven Modul は利用しない。これらのモジュールはユーザ行動と合わせて通信のシミュレーションも行う場合に必要となる。

2. シナリオ作成例

シナリオの作成に解説する。典型的なシナリオの作成として以下のシナリオを作成する方法を示す。

- スクラッチによる簡易シナリオ
- 交通流シナリオ
- 実地図に基づく都市部シナリオ
- 避難行動シナリオ

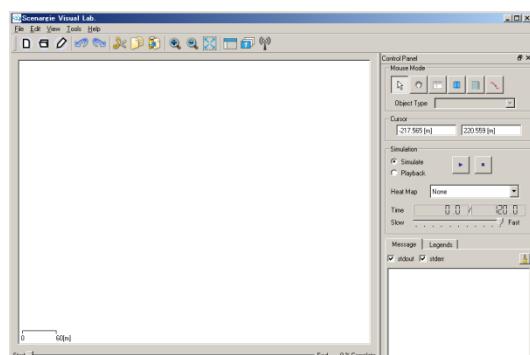
2.1. スクラッチによる簡易シナリオ

エージェントの移動行動をシミュレーションするための最低限のシナリオを作成する。

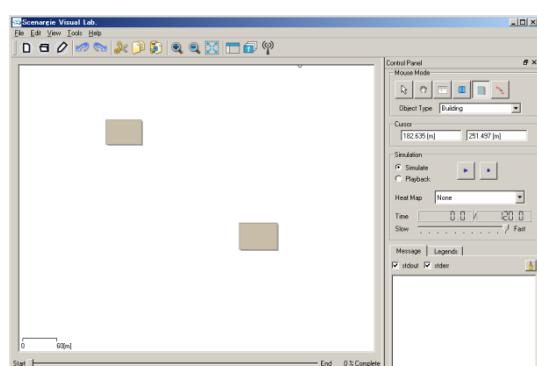
シナリオ概要

人:1人
建物:2個
道路:1本
シミュレーション内容:1人のエージェントの建物間の移動

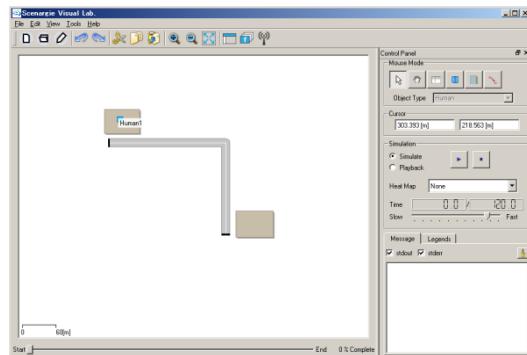
1) [File]->[New]により新規作成する。



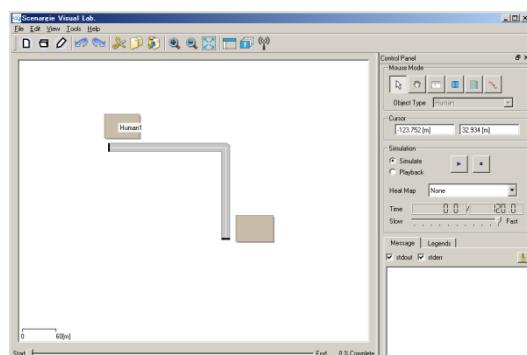
2) 初期配置場所と目的地用に建物を作成する。



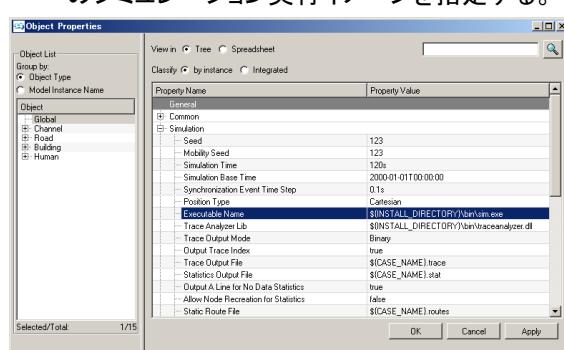
3) 初期配置と目的地をつなぐ道路を作成する。(建物に直接接続する必要はない)



4) Human エージェントを 1 人配置する。



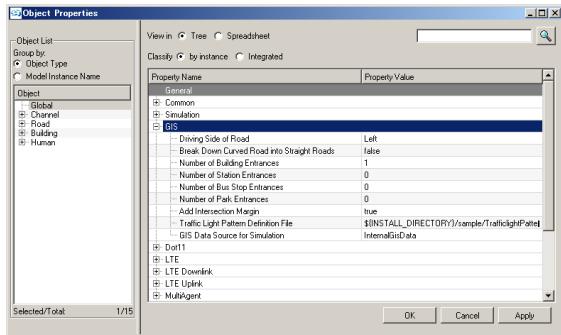
5) [Tools]->[ObjectProperties]の Executable File Name よりマルチエージェントシミュレーション用のシミュレーション実行イメージを指定する。



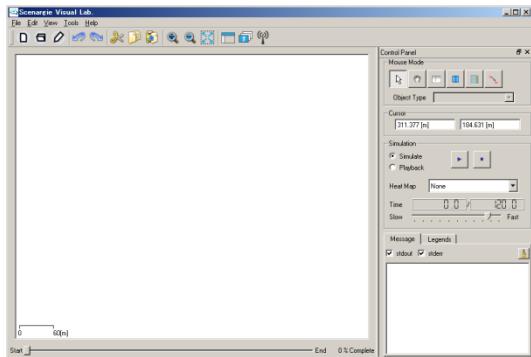
6) [Tools]->[ObjectProperties]より GIS グローバルの設定を行う。

パラメータ名	値
Break Down Curved Road into Straight Roads	false
Number of Building Entrances	1

Add Intersection Margin	true
-------------------------	------



7) Object Properties で OK を選択後、シミュレーションを実行する。



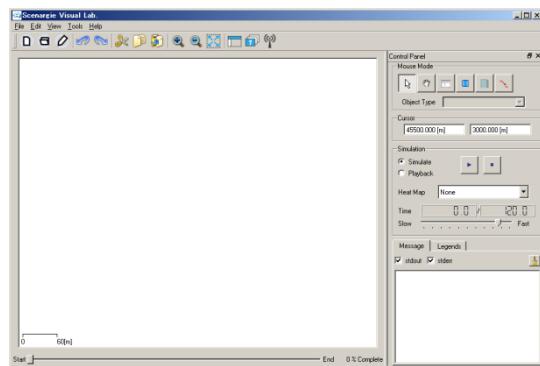
2.2. 交通流シナリオ

交通流シミュレーション実施するためのシナリオを作成する。

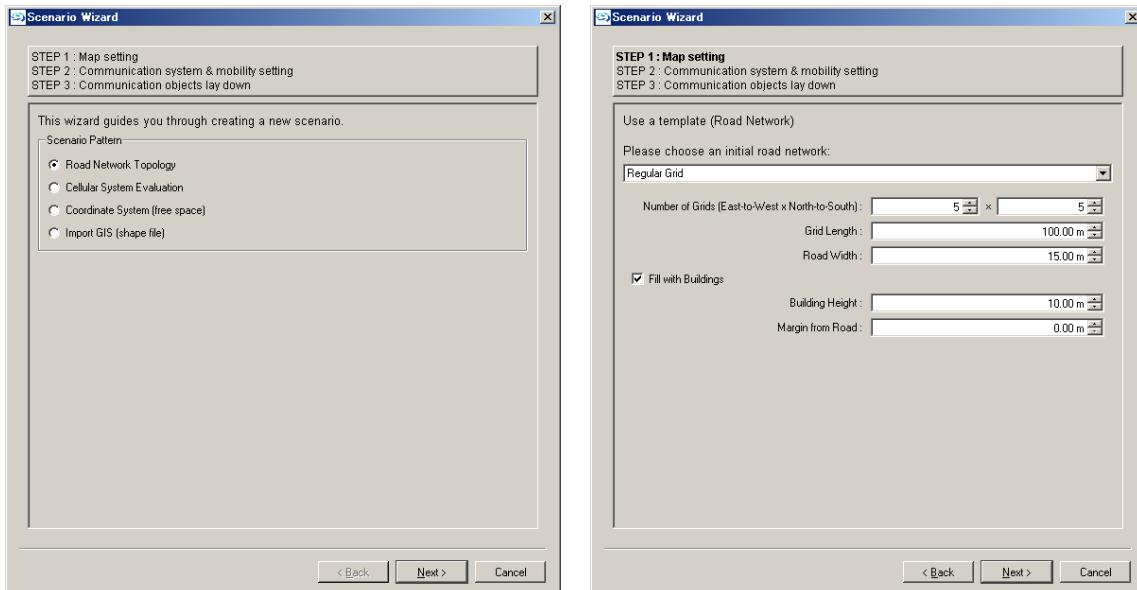
シナリオ概要

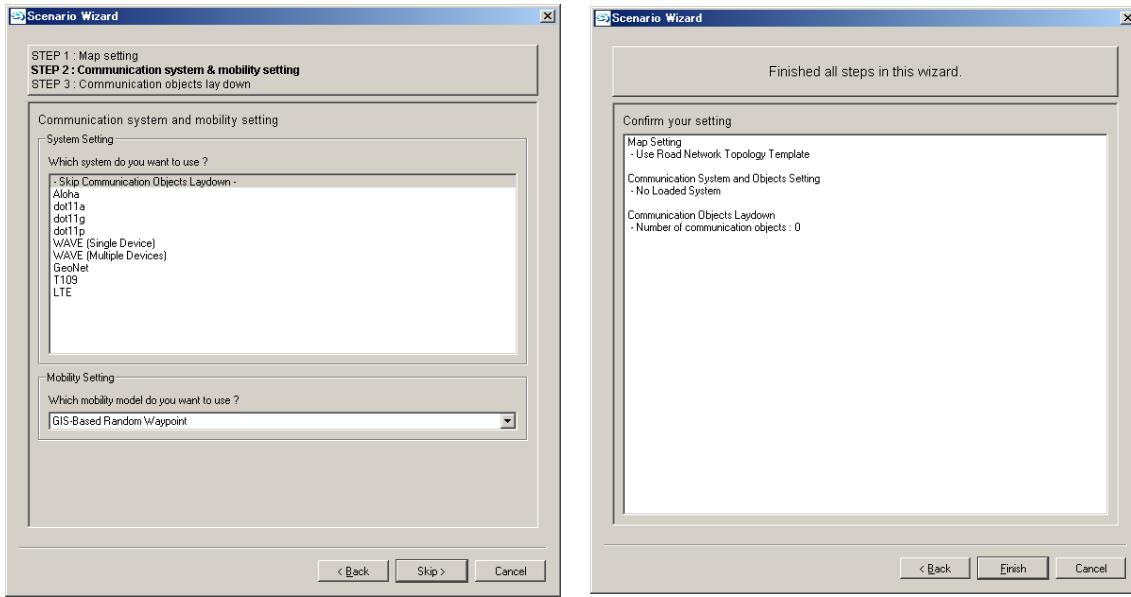
人:50 人
車両:50 台
建物:27 個
道路:25 本
シミュレーション内容:50 人のエージェントの車両による移動

1) [File]->[New]により新規作成する。

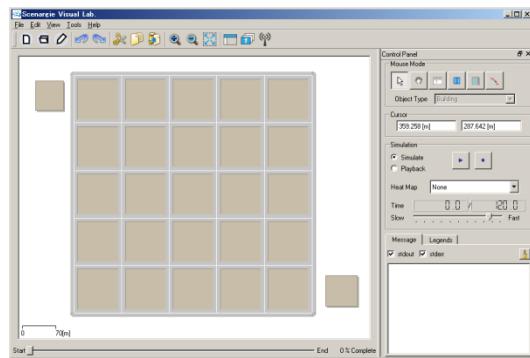


2) [File]->[Scenario Wizard]より、格子状の道路を 5x5 で設定する。Communication Object は配置しない。

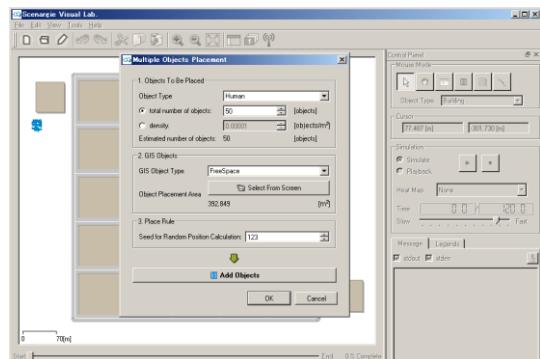




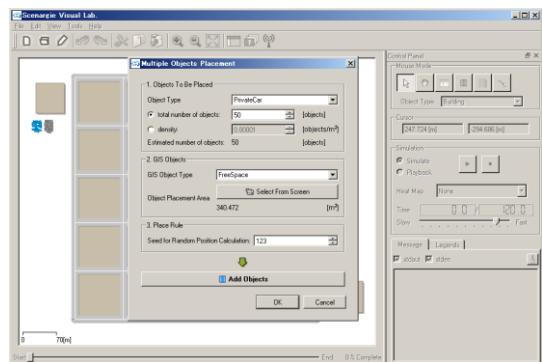
3) 初期配置場所と目的地用に建物を作成する。



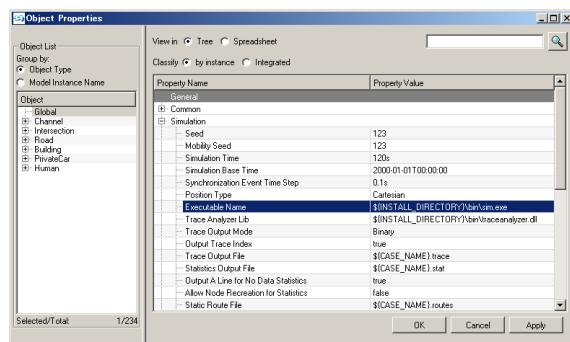
4) [Tools] -> [Multiple Objects Placement] より、Human エージェントを 50 人配置する。



5) PrivateCar を 50 台配置する。

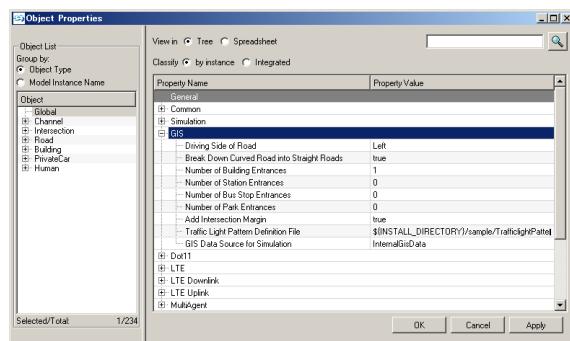


6) [Tools]->[ObjectProperties]の Executable File Name よりマルチエージェントシミュレーション用のシミュレーション実行イメージを指定する。



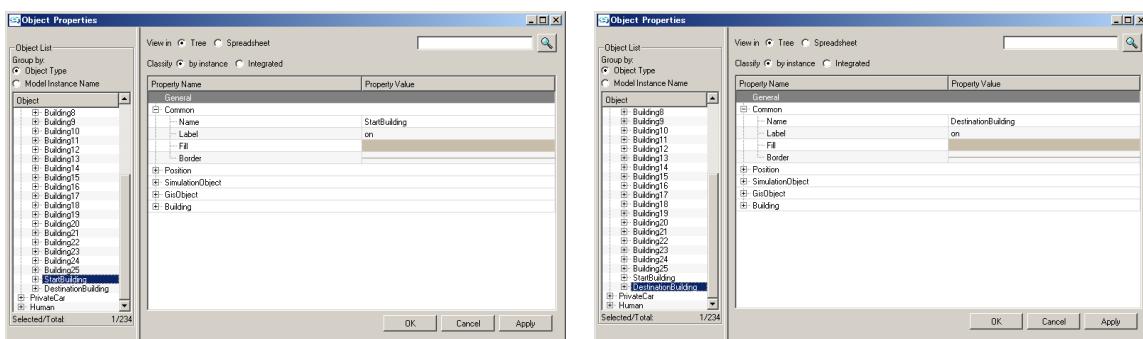
7) [Tools]->[ObjectProperties]より GIS グローバルの設定を行う。

パラメータ名	値
Break Down Curved Road into Straight Roads	false
Number of Building Entrances	1
Add Intersection Margin	true



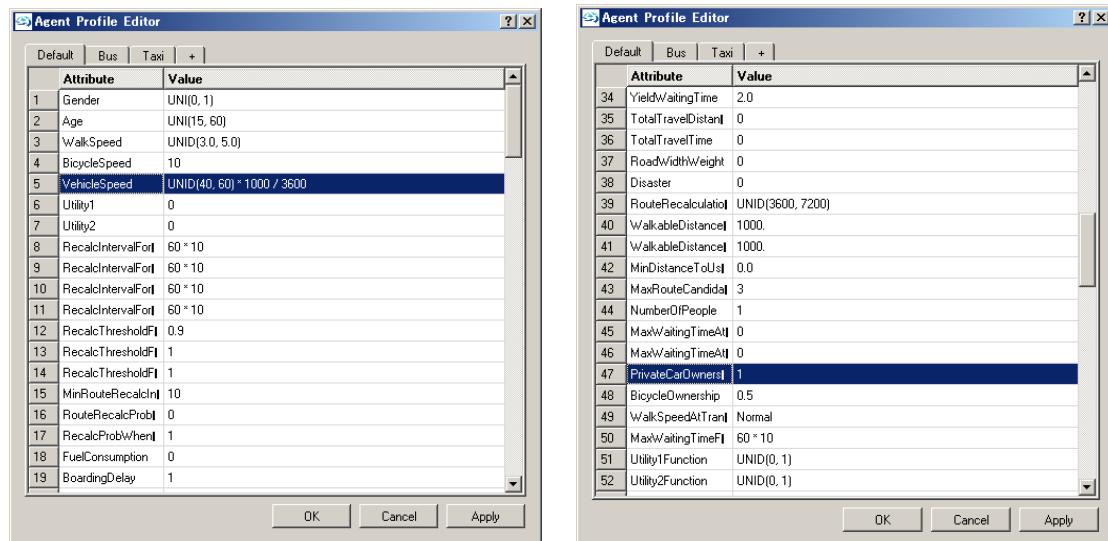
8) [Tools]->[ObjectProperties]より建物の名称を変更する。

パラメータ名	オブジェクト	値
Name	Building26	StartBuilding
	Building27	DestinationBuilding
Label	Building26/Building27	on



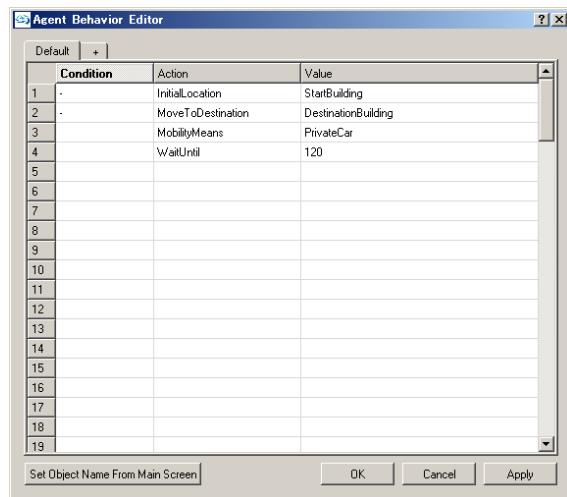
9) [Tools]->[Multi-Agent Settings]->[Agent Profile Editor]よりプロファイルの設定を行う。

パラメータ名	値
VehicleSpeed	UNID(40, 60)*1000/3600
PrivateCarOwnership	1

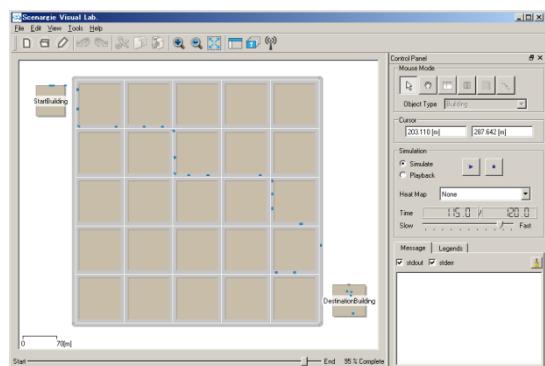


10) [Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]より行動の設定を行う。

- InitialLocation の Value を StartBuilding に設定
- MoveToDestination の Value を DestinationBuilding に設定
- Destination の次の行に、Condition を空欄として MobilityMeans を PrivateCar に設定



11) シミュレーションを実行する。



2.3. 各種行動モデルと GIS オブジェクトを利用したシナリオ

全ての行動モデルを利用し、各 GIS オブジェクトを利用したシミュレーション実施するためのシナリオを作成する。

シナリオ概要

人:200 人

車両:50 台程度

タクシー:5 台

バス:3 台

電車:3 台

建物:27 個

道路:25 本

信号:2 機

入り口:2 つ

公園:1 つ

バス停:3 つ

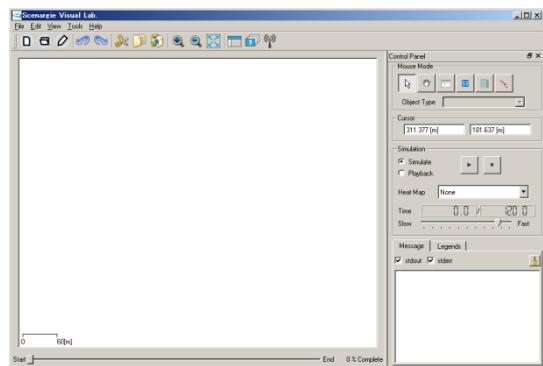
駅:3 つ

線路:2 本

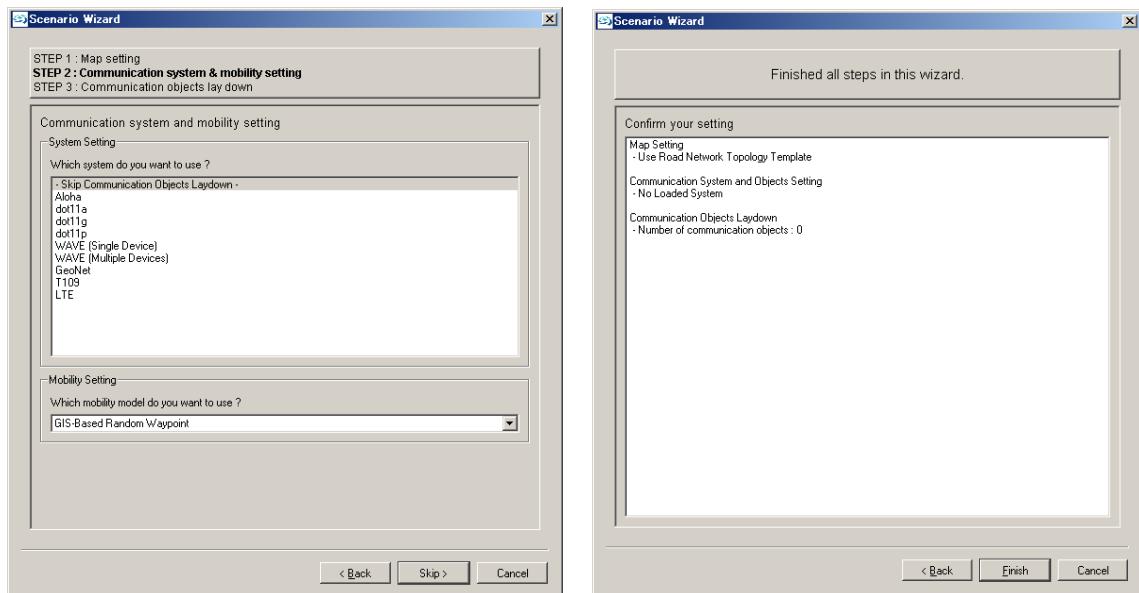
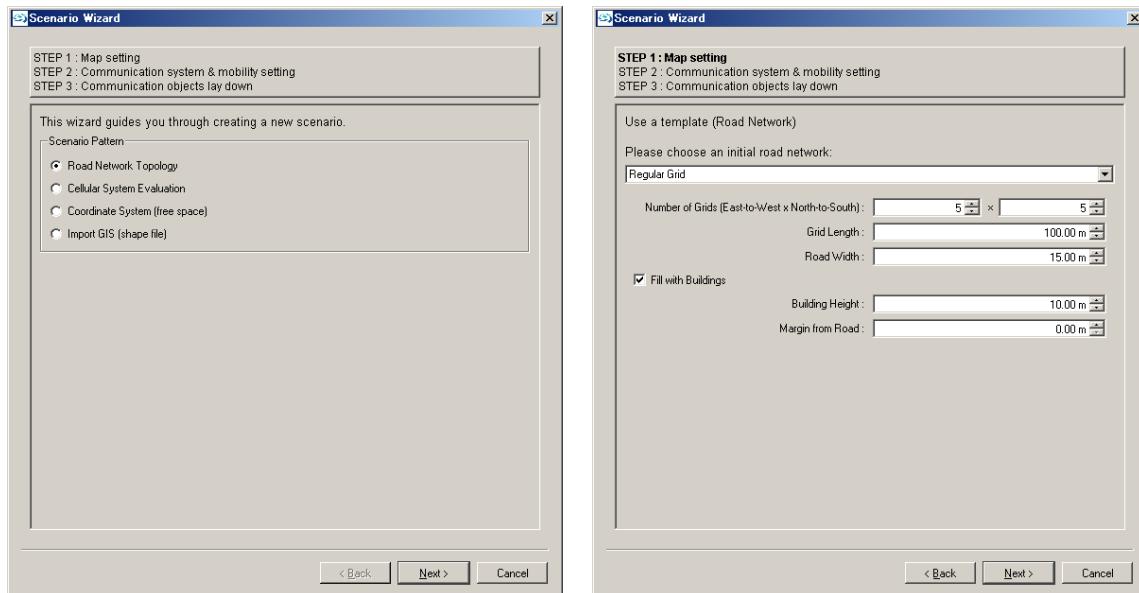
エリア:1 つ

シミュレーション内容:200 人のエージェントが各種行動モデルを利用して移動

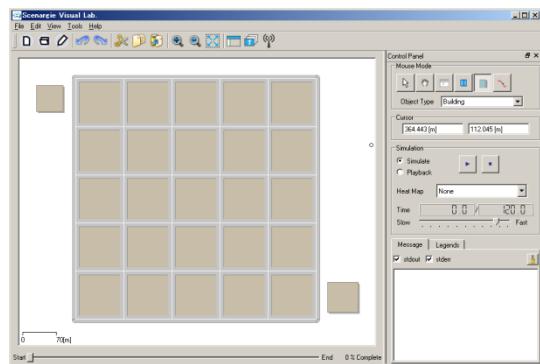
- 1) [File]->[New]により新規作成する。



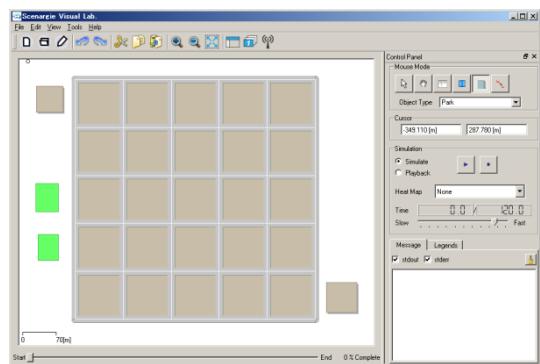
- 2) [File]->[Scenario Wizard]より、格子状の道路を 5x5 で設定する。Communication Object は配置しない。



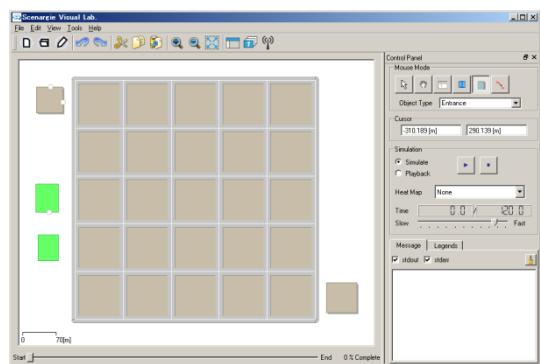
3) 初期配置場所と最初の目的地用に建物を作成する。



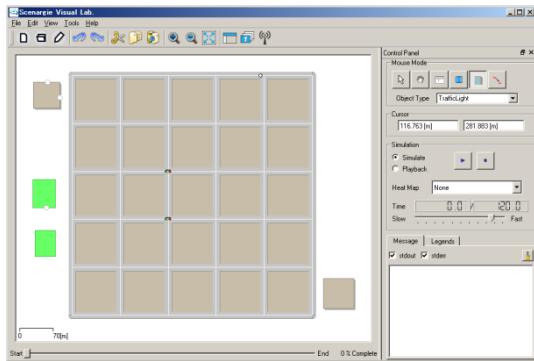
4) 最終目的地用の公園を作成する。



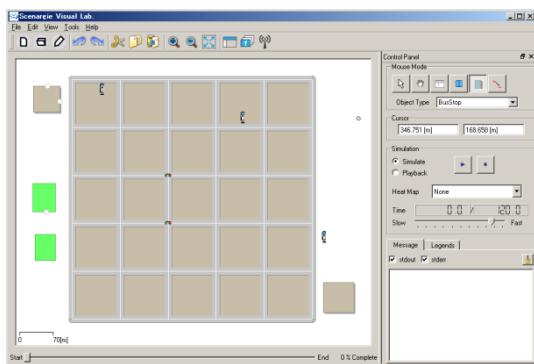
5) 建物、公園に入り口を作成する。



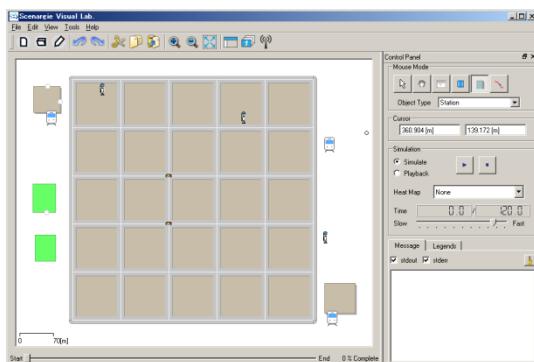
6) 交差点の中心に信号機を追加する。



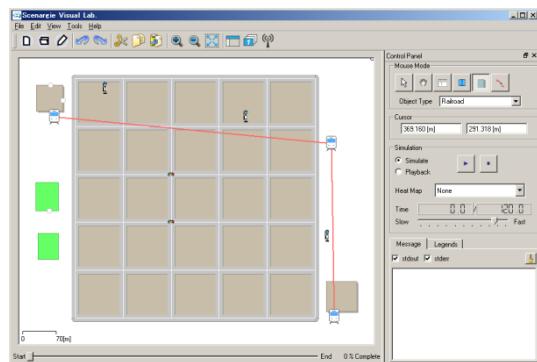
7) バス停を配置する。



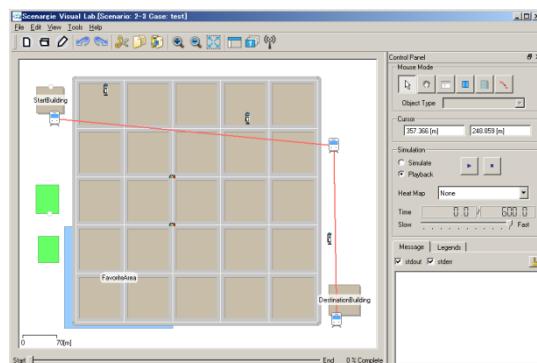
8) 駅を配置する。



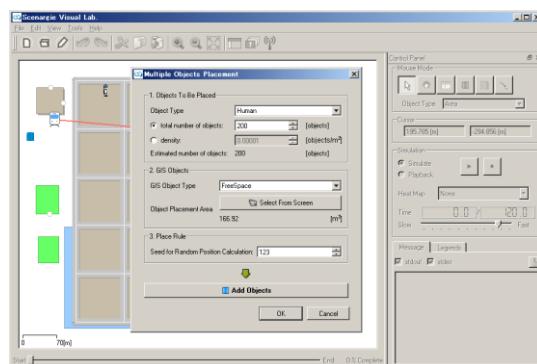
9) 駅間を線路で接続する。全ての駅を一本の線路で繋がずに、駅間で別々の線路を作成する。



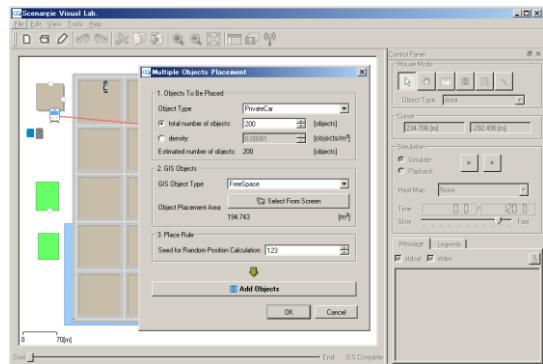
10) エリアを作成する。



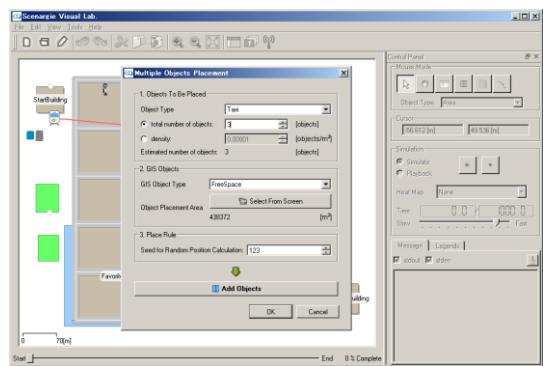
11) Human エージェントを 200 人配置する。



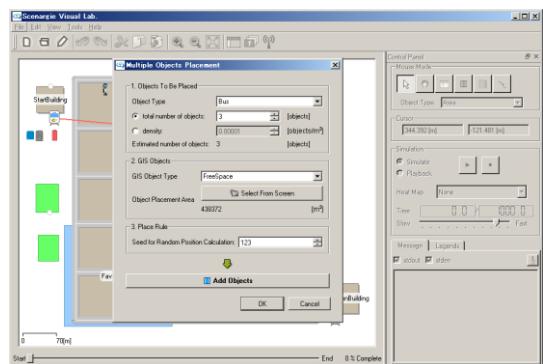
12) PrivateCar を 200 台配置する。



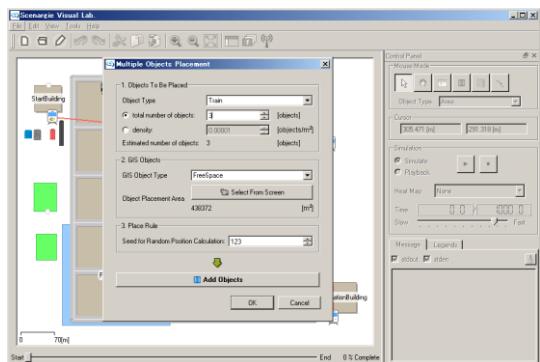
13) Taxi を 3 台配置する。



14) Bus を 3 台配置する。

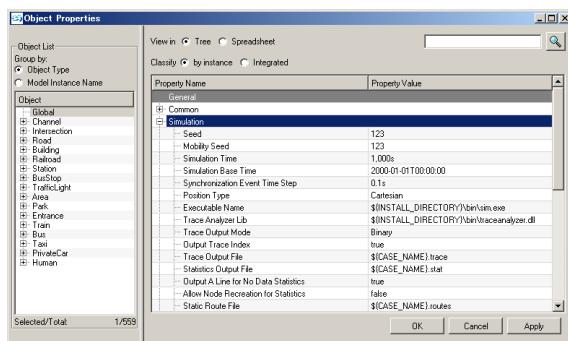


15) Train を 3 台配置する。



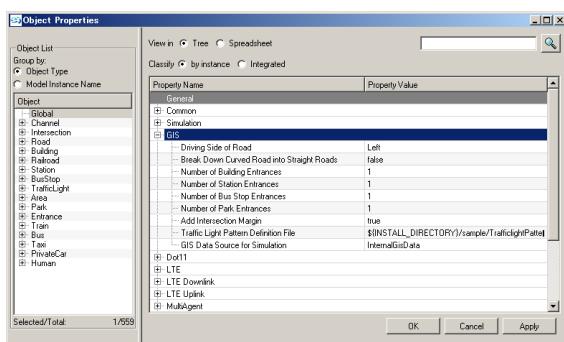
16) [Tools]->[ObjectProperties]よりシミュレーショングローバルの設定を行う。

パラメータ名	値
Simulation Time	1000s
Executable File Name	マルチエージェントシミュレーション用のシミュレーション実行イメージのパス



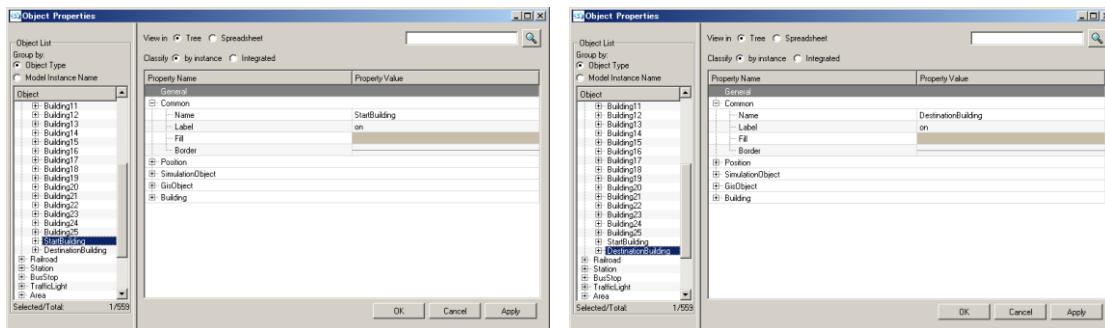
17) [Tools]->[ObjectProperties]より GIS グローバルの設定を行う。

パラメータ名	値
Break Down Curved Road into Straight Roads	false
Number of Building Entrances	1
Number of Station Entrances	1
Number of Bus Stop Entrances	1
Number of Park Entrances	1
Add Intersection Margin	true



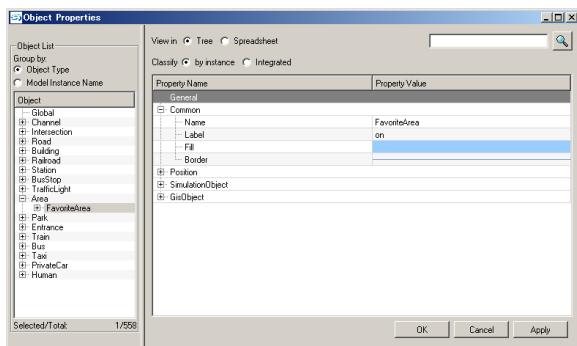
18) [Tools]->[ObjectProperties]より建物の名称を変更する。

パラメータ名	オブジェクト	値
Name	Building26	StartBuilding
	Building27	DestinationBuilding
Label	Building26/Building27	on



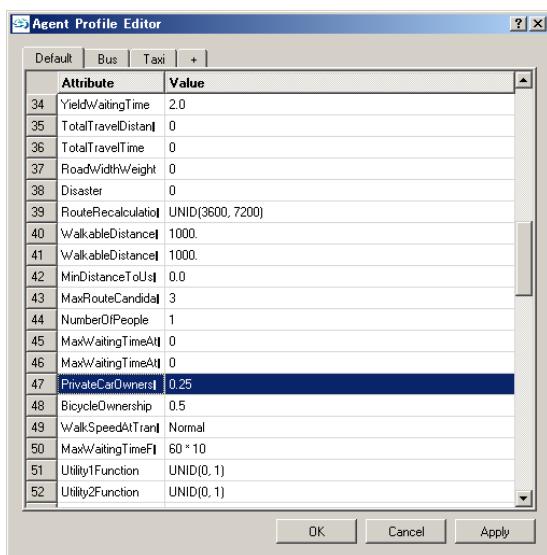
19) [Tools]->[ObjectProperties]よりエリアの名称を変更する。

パラメータ名	オブジェクト	値
Name	Area1	FavoriteArea
Label	Area1	on
Color	Area1	赤色等



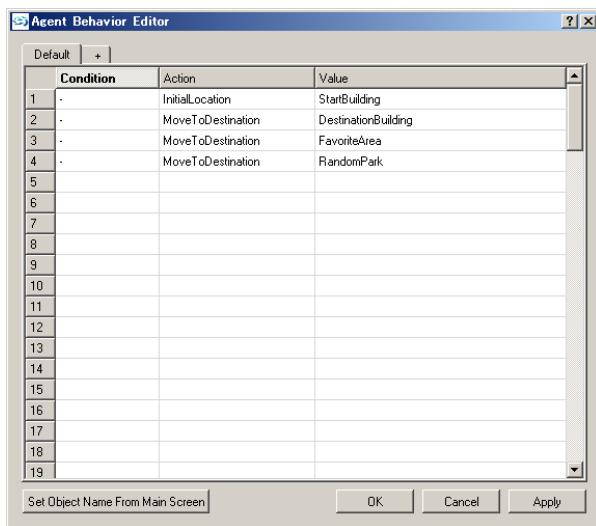
20) [Tools]->[Multi-Agent Settings]->[Agent Profile Editor]よりプロファイルの設定を行う。

パラメータ名	値
PrivateCarOwnership	0.25



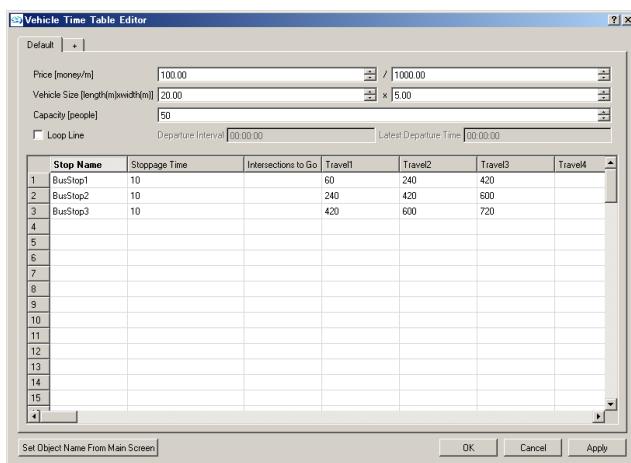
21) [Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]より行動の設定を行う。

- InitialLocation の Value を StartBuilding に設定
- MoveToDestination の Value を DestinationBuilding に設定
- 次の行に、Condition を"-“として FavoriteArea の Destination を追加
- 次の行に、Condition を"-“として RandomPark の Destination を追加



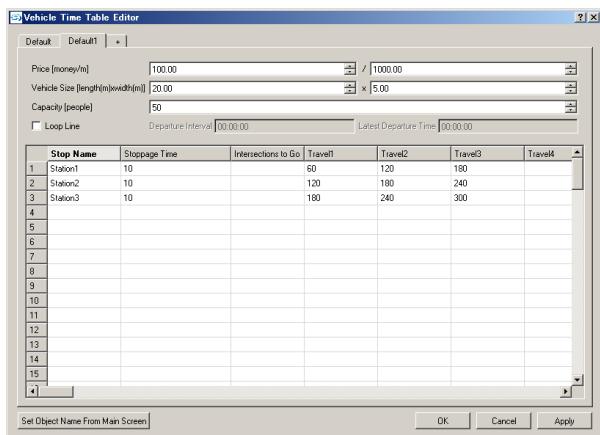
22) [Tools]->[Multi-Agent Settings]->[Vehicle Time Table Editor]よりバスのスケジュール設定を行う。

- Stop Name に BusStop1、BusStop2、BusStop3 を設定
- Stoppage Time に 10[秒]を設定
- Travel1 に 60[秒]、240[秒]、420[秒]を設定
- Travel2 に 240[秒]、420[秒]、600[秒]を設定
- Travel3 に 420[秒]、600[秒]、780[秒]を設定
- Capacity を 50 人に設定

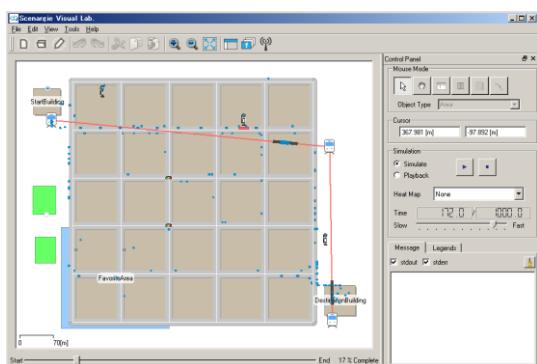


23) [Tools]->[Multi-Agent Settings]->[Vehicle Time Table Editor]より電車のスケジュール設定を行う。

- “+”で新しい運行ラインを追加
- Stop Name に Station1、Station2、Station3 を設定
- Stoppage Time に 10[秒]を設定
- Travel1 に 60[秒]、120[秒]、180[秒]を設定
- Travel2 に 120[秒]、180[秒]、240[秒]を設定
- Travel3 に 180[秒]、240[秒]、300[秒]を設定
- Capacity を 50 人に設定



24) シミュレーションを実行する。



2.4. 災害発生シナリオ

2.3 の各行動モデルと GIS オブジェクトを配置したシナリオを元に作成する。

シナリオ概要

人:200 人

車両:50 台程度

タクシー:5 台

バス:3 台

電車:3 台

建物:27 個

道路:25 本

信号:2 機

入り口:2 つ

公園:1 つ

バス停:3 つ

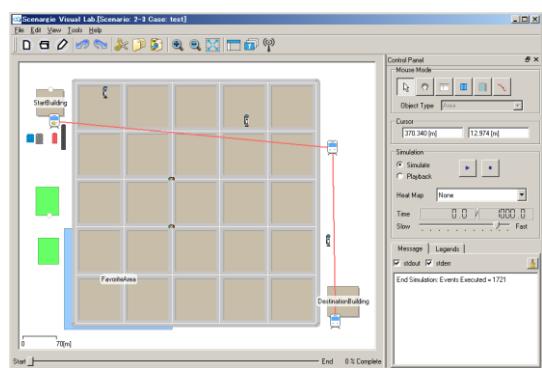
駅:3 つ

線路:2 本

エリア:1 つ

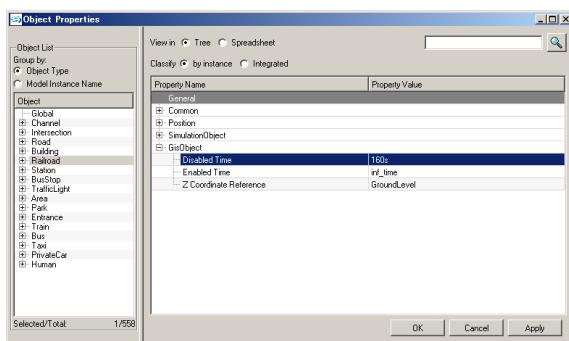
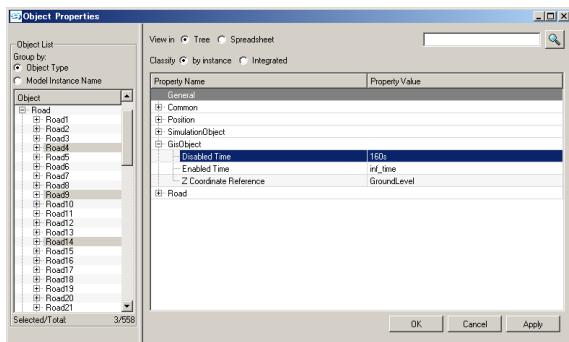
シミュレーション内容:200 人のエージェントが各種行動モデルを利用して移動、160 秒の時点で災害が発生し、道路、線路が被災して利用不可となる。

1) 2.3 に従ってシナリオを作成する。



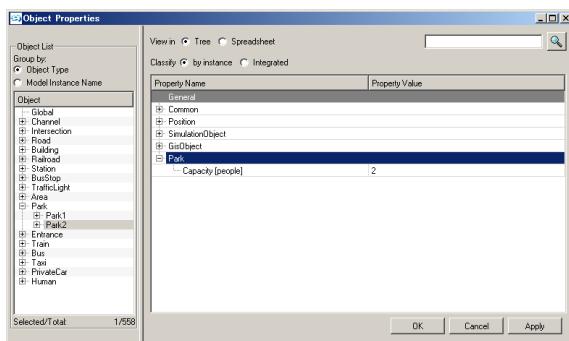
2) [Tools]->[ObjectProperties]より災害発生時刻に道路、線路を無効化する。

- Road4、Road9、Road14
- Railroad2



3) [Tools]->[ObjectProperties]より避難所として利用する公園に定員を設定する。

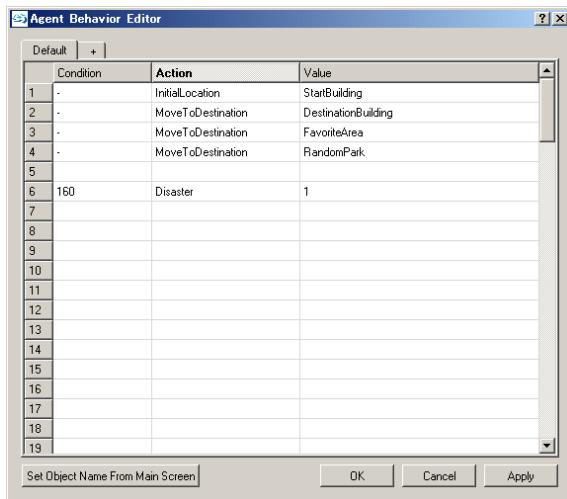
- Park1:1000 人
- Park2:2 人



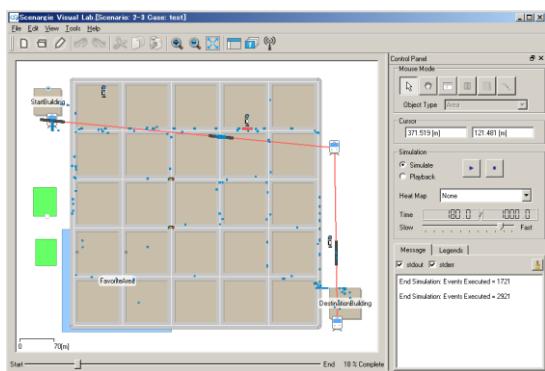
4) [Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]より災害用のプロファイルの設定を行う。公園(避難所)に人がとどまるようにし、災害発生時刻に災害モードのフラグを 1 にする。(災害時に道路の任意の場所を歩行可能とする場合)

- 最終の MoveToDestination の下に Condition を空欄として 1000 秒の Wait 時間を設定する。

- 160 秒の時点で Disaster の値を 1 にする。



5) シミュレーションを実行する。



2.5. 実地図に基づく都市部シナリオ

実地図に基づく都市部シナリオを作成する。

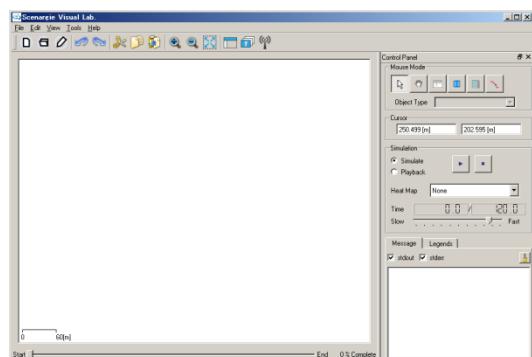
シナリオ概要

人:100 人
車両:50 台程度
地図:銀座エリア
シミュレーション内容:200 人のエージェントが各種行動モデルを利用して移動

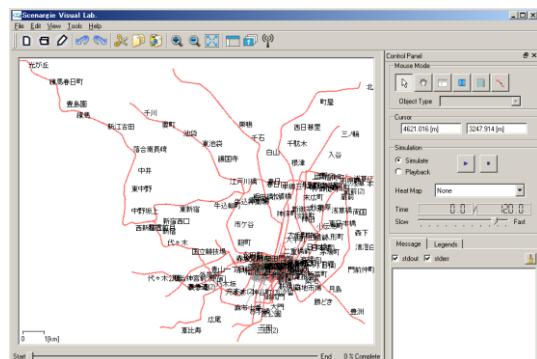
1) OpenStreetMap より銀座エリアの地図を入手する。

- OpenStreetMap のウェブサイトにアクセスする。
<http://www.openstreetmap.org/?bbox=139.76261,35.66872,139.77005,35.67327>
- エクスポートタブより、OpenStreetMap XML データを選択し、map.osm ファイルにエクスポートする。

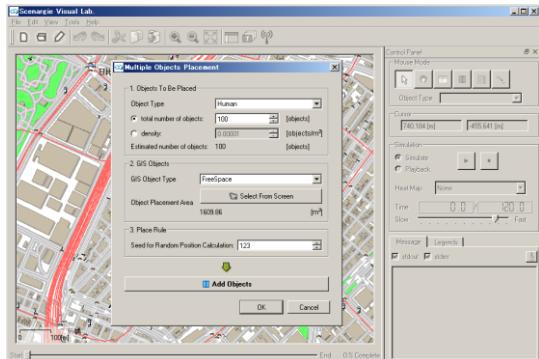
2) [File]->[New]により新規作成する。



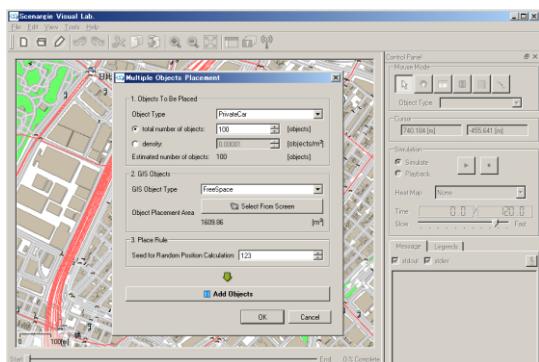
3) [File]->[Import]->[Open Street Map XML (.osm, .xml)]を選択し、OpenStreetMap のウェブサイトよりエクスポートした map.osm をインポートする。



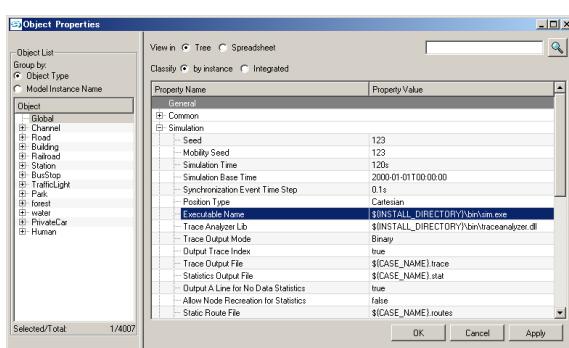
4) Human エージェントを 100 人配置する。



5) PrivateCar を 100 台配置する。



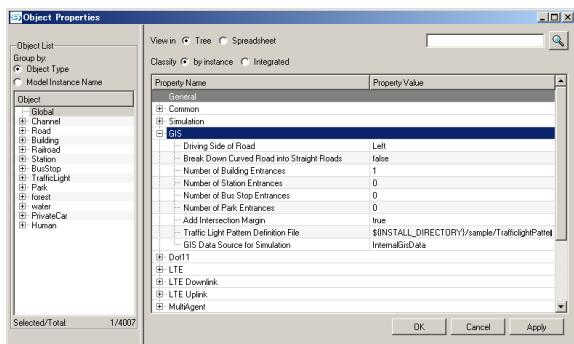
6) [Tools]->[ObjectProperties]の Executable File Name よりマルチエージェントシミュレーション用のシミュレーション実行イメージを指定する。



7) [Tools]->[ObjectProperties]より GIS グローバルの設定を行う。

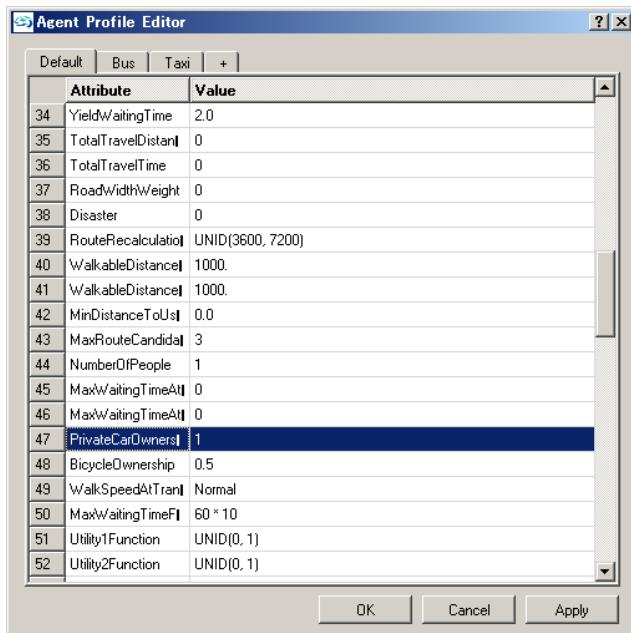
パラメータ名	値
Break Down Curved Road into Straight Roads	false

Number of Building Entrances	1
Add Intersection Margin	true



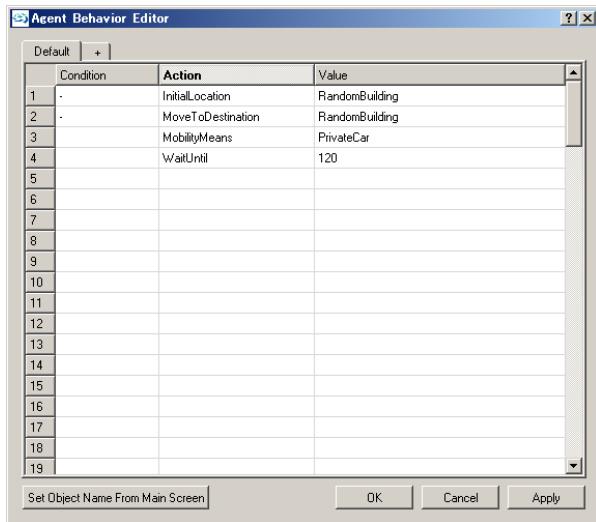
8) [Tools]->[Multi-Agent Settings]->[Agent Profile Editor]よりプロファイルの設定を行う。

パラメータ名	値
PrivateCarOwnership	1

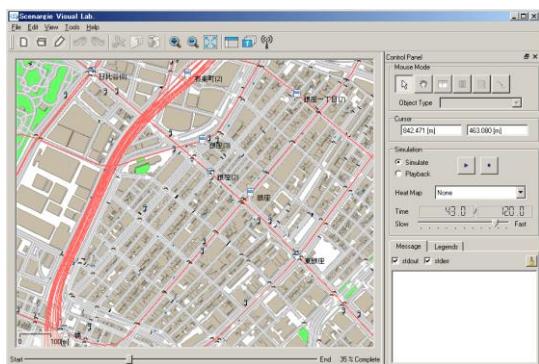


9) [Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]より行動の設定を行う。

- PreferedMobilityMeans の Value を PrivateCar に設定



10) シミュレーションを実行する。



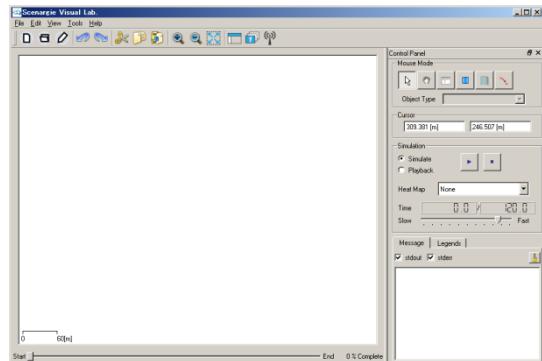
2.6. 通信機を利用したシナリオ

通信機を持ったエージェントを利用したシミュレーション実施するためのシナリオを作成する。

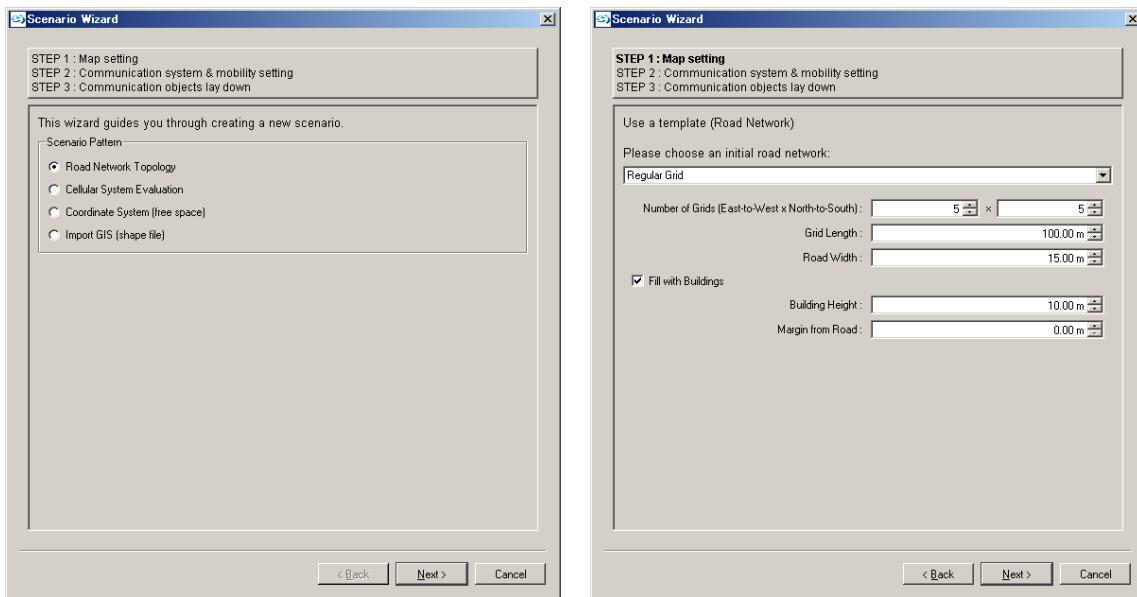
シナリオ概要

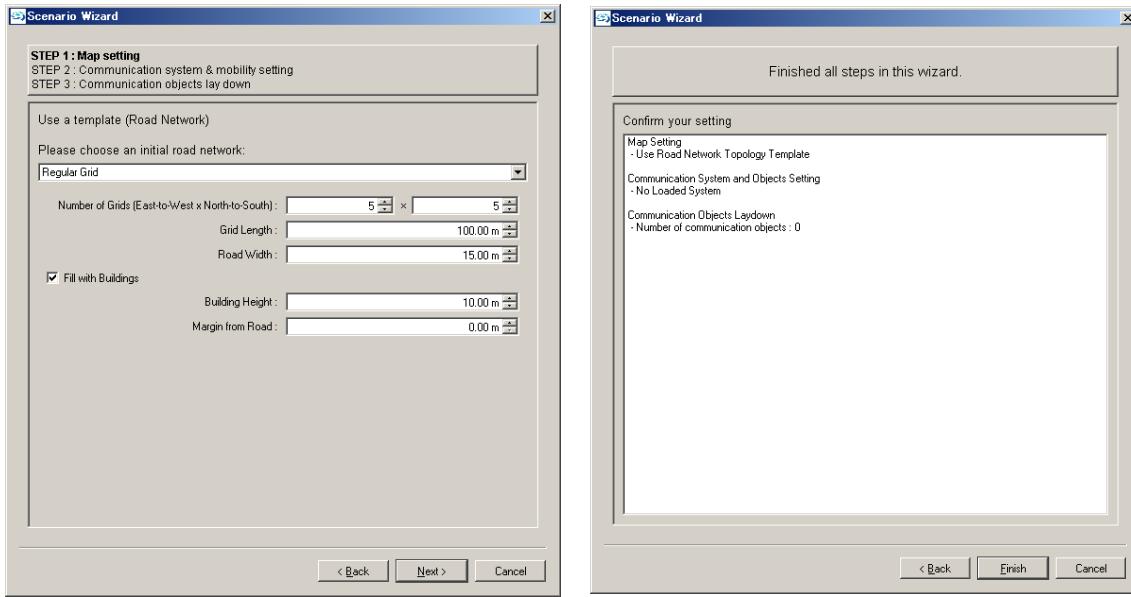
Dot11 の通信機を持ったエージェント:20 人
シミュレーション内容:20 人の通信エージェントの移動

- 1) [File]->[New]により新規作成する。

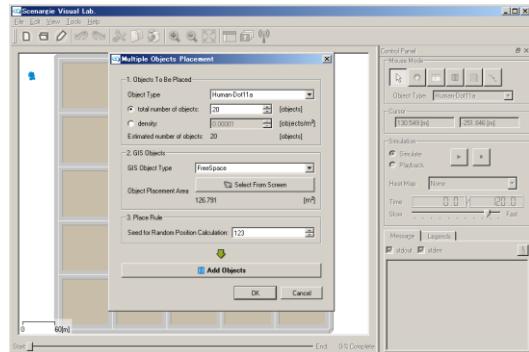


- 2) [File]->[Scenario Wizard]より、格子状の道路を 5x5 で設定する。Communication Object は配置しない。

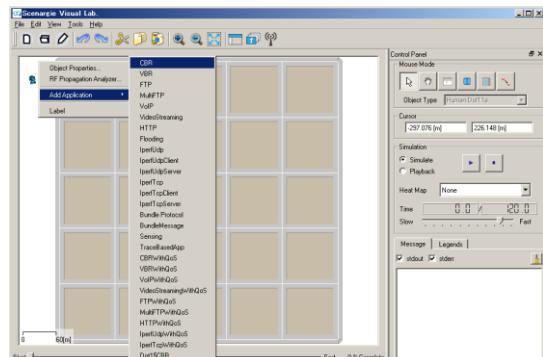




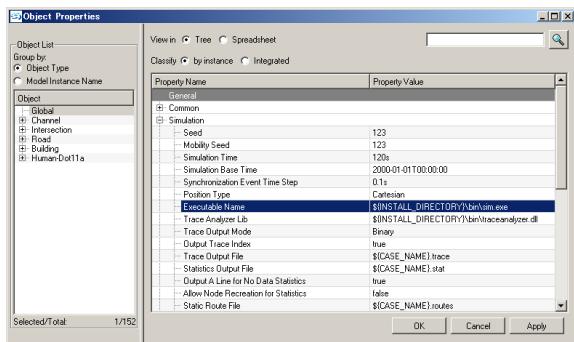
3) Human-Dot11a エージェントを 20 人配置する。



4) Human-Dot11a エージェントに CBR を追加する。

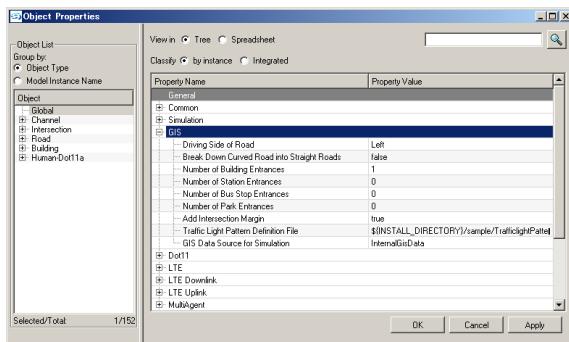


5) [Tools]->[ObjectProperties]の Executable File Name より Dot11+マルチエージェントシミュレーション用のシミュレーション実行イメージを指定する。

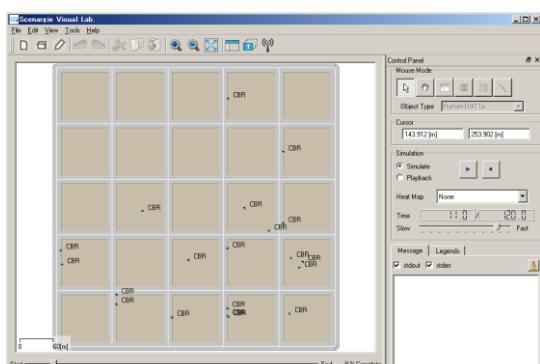


6) [Tools]->[ObjectProperties]より GIS グローバルの設定を行う。

パラメータ名	値
Break Down Curved Road into Straight Roads	false
Number of Building Entrances	1
Add Intersection Margin	true



7) シミュレーションを実行する。

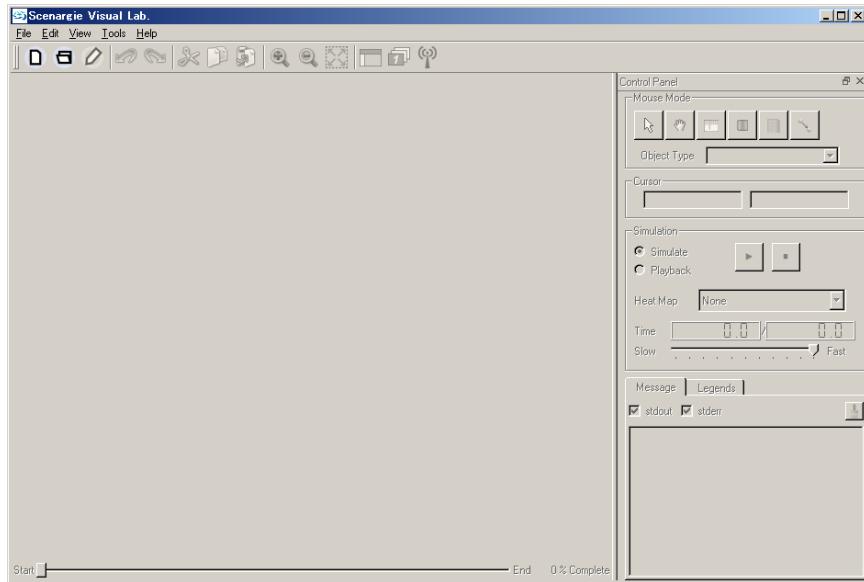


3. シナリオ作成

Scenragie Visual Lab からシナリオ作成し、シミュレーションを実行する方法を解説する。

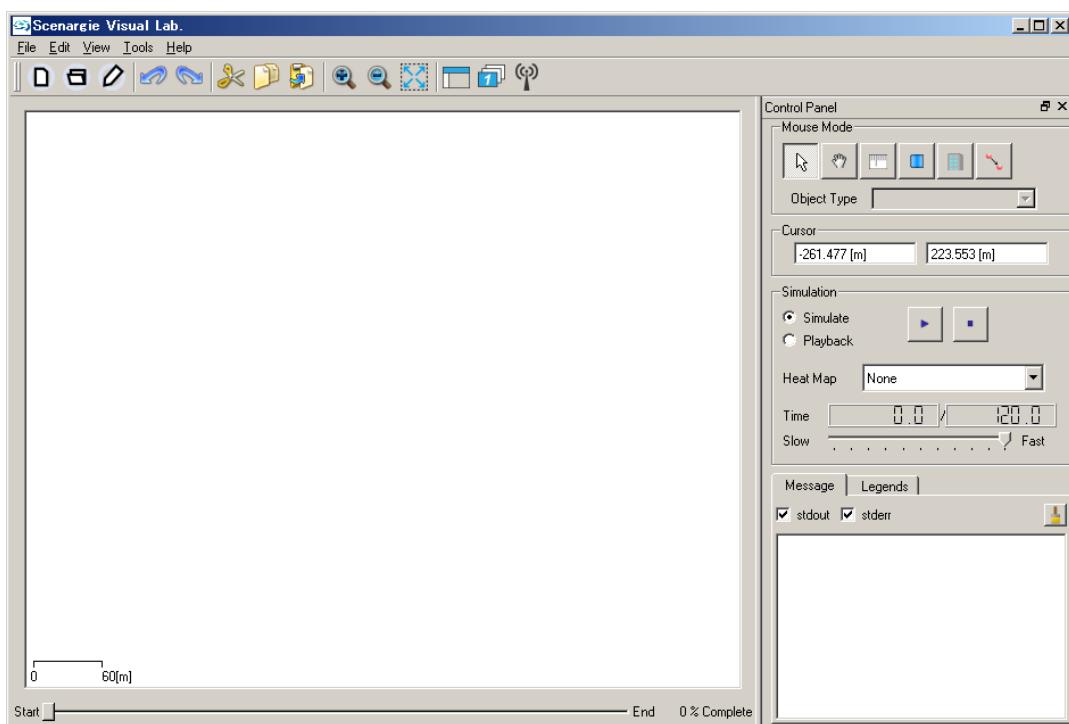
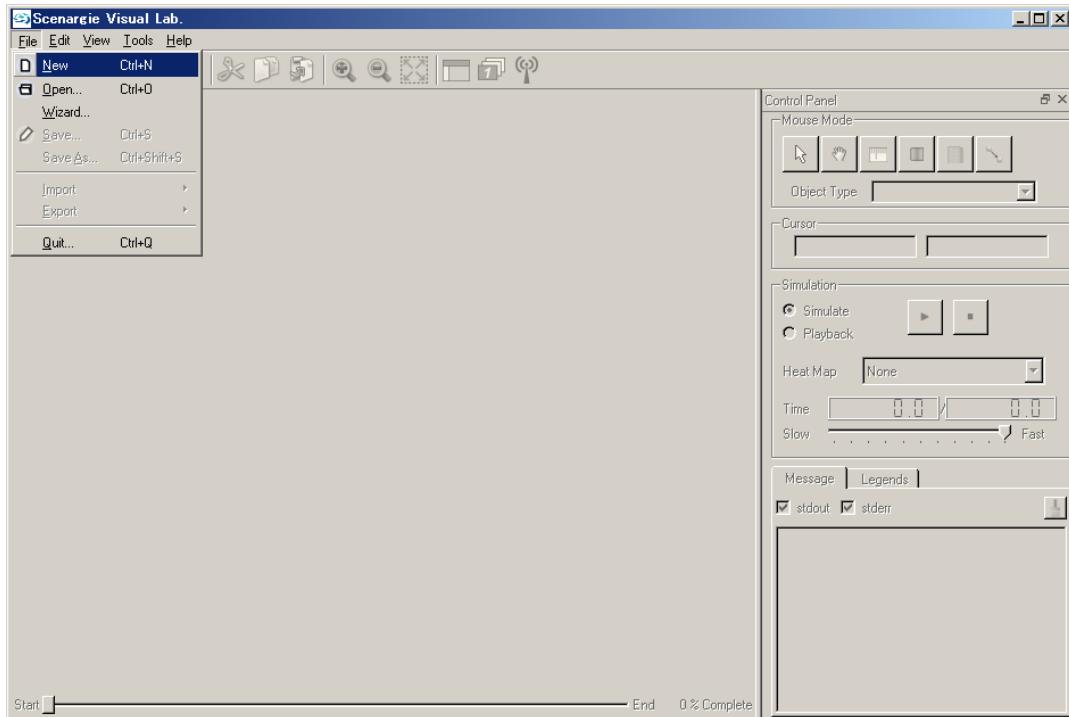
3.1. Visual Lab の起動

Visual Lab を起動する。



3.2. 新規シナリオ作成

[File]->[New]より新規にシナリオを作成する。

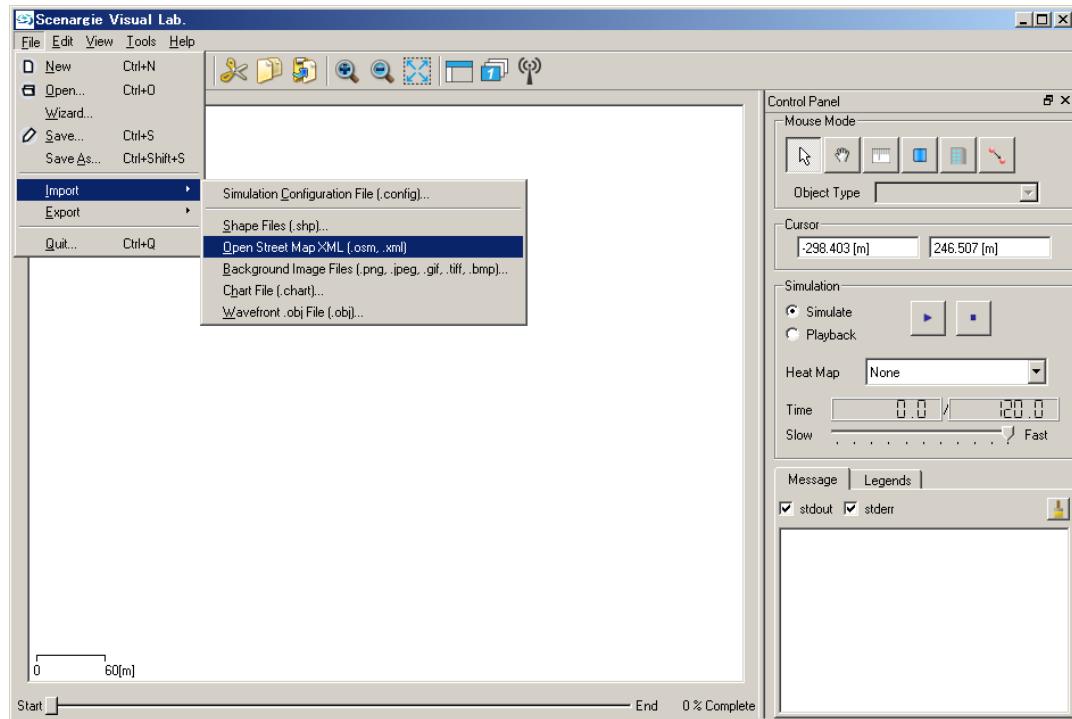


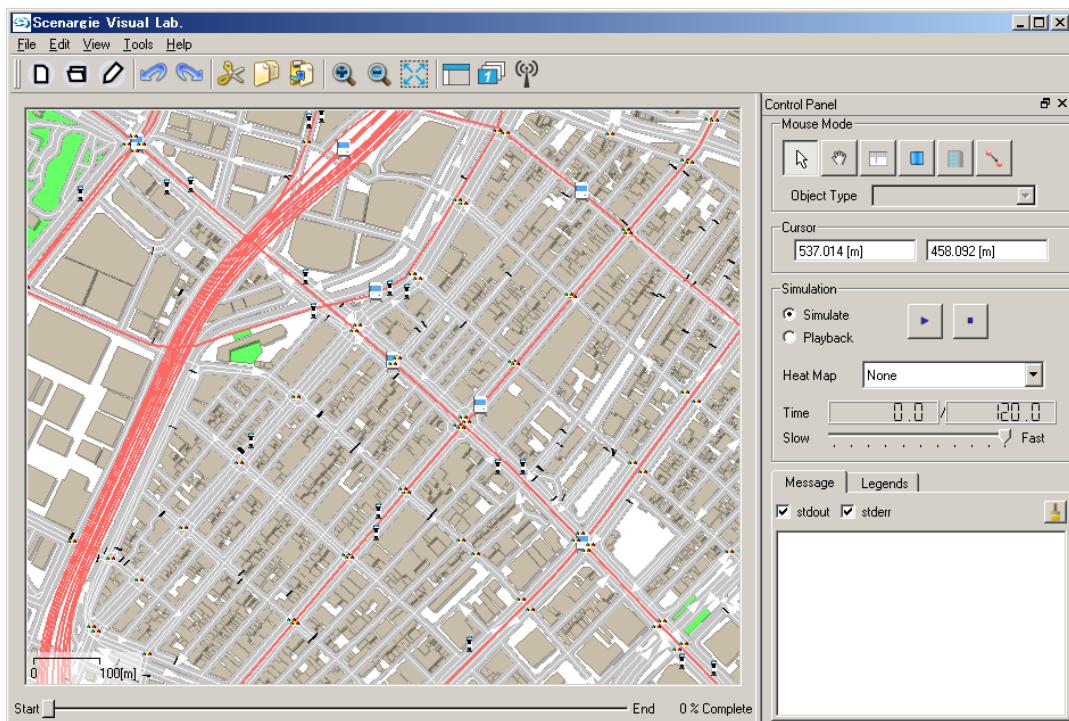
3.2.1.GIS 情報の作成

GIS 情報の作成には以下の三通りの方法がある。

- OSM ファイルのインポート
- シェープファイルのインポート
- マニュアルによる作成
- OSM ファイルのインポート

地図情報として OpenStreetMap の利用が可能である。OpenStreetMap の地図を利用する場合、[File] -> [New]で新規作成を行った後、メニューの[File] -> [Import] -> [Open Street Map XML]より OSM ファイルの指定を行う。

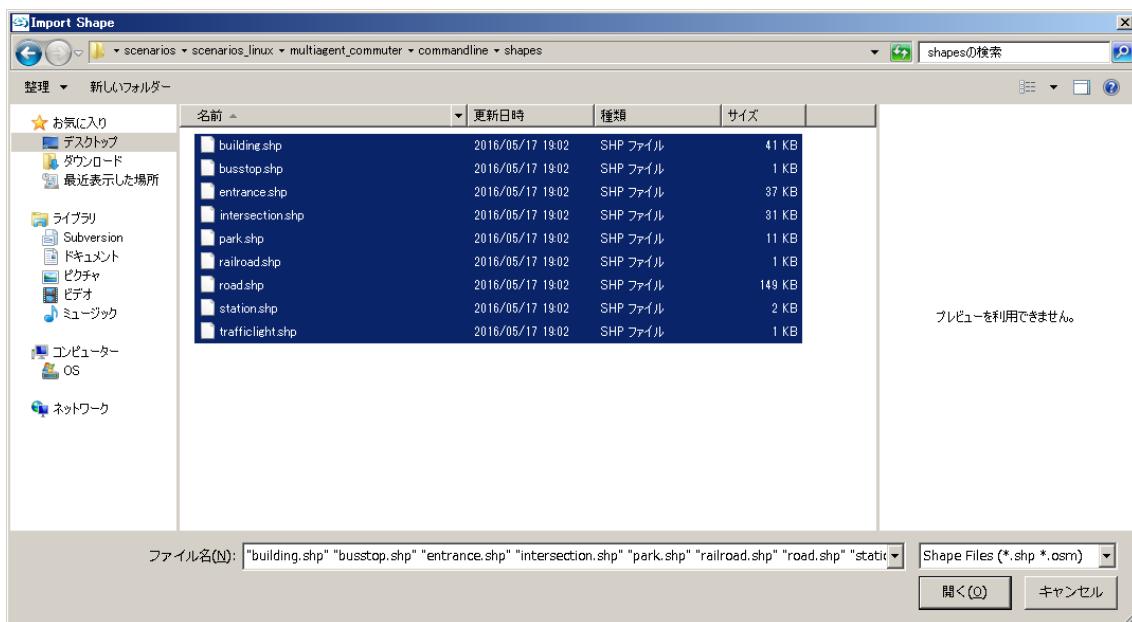
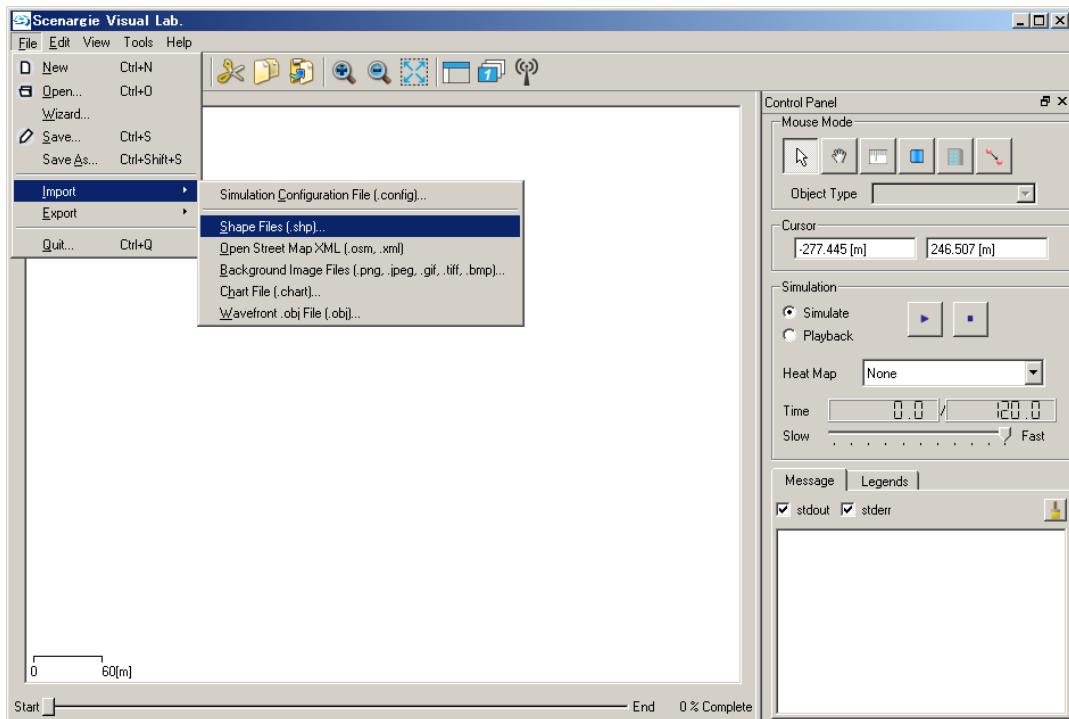




OSM ファイルのインポートを行った場合、エリアは OSM ファイル(XML 形式)の 3 行目に記述されるエリア情報を利用する。エリアは OSM ファイル内の GIS オブジェクトを全て含んだ領域となる。読み込みを行った GIS オブジェクトはデフォルトで Display モードとなる。

- シェープファイルのインポート

地図情報として国土地理院の地図等に利用されているシェープ形式(shp 形式)の地図を利用可能である。シェープファイルの地図を利用する場合、[File] -> [New]で新規作成を行った後、メニューの [File] -> [Import] -> [Open Street Map XML]より OSM ファイルの指定を行う。



シェープファイルのインポートでは一度に複数のファイルをインポートすることが可能である。この時、エリア情報は全てのシェープファイルに含まれるエリア情報を統合し、さらに中心点を(0,[m] 0[m])にオフセットしてインポート処理を行うため、複数のシェープファイルを利用するがあらかじめ分かれている場合は、それらのファイルを一度に指定して読み込むことを推奨する。

ファイルを指定した後、各ファイルをどのオブジェクトタイプで読み込むかの指定を行う。デフォルトではファイル名に含まれる文字列よりオブジェクトタイプの指定が行われる。マルチエージェントシミュレーションで利用されるオブジェクトは道路、交差点、信号、バス停、建物、公園、線路、駅、入り口、エリアである。

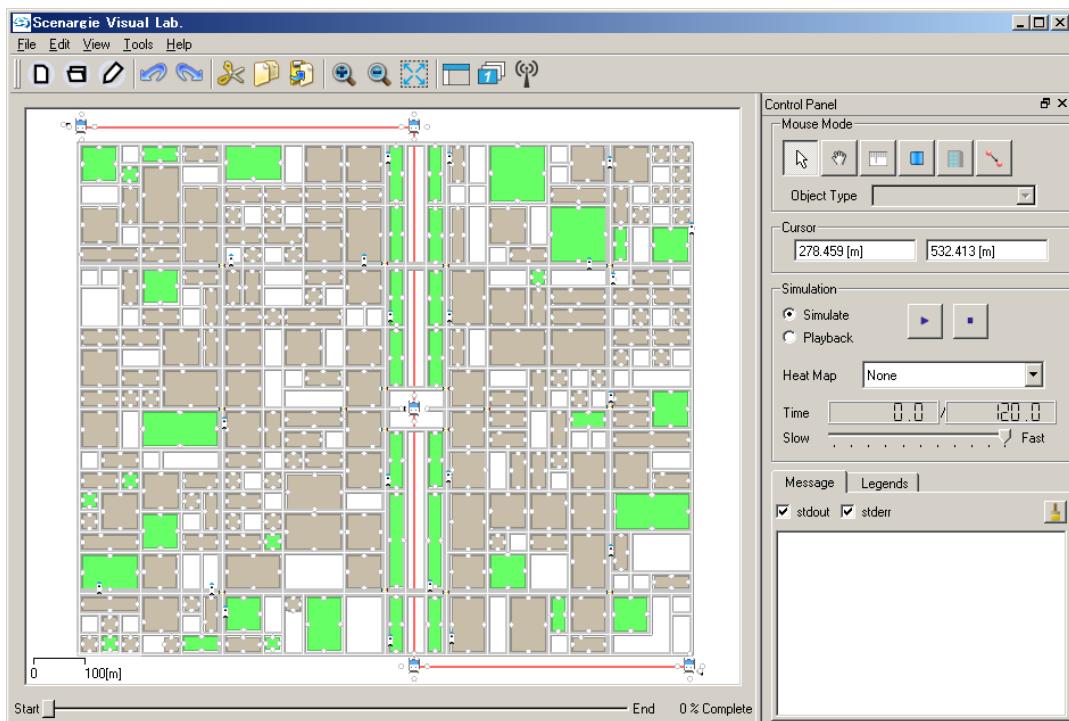
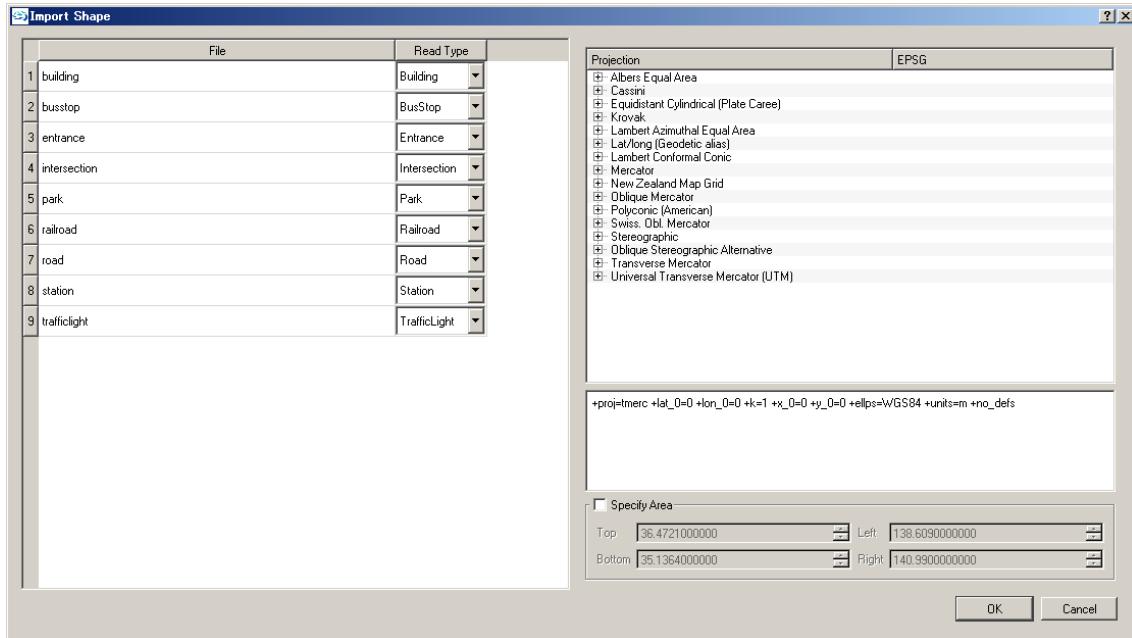
Read Type	オブジェクト	シェープタイプ	ファイル名からの推定
Road	道路	ARC/ARCZ	“road”
Intersection	交差点	POINT/POINTZ	“intersection”
TrafficLight	信号	POINT/POINTZ	“trafficlight”
BusStop	バス停	POINT/POINTZ	“busstop”
Building	建物	POLYGON/POLYGONZ	“building”
Park	公園	POLYGON/POLYGONZ	“park”
Railroad	線路	ARC/ARCZ	“rail”
Station	駅	POLYGON/POLYGONZ	“station”
Entrance	入り口	POINT/POINTZ	“entrance”
Area	エリア	POLYGON/POLYGONZ	“area”
Wall	壁	ARC/ARCZ	“wall”
Ground	地面	POINTZ	“ground”
Tree	樹木	POINTZ	“tree”
POI	POI (Point of Interest)	POINTZ	“poi”
Other	その他	POINT/POINTZ/ ARC/ARCZ/ POLYGON/POLYGONZ	

Other タイプでの読み込みを行ったオブジェクトについては、ファイル名を用いて新規のオブジェクトタイプを作成し、シミュレーションで利用しない表示のみのオブジェクトとしてインポートする。

シェープファイルと共に、プロジェクトを指定した”.prj”ファイルが存在する場合、シェープファイル指定時にプロジェクトの値も読み込まれる。プロジェクトファイルが存在しない場合、デフォルトでは次のプロジェクトが設定される。

```
+proj=tmerc +lat_0=0 +lon_0=0 +k=1 +x_0=0 +y_0=0 +ellps=WGS84 +units=m  
+no_defs
```

読み込み時のプロジェクトを変更する場合は、Projection のツリーから選択する。



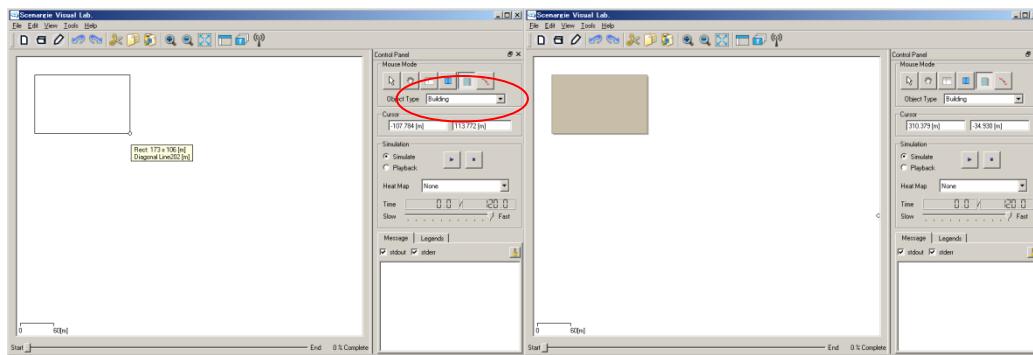
読み込みを行った GIS オブジェクトのデフォルト編集設定は Options の "Default GIS Object Edit Mode" に従う。

- マニュアルによる作成

マウス操作により GIS オブジェクトを配置可能である。マニュアル作成した GIS オブジェクトのデフォルトの編集設定は Options の”Default GIS Object Edit Mode”に従う。

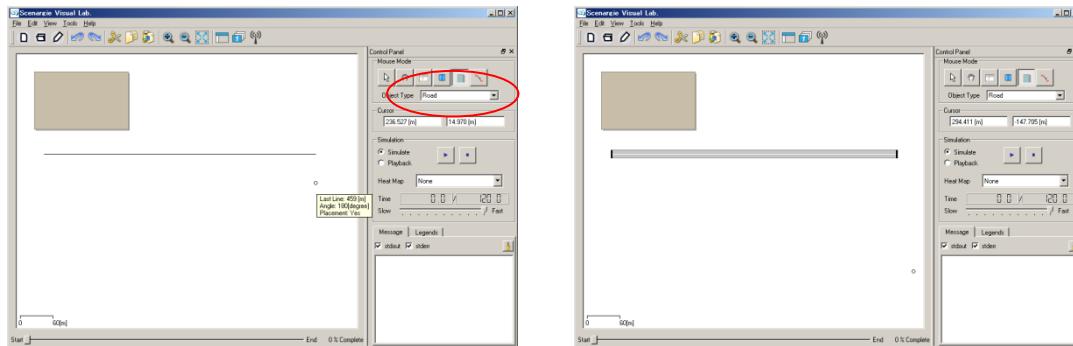
- 建物の作成

Object Type より Building を選択し、マウスドラッグまたはマウスクリック操作により建物ポリゴンを作成する。配置した建物はエージェントの初期位置、目的地として設定が可能である。



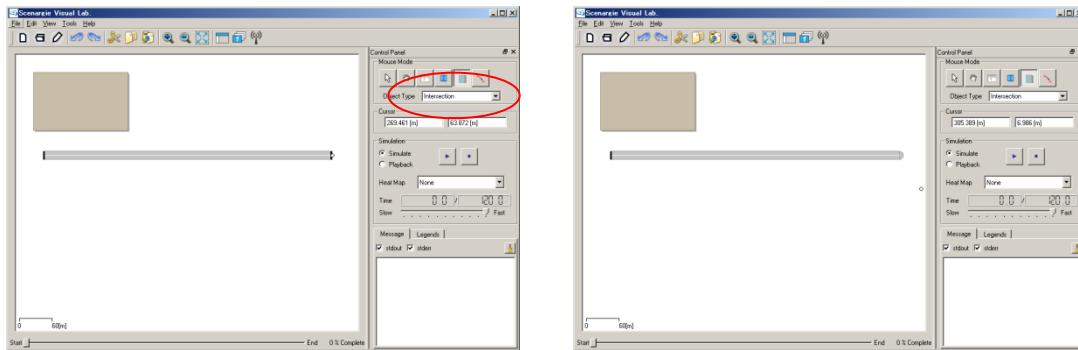
- 道路の作成

Object Type より Road を選択し、マウスクリック操作により道路線を作成する。エージェントは配置した道路上を移動可能である。



- 交差点の作成

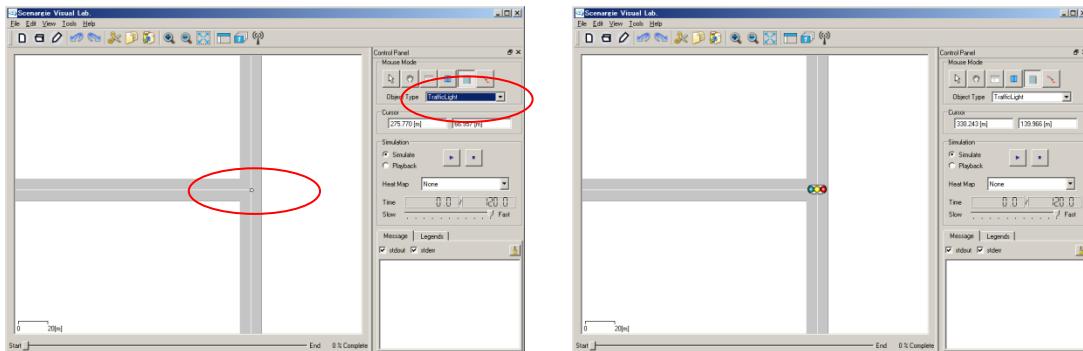
Object Type より Intersection を選択し、マウスクリック操作により交差点を作成する。交差点は道路の端点に作成可能である。交差点には信号が設定可能である。



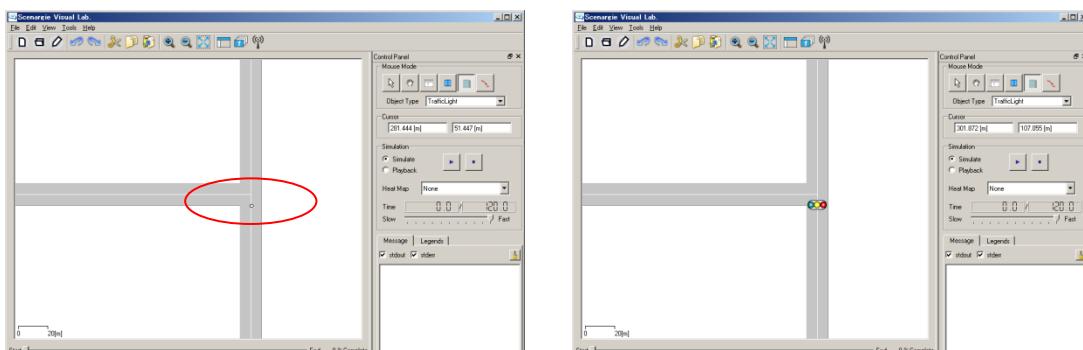
- 信号の作成

Object Type より TrafficLight を選択し、マウスクリック操作により信号を作成する。信号は交差点上または交差点と接続している道路端にのみ配置が可能である。交差点上に配置した場合は、信号はその交差点に接続している道路をグループ化し、グループ単位で順番に信号の切り替えを行う。道路端に配置した場合、その道路から交差点に進入する車線に対してのみ有効な信号となる。

交差点の中央に配置する場合

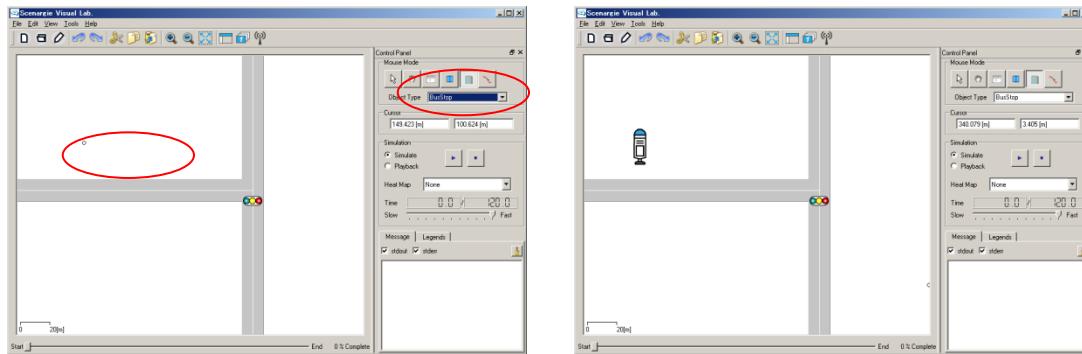


道路端に配置する場合



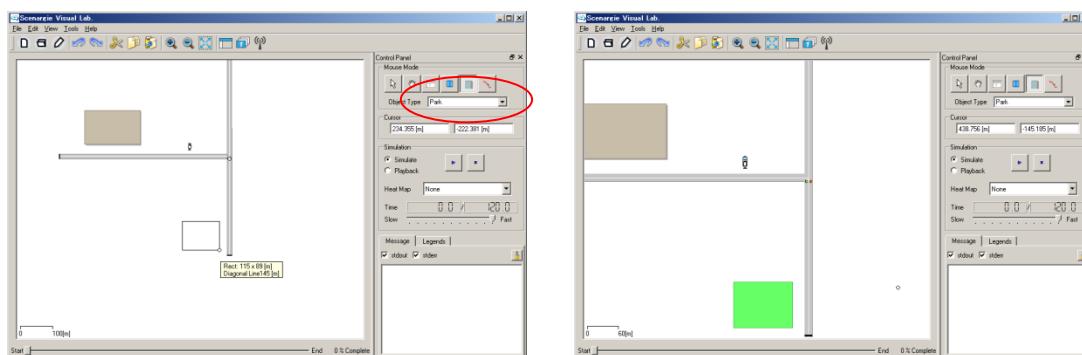
- バス停の作成

Object Type より BusStop を選択し、マウスクリック操作によりバス停を作成する。バス停はバスの停車する場所としてエージェントタイムテーブル設定ファイルで指定可能である。



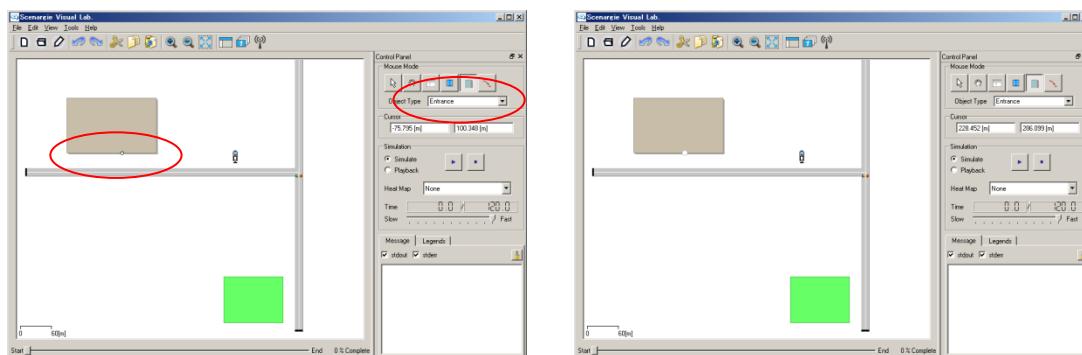
- 公園の作成

Object Type より Park を選択し、マウスドラッグまたはマウスクリック操作により公園を作成する。配置した公園はエージェントの初期位置、目的地として設定が可能である。



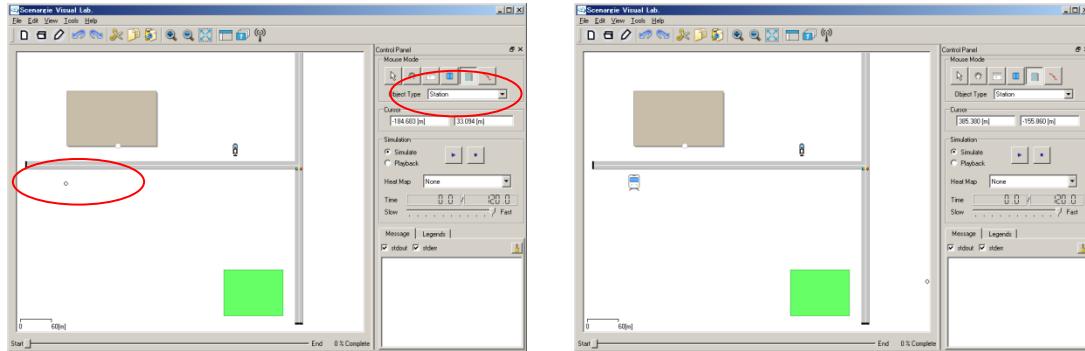
- 入り口の作成

Object Type より Entrance を選択し、マウスクリック操作により入り口を作成する。入り口は建物、公園、駅に対して設置可能である。エージェントは入り口からの出入りを行う。



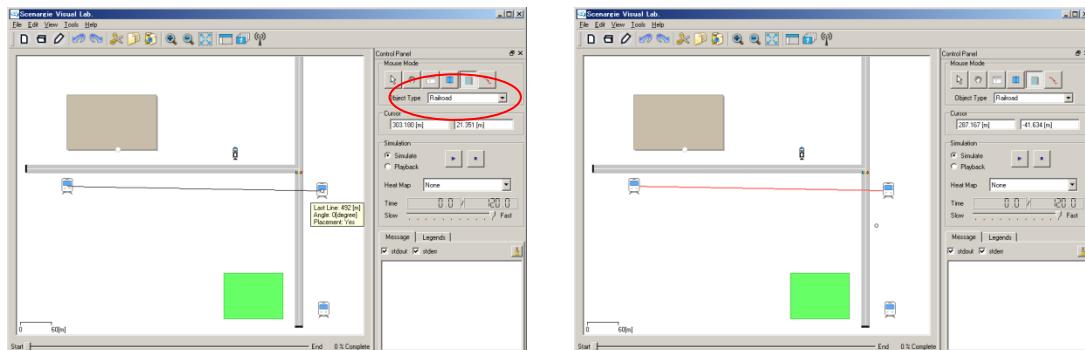
- 駅の作成

Object Type より Station を選択し、マウスクリック操作により駅を作成する。駅は電車の停車する場所としてエージェントタイムテーブル設定ファイルで指定可能である。



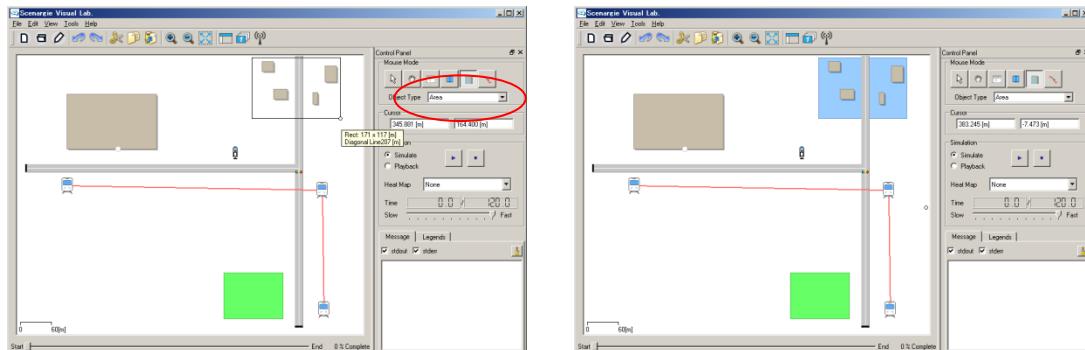
- 線路の作成

Object Type より Railroad を選択し、マウスクリック操作により線路を作成する。線路により駅同士での接続を設定可能である。



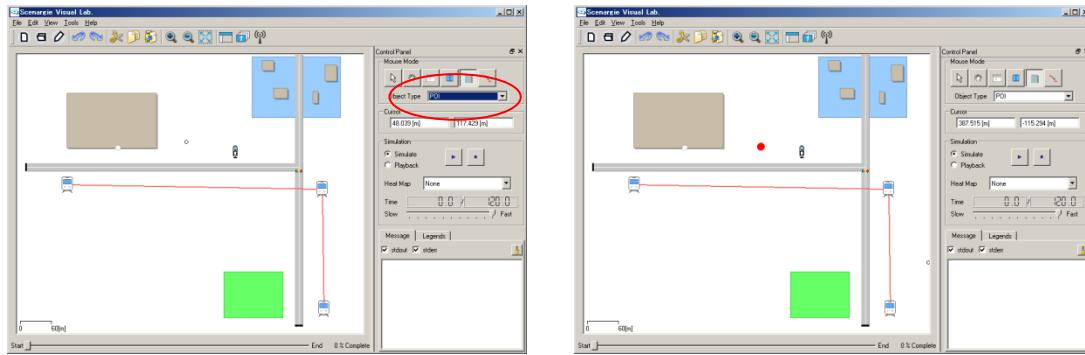
- エリアの作成

Object Type より Area を選択し、マウスドラッグまたはマウスクリック操作によりエリアを作成する。配置したエリアはエージェントの初期位置、目的地として設定が可能である。



- POI の作成

Object Type より POI を選択しマウスクリック操作により POI を作成する。配置した POI はエージェントの初期位置、目的地として設定が可能である。



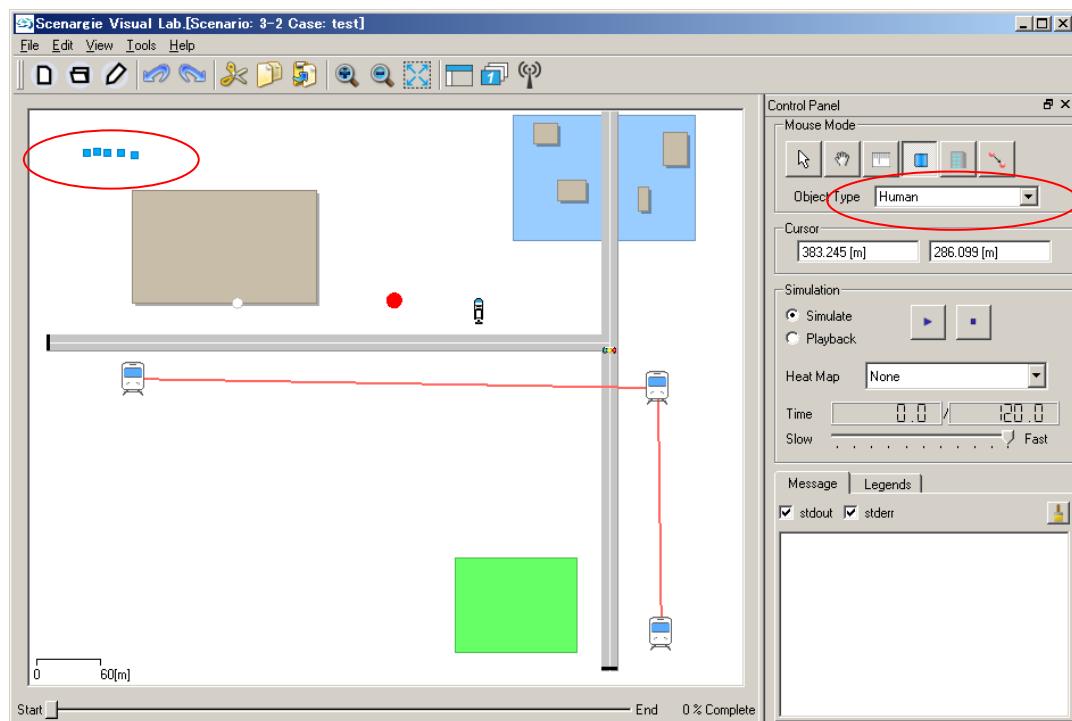
3.2.2. エージェントの配置

エージェントの配置は以下の二通りの方法がある。

- マニュアルによる配置
 - Multiple Objects Placement による配置
 - マニュアルによる作成
- マウス操作によりエージェントを配置可能である。

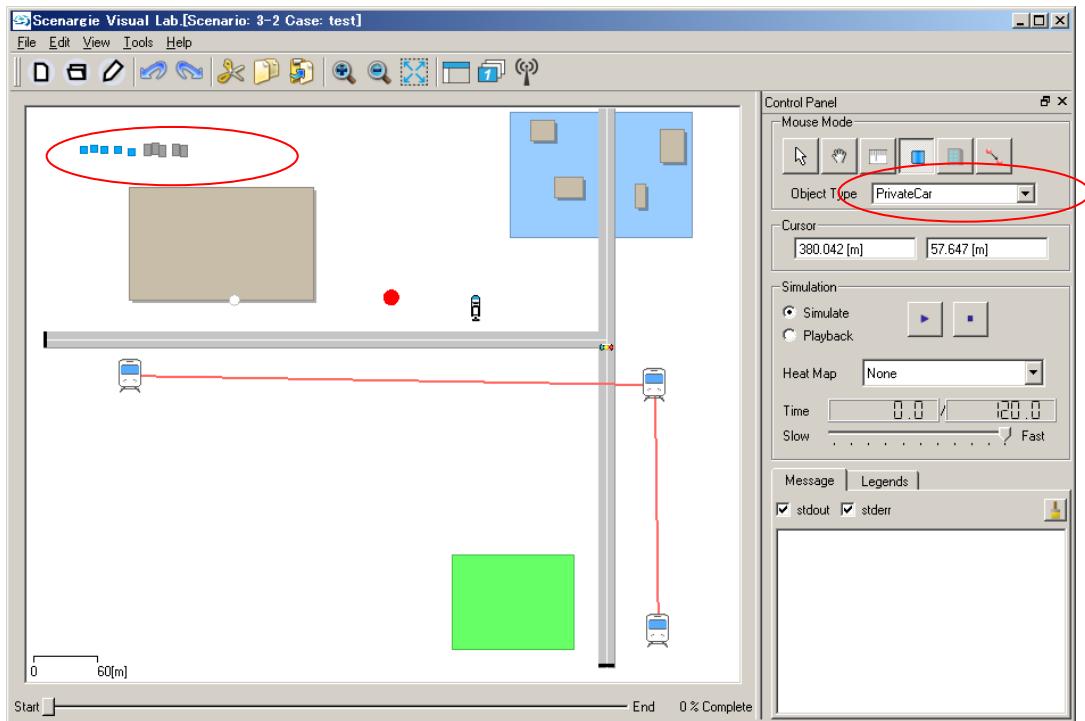
- Human

Object Type より Human を選択し、マウスクリック操作により Human エージェントを配置する。Human の初期位置は Behavior で指定した "InitialLocation" となるため、マニュアルで配置した場所とシミュレーションの開始時の初期位置は異なる。



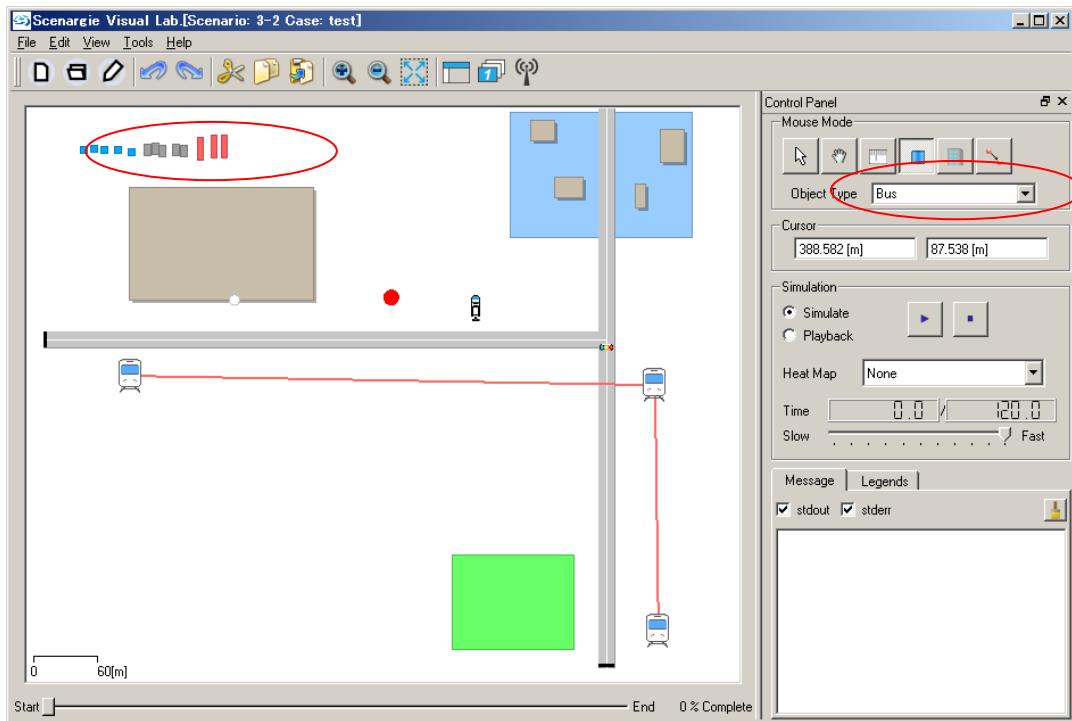
- PrivateCar

Object Type より PrivateCar を選択し、マウスクリック操作により PrivateCar エージェントを配置する。PrivateCar は車両を保持しているエージェントに対して割り当てられ、そのエージェントの初期位置(建物または公園)の入り口に配置される。マニュアルで配置した場所とシミュレーションの開始時の初期位置は異なる。



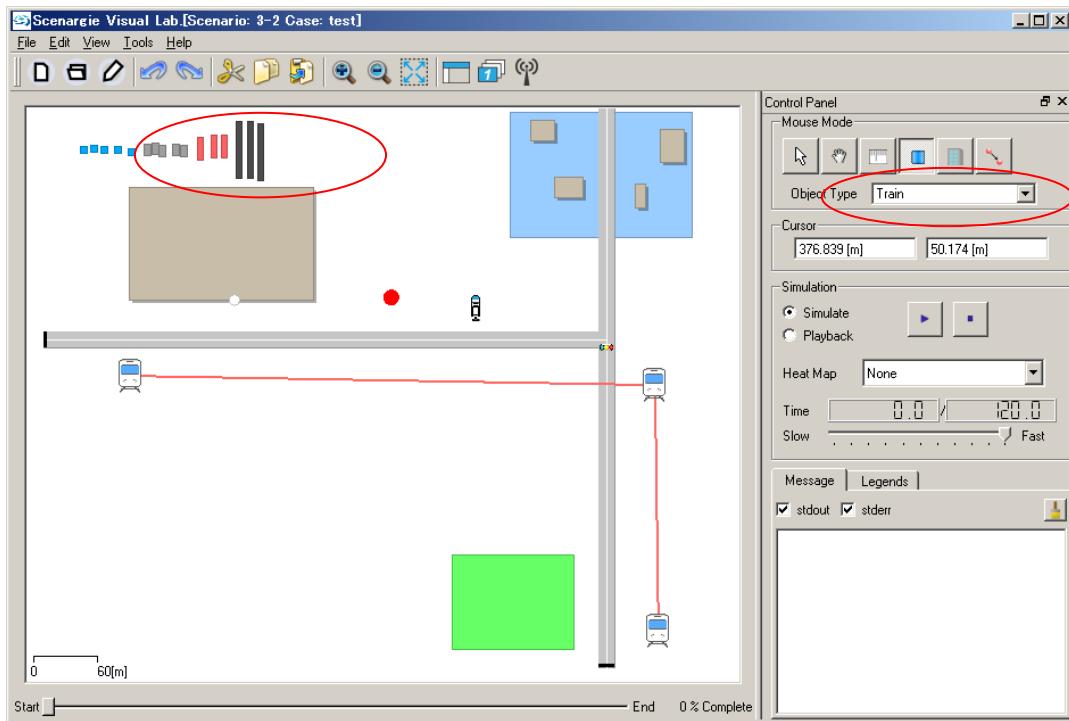
- Bus

Object Type より Bus を選択し、マウスクリック操作により Bus エージェントを配置する。バスはエージェントタイムテーブル設定ファイルで指定された運行スケジュールに従って移動を行う。スケジュールに従ったバス停から登場し終点のバス停で消滅するため、マニュアルで配置した場所とシミュレーションの開始時の初期位置は異なる。また、エージェントタイムテーブル設定ファイルで指定された数よりもバスの台数が少ない場合、エラーとなる。



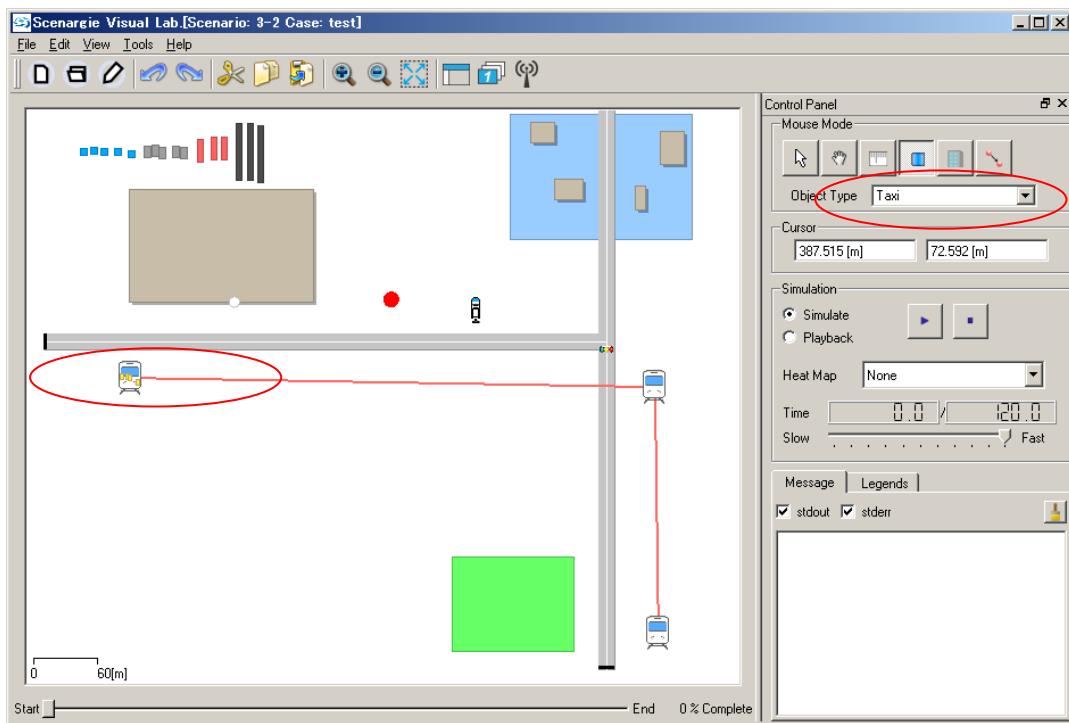
- Train

Object Type より Train を選択し、マウスクリック操作により Train エージェントを配置する。電車はエージェントタイムテーブル設定ファイルで指定された運行スケジュールに従って移動を行う。スケジュールに従った駅から登場し終点の駅で消滅するため、マニュアルで配置した場所とシミュレーションの開始時の初期位置は異なる。また、エージェントタイムテーブル設定ファイルで指定された数よりも電車の台数が少ない場合、エラーとなる。



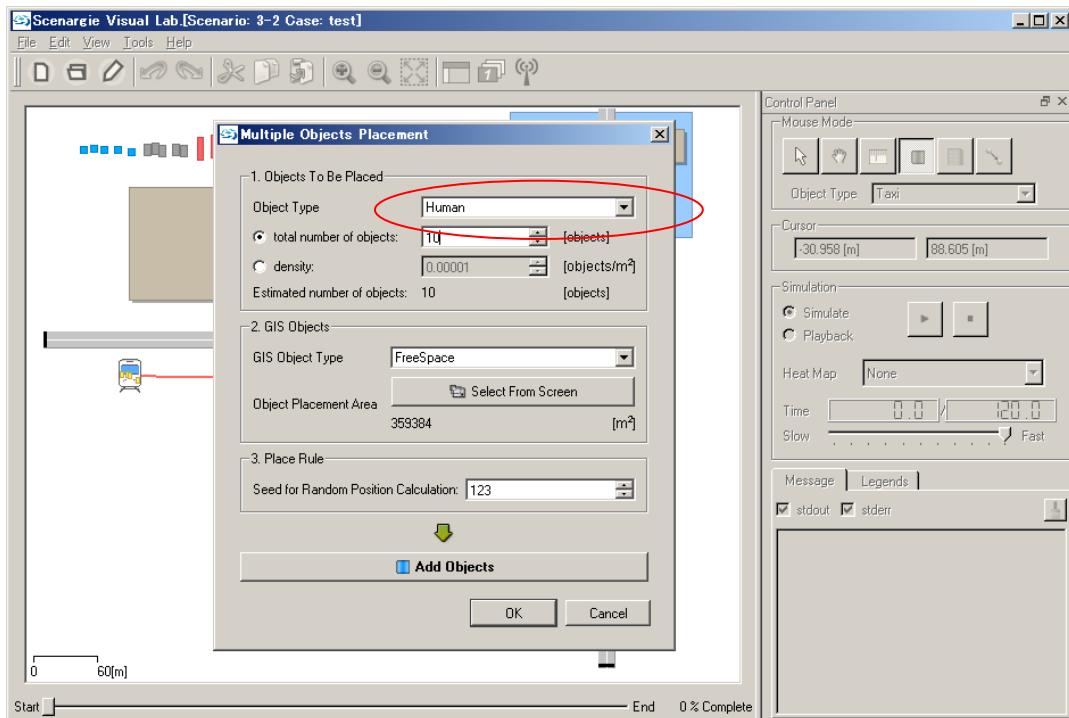
- Taxi

Object Type より Taxi を選択し、マウスクリック操作により Taxi エージェントを配置する。タクシーは、駅、バス停、公園、建物、POI のいずれかのオブジェクト上に配置する。配置したタクシーはその GIS オブジェクト上から発生する。



- Multiple Objects Placementによる配置

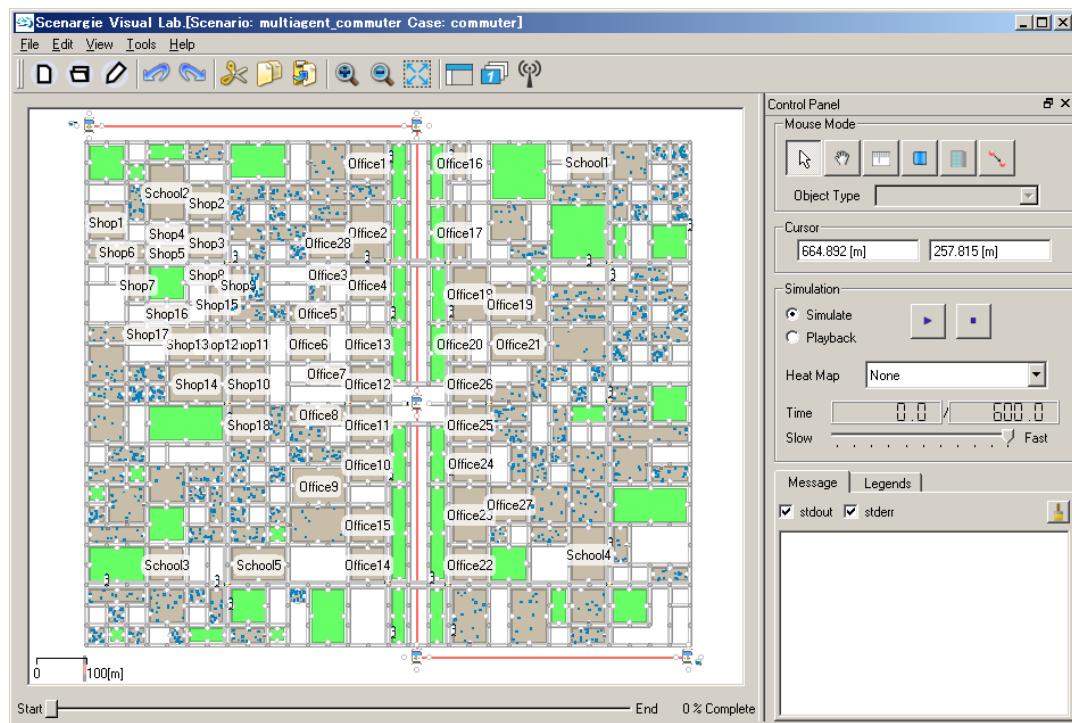
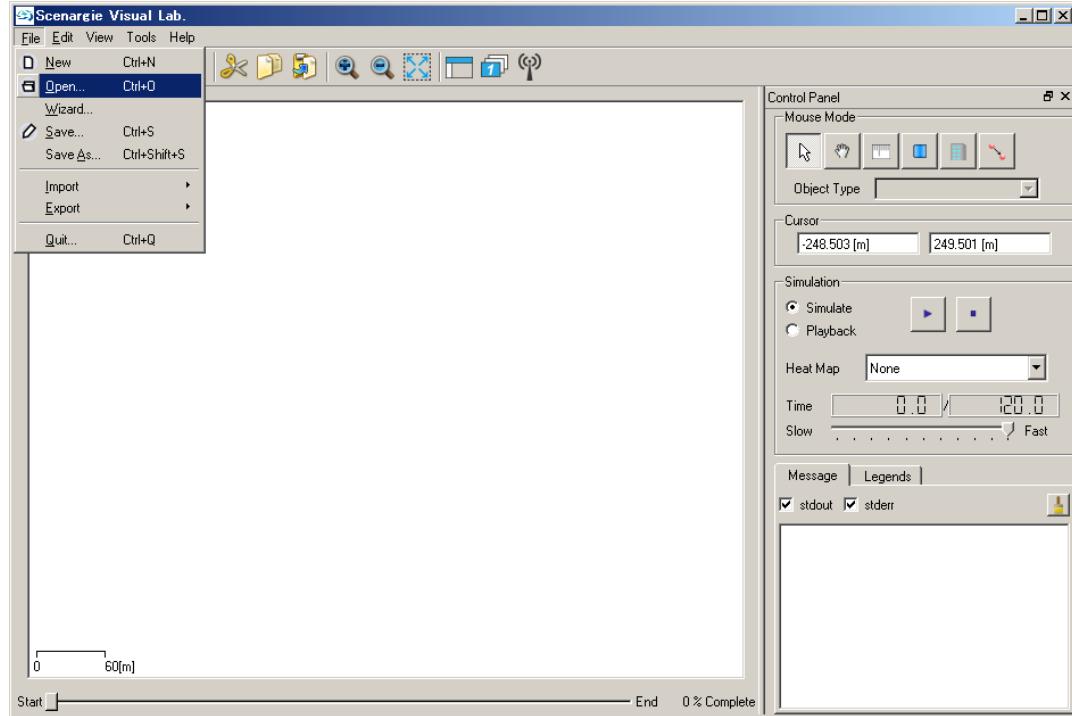
メニューの[Tools] -> [Multiple Objects Placement]より複数のエージェントを一度に配置が可能である。Objects To Be Placed の Object Type より Human、PrivateCar、Taxi、Bus、Train を選択可能である。



3.3. 既存シナリオの読み込み

3.3.1. ケースファイル

メニューの[File]->[Open]より、既存のシナリオ(“.case”ファイル)を読み込み可能である。

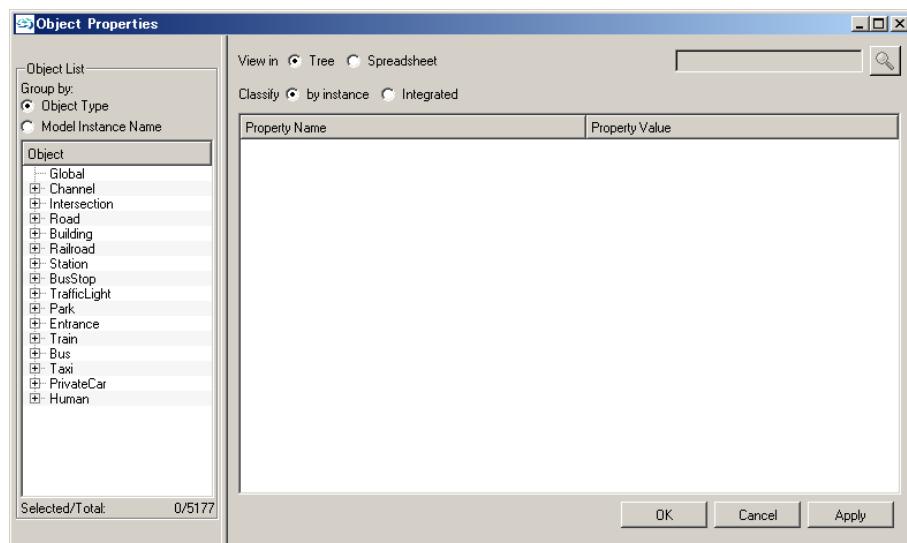
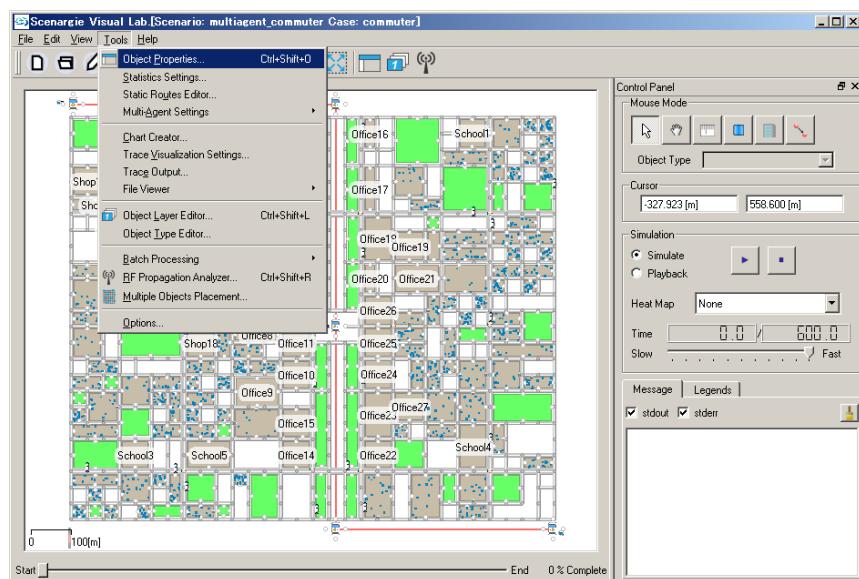


3.3.2. シミュレーション設定ファイル”.config”的インポート

Visual Lab の[メニュー] -> [File] -> [Import] -> [Simulation Configuration File]でシミュレーション用のファイルをインポート可能である。

3.4. シナリオのプロパティ設定

メニューの[Tools] -> [Object Properties]により設定情報を確認可能である。



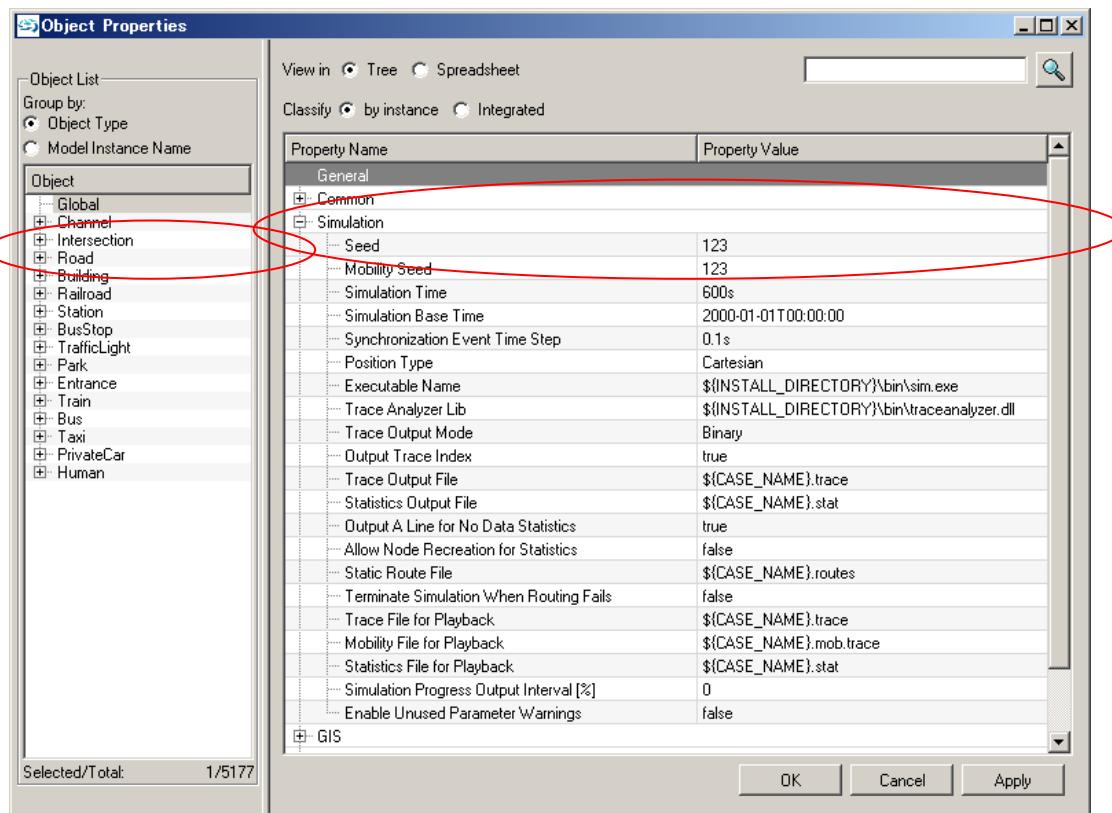
3.5. グローバル設定

グローバル設定では以下の項目を設定する。

- シミュレーショングローバル設定
- マルチエージェントグローバル設定
- GIS グローバル設定

3.5.1. シミュレーショングローバル設定

シミュレーショングローバル設定では、シミュレーション全般のプロパティ設定を行う。



プロパティ	プロパティ名	シミュレーションプロパティ名	値
モビリティ乱数 シード	Mobility Seed	mobility-seed	整数
シミュレーション時間	Simulation Time	simulation-time	時間
基準時刻	Simulation Base Time	simulation-base-time	時間
同期間隔	Synchronization Event Time Step	time-step-event-synchronization-step	時間

位置タイプ	Position Type	sim-postype	
トレース出力モード	Trace Output Model	trace-binary-output	Binary/ Text
トレースインデックス	Output Trace Index	trace-index-output	boolean
トレース出力ファイル	Trace Output File	trace-output-file	ファイル名
統計情報出力ファイル	Statistics Output File	statistics-output-file	ファイル名
空の統計情報出力	Output A Line for No Data Statistics	statistics-output-for-no-data	boolean
IP 経路情報ファイル	Static Route File	network-static-route-file	ファイル名
ルーティング失敗時のシミュレーション終了処理	Terminate Simulation When Routing Fails	network-terminate-sim-when-routing-fails	boolean
シミュレーション実行ファイル名	Executable Name	executable-name	ファイル名
プレイバックトレースファイル名	Trace File for Playback	trace-file-for-playback	ファイル名
移動プレイバックトレースファイル名	Mobility File for Playback	mobility-file-for-playback	ファイル名
統計プレイバックファイル名	Statistics File for Playback	statistics-file-for-playback	ファイル名
シミュレーション経過出力間隔	Simulation Progress Output Interval [%]	progress-sim-time-output-interval-percents	実数
トレースアナライザライブラリ	Trace Analyzer Lib	trace-analyzer-lib	ファイル名
未使用プロパティに対する警告出力	Enable Unused Parameter Warnings	enable-unused-parameter-warnings	boolean

1) 亂数シード

乱数シードは乱数生成器のシードとして利用する。同一のシードを利用する限り、シミュレーション実行時には必ず同じ順序で乱数が生成されるため、シミュレーション結果も同一となる。

2) シミュレーション時間

シミュレーション時間を設定する。エージェントタイムテーブル設定ファイルにおいてシミュレーション時間以降のスケジュールはスキップされる。スキップされた車両を利用する経路はエージェントの経路計算では考慮されない。

エージェントタイムテーブル設定ファイルはシミュレーション時間の 0 秒を基準とする。

3) 同期間隔

エージェントの行動、GIS 情報の同期間隔を設定する。エージェントは同期間隔で指定された間隔で互いの状態を認識し、その情報を元に移動を行うため、エージェントの移動を細かくシミュレーションする場合は、同期間隔を 0.1 秒等の小さめの値に設定する。また、この間隔はシミュレーションの実行速度に影響を与えるため、シミュレーションを早く実行する場合や、行動粒度を大きくしても問題ないシミュレーションを実施する場合は数秒単位程度の値としても良い。

同期間隔を変更した場合は、エージェントの周囲の状況の認識粒度が変化するため、エージェントの行動も変化する。

4) バイナリトレース

バイナリトレースを設定する。Visual Lab のプレイバックではバイナリトレースを利用するため、シミュレーションのプレイバックを行う場合には `true` に設定する。トレース情報をテキストで確認したい場合は `false` に設定する。

5) トレース出力ファイル

トレース情報の出力先を設定する。トレース出力ファイルに出力される情報は各エージェントまたは GIS オブジェクトの Trace Tags 設定で指定された情報(7.2.3 トレース設定)となる。

6) 統計情報出力ファイル

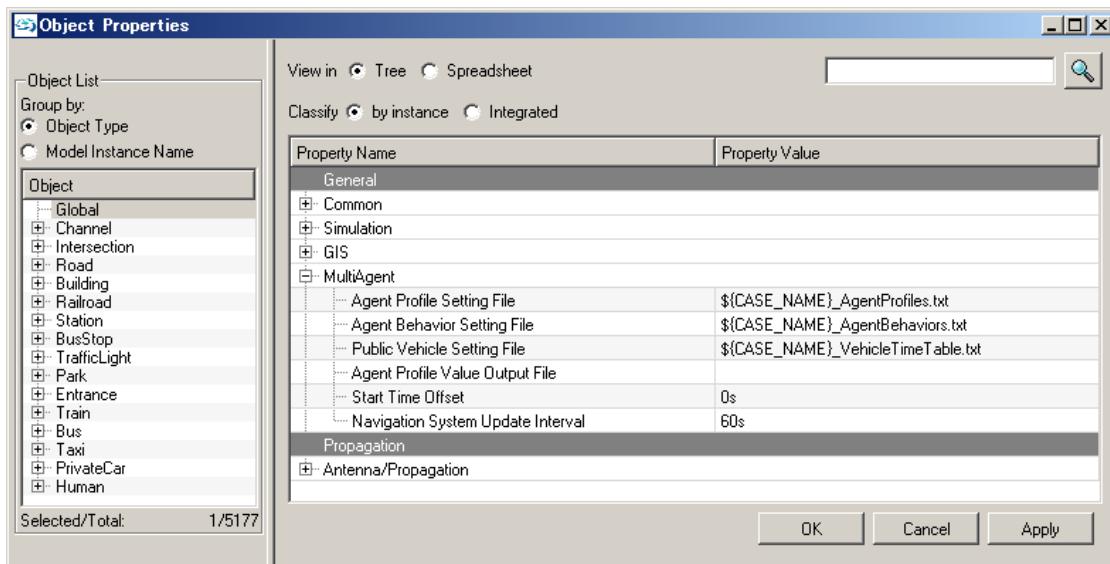
統計情報の出力先を設定する。統計情報出力ファイルに出力される情報は各エージェントまたは GIS オブジェクトについて Statistics Settings 設定で指定された情報(7.2.2 統計値設定)となる。

7) シミュレーション実行ファイル

シミュレーションを実行する実行イメージを設定する。マルチエージェントシミュレーション用のシミュレーション実行イメージを指定していない場合エラーとなる。

3.5.2. マルチエージェントグローバル設定

マルチエージェントグローバル設定では、マルチエージェント全般のプロパティ設定を行う。



プロパティ	プロパティ名	シミュレーションプロパティ名	値
エージェントプロファイル定義ファイル	Agent Profile Setting File	multiagent-profile-file	ファイルパス
エージェント行動定義ファイル	Agent Behavior Setting File	multiagent-behavior-file	ファイルパス
エージェントタイムテーブル設定ファイル	Public Vehicle Setting File	gis-public-vehicle-file	ファイルパス
プロファイル値出力ファイル	Agent Profile Value Output File	multiagent-profile-value-output-file	ファイルパス
エージェントタイムテーブル設定オフセット	Start Time Offset	multiagent-start-time	時間
マルチエージェントシミュレーションスレッド数	Number Threads	number-data-parallel-threads-for-multiagent	整数(1以上)
GIS 統計情報	Navigation System Update	multiagent-navigation-system-update	時間

更新間隔	Interval	-interval	
------	----------	-----------	--

1) エージェントプロファイル定義ファイル

エージェントプロファイルを定義したファイルを設定する。各エージェントのプロファイル設定はこのファイルに定義されたプロファイルタイプから選択可能とする。[Tools]->[Multi-Agent Settings]->[Agent Profile Editor]を利用してプロファイル定義を変更する場合、エージェントプロファイル定義ファイルで指定されたファイルに対して読み書きを行う。

2) エージェント行動定義ファイル

エージェント行動を定義したファイルを設定する。各エージェントの行動設定はこのファイルに定義された行動タイプから選択可能とする。[Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]を利用して行動定義を変更する場合、エージェント行動定義ファイルで指定されたファイルに対して読み書きを行う。

3) エージェントタイムテーブル設定ファイル

エージェントタイムテーブル設定を定義したファイルを設定する。[Tools]->[Multi-Agent Settings]->[Vehicle Time Table Editor]を利用して交通機関つけじゅうる定義を変更する場合、エージェントタイムテーブル設定ファイルで指定されたファイルに対して読み書きを行う。

4) エージェントタイムテーブル設定オフセット

エージェントタイムテーブル設定の開始時刻をオフセットして開始する場合に設定する。例えばオフセットを 60[秒]と設定した場合、全てのエージェントタイムテーブル設定が 60 秒早く実行される。オフセットにより、負の時刻から運転を開始することになったスケジュールはシミュレーションから除外する。

5) マルチエージェントシミュレーションスレッド数

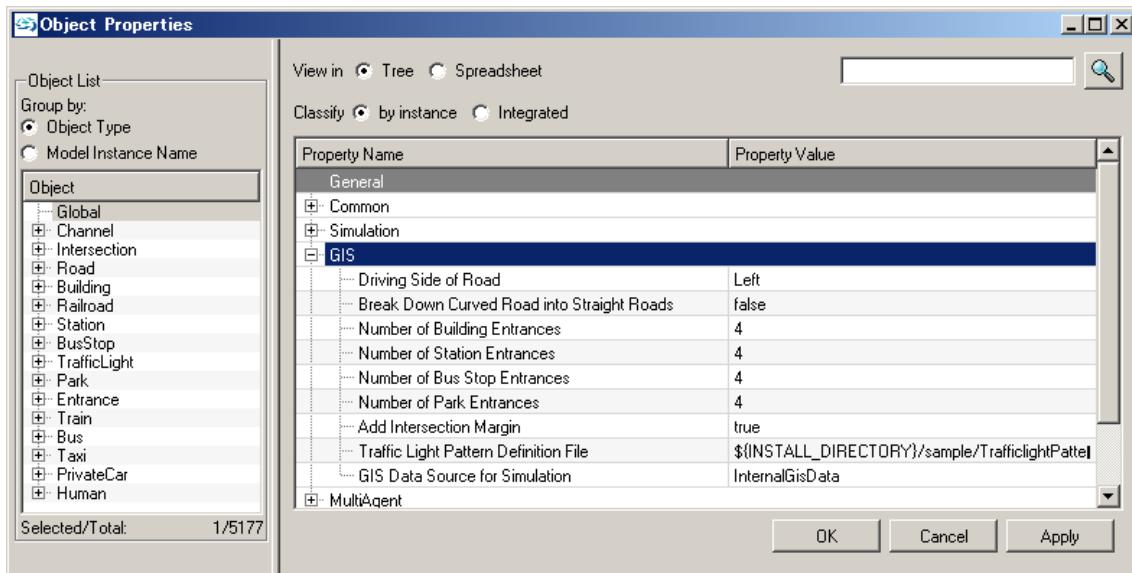
マルチエージェントシミュレーションにより利用する最大スレッド数を設定する。本プロパティによる複数スレッドを利用した計算はマルチエージェントシミュレーション(エージェントの行動シミュレーション)にのみ適用される。(マルチエージェントシミュレーションと同時に通信のシミュレーションも実行している場合、通信側のシミュレーションには適用されない。)

6) GIS 統計情報更新間隔

経路検索に利用する、道路上の統計情報(道路上の歩行者数と車両数)の情報更新間隔を設定する。

3.5.3.GIS グローバル設定

GIS グローバル設定では、GIS 全般のプロパティ設定を行う。



プロパティ	プロパティ名	シミュレーションプロパティ名	値
走行車線	Driving Side of Road	gis-road-driving-side	left/right
道路の分割	Break Down Curved Road into Straight Roads	gis-los-break-down-cureved-road-into-straight-roads	false
建物への最小の入り口数	Number of Building Entrances	gis-number-entrances-to-building	整数
駅への最小の入り口数	Number of Station Entrances	gis-number-entrances-to-station	整数
バス停への最小の入り口数	Number of Bus Stop Entrances	gis-number-entrances-to-busstop	整数
公園への最小の入り口数	Number of Park Entrances	gis-number-entrances-to-park	整数
交差点領域の作成	Add Intersection Margin	gis-road-set-intersection-margin	true
信号パターン定義ファイル	Trafficlight Pattern Definition File	gis-trafficlight-pattern-definition-file	ファイル名

1) 走行車線

車両(Taxi、PrivateCar、Bus)の走行車線を設定する。

2) 道路の分割

道路を頂点毎に分割し、分割した点に交差点を作成するかの設定を行う。マルチエージェントシミュレーションの場合は false に設定する。

3) 建物/駅/バス停/公園への最小の入り口数

建物/駅/バス停/公園へに対しての最小入り口数を設定する。建物/駅/バス停/公園に配置済みの入り口数が最小入り口数を満たしていないければ自動で入り口を作成する。入り口を設定していない建物/駅/バス停/公園には進入することが出来ない。また、シミュレーション途中で入り口への経路が無くなってしまった場合も進入不可となる。

4) 交差点領域の作成

道路に対して交差点領域を作成するかの設定を行う(7.7.1 道路ファイル)。マルチエージェントシミュレーションの場合は true に設定する。

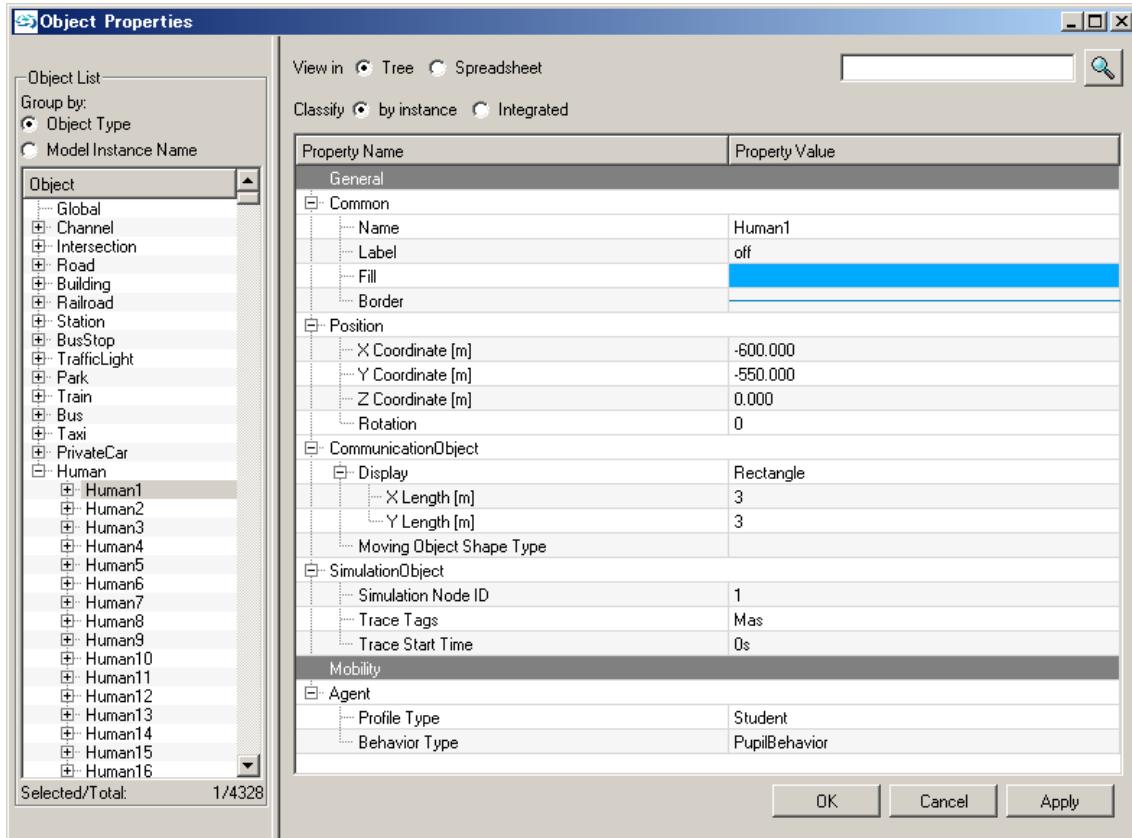
5) 信号パタン定義ファイル

信号パタン定義ファイルを指定する。信号パタン定義ファイル内に指定されている信号パタンは、各信号オブジェクトの制御パタン(Switching Pattern Type)を"Predefined"に指定した際に利用可能である。

3.6. エージェント設定

エージェント設定では以下の項目を設定する。

- プロファイル設定
- 行動設定



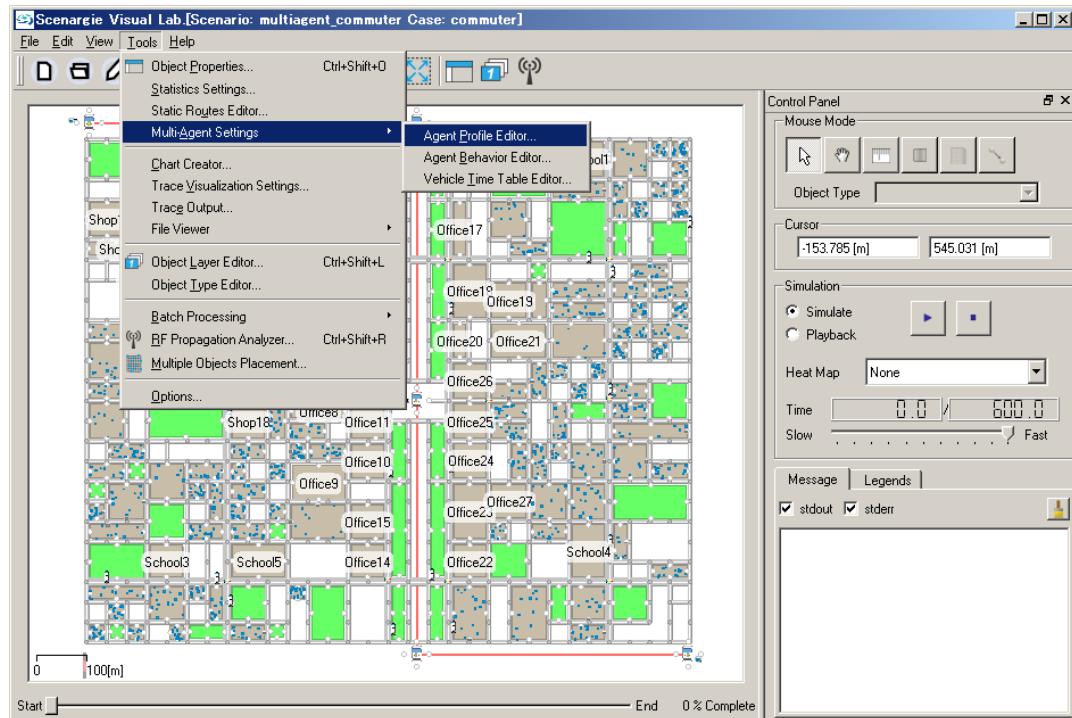
プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	無し
ラベル ON/OFF	Label	on/off	無し
色	Color	色	無し
X 座標	Position X Meters	実数	無し
Y 座標	Position Y Meters	実数	無し
Z 座標	Position Z Meters	実数	無し
回転	Rotation	実数	無し
表示	Display	Rect/StaticRect/Ic on	無し

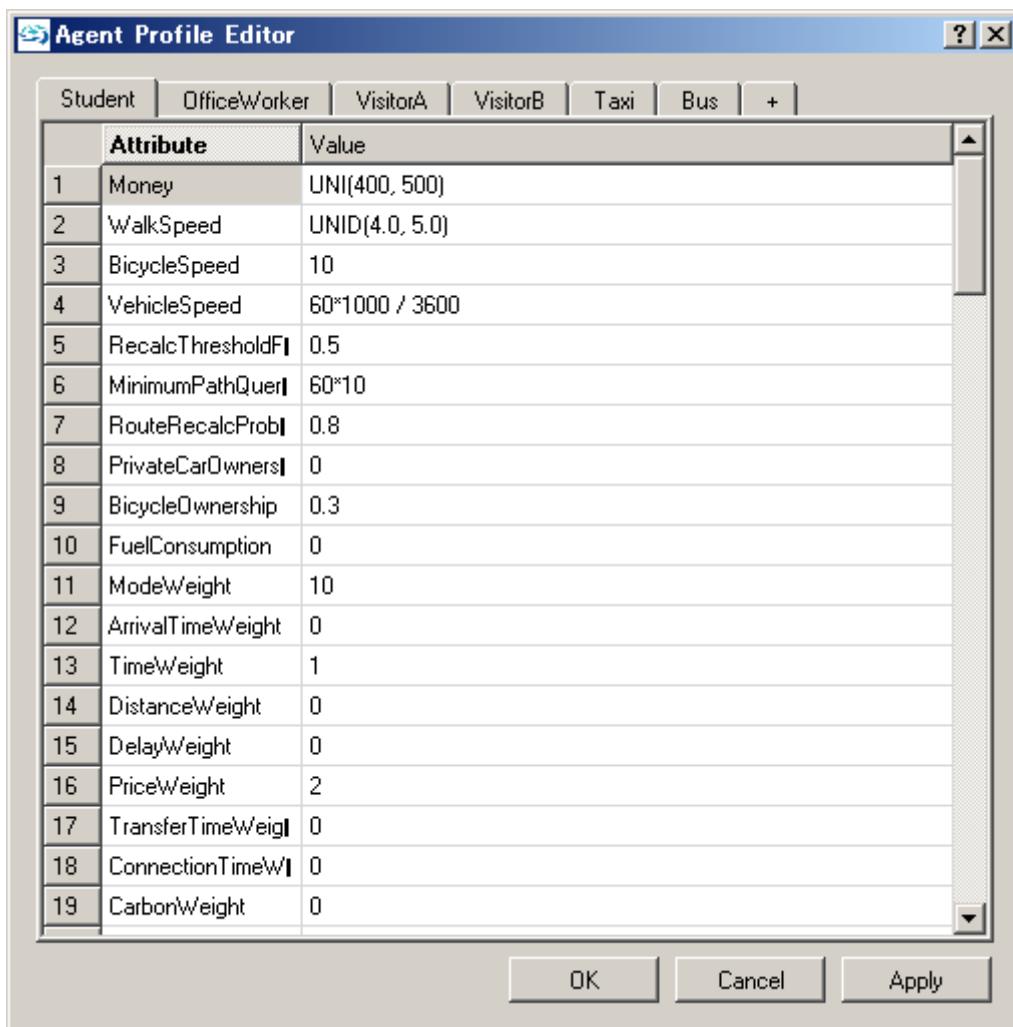
x 方向の長さ	X Length [m]	実数	無し
y 方向の長さ	Y Length [m]	実数	無し
アイコンパス	Icon Path	ファイルパス	無し
形状	Moving Object Shape Type	文字列	無し
トレースタグ	Trace Tags	文字列	Mas トレースタグを設定した場合、トレース出力に情報が出力される
トレース開始時刻	Trace Start Time	時間	有り
プロファイルタイプ	Profile Type	文字列	有り
行動タイプ	Behavior Type	文字列	有り

3.6.1. プロファイル設定

エージェントのプロファイルタイプを設定する。プロファイルタイプはマルチエージェントグローバル設定のエージェントプロファイル定義ファイルで指定されたファイル内で定義された文字列を利用可能である。

プロファイルタイプ定義の編集は場合、[Tools]->[Multi-Agent Settings]->[Agent Profile Editor]より行う。





タブ名にプロファイル名を指定し、タブ単位でプロファイルタイプを定義する。プロファイルのパラメータは Attribute として指定し、各項目について文字列で値の指定を行う。Taxi、Bus のプロファイルタイプはシミュレーションにおいて必須のプロファイルタイプであるため、タクシー、バスの有無に関わらず定義は常に残しておく。Attribute と Value の両方に値を含めた行が有効となり、未設定のセルを含む不十分な項目についてはスキップして処理を行う。

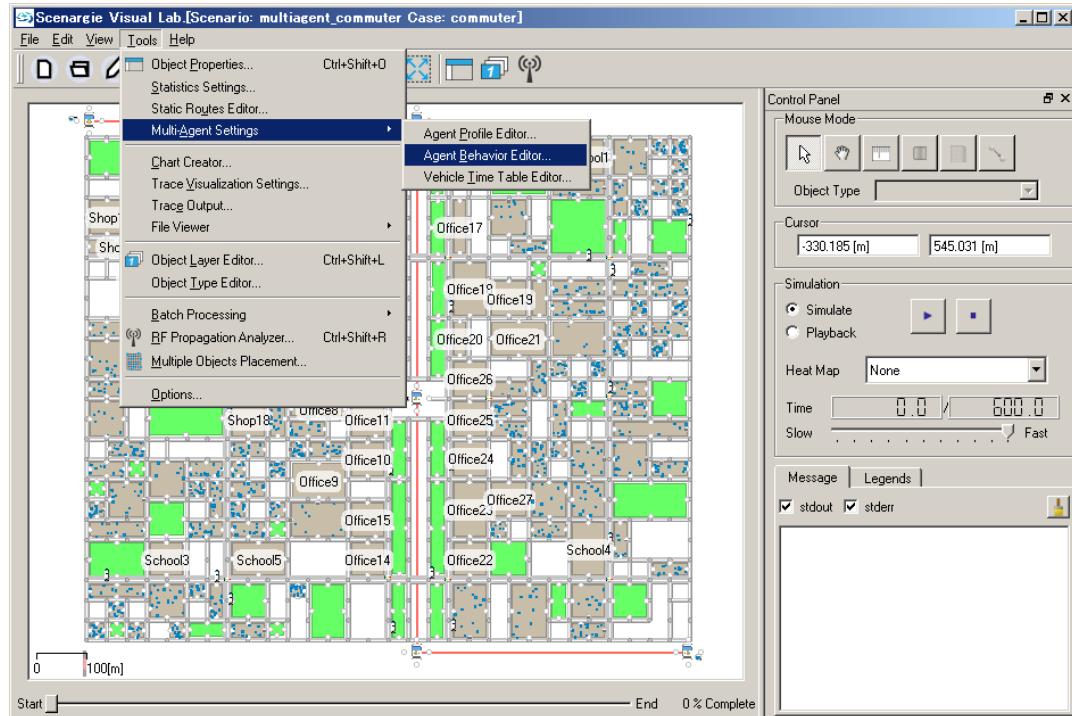
OK または Apply を選択した時点で、エージェントプロファイル定義ファイルの内容が上書きされる。

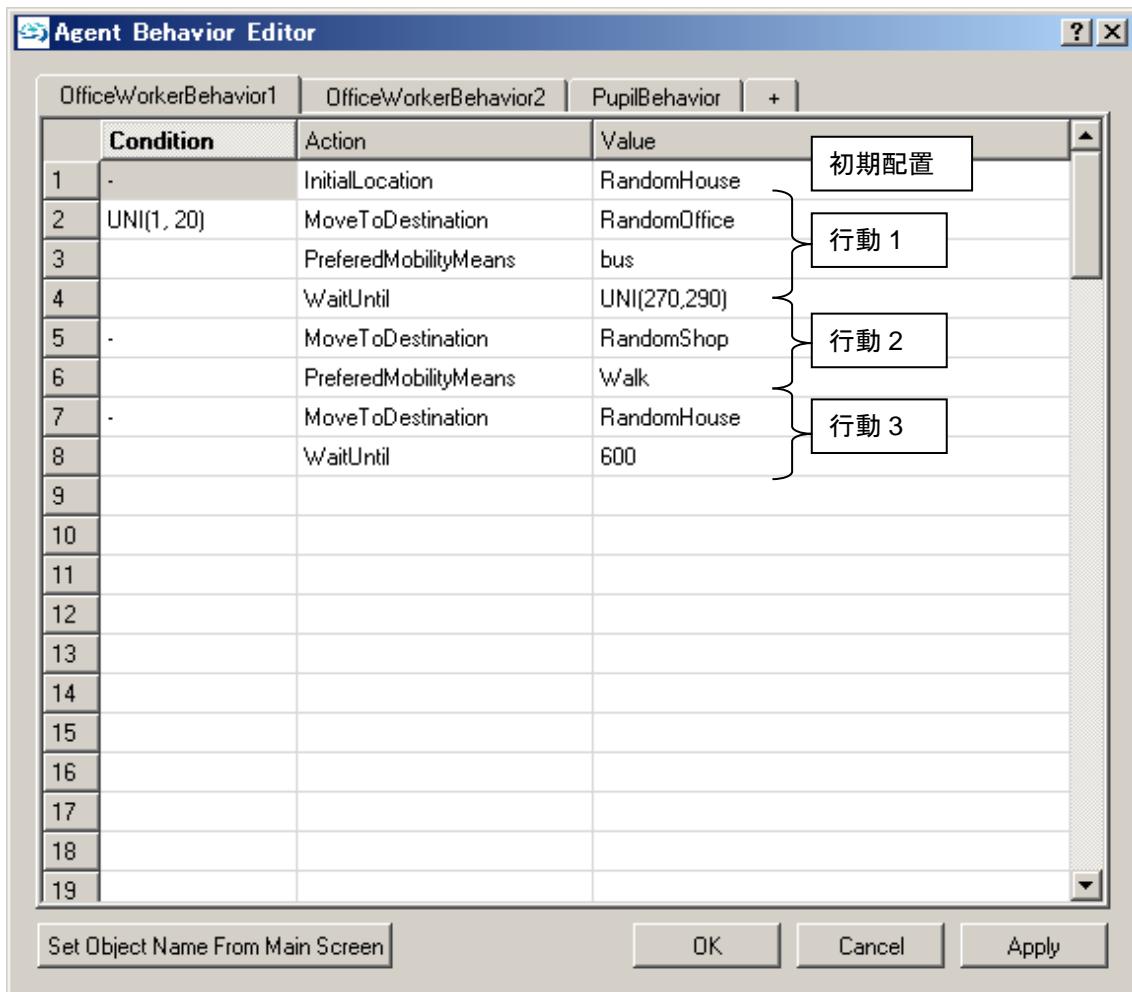
プロファイル値の詳細は「7.3 エージェントプロファイル定義ファイル」を参照。

3.6.2. 行動設定

エージェントの行動タイプを設定する。行動タイプはマルチエージェントグローバル設定のエージェント行動定義ファイルで指定されたファイル内で定義された文字列を利用可能である。

行動タイプ定義の編集は場合、[Tools]->[Multi-Agent Settings]->[Agent Behavior Editor]より行う。





タブ名に行動タイプ名を指定し、タブ単位で行動タイプを定義する。Conditionには行動時刻、Actionには行動、Valueには行動に渡す値を設定する。行動は時系列順に上の行から指定を行う。行動時刻に"-"-を設定した場合は、前の行動が終わったらすぐに行動を開始する。初期配置に"-"を設定した場合はシミュレーション時間 0 秒の時点での初期配置が行われる。

目的地をメイン画面上からクリック選択する場合、MoveToDestination の Value のセルを選択した後 Set Object Name From Main Screen のボタンを選択し、メイン画面上の目的地をクリックする。

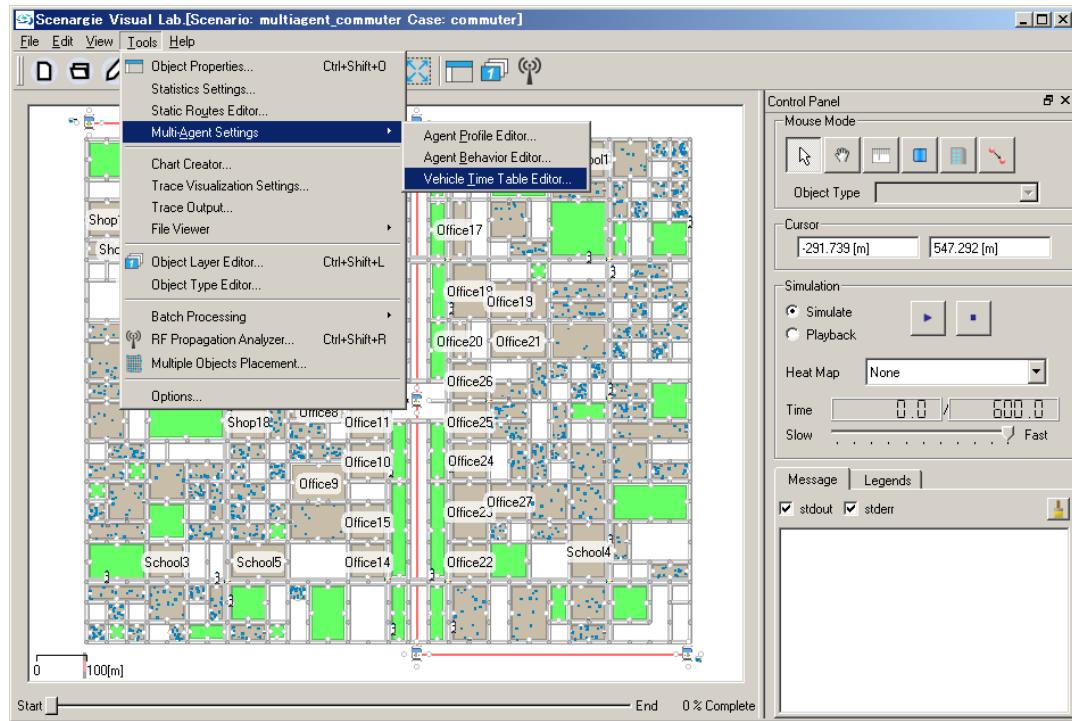
行動に対して条件を付加する場合、Condition の欄は空欄として、Action と Value を続けて記述する。Action と Value の両方に値を設定した行が有効となり、未設定のセルを含む不十分な項目については処理をスキップする。

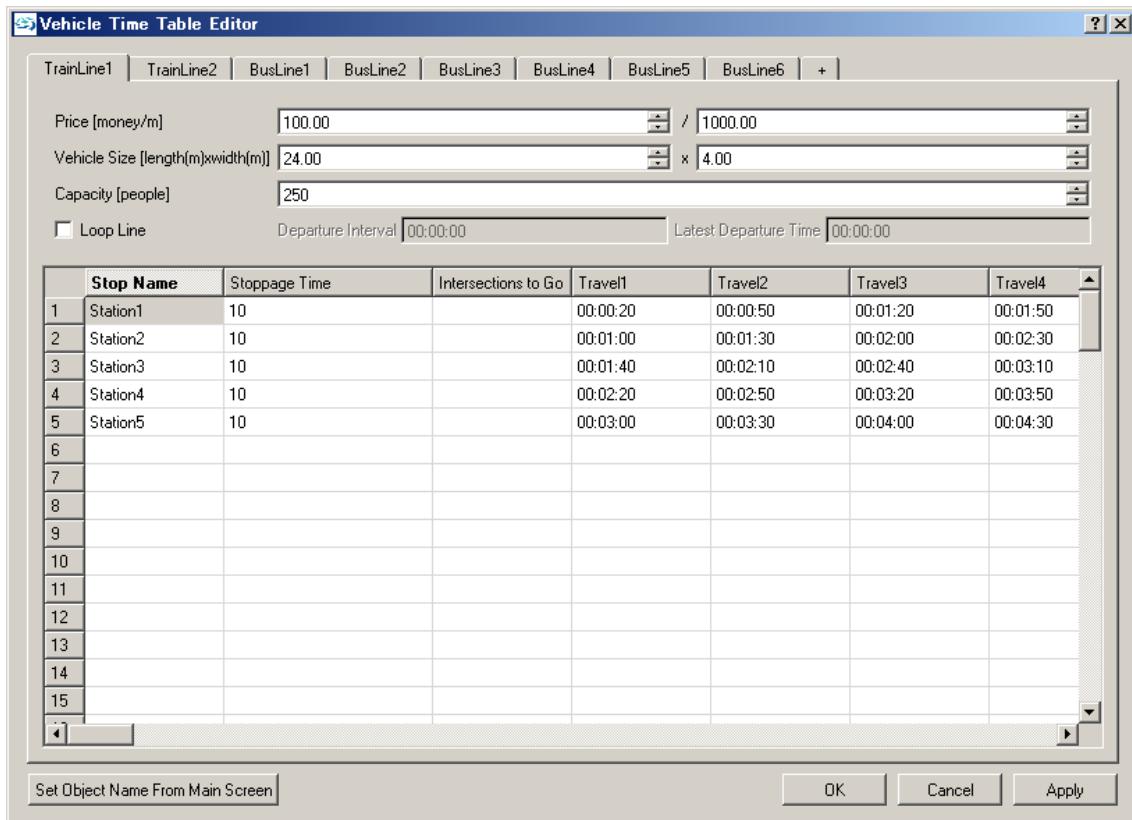
OK または Apply を選択した時点で、エージェント行動定義ファイルの内容が上書きされる。

行動定義の詳細は「7.4 エージェント行動定義ファイル」を参照。

3.7. エージェントタイムテーブル設定設定

信号機制御の機能を利用する場合、各信号機の Switching Pattern Type を Predefined に設定し、Predefined Pattern Name に信号パターン設定ファイルで指定したパターン名を指定する。





タブ名に運行ライン名を指定し、タブ単位で運行ラインを定義する。Price は距離あたりの運賃、Vehicle Size はシミュレーション上での車両の大きさ、Capacity は車両あたりの乗車定員を示す。

Stop Name に車両の停車する駅/バス停を停車順に設定する、バスの路線の場合にはバス停名、電車の路線の場合には駅名を指定する。バス停名と駅名が混在している場合には設定完了時にエラー表示が出力される。駅/バス停をメイン画面上からクリック選択する場合、Stop Name のセルを選択した後 Set Object Name From Main Screen のボタンを選択し、メイン画面上の目的地を停車する順にクリックする。バスの路線の場合、バス停選択後、交差点をクリックすることで(複数可)、次のバス停までに経由する交差点を指定可能である。

Stoppage Time は各駅に停車する時刻[秒]示す。

Intersections To Go Through は運行ラインがバス路線の場合に、指定した行のバス停から次の行のバス停までの移動において、経由すべき交差点を”.”(コロン)区切りで経由順に指定可能である。

Travel1 から TravelN までが運行ラインの運行スケジュールであり、一つの Travel が一つの運行スケジュールを示す。各 Travel について停車する駅/バス停への到着時刻を設定する。

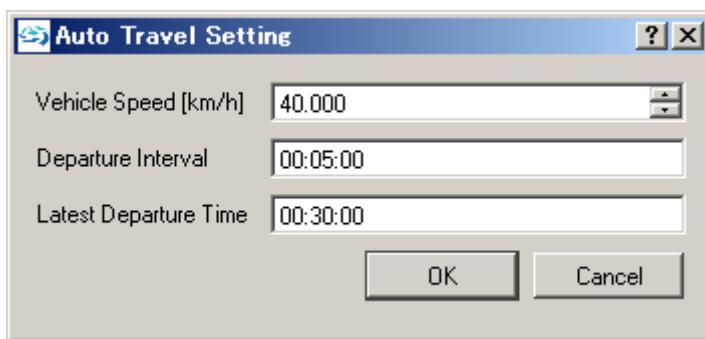
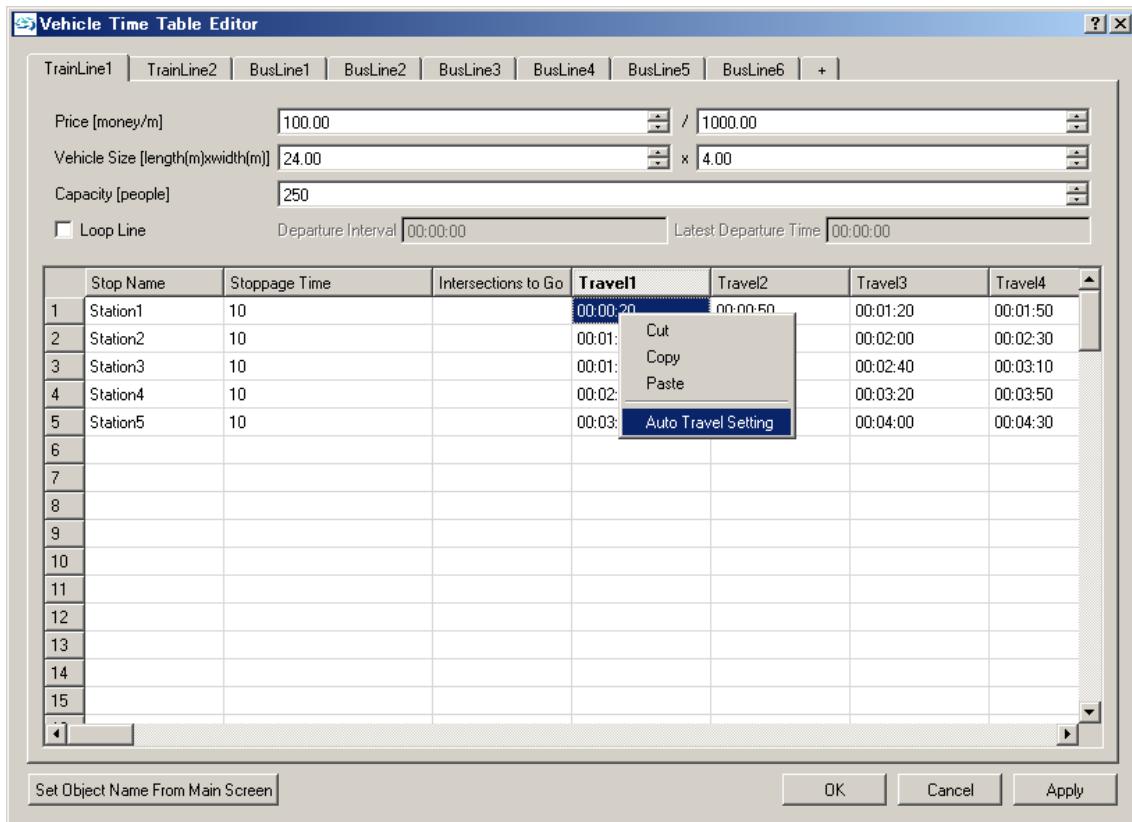
Stop Name、Stoppage Time、TravelNにおいて、未設定のセルを含む不十分な項目については処理をスキップする。

OK または Apply を選択した時点で、エージェント行動定義ファイルの内容が上書きされる。

行動定義の詳細は「7.5 エージェントタイムテーブル設定ファイル」を参照。

- 時刻表の自動設定

自動で時刻設定を行いたい Travel の列について、最初の駅(バス停)での時刻を設定している場合、右クリックより、Auto Travel Setting が利用可能である。



Vehicle Speed [km/h]: 駅(バス停)間での車両の速度 (自動設定用に到着時刻を推定するための速度で、シミュレーション時には参照される値ではない)

Departure Interval: 発車間隔

Latest Departure Time: 最終の発車時刻

Auto Travel Setting を実行すると、基準の Travel の時刻から Latest Departure Time の時刻分まで、Travel の列が補間される。各 Travel は Departure Interval で指定した間隔で追加される。Travel 内での各駅(バス停)での時刻は、各駅(バス停)の直線距離を Vehicle Speed で移動する時間に Stoppage Time(指定がない場合は 0)を加算した時刻で補間される。

- 環状線設定

環状線設定を行う場合、Loop Line のチェックを有効とする。環状線は、Travel1 の出発時刻から、Departure Interval で指定した間隔で、Latest Departure Time まで運行が行われる。車両は Departure Interval 毎に、始発駅(バス停)から発生するが、出発時刻に周回を終えた車両が存在する場合には、その車両がそのまま次に出発する車両として利用される。

環状線の場合には、最初の StopName と最後の StopName が一致していなければならない。また、Travel の設定は一つだけ(Travel1 のみ)の設定としなければならない。

Departure Interval にも 0 を指定した場合、一台の車両で、Latest Departure Time まで運行を実施する。

また Latest Departure Time が 0 の場合、シミュレーションの終了時刻まで運行を実施する。

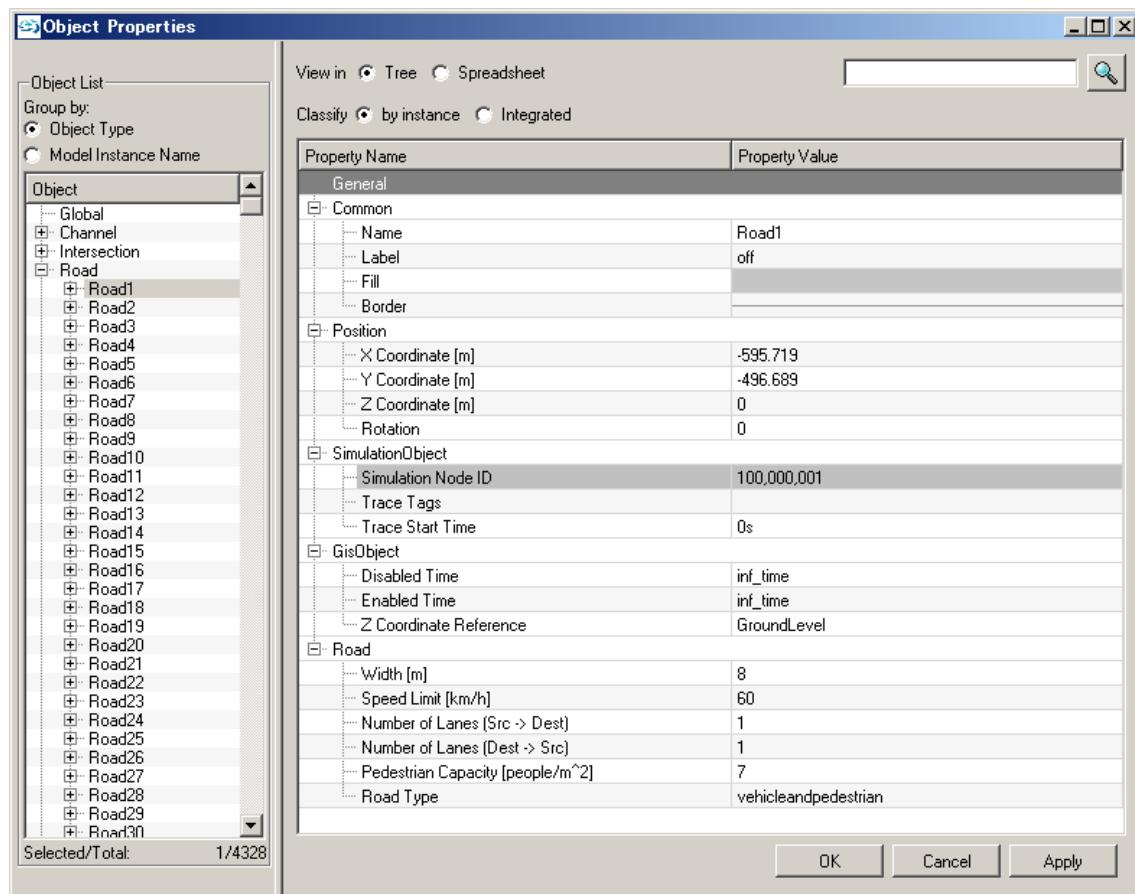
3.8. GIS オブジェクト設定

GIS オブジェクト設定では以下の項目を設定する。

- 道路設定
- 交差点設定
- 信号設定
- バス停設定
- 建物設定
- 公園設定
- 駅設定
- 入り口設定
- エリア設定

3.8.1.道路設定

道路設定では、各道路についてのプロパティ設定を行う。



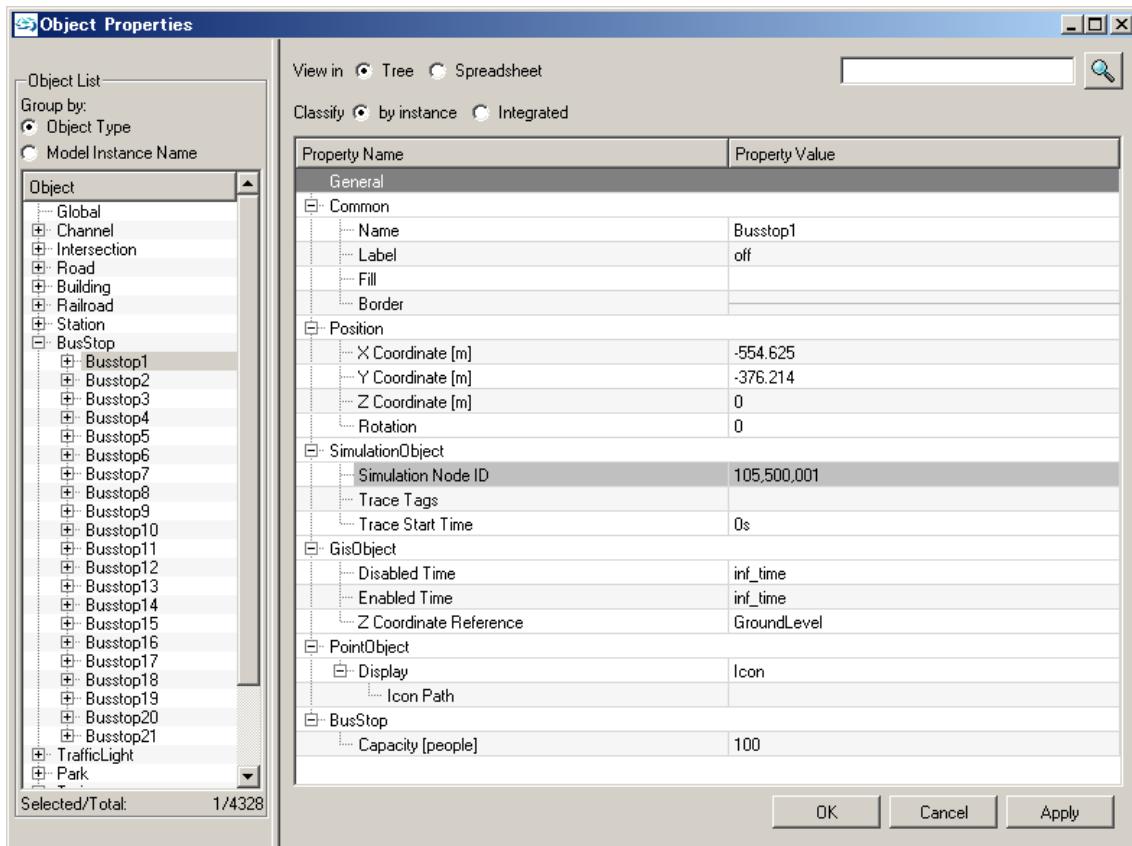
プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	無し
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	有り
ID	Simulation Node ID	整数(シミュレーション時に自動で割り当て、編集不可)	有り
トレースタグ	Trace Tags	文字列	Gis トレースタグを設定した場合、ト レース出力に Enabled/Disabled の 情報が出力される
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	有り
有効化時刻	Enabled Time	時間	有り
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
道路幅	Width [m]	実数	有り
制限速度	Speed Limit [km/h]	実数	有り
始点から終点向きの車線数	Number of Lanes (Src -> Dest)	整数	有り
終点から始点向きの車線数	Number of Lanes (Dest -> Src)	整数	有り
歩行者のキャパシティ	Pedestrian Capacity [people/m^2]	実数	有り
道路種別	Road Type	Vehicle AndPedestrian/Pe	有り

		destrianOnly/VehicleOnly	
--	--	--------------------------	--

Number of Lanes (Src -> Dest)または、Number of Lanes (Dest -> Src)のいずれかを0とした場合には、車線数を指定した向きについて一方通行となる。Number of Lanes (Src -> Dest)およびNumber of Lanes (Dest -> Src)の両方を0とすることは不可である。

3.8.2. 交差点設定

交差点のプロパティ設定は、マルチエージェントのシミュレーションに影響しない。(※ただし、交差点の位置を変更した場合は道路網のトポロジが変わり、シミュレーションに影響が与える場合がある)

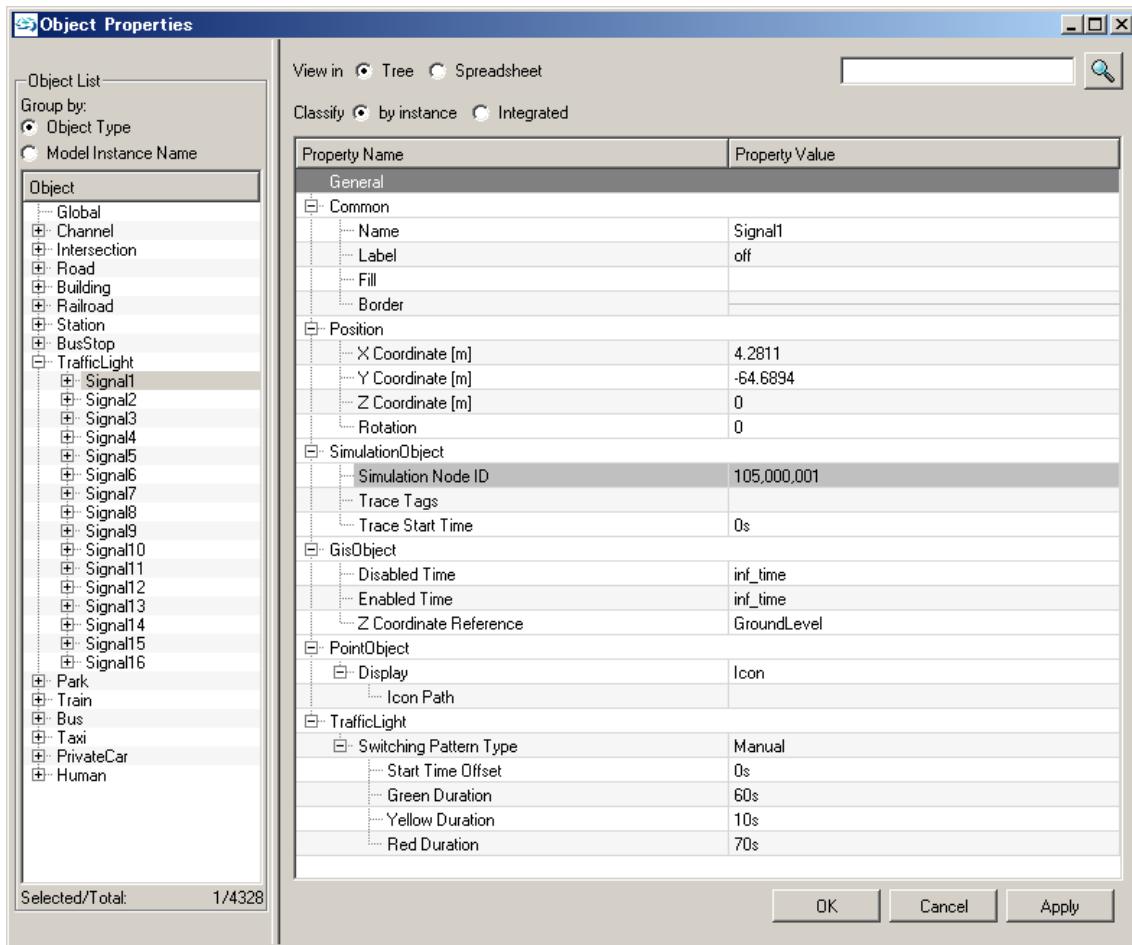


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	無し
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション時に自動で割り当て、編集不可)	有り

トレースタグ	Trace Tags	文字列	無し
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
交差点半径	Radius [m]	実数	無し
ITS Extension Module 利用時 のMATESモビリ ティモデルの車 両の流入量	Node Generation Rate [node/s]	実数	無し

3.8.3.信号設定

信号設定では、各信号についてのプロパティ設定を行う。交差点や道路の移動により、本来信号の配置されるべき場所からはずれてしまった信号は、シミュレーションにおいて無視される。



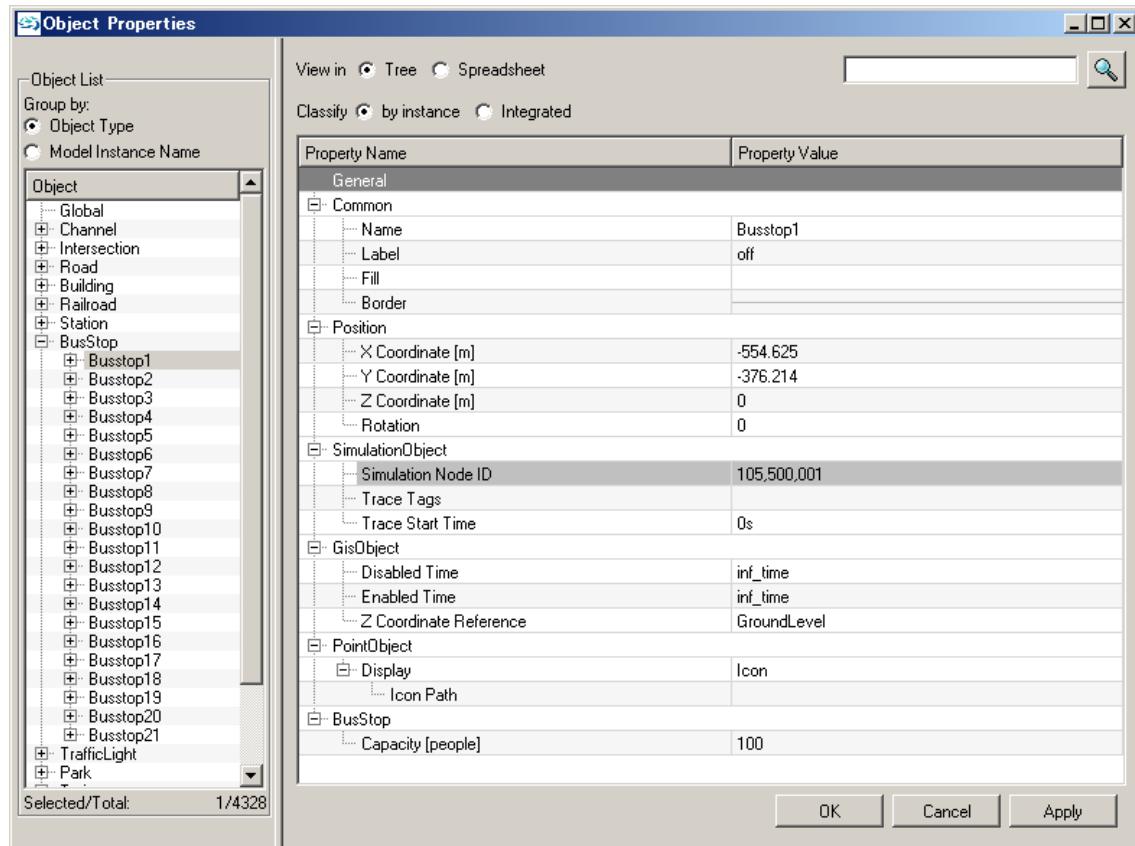
プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	無し
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション)	有り

		ン時に自動で割り 当て、編集不可)	
トレースタグ	Trace Tags	文字列	Gis トレースタグを設定した場合、ト レース出力に Enabled/Disabled の 情報が出力される
トレース開始時 刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	有り
有効化時刻	Enabled Time	時間	有り
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し
制御パターン種別	Switching Pattern Type	Manual/Predefine d	有り
開始時刻オフセ ット	Start Time Offset	時間	有り
青色時間	Green Duration	時間	有り
黄色時間	Yellow Duration	時間	有り
赤色時間	Red Duration	時間	有り
制御パターン名	Predefiend Pattern Name	文字列	有り

制御パターン名は GIS グローバル設定の信号パターン定義ファイル内の定義名を利用可能である。無効化状態の信号である場合、エージェントは信号が無いものとして交差点エリアでの挙動を行う。

3.8.4. バス停設定

バス停設定では、各バス停について収容人数の設定を行う。

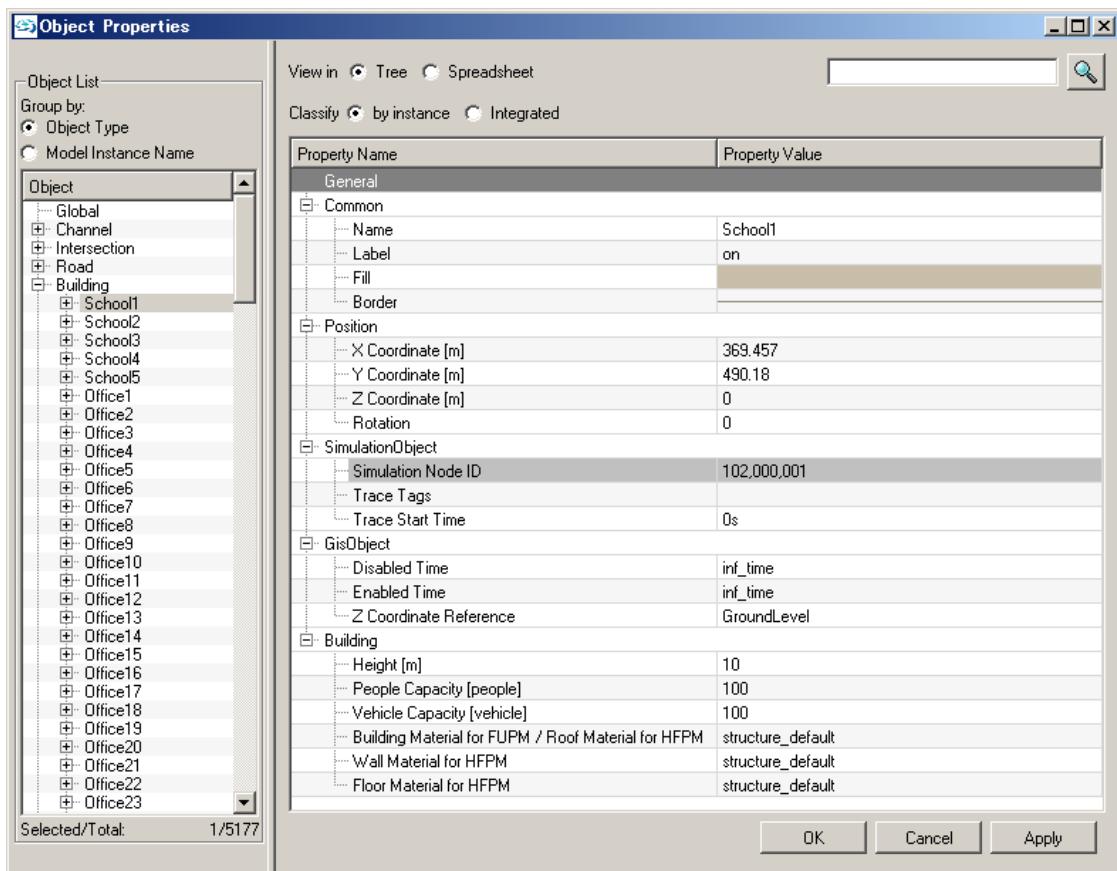


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	公共エージェントタイムテーブル設定設定で利用している名称のバス停の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション時に自動で割り)	有り

		当て、編集不可)	
トレースタグ	Trace Tags	文字列	無し
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し
最大収容人数	Capacity [people]	整数	有り

3.8.5. 建物設定

建物設定では、各建物について収容人数の設定等を行う。

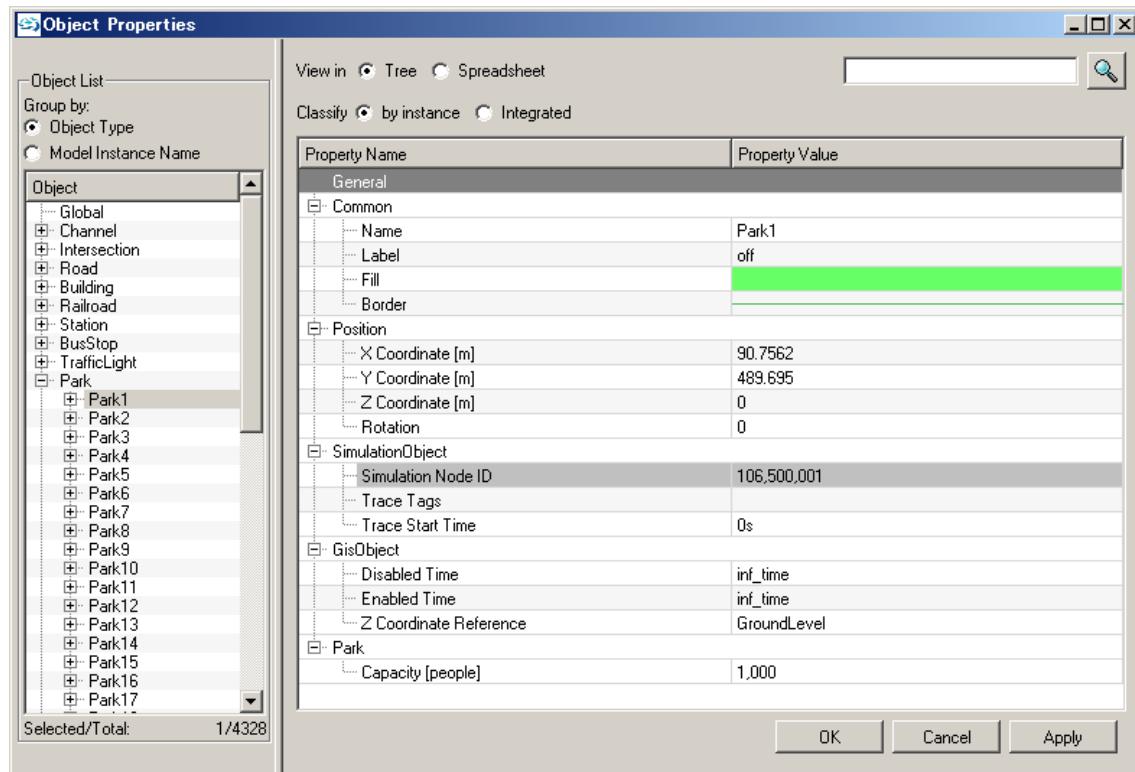


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	エージェント行動定義ファイルで利用している名称の建物の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	有り
ID	Simulation Node ID	整数(シミュレーション)	有り

		ン時に自動で割り 当て、編集不可)	
トレースタグ	Trace Tags	文字列	無し
トレース開始時 刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
屋根の材質	Building Material for FUPM / Roof Material for HFPM	文字列	無し
壁の材質	Wall Material for HFPM	文字列	無し
床の材質	Floor Material for HFPM	文字列	無し
人の最大収容人 数	People Capacity [people]	整数	有り
車の最大収容数	Vehicle Capacity [vehicle]	整数	有り

3.8.6.公園設定

公園設定では、各公園について収容人数の設定を行う。

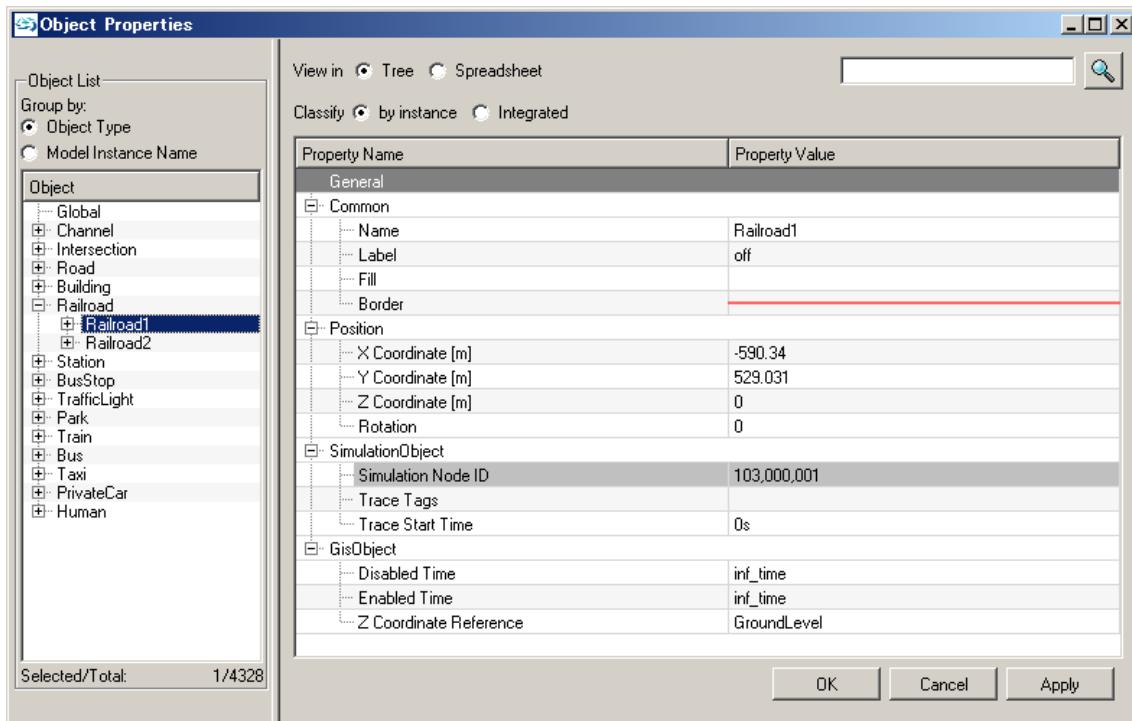


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	エージェント行動定義ファイルで利用している名称の公園の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	有り
ID	Simulation Node ID	整数(シミュレーション)	有り

		ン時に自動で割り 当て、編集不可)	
トレースタグ	Trace Tags	文字列	Gis トレースタグを設定した場合、ト レース出力に Enabled/Disabled の 情報が出力される
トレース開始時 刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
最大収容人数	Capacity [people]	整数	有り

3.8.7.線路設定

線路設定では、各線路についての設定を行う。

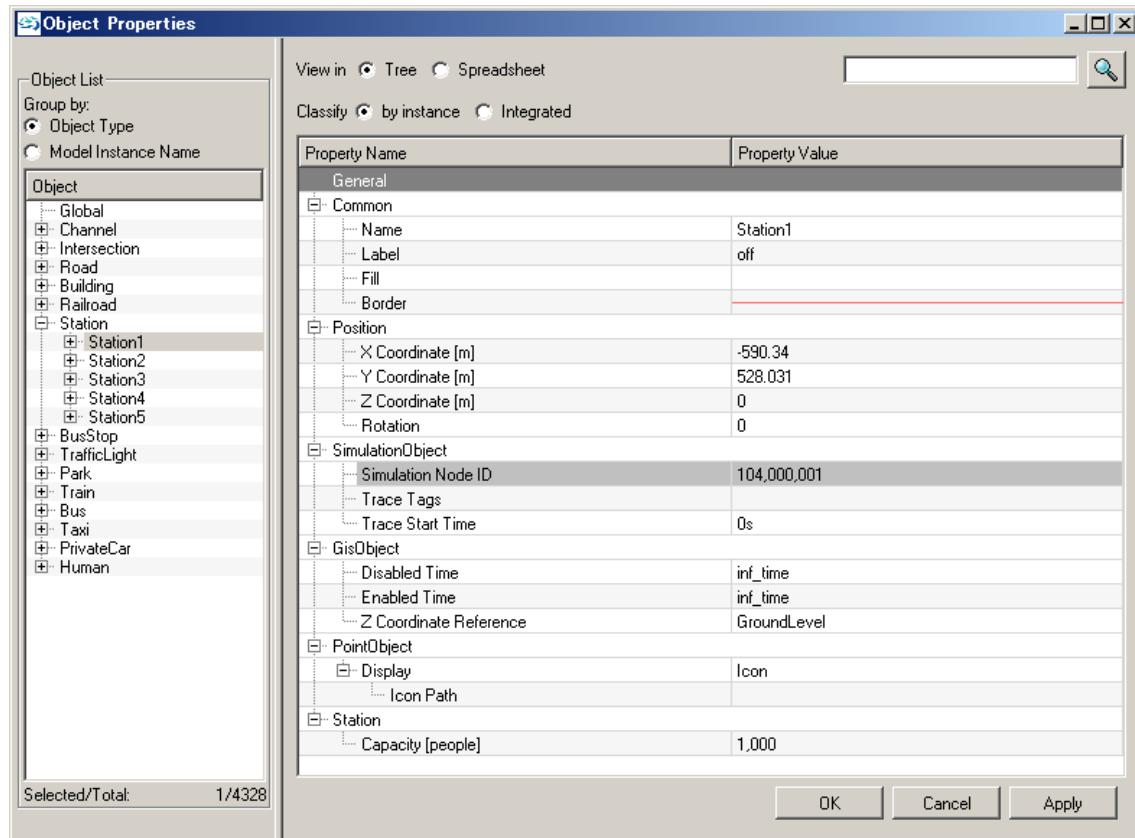


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	無し
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	有り
ID	Simulation Node ID	整数(シミュレーション時に自動で割り当て、編集不可)	有り
トレースタグ	Trace Tags	文字列	Gis トレースタグを設定した場合、トレース出力に Enabled/Disabled の

			情報が出力される
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	有り
有効化時刻	Enabled Time	時間	有り
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り

3.8.8. 駅設定

駅設定では、各駅について収容人数の設定を行う。

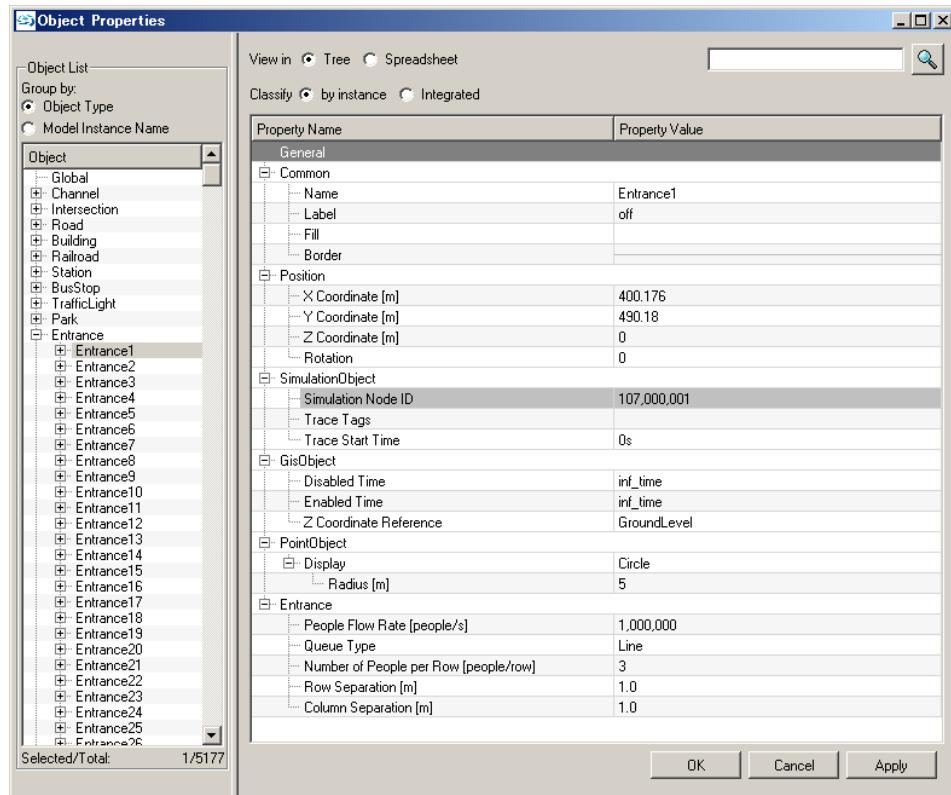


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	公共エージェントタイムテーブル設定設定で利用している名称の駅の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション時に自動で割り)	有り

		当て、編集不可)	
トレースタグ	Trace Tags	文字列	Gis トレースタグを設定した場合、トレース出力に Enabled/Disabled の情報が出力される
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し
最大収容人数	Capacity [people]	整数	有り

3.8.9. 入り口設定

入り口設定では、各入り口についての座標の変更が可能である。

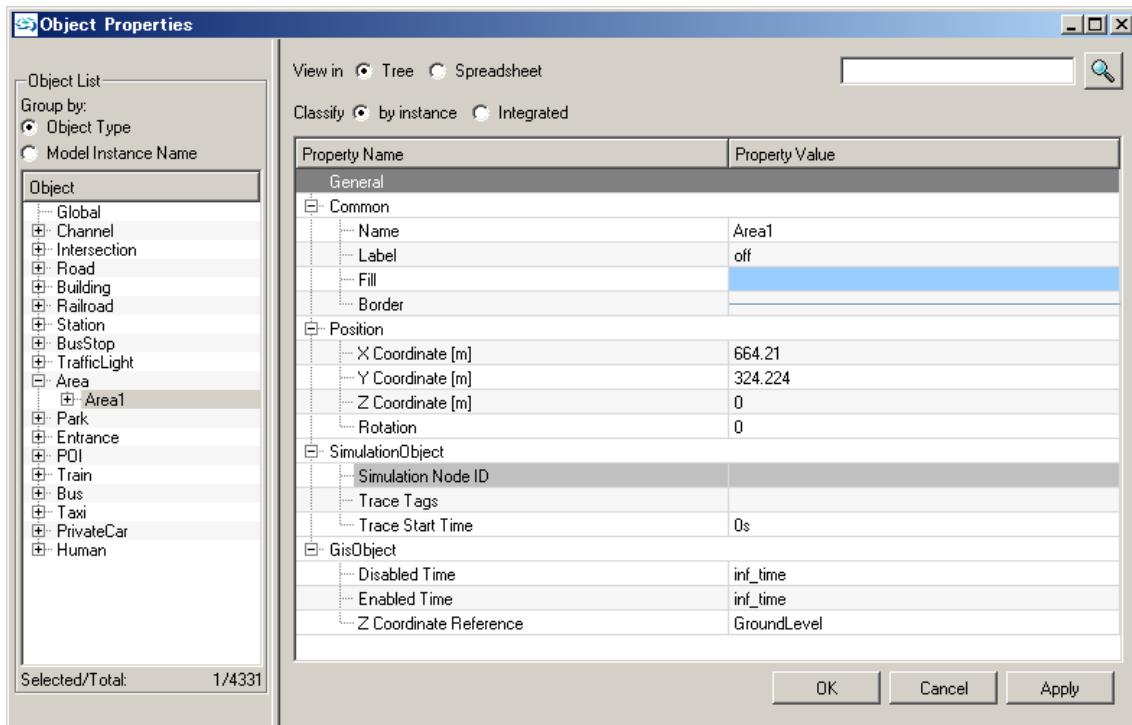


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	エージェント行動定義ファイルで利用している名称の建物の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション時に自動で割り当て、編集不可)	有り

トレースタグ	Trace Tags	文字列	無し
トレース開始時刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し

3.8.10. エリア設定

エリア設定では、各エリアの名称、色、座標の変更が可能である。

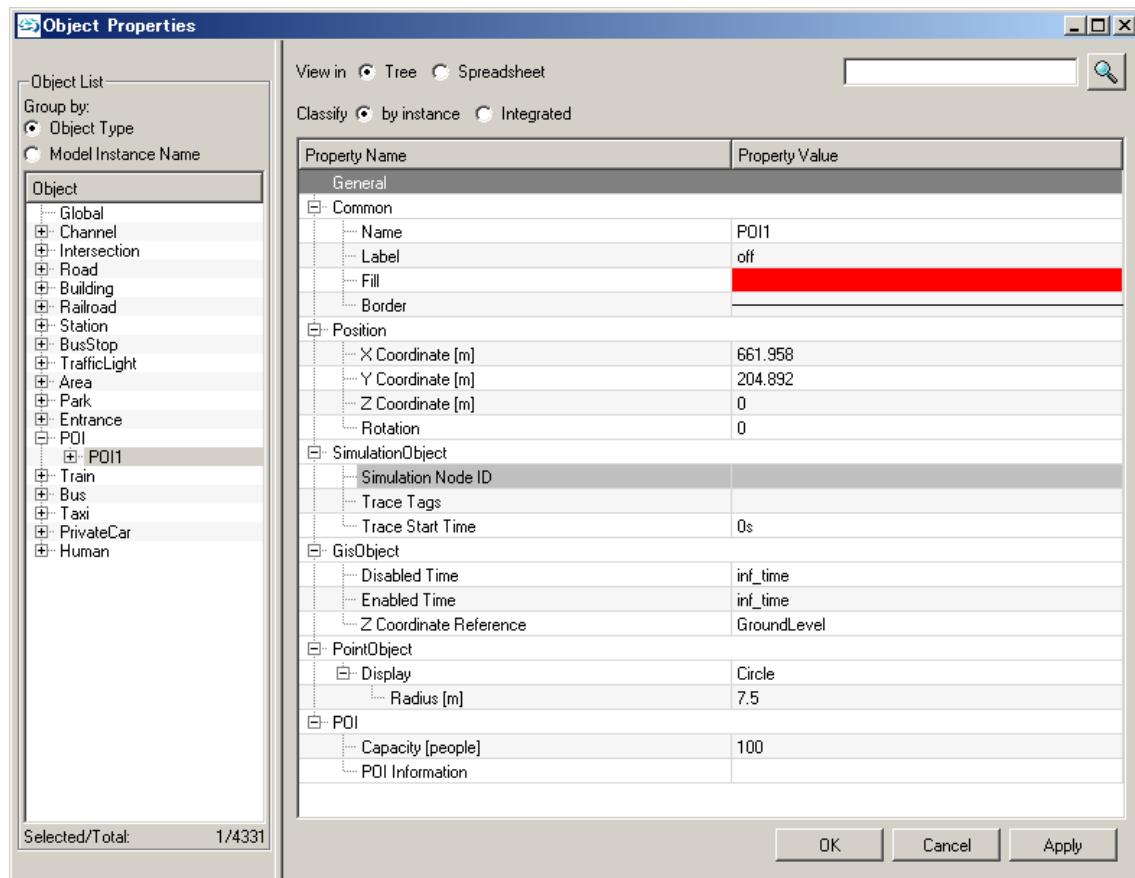


プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	エージェント行動定義ファイルで利用している名称の建物の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	座標の変更によりエリア内の建物が変化する場合影響あり
Y 座標	Position Y Meters	実数	座標の変更によりエリア内の建物が変化する場合影響あり
Z 座標	Position Z Meters	実数	無し
回転	Rotation	実数	回転によりエリア内の建物が変化する場合影響あり
ID	Simulation Node ID	整数(シミュレーション)	有り

		ン時に自動で割り 当て、編集不可)	
トレースタグ	Trace Tags	文字列	無し
トレース開始時 刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	無し
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し
入口での一秒当 たりの最大流入 可能人数 [人/s]	Peopl Flow Rage [people/s]	実数	有り
入口待ちのキュー 一種別	Queue Type	Line	有り
入口待機での列 あたりの人数 [人/列]	Number of People per Row [people/row]	整数	無し
入口待機列の間 隔 [m]	Row Separation [m]	実数	他 Agent の行動には影響無し (※入り口待ち中の位置は変化する ため、通信シミュレーションの結果 には影響がある)
入口待機行の間 隔 [m]	Column Separation [m]	実数	他 Agent の行動には影響無し (※入り口待ち中の位置は変化する ため、通信シミュレーションの結果 には影響がある)

3.8.11. POI 設定

POI 設定では、各 POI の情報を変更可能である。



プロパティ	プロパティ名	値	マルチエージェントシミュレーションへの影響
名称	Name	文字列	エージェント行動定義ファイルで利用している名称の建物の場合、影響有り
ラベル ON/OFF	Label	on/off	無し
色	Fill	色	無し
枠色	Border	色	無し
X 座標	Position X Meters	実数	有り
Y 座標	Position Y Meters	実数	有り
Z 座標	Position Z Meters	実数	有り
回転	Rotation	実数	無し
ID	Simulation Node ID	整数(シミュレーション)	有り

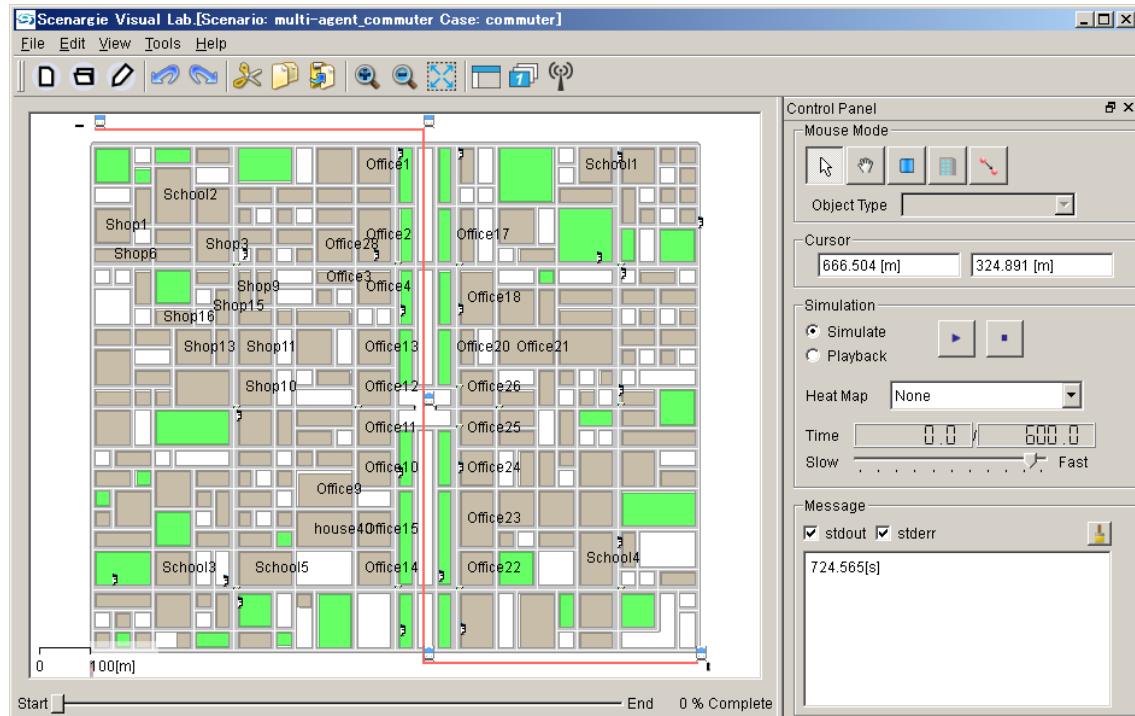
		ン時に自動で割り 当て、編集不可)	
トレースタグ	Trace Tags	文字列	無し
トレース開始時 刻	Trace Start Time	時間	無し
無効化時刻	Disabled Time	時間	無し
有効化時刻	Enabled Time	時間	無し
基準の高さ	Z Coordinate Reference	GroundLevel/Sea Level	有り
表示設定	Display	Circle/Icon	無し
アイコンパス	Icon Path	ファイルパス	無し
円表示半径	Radius [m]	実数	無し
最大収容人数	Capacity [people]	整数	有り
情報	POI Information	文字列	無し(標準のモデルでは影響なし)

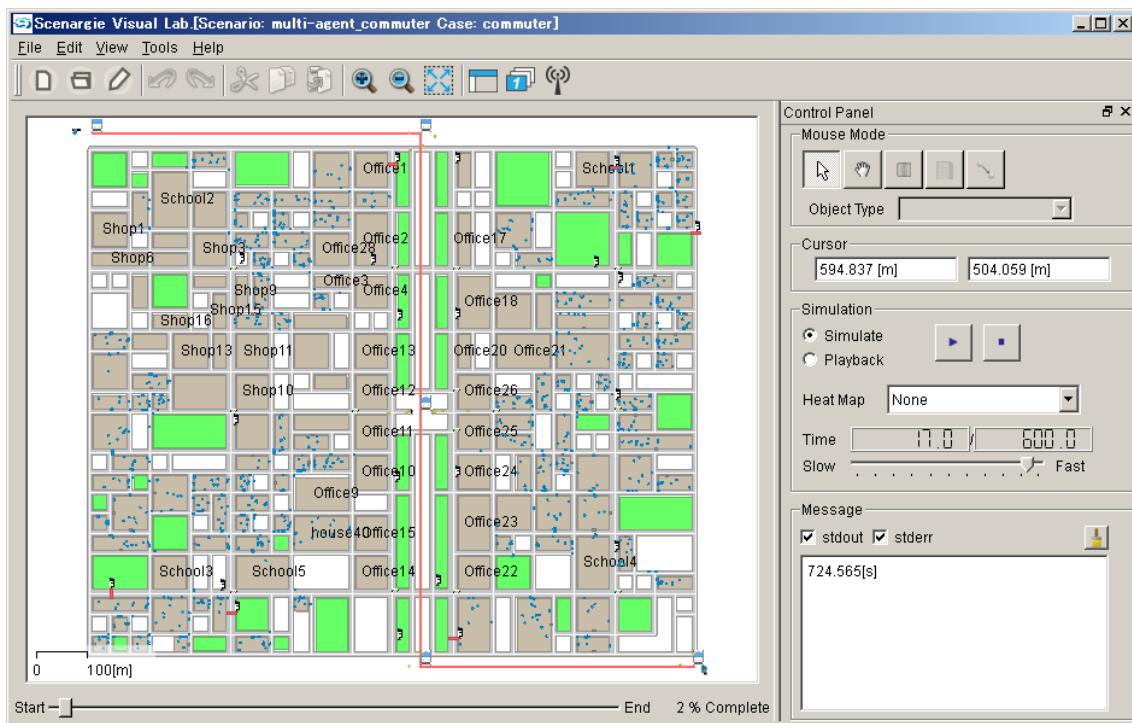
4. シミュレーション

シミュレーションの実行はメインウインドウの右側のコントロールパネルの実行ボタンから実行する。

4.1. シミュレーションの実行

シミュレーションの実行はメインウインドウの右側のコントロールパネルの実行ボタンから実行する。

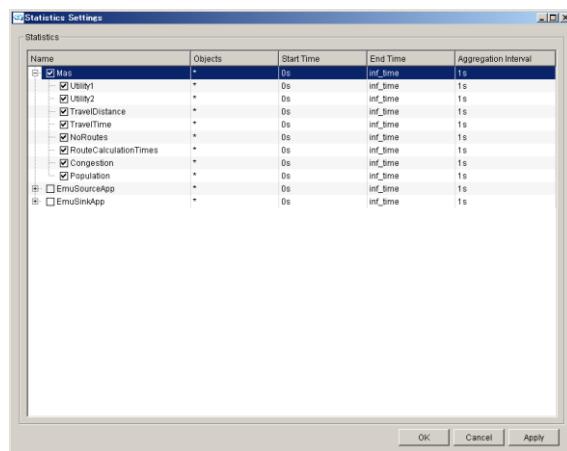
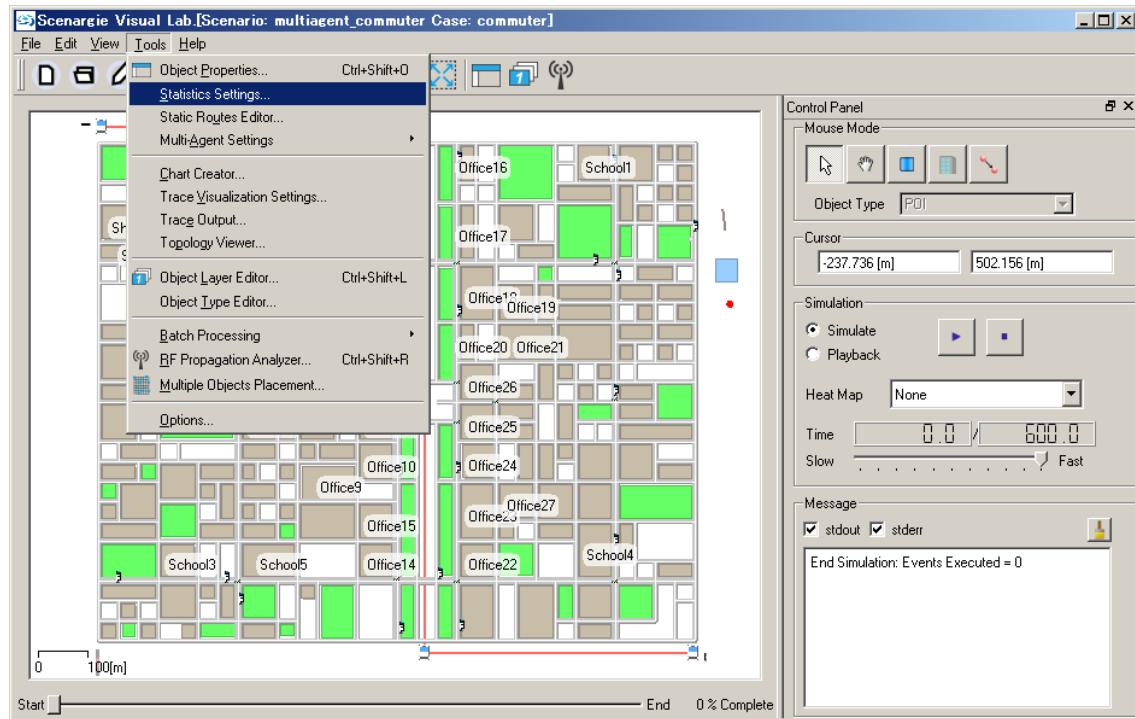




シミュレーションが開始するとエージェントは設定された情報を元に行動を行う。移動状態はメイン画面上で確認可能である。

4.2. 統計値設定

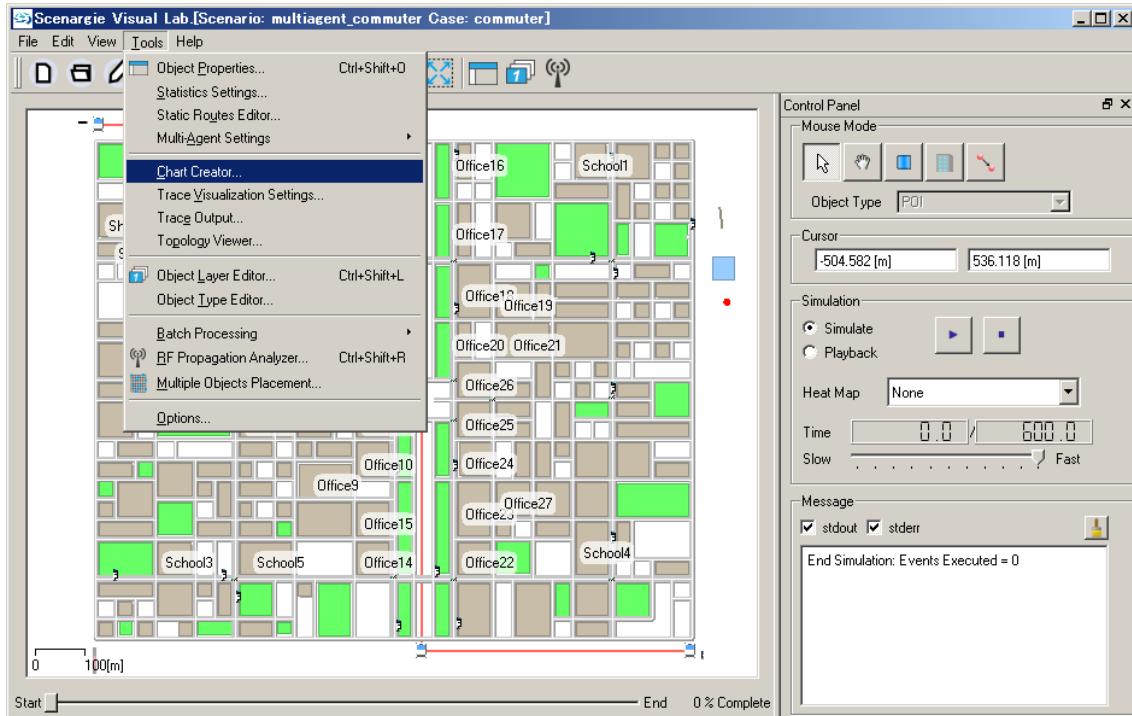
統計値は[Tools]->[StatisticsSetting]で Mas を設定可能である。統計値の設定を行う場合、統計対象のオブジェクトがシナリオ上に配置されていなければならない。



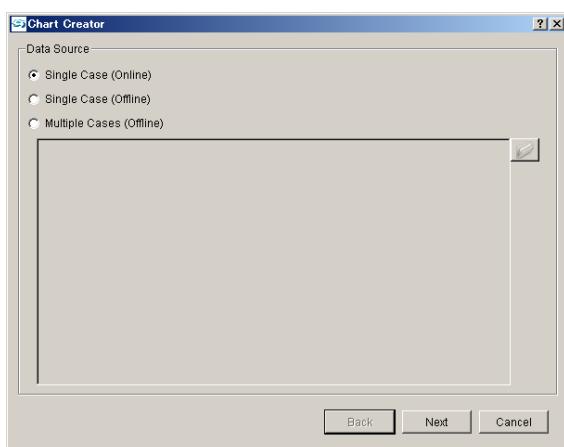
マルチエージェント用のトレース情報は"Mas"である。設定を行った統計値は、シミュレーショングローバル設定の統計情報出力ファイルに出力される。

4.3. 統計値表示

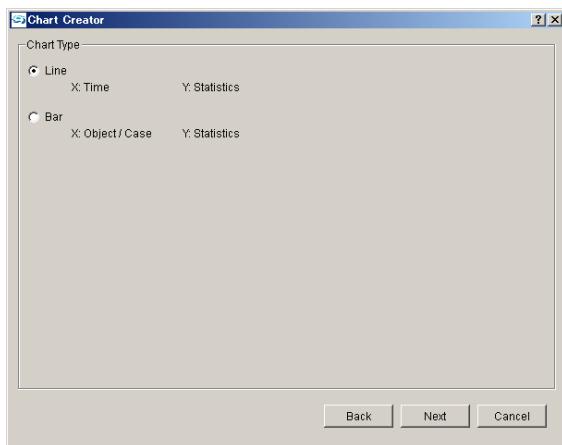
Chart Creator でチャートウィンドウを表示した状態でシミュレーションを実行することで、エージェントまたは GIS 情報の統計値をグラフ表示可能である。[Tools]->[Chart Creator]よりチャートウィンドウの作成を行う。



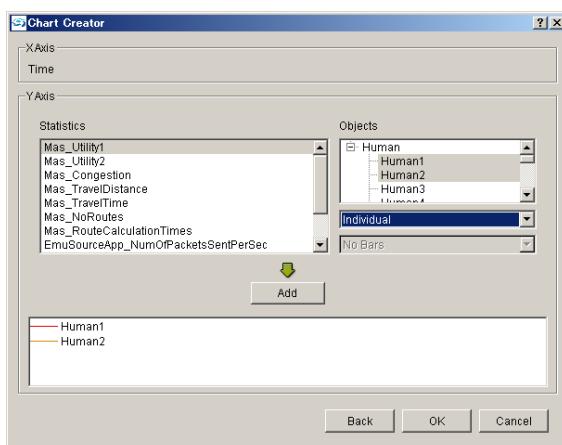
- 1) 表示する統計データの出力元(Data Source)を選択する。シミュレーションを実行しながらグラフを表示する場合は Single Case(Online)を選択する。(出力済みの統計ファイル“.stat”を表示する場合は Single Case (Offline)を選択する。)



- 2) グラフの種別(Chart Type)を選択する。時系列に沿って値を表示する場合は Line を選択する。



- 3) Statistics より統計値を選択し、Objects より統計値を表示するオブジェクトを選択する。オブジェクト選択後、Select Counting Type のボックスより統計方法を選択する。統計方法を Average または Median とした場合は、さらに統計オプションを選択可能である。統計値・オブジェクト・統計方法・(統計オプション)を選択した後、Add によりグラフへの追加が行われる。



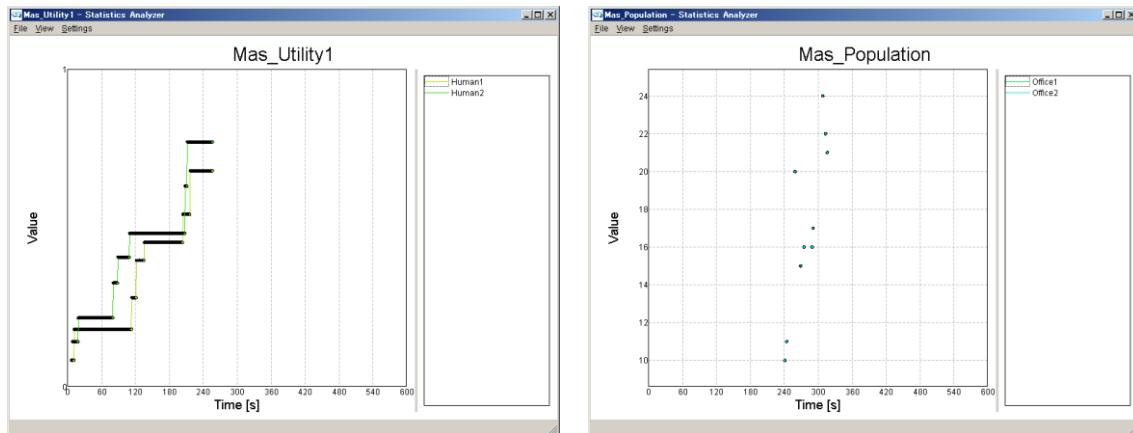
統計値の種類は、「7.2.2 統計値設定」を参照。

統計方法

- Individual: 個別に統計
- Average: 平均値
- Median: 中央値
- Total: 合計値

統計オプション

- Upper and Lower Limits: 上限値・下限値の表示
- 99% Confidence Limits: 99%信頼区間の表示
- 95% Confidence Limits: 95%信頼区間の表示



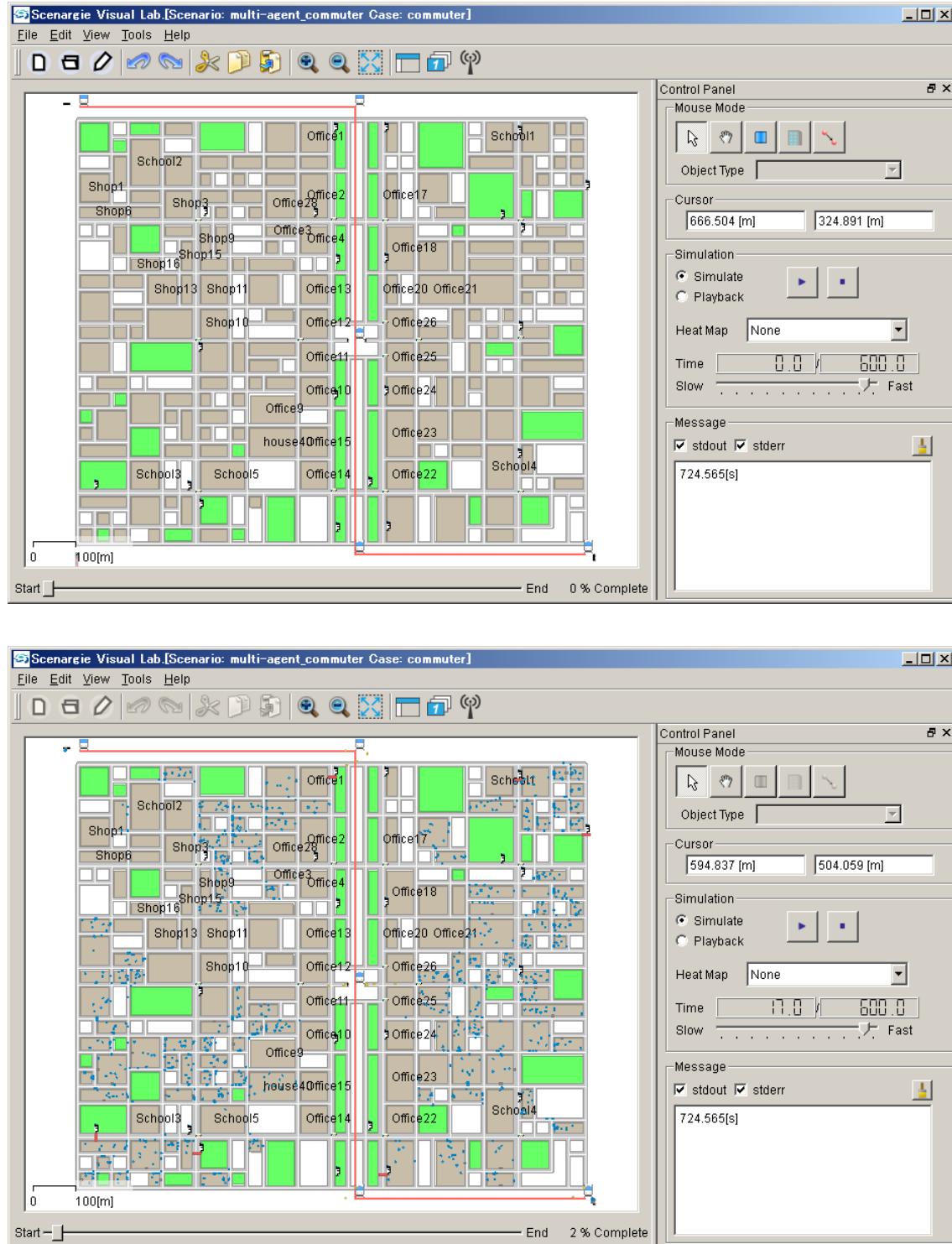
4.4. 編集したシナリオのコマンドラインからのシミュレーション実行

シミュレーションの実行に関して、Visual Lab を使用せずにコマンドラインからシミュレーションを実行することも可能である。コマンドラインからシミュレーションを実行した場合、Visual Lab とのデータのやりとりが必要ないため、実行速度が向上する。

シミュレーションのコマンドラインからの実行は、Visual Lab の[メニュー] -> [File] -> [Export] -> [Simulation Configuration File]で出力したファイルをシミュレーション実行ファイルの引数に渡すことで可能である。エクスポートにより出力されるファイルは、シミュレーションの入力ファイルである".config"ファイルと GIS 情報を含めた shape ファイルとなる。Object Properties で設定した情報は".config"に、GIS に関する情報は shape ファイルに出力される。

5. プレイバック

一度シミュレーションを実行した後、コントロールパネルの Playback をチェックした後実行する。

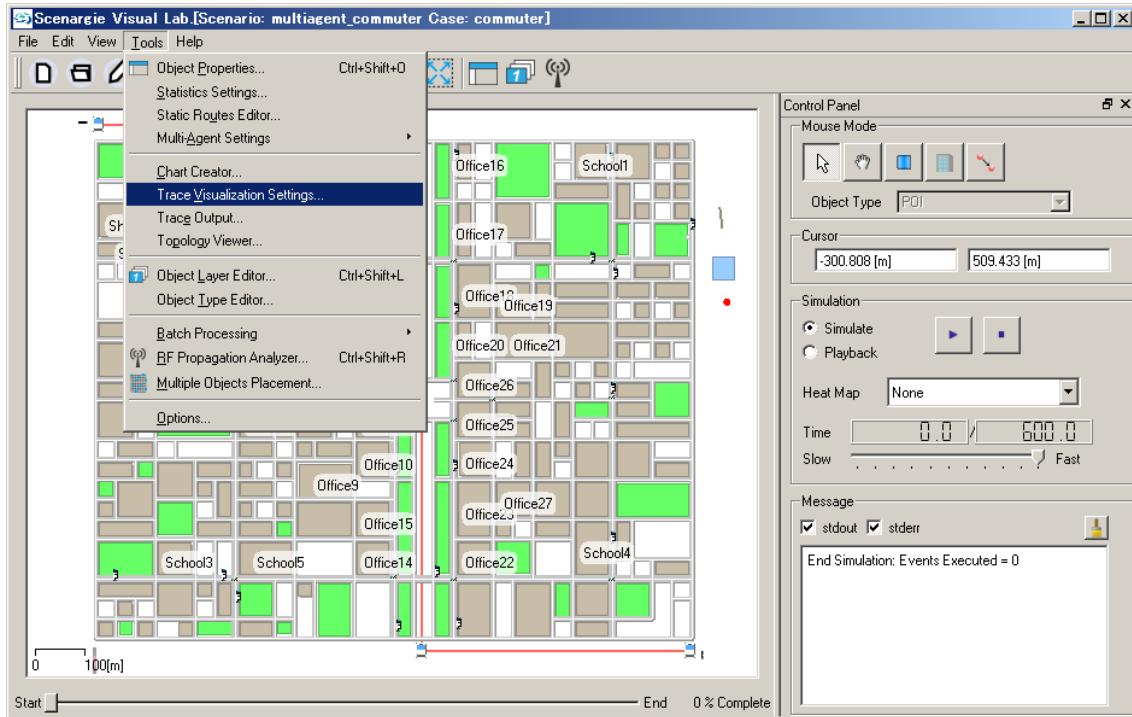


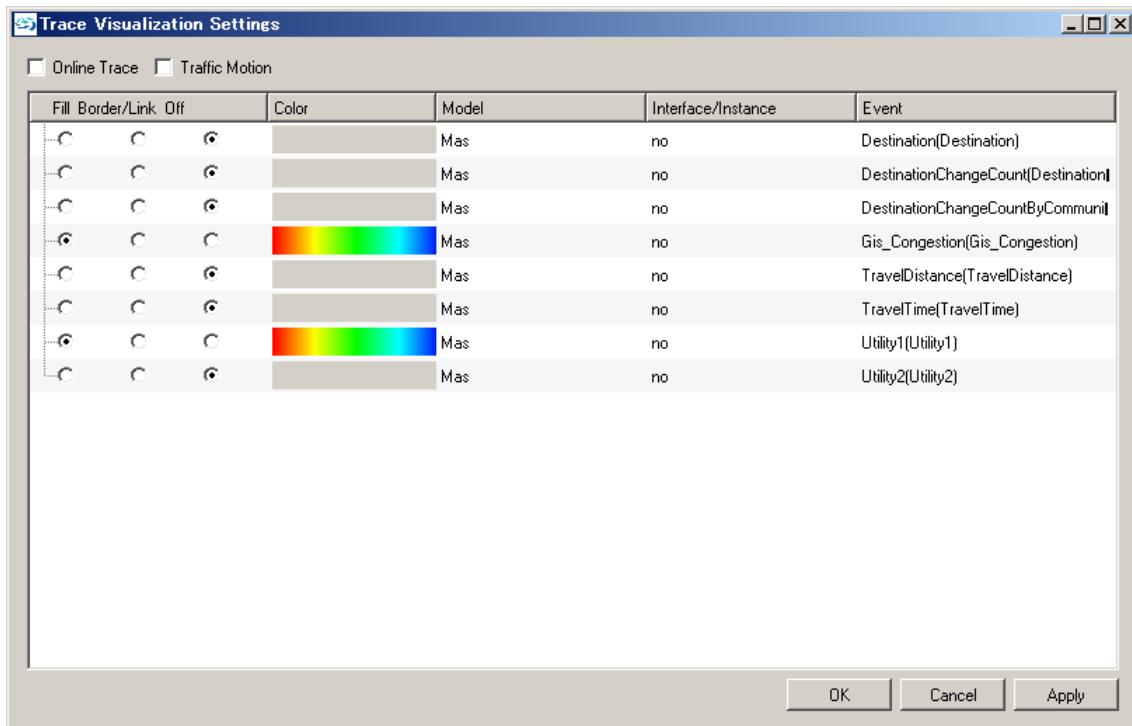
5.1. トレース表示

プレイバック実行時にトレース情報を表示可能である。トレース表示を行う場合、シミュレーションの設定でエージェントまたは GIS オブジェクトのトレースタグを有効化し、シミュレーションを行う必要がある。

トレース情報の設定は「7.2.3 トレース設定」を参照

トレース表示の設定は[Tools]->[Trace Visualization Settings]より行う。

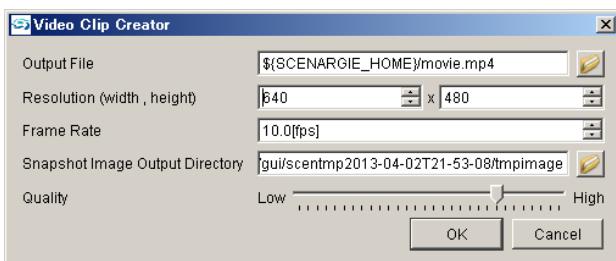
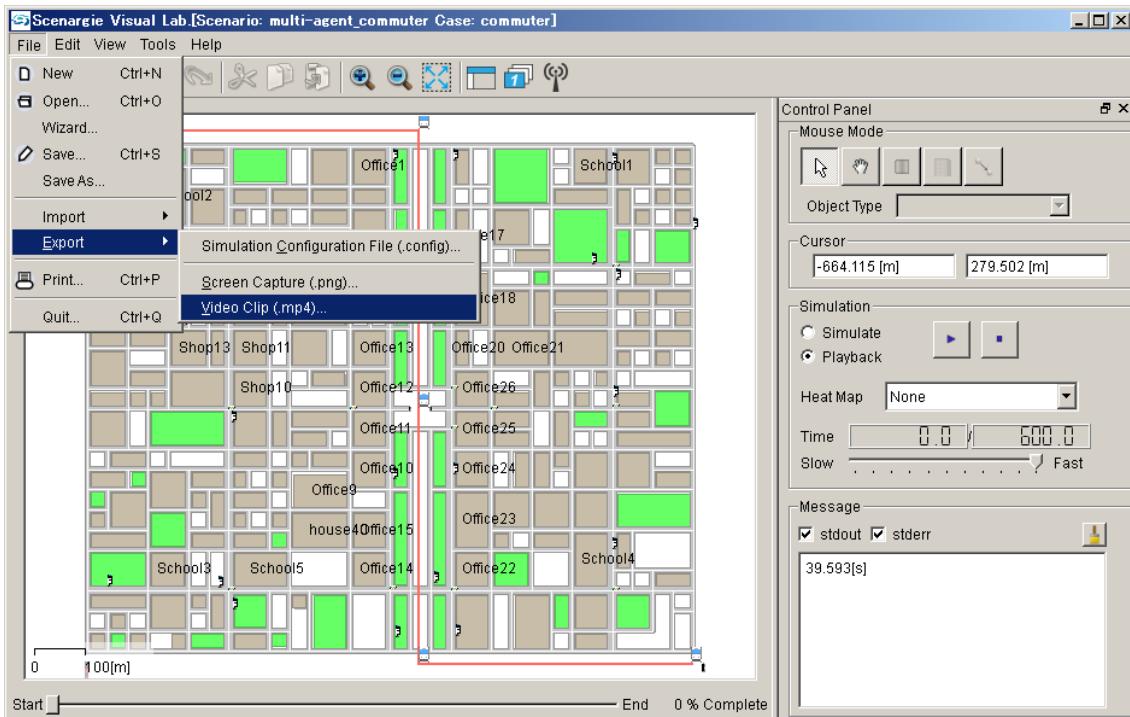




Settings の Event がイベントの種類を示す。トレース表示を行いたいイベントについて Inside または Border を選択し、色を決定する。色は値の範囲とそれに対応する色で設定可能である。

5.2. ビデオクリップの作成

[File]->[Export]->[Video Clip]によりプレイバックを mp4 形式のビデオクリップに出力することが可能である。

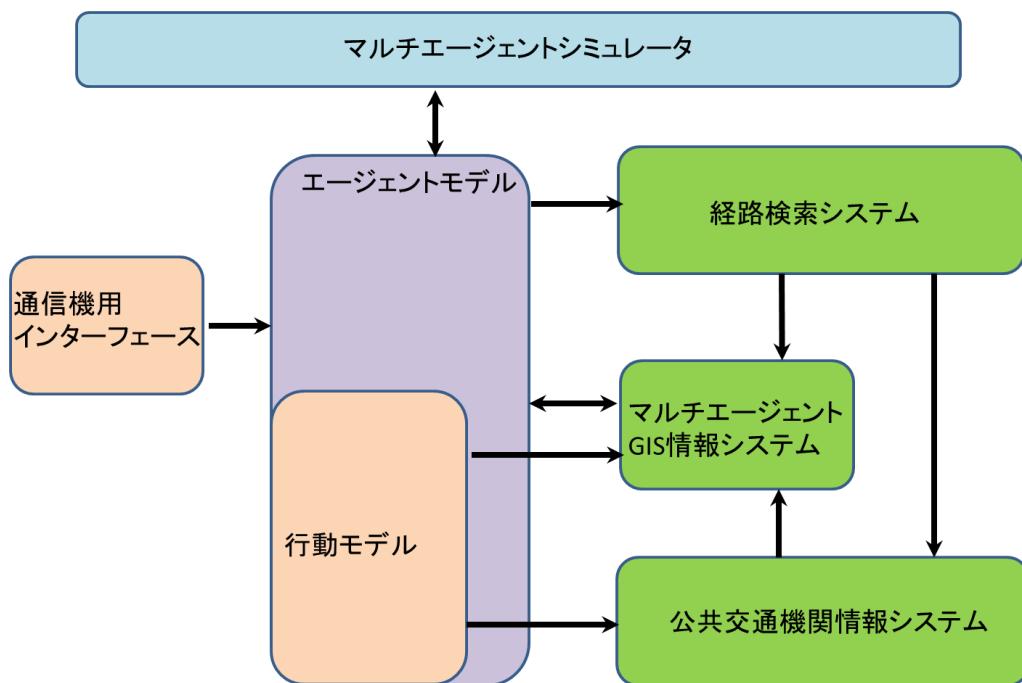


6. 機能構成

マルチエージェントシミュレーション機能は以下の機能で構成される。

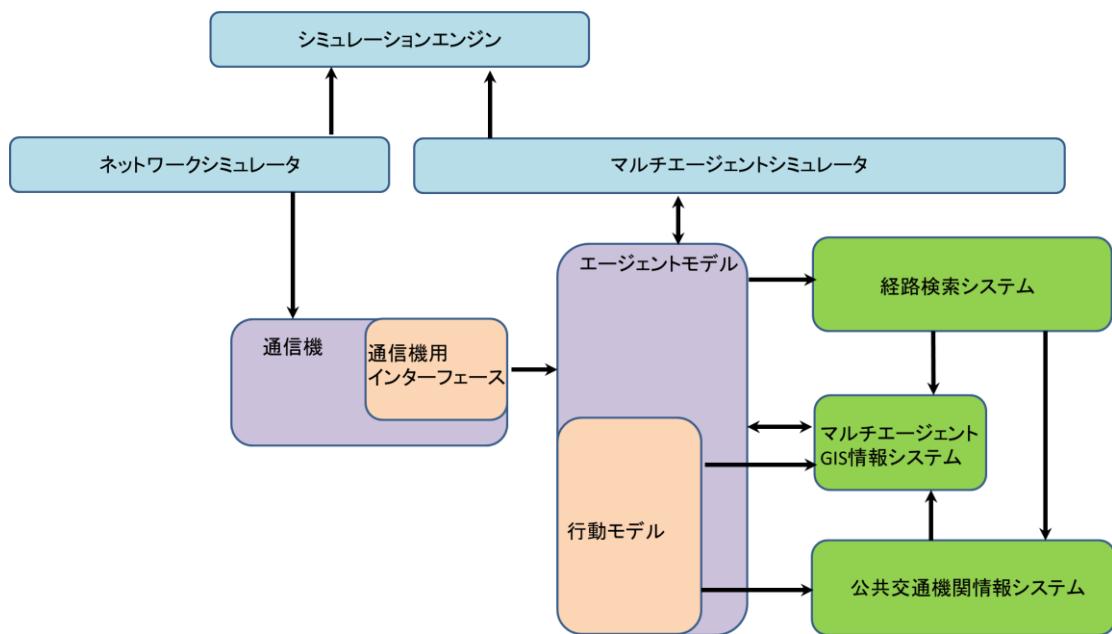
- マルチエージェントシミュレータ
- エージェントモデル
- 行動モデル
- マルチエージェント GIS 情報システム
- 経路検索システム
- 公共交通機関情報システム
- 通信機インターフェース

各機能間の関係を図に示す。



マルチエージェントシミュレータ、経路検索システム、マルチエージェント GIS 情報システム、公共交通機関情報はシミュレーション中でグローバルに一つ存在し、エージェントモデルとそのエージェントの行動モデルは、シミュレーション中に存在するエージェントの数分存在する。

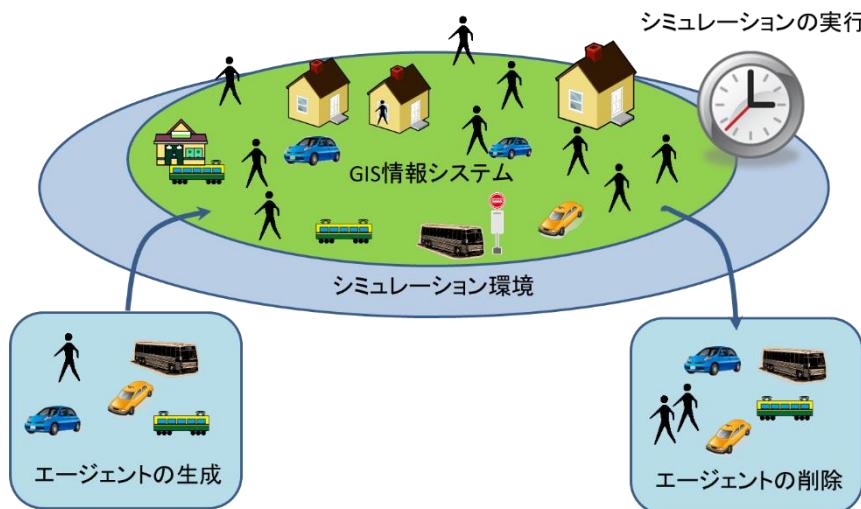
さらに、通信機と連携してシミュレーションを行う場合の機能構成を図に示す。



マルチエージェントシミュレータとネットワークシミュレータは独立してシミュレーションを実施し、同期時刻のタイミングで情報の同期を行う。ネットワークシミュレータ内の通信機がエージェントと関連付けられる場合、通信機用インターフェースを介してデータのやりとりを行う。

6.1. マルチエージェントシミュレータ

マルチエージェントのシミュレーションを行うシミュレータであり、時系列に沿ったエージェントの行動シミュレーションを行う機能である。シミュレーションの機能は大きく分けてエージェントの生成/削除、シミュレーションの実行から構成される。



6.1.1. エージェントの生成/削除

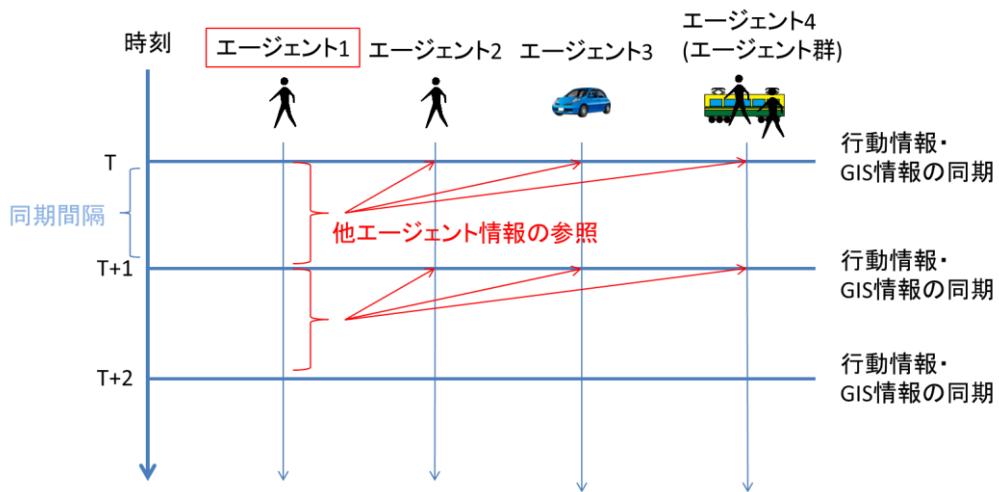
Human、PrivateCar、Bus、Train、Taxi のエージェントを生成/削除する。エージェントの行動が開始する際、シミュレーション環境上にエージェントの生成を行い、エージェントの行動が全て終了した際に、エージェントの削除を行う。エージェントの生成/削除のタイミングは、あらかじめ定めたシミュレーションの同期間隔の単位で実施する。

生成され、その後削除されたエージェントは、再びシミュレーション中に現れる事はない。また、同一のエージェント(同一の識別子のエージェント)がシミュレーション環境に複数存在することはない。

6.1.2. シミュレーションの実行

時系列に従って、同期間隔でシミュレーションの実行を行う。ここで、シミュレーションの実行とはエージェントの行動の実行と同義である。マルチエージェントシミュレータは、シミュレーション環境中に存在する各エージェントに関して、同時間隔の時間分の行動の実行を行い、同期間隔毎に、エージェント間での行動情報、GIS 情報の同期を行う。

行動の実行中に他のエージェントの情報の参照が必要である場合、最後に行動情報・GIS 情報の同期を行った時刻のエージェントの情報を利用する。



例えば図のようにエージェント1～4が存在し、時刻TからT+1までのエージェント1の行動のシミュレーションを実行する場合、エージェント1が参照可能なエージェント2～4の情報は時刻Tでのエージェント2～4の情報である。エージェント1の自分自身に対する情報は時刻Tから時刻T+1への行動について、動的な瞬時値を参照する。

6.2. エージェントモデル

シミュレーションに参加可能なエージェントの種別として、Human、PrivateCar、Bus、Train、Taxi を定義している。各エージェントはプロファイルと行動を保持し、自身の判断または事前に決定されたエージェントタイムテーブル設定に従って行動を行う。

6.2.1. エージェント種別

エージェントの種別を解説する。

- Human エージェント

人の行動を模擬するエージェントであり、マルチエージェントシミュレーションでの基盤を構成するエージェントである。Human エージェント一つが一人の人に対応し、行動モデルの変化により、道路の歩行、自家用車への乗車、バス・電車への乗車、タクシーへの乗車といった行動を行う。

Human エージェントの行動は他の Human エージェント、PrivateCar エージェント、Bus エージェント、Taxi エージェントに対して影響を与える。

- PrivateCar エージェント

自家用車を模擬するエージェントであり、Human エージェントが車両に乗り込むことで、道路上を移動する。

道路上を移動する際は道路上の他の車両に追従して移動し、道路を歩行する Human エージェント、他の PrivateCar エージェント、Bus エージェント、Taxi エージェントに対して影響を与える。

- Bus エージェント

バスを模擬するエージェントであり、あらかじめ定められたバスのスケジュールに従って、道路上を走行しバス停間の移動を行う。バス停で停車している場合、バスを待っている Human エージェントを定員数まで乗せることが可能である。尚、道路の混雑状況によってはあらかじめ定められたスケジュールよりも遅れて走行を行う場合がある。

移動する際は道路上の他の車両に追従して移動し、道路を歩行する Human エージェント、他の Bus エージェント、PrivateCar エージェント、Taxi エージェントに対して影響を与える。

- Train エージェント

電車を模擬するエージェントであり、あらかじめ定められた電車のスケジュールに従って、線路上を走行し駅間の移動を行う。駅で停車している場合、電車を待っている Human エージェントを定員数まで乗せることが可能である。シミュレーション中に線路が利用不可になった場合を除いて、Train エージェントは必ずスケジュールに従って移動を行う。

移動する際は他のエージェントの挙動に関係なく線路上を等速で移動する。

- Taxi エージェント

タクシーを模擬するエージェントであり、Human エージェントがタクシーを利用する場合、タクシーが配置されている GIS オブジェクトから発生する。Taxi エージェントは発生地点からタクシーを待っている Human エージェントがいる場所まで移動し、Human エージェントを乗せ目的地まで移動し Human エージェントを車両から降ろす。空車になった Taxi エージェントは自身が発生地点に戻り次のタクシーの利用を待つ。

道路上を移動する際は道路上の他の車両に追従して移動を行い、道路を歩行する Human エージェント、他の Taxi エージェント、PrivateCar エージェント、Bus エージェントに対して影響を与える。

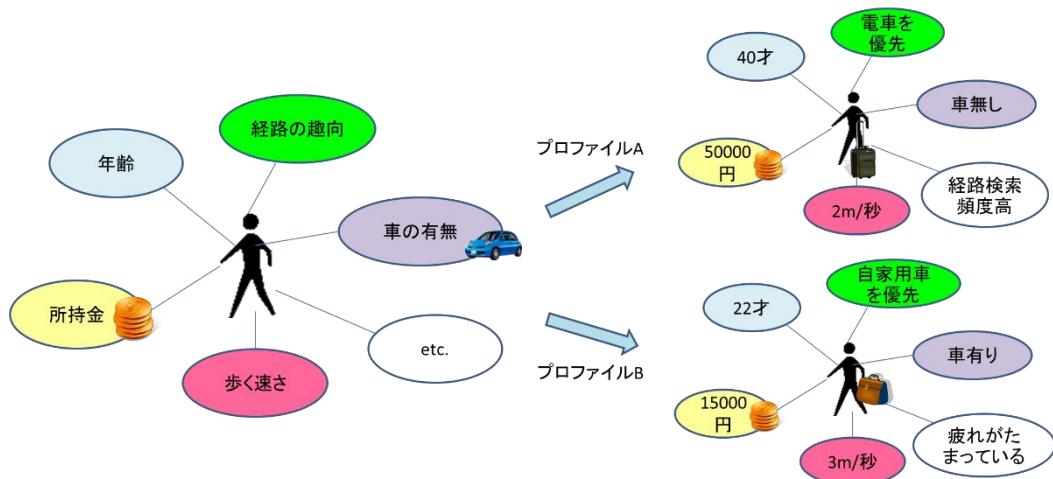
エージェント種別ごとに、そのエージェントの行動が他エージェントの行動に直接的に影響を与えるか否かの関係を図に示す。

他エージェント に対しての影響	Human 	Private Car 	Bus 	Train 	Taxi
Human 	○	○	○	-	○
PrivateCar 	○	○	○	-	○
Bus 	○	○	○	-	○
Train 	○	-	-	-	-
Taxi 	○	○	○	-	○

各エージェントは入力されたプロファイルと行動に従って、移動と状態の更新を行う。

6.2.2. エージェントプロファイル

各エージェントには、属性や嗜好や状態などのプロファイルを割り当てることが可能である。プロファイル内の各項目はシミュレーション中に値を変更することも可能である。



プロファイルは何種類でも定義が可能である。また、プロファイルに含まれる項目は任意であり、マルチエージェントシミュレーションの目的に応じて何項目でも定義が可能である。プロファイルに含まれる項目の中で、行動モデルに影響を与えるプロファイル項目や汎用的なプロファイル項目は事前に定義済みである。

プロファイルとエージェントは一对多の関係で関連付けが可能で、用途に応じてエージェント毎に個々にプロファイルを設定することも、複数のエージェントに対して共通のプロファイルを設定すること也可能である。プロファイルの値には定数だけではなく、分布を設定することも可能である。分布を設定した場合は各エージェントがそれぞれ疑似乱数を生成し、プロファイルの値として指定の分布に従った値を利用する。

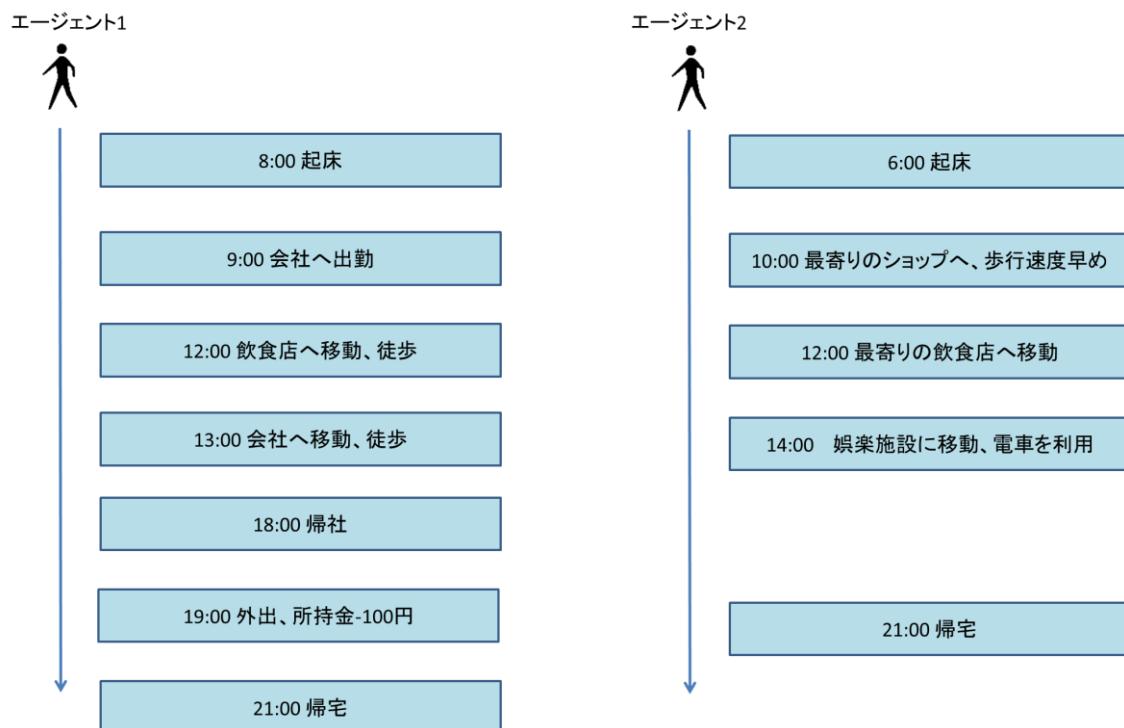
定義済みプロファイルを示す。

- 年齢
- 歩行速度
- 自転車速度
- 車両の最大速度
- エージェントのユーティリティ
- 経路の再計算トリガ
- 公共交通機関に乗り遅れた場合の経路再計算確率
- 車両のガソリンコスト
- バスの乗り込み時間
- ドライバの歩行者認識確率
- ドライバの前方車両との時間間隔(ヘッドウェイ)

- ドライバの最小車間距離
- ドライバの最大加速度/減速度
- 車両の最大の減速度
- ドライバの加速度指数
- 車線変更による後方車両の減速度の閾値
- カーブでの最小速度
- ドライバの車線変更加速度閾値
- IDM モデル[1]における s1
- ドライバの交差点での譲り合いの際に他車両に対して譲歩する時間
- 譲歩時に待つ時間
- 経路計算時の道路幅の重み
- 災害フラグ(0:-、1:歩行行動時に道路全体を利用可能)災害状態の設定
- 経路再計算間隔
- 許容可能な車までの歩行距離
- 許容可能な駅・バス停までの歩行距離
- 車での最小の移動距離
- 最大経路候補数
- 人數
- 入り口での最大待ち時間
- 車の所有率
- 自転車の所有率
- 公共交通機関の乗り換え時間
- 公共交通機関のきっぷの種類

6.2.3.エージェント行動

各エージェントに対して、ユーザ行動を割り当てることが可能である。ユーザ行動は何項目でも割り当てることが可能である。エージェントは指定された行動順に従って状態の変更と移動行動を実施する。



利用可能な行動の種類は次の通りである。

- 初期位置を指定する
- 目的地を定めて移動する
 - 行動の開始時刻を指定する
 - 複数の目的地の中から最寄りの目的地を選択して移動する
 - 移動に利用する行動モデルを指定する
 - 公共交通機関の時刻を指定して経路を検索する
- プロファイルの値を変更する

6.3. 行動モデル

エージェントは目的地までの移動について、各自が判断した行動モデルに従って移動を行う。行動モデルには次の種類がある。

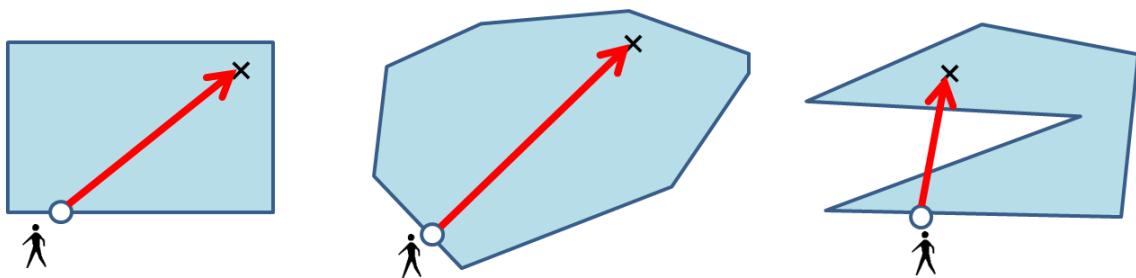
- 自由歩行
- 道路歩行
- 自転車
- 自家用車ドライバ
- バスドライバ
- バスゲスト
- 電車ドライバ
- 電車ゲスト
- タクシードライバ
- タクシーゲスト

各エージェントが利用可能な行動モデルを表に示す。

	Human 	Private Car 	Bus 	Train 	Taxi 
自由歩行	○	-	-	-	-
道路歩行	○	-	-	-	-
自転車	○	-	-	-	-
自家用車ドライバ	○	-	-	-	-
バスドライバ	-	-	○	-	-
バスゲスト	○	-	-	-	-
電車ドライバ	-	-	-	○	-
電車ゲスト	○	-	-	-	-
タクシードライバ	-	-	-	-	○
タクシーゲスト	○	-	-	-	-

6.3.1. 自由歩行

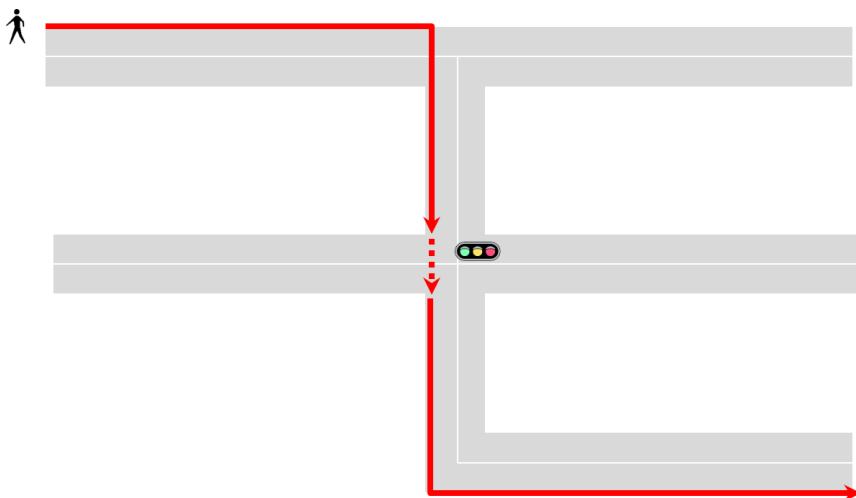
建物または公園内において、目的地まで直線移動する。自由歩行は他のエージェントの影響を受けず、また影響を与えない。



6.3.2. 道路歩行

車/歩行者共用の道路である場合、道路端を歩行する。歩行者専用の道路である場合は道路上の任意の場所を歩行する。信号の無い交差点の場合は、車より優先的に交差点を横断する。信号の有る交差点の場合、信号に従って交差点の横断を行う。車両専用の道路は歩行不可である。

移動予定の道路が通行不可となっていた場合は、経路を再検索して目的地まで移動可能な経路・行動モデルを利用する。



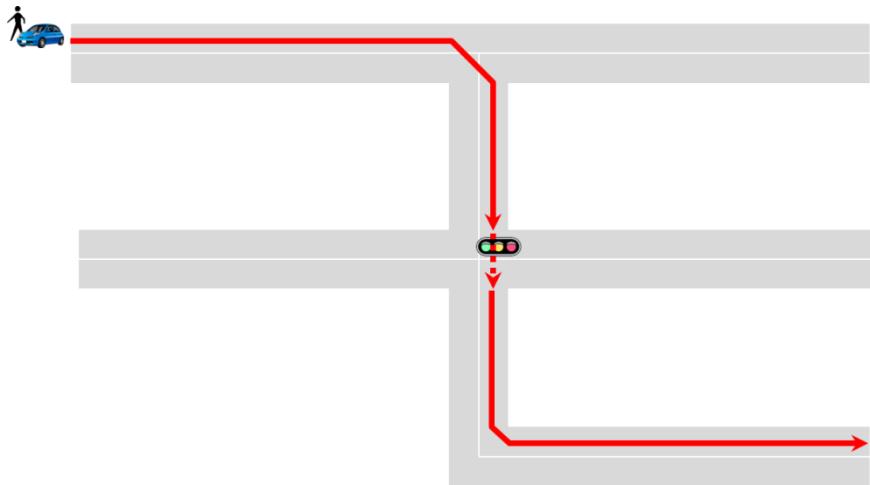
6.3.3. 自転車

道路歩行と同様の挙動をとる行動モデルである。移動速度には歩行速度ではなく自転車の速度を利用する。

6.3.4. 自家用車ドライバ

車/歩行者共用の道路または車専用の道路の車線の中央を走行する。自家用車のドライバは前を走行している車両の状態を見て安全に運転可能な車間距離を維持して運転を行う。[1]信号の無い交差点の場合は、歩行者または自転車が交差点を横断していなければ交差点を通過する。信号の有る交差点の場合、信号に従って交差点の通過を行う。

移動予定の道路が通行不可となっていた場合は、経路を再検索して目的地まで移動可能な経路を利用する。



6.3.5.バスドライバ

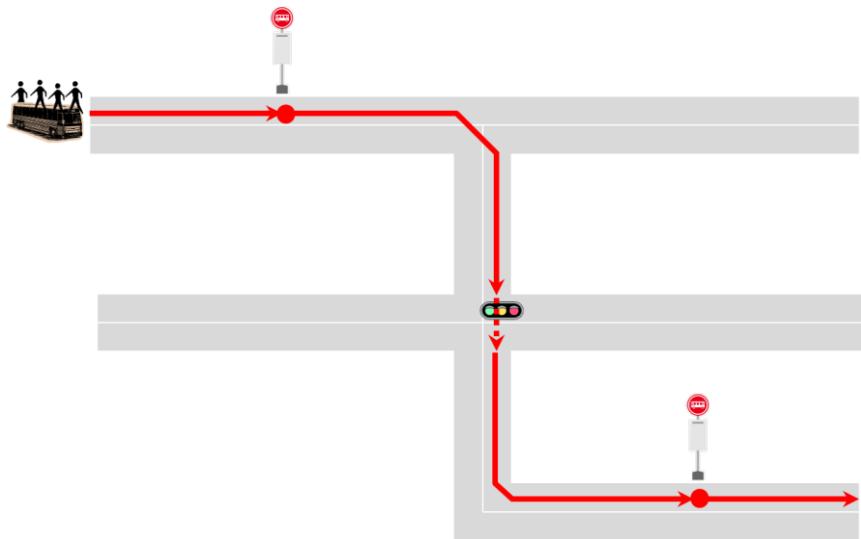
バスドライバは運行スケジュールに従ってバス停間をバスで移動する。バスの運転は自家用車ドライバと同一の制御を行う。バス停で停車している際、他の乗用車はバスを追い抜き可能とする。

6.3.6.バスゲスト

道路上に指定されているバス停で乗車予定のバスが到着するのを待つ。バスが到着した場合は先に待っている人から順にに乗車する。乗車には各 Human エージェントに設定した乗車時間を要する。乗車後は Human エージェント毎に車両内のランダムな位置まで歩行し、降車予定のバス停にバスが到着するのを待つ。バスが降車予定のバス停に到着した場合、降車する。乗り換えが必要な場合は次の駅またはバス停まで歩行し、さらに次のバスの到着を待つ。

利用を予定していたバスより先に同一の運行ラインのバスが到着した場合は、利用を予定していたバスではなく先に到着したバスに乗車する。

利用を予定したバスが満員になって乗車出来なかった場合や、バス停に到着したが利用を予定していたバスが既に出発していて乗車出来なかった場合は、経路を再計算し別の経路を利用可能であればそちらの経路を利用する。



6.3.7. 電車ドライバ

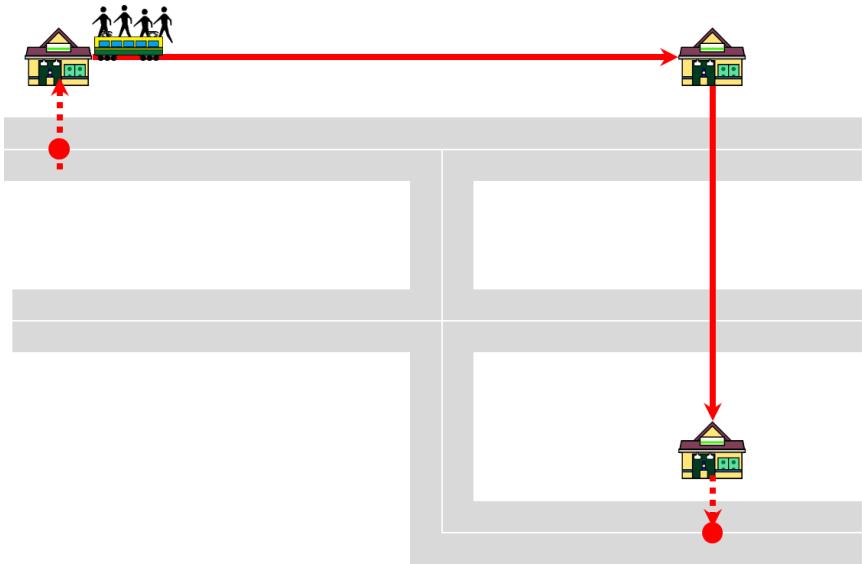
電車の運行スケジュールに従って電車を運行する。

6.3.8. 電車ゲスト

経路に定めた駅で乗車予定の電車が到着するのを待つ。電車が到着した場合は乗車に要する時間は0として先に待っている人から順に乗車する。乗車後は Human エージェント毎に車両内のランダムな位置まで歩行し、降車予定の駅に電車が到着するのを待つ。電車が降車予定の駅に到着した場合、降車する。乗り換えが必要な場合は次の駅または駅まで歩行し、さらに次の電車の到着を待つ。

利用を予定していた電車より先に同一の運行ラインの電車が到着した場合は、利用を予定していた電車ではなく先に到着した電車に乗車する。

利用を予定した電車が満員になって乗車出来なかった場合や、駅に到着したが利用を予定していた電車が既に出発していて乗車出来なかった場合は、経路を再計算し別の経路を利用可能であればそちらの経路を利用する。

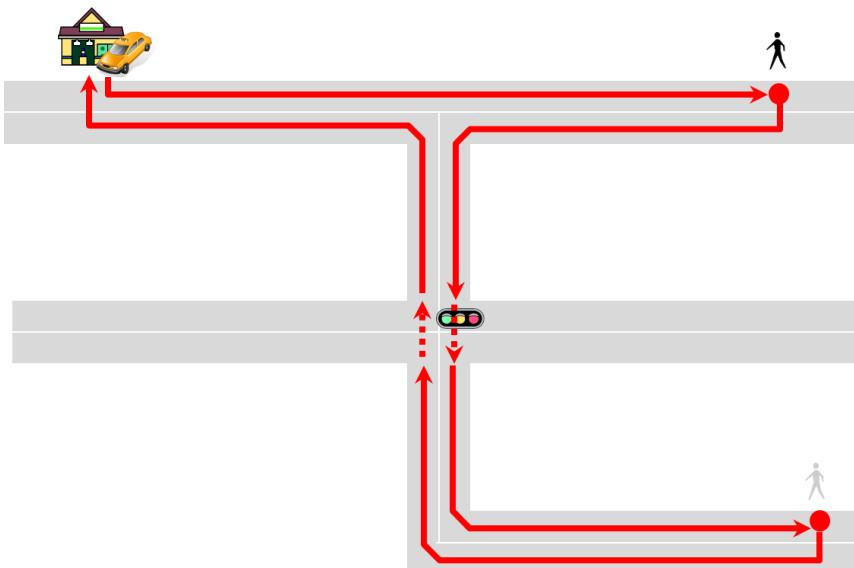


6.3.9. タクシードライバ

タクシードライバは利用者までタクシーで移動し、利用者を目的地まで運ぶ。車の運転は自家用車ドライバと同一の制御を行う。

6.3.10. タクシーゲスト

停止してタクシーが来るのを待ち、タクシーが到着したらタクシーに乗車する。タクシーが目的地に着いたら降車する。



6.4. マルチエージェント GIS 情報システム

マルチエージェントシミュレーションではシミュレーション内の様々な GIS 情報を利用しエージェントの行動シミュレーションを行う。マルチエージェントシミュレーションで利用する GIS 情報には次の種類がある。

- 道路
- 交差点
- 信号
- バス停
- 建物
- 公園
- 線路
- 駅
- 入り口
- エリア
- POI

各エージェントの行動モデルについて、直接的に利用を行う GIS オブジェクトを表に示す。

エージェント	行動モデル	道路	交差点	信号	バス停	建物	公園	線路	駅	入り口	エリア	POI
 Human	自由歩行	-	-	-	-	○	○	-	-	○	-	○
	道路歩行	○	○	○	-	-	-	-	-	○	-	-
	自転車	○	○	○	-	-	-	-	-	-	-	-
	自家用車ドライバ	○	○	○	-	-	-	-	-	-	-	-
	バスゲスト	-	-	-	○	-	-	-	-	○	-	-
	電車ゲスト	-	-	-	-	-	-	-	○	○	-	-
	タクシーゲスト	-	-	-	-	-	-	-	-	-	-	-
 PrivateCar	-	-	-	-	-	-	-	-	-	-	-	-
 Bus	バスドライバ	○	○	○	○	-	-	-	-	-	-	-
 Train	電車ドライバ	-	-	-	-	-	-	○	○	-	-	-
 Taxi	タクシードライバ	○	○	○	-	-	-	-	○	-	-	-

※エリアは GIS オブジェクトの集合を定義する概念的な GIS オブジェクトである。

6.4.1. 道路

エージェントは目的地までの移動の際に、道路上を移動することが可能である。道路の接続情報はエージェントの経路計算に利用される。

道路は有効状態/無効状態の指定が可能で、初期状態では有効状態となっている。無効状態となった場合利用不可となり、利用不可となった道路を通過しようとするエージェントは経路の再計算を行う。道路歩行と自転車の行動モデルで道路に進入するエージェントに対して、道路は定員数を保持している。定員以上のエージェントは同時に道路に進入することが出来ない。進入しようとした道路が定員となっている場合は、道路端で待機し定員に空きが出るのを待つ。

6.4.2. 交差点

交差点は、複数の道路を接続する点で信号を利用可能である。

6.4.3. 信号

交差点上に配置される車両用の信号であり、信号の配置された交差点を通過または横断しようとするエージェントの行動に影響を与える。信号は青色、黄色、赤色の状態があり、車両は青色または黄色の場合にのみ通過可能である。歩行者の横断は歩行する道路の信号が赤色の場合に可能である。有効状態/無効状態の指定が可能で、初期状態では有効状態となっている。無効状態となった場合表示の故障となり、交差点を通過しようとするエージェントは信号が無いものとして移動を行う。

6.4.4. バス停

バス停は、道路上でバスが停車可能な点となる。他の車両は、バス停に停車しているバスの追い抜きが可能である。

6.4.5. 建物

建物は Human エージェントが内部に進入が可能な施設で、エージェントの行動の初期位置または目的地として利用可能である。建物には定員があり、定員以上の人数は同時に建物に入ることが出来ない。定員に達して建物に入ることが出来なかった場合は現在の行動をスキップし、行動リストの次の行動を行う。ただし目的地としてグループ化された目的地が指定されている場合には、グループ化された目的地のうちの他の目的地に向かって移動する。

6.4.6. 公園

公園は Human エージェントが内部に進入が可能なエリアで、エージェントの行動の初期位置または目的地として利用可能である。公園には定員があり、定員以上の人数は同時に公園に入ることが出来ない。定員に達して建物に入ることが出来なかった場合は現在の行動をスキップし、行動リストの次の行

動を行う。ただし目的地としてグループ化された目的地が指定されている場合には、グループ化された目的地のうちの他の目的地に向かって移動する。

6.4.7. 線路

駅の接続を示す線である。電車は線路上のみを移動可能である。

有効状態/無効状態の指定が可能で、初期状態では有効状態となっている。無効状態となった場合。その線路を利用する全ての路線の電車が停止する。

6.4.8. 駅

駅は、電車が停車可能な点となる。駅には定員があり、定員以上の人数は同時に駅に入ることが出来ない。定員に達して駅に入ることが出来なかった場合は駅の入り口で待機し定員に空きが出るのを待つ。

6.4.9. 入り口

入り口は、道路と建物、公園、駅をつなぐ点である。Human エージェントは入り口を通って出入りを行う。建物または公園に対して追加した入り口には、流入レートを設定することで入り口待ちによる人流の制御が可能である。

6.4.10. エリア

エリアは GIS オブジェクトの集合を定義した概念である。エリアで指定した領域内の全ての建物はそのエリアに所属するものとして、エージェントの行動の初期位置または目的地として利用可能である。

6.4.11. POI

POI は Human エージェントが進入が可能な点で、エージェントの行動の初期位置または目的地として利用可能である。POI には定員があり、定員以上の人数は同時に入ることが出来ない。定員に達して POI に入ることが出来なかった場合は現在の行動をスキップし、行動リストの次の行動を行う。ただし目的地としてグループ化された目的地が指定されている場合には、グループ化された目的地のうちの他の目的地に向かって移動する。

POI が建物(または公園)内に配置されている場合、その建物への出入りは入り口を経由し、その後入り口から POI への移動(または POI から入り口への移動)を行なう。

6.5. 経路検索システム

各エージェント毎に、エージェントのプロファイルから利用可能な交通手段を対象として、経路の検索を行う。経路検索システムでは次の経路を経路候補として選択する。

- 公共交通機関経路(全検索)
- 公共交通機関経路(制限付き)
- 道路歩行経路
- 自転車経路
- 自家用車経路
- タクシー経路

6.5.1.公共交通機関経路(全検索)

全ての駅とバス停の各ペアに対して、エージェント行動で指定した ArrivalTime と DepartureTime を参考として全ての経路を検索する。

経路の検索は乗り換え回数が少ない順、運賃が安い順、移動時間が短い順についてそれぞれ検索を行い、経路候補に含める。

6.5.2.公共交通機関経路(制限付き)

始点の最寄りの駅またはバス停(合計で最大 3 個)と、終点の最寄りの駅またはバス停(最大 3 個)の各ペアに対して、エージェント行動で指定した ArrivalTime と DepartureTime を参考として、最大 10 回の乗り換えで経路を検索する。

経路の検索は乗り換え回数が少ない順、運賃が安い順、移動時間が短い順についてそれぞれ検索を行い、経路候補に含める。

6.5.3.道路歩行経路

始点から終点までの歩行者・自転車が通行可能な道路上の全ての経路について、歩行者または自転車による道路の混雑度を考慮して各道路の通過時間を算出する。終点に到達するまでの道路の通過時間を合計した値が最も短い経路を利用する。つまり、道路上に他の歩行者・自転車が存在しない場合は始点と終点を最短距離で結ぶ道路上を移動し、道路上に歩行者・自転車が存在し道路毎に混雑度が異なっている場合は、移動距離が短くなるべく混雑していない経路を利用する。

6.5.4.自転車経路

道路歩行経路と同一のアルゴリズムで経路を検索する。ただし、移動速度は自転車の移動速度とする。

6.5.5.自家用車経路

始点から終点までの車両が通行可能な道路上の全ての経路について、車両による道路の混雑度を考慮して各道路の通過時間を算出する。終点に到達するまでの道路の通過時間を合計した値が最も短い経路を利用する。つまり、道路上に車両が存在しない場合は始点と終点を最短距離で結ぶ道路上を移動し、道路上に車両が存在し道路毎に混雑度が異なっている場合は、移動距離が短くなるべく混雑していない経路を利用する。

6.5.6.タクシー経路

自家用車と同一のアルゴリズムで経路を検索する。ただし、移動速度等の経路検索に利用するプロファイルはタクシーのプロファイルである。

6.6. 公共交通機関情報システム

交通機関情報システムは、マルチエージェントシミュレーションにおいて電車・バスのスケジュールを管理する。交通機関情報システムにより、実環境と同様の電車路線・バス路線と路線ごとの運行スケジュールを定義することが可能である。

××線

	スケジュール1	スケジュール2	スケジュール3	スケジュール4
東京	10:00	10:05	10:10	10:15
銀座	10:03	10:08	10:13	10:18
品川	10:08	10:13	10:18	10:23
六本木	10:14	10:19	10:24	10:29

○○線

	スケジュール1	スケジュール2	スケジュール3	スケジュール4
新宿	10:00	10:06	10:12	10:18
江戸川橋	10:05	10:11	10:17	10:23
駒込	10:09	10:15	10:21	10:27
日暮里	10:12	10:18	10:24	10:30

運行スケジュールを管理するために、交通機関情報システムは次の情報を保持する。

- 駅、バス停の接続/乗り換え情報の管理
- 電車またはバスの路線
- 車両単位での運行スケジュール
- 車両の大きさと最大の乗客数

交通機関情報システムはスケジュールで定められた電車・バスの発生時刻に合わせて電車・バスの生成/消滅を行い、バスに遅延がある場合は、運行ライン単位でバスの遅延情報を保持する。

電車・バスを利用した経路検索の場合にも交通機関情報システムを利用する。経路検索システムは、出発事項および到着時刻で条件を設定して経路を検索可能である。また、経路の検索は乗り換え回数、運賃、移動時間をメトリックとする。

6.7. 通信機インターフェース

通信機インターフェースはエージェントに通信機を関連付ける場合に利用する。通信機インターフェースは通信機に次の機能を提供する。

- 通信機がエージェントと関連付けられているかの判定
- エージェントが有効かの判定
- 位置情報の取得
- 目的地の変更
- 進入が不可能な目的地の追加
- 経路の再計算
- 行き止まり地点に到達したことの通知
- 目的地に到達したことの通知
- 目的地に進入したことの通知
- 目的地に進入不可であることの通知

7. シナリオ構成

シナリオ作成について解説する。

7.1. 構成ファイル

シナリオは次のファイルによって構成される。シナリオの編集を行う際にはこれらのファイルの編集を行う。

シミュレーション実行用ファイル

ファイル	内容
コンフィグレーションファイル (xxx.config)	シミュレーション設定
エージェントプロファイル定義ファイル (AgentProfiles.txt)	エージェントのプロファイル定義
エージェント行動定義ファイル (AgentBehaviors.txt)	エージェントの行動定義
エージェントタイムテーブル設定ファイル (AgentTimeTable.txt)	公共交通機関のスケジュール
統計情報ファイル (xxx.stat)	統計情報出力
トレースファイル (xxx.trace.bin)	位置情報、混雑度、満足度等のトレース出力
道路ファイル(road.shp)	道路情報
信号ファイル(trafficlight.shp)	信号情報
バス停ファイル(busstop.shp)	バス停情報
建物ファイル(building.shp)	建物情報
公園ファイル(park.shp)	公園情報
線路ファイル(railroad.shp)	線路情報
駅ファイル(station.shp)	駅情報
入り口ファイル(entrance.shp)	入り口情報
エリアファイル(area.shp)	エリア情報
交差点ファイル(intersection.shp)	交差点情報
信号パターン定義ファイル(TrafficlightPattern.txt)	信号パターン定義ファイル
POI定義ファイル(poi.shp)	POI情報

Visual Lab 実行用シナリオファイル

ファイル	内容
ケース名.case	シナリオ情報
ケース名.property	オブジェクトのプロパティ情報
ケース名_Human.layer	人レイヤ情報

ケース名_Area.layer	エリアレイヤ情報
ケース名_Building.layer	建物レイヤ情報
ケース名_Bus.layer	バスレイヤ情報
ケース名_BusStop.layer	バス停レイヤ情報
ケース名_Channel.layer	チャネルレイヤ情報
ケース名_Global.layer	グローバルレイヤ情報
ケース名_Intersection.layer	交差点レイヤ情報
ケース名_Park.layer	公園レイヤ情報
ケース名_Railroad.layer	線路レイヤ情報
ケース名_Road.layer	道路レイヤ情報
ケース名_TrafficLight.layer	信号レイヤ情報
ケース名_Station.layer	駅レイヤ情報
ケース名_Eintrancve.layer	入り口レイヤ情報
ケース名_Taxi.layer	タクシーレイヤ情報
ケース名_Train.layer	電車レイヤ情報
ケース名_PrivateCar.layer	車レイヤ情報
ケース名_Wall.layer	壁レイヤ情報
ケース名_POI.layer	POI レイヤ情報

7.2. コンフィグレーションファイル

シミュレーション入力用のシナリオファイルで、シミュレーションのシードやシミュレーション時間といった各種プロパティの定義が可能である。

- ファイル構文
 - 一行につき一プロパティ記述する
 - 文字コード・改行コードはシミュレーション実行環境に合わせる。
 - 空行はスキップされる
 - 「#」で開始する行はコメント行としてスキップされる

7.2.1. プロパティ

コンフィグレーションファイルで設定可能なプロパティー一覧を示す。

パラメータ	プロパティ名	値	参考値
モビリティ乱数シー ド	mobility-seed	整数	123
シミュレーション時 間	simulation-time	時間	100s
シミュレーションの ステップ間隔	time-step-event-synchronization-step	時間	1s
トレース出力ファイ ル	trace-output-file	ファイル名	test.trace
トレース出力モード	trace-output-mode	Binary/Text	binary
統計情報出力ファ イル	statistics-output-file	ファイル名	test.stat
値無し統計値の出 力設定	statistics-output-for-no-data	true/false	true
シミュレーション進 捲状況の出力	progress-sim-time-output-interval-perce nts	実数	10
車道の進行方向	gis-road-traffic-direction	left/right	left
道路の分割	gis-los-break-down-curved-road-into-str aight-roads	true/false	false
建物への最低限 の入り口の数	gis-number-entrances-to-building	整数	1
駅への最低限の 入り口の数	gis-number-entrances-to-station	整数	1

バス停への最低限の接続の数	gis-number-entrances-to-busstop	整数	1
公園への最低限の入り口の数	gis-number-entrances-to-park	整数	1
交差点領域の作成	gis-road-set-intersection-margin	true/false	true
信号パターン定義ファイル	gis-trafficlight-pattern-definition-file	文字列	TrafficlightPattern.txt
エージェントプロファイルの定義ファイル	multiagent-profile-file	ファイル名	AgentProfiles.txt
エージェント行動の定義ファイル	multiagent-behavior-file	ファイル名	AgentBehaviors.txt
エージェントタイムテーブル設定ファイル	gis-public-vehicle-file	ファイル名	VehicleTable.txt
交通機関の開始オフセット	multiagent-start-time	時間	0
実行スレッド数	number-data-parallel-threads-for-multiagent	整数	1
プロファイルタイプ	multiagent-profile-type	プロファイル定義ファイルで定義される文字列	Default
行動タイプ	multiagent-behavior-type	行動定義ファイルで定義される文字列	Default
GIS(Shape)ファイルのパス	gis-objects-file-path	ディレクトリ名	shapes/
GIS(Shape)ファイルのファイル名	gis-object-files	ファイル名	road.shp
統計値設定	statistics-configuration-file	ファイル名	test.statconfig
トレース設定	trace-enabled-tags	文字列	Max Gis
GIS オブジェクトの無効化	gisobject-disable-time	時刻	inf_time
GIS オブジェクトの	gisobject-enable-time	時刻	inf_time

有効化			
-----	--	--	--

7.2.2. 統計値設定

統計値設定と出力される統計値を示す。

統計対象	統計値	統計名	統計種別
エージェント	ユーティリティ1	Mas_Utility1	変動値
	ユーティリティ2	Mas_Utility2	変動値
	総移動距離	Mas_TravelDistance	変動値
	総移動時間	Mas_Traveltime	変動値
	経路計算回数	Mas_NoRoutes	積算値
	経路無し回数	Mas_RouteCalculationtimes	積算値
道路/建物/公園/駅/ バス停	混雑度	Congestion	積算値
建物/公園	滞在人数	Population	積算値

混雑度はエージェント数[人]/オブジェクトのポリゴン面積[m^2]で算出する。(※バス停のみ面積を1[m^2]として算出する)

7.2.3. トレース設定

トレース設定と出力されるトレース情報を示す。

トレース対象	トレース設定タグ	トレース値	イベント名	値
エージェント	Mas	ユーティリティ1	Utility1	実数
		ユーティリティ2	Utility2	実数
		総移動距離	TravelDistance	実数
		総移動時間	TravelTime	実数
		目的地変更回数	DestinationChangeCount	実数
		通信による目的地変更回数	DestinationChangeCountByCommunication	
		目的地名	Destination	文字列
道路/建物/公園/ 駅/バス停	Mas	混雑度	Gis_Congestion	実数
道路/建物/公園/	Gis	オブジェクト有効/無	State	Enabled/

駅/線路/信号		効		Disabled
---------	--	---	--	----------

ファイル入力の例を示す。

```
#Instance general
#Component Simulation
seed = 123
simulation-time = 600.000000000
time-step-event-synchronization-step = 0.100000000
trace-output-mode = Text
trace-index-output = true
trace-output-file = simulation.trace
statistics-output-file = simulation.stat
statistics-output-for-no-data = true
network-terminate-sim-when-routing-fails = false
progress-sim-time-output-interval-percents = 5.000000000
enable-unused-parameter-warnings = true

#Component GIS
gis-road-driving-side = left
gis-los-break-down-cureved-road-into-straight-roads = false
gis-number-entrances-to-building = 4
gis-number-entrances-to-station = 4
gis-number-entrances-to-busstop = 4
gis-number-entrances-to-park = 4
gis-road-set-intersection-margin = true

#Component Antenna/Propagation
number-data-parallel-threads-for-propagation = 1

#Component MultiAgent
multiagent-profile-file = simulation_AgentProfiles.txt
multiagent-behavior-file = simulation_AgentBehaviors.txt
gis-public-vehicle-file = simulation_VehicleTimeTable.txt
multiagent-start-time = 0.000000000
number-data-parallel-threads-for-multiagent = 1
```

```

multiagent-navigation-system-update-interval = 60.000000000

[2161-2195] is-member-of = TrainObjectType
[2266-2367] is-member-of = BusObjectType
[2196-2265] is-member-of = TaxiObjectType
[2001-2160] is-member-of = PrivateCarObjectType
[1-2000] is-member-of = HumanObjectType
#Component SimulationObject
[1-100] trace-enabled-tags = Mas
[1-2367,100000001-100001001,101000001-101000704,102000001-102000162,103000001
-103000002,104000001-104000005,105000001-105000016,105500001-105500021,106500
001-106500041] trace-start-time = 0.000000000

#Component GisObject
[100000001-100001001,102000001-102000162,103000001-103000002,104000001-104000
005,105000001-105000016,106500001-106500041] gisObject-disable-time = inf_time
[100000001-100001001,102000001-102000162,103000001-103000002,104000001-104000
005,105000001-105000016,106500001-106500041] gisObject-enable-time = inf_time

#Component Agent
[1-500] multiagent-profile-type = Student
[2001-2160] multiagent-profile-type = PrivateCar
[2161-2195] multiagent-profile-type = Train
[501-2000] multiagent-profile-type = OfficeWorker
[2266-2367] multiagent-profile-type = Bus
[2196-2265] multiagent-profile-type = Taxi
[2001-2367] multiagent-behavior-type = None
[1-500] multiagent-behavior-type = PupilBehavior
[501-1250] multiagent-behavior-type = OfficeWorkerBehavior1
[1251-2000] multiagent-behavior-type = OfficeWorkerBehavior2

gis-object-position-in-latlong-degree = false
gis-object-file-path = shapes/
gis-object-files = intersection.shp road.shp trafficlight.shp busstop.shp

```

```
building.shp railroad.shp station.shp park.shp
```

```
statistics-configuration-file = simulation.statconfig
```

7.3. エージェントプロファイル定義ファイル

エージェントプロファイル定義ファイルでは各エージェントが保持する属性を設定可能である。設定可能な項目を示す。

- プロファイルタイプ定義
- パラメータ設定
- 計算式の設定
- Utility
- 経路コスト

- ファイル構文
 - 一行につき一項目記述する
 - 文字コード・改行コードはシミュレーション実行環境に合わせる。
 - 空行はスキップされる
 - 「#」で開始する行はコメント行としてスキップされる

7.3.1. プロファイルタイプ定義

各プロファイルタイプに対して"ProfileType:"の予約語から記述を始める。"ProfileType"以降に記述されたパラメータ設定および計算式の設定は、次に新しい"ProfileType"を記述した行が表れるまで、そのプロファイルタイプの設定となる。

ProfileType:<プロファイルタイプ名>

7.3.2. パラメータ設定

各プロファイルタイプが保持する値の初期値を設定可能である。値は実数または文字列での指定を行う。実数値のパラメータは計算式での値の指定も可能である。パラメータ設定は次の形式で行う。

<パラメータ名> = <値 or 式>

各種パラメータは行動定義ファイル内で指定することでシミュレーション中の任意の時刻や行動の前後のタイミングで値を変更可能である。

予約済みのパラメータを以下に示す。

パラメータ名	値	補足	設定省略時の初期値

Age	実数	年齢	0
WalkSpeed	実数	歩行速度[m/s]	0
BicycleSpeed	実数	自転車速度[m/s]	0
VehicleSpeed	実数	車両の最大速度[m/s]	60 / 3.6
Utility1	実数	ユーティリティ1	0
Utility2	実数	ユーティリティ2	0
RecalcIntervalForLastViaPointDelay	実数	経路再計算トリガ(最後に訪れた via ポイントでの遅延に対するトリガ)	60*10
RecalcIntervalForNextViaPointDelay	実数	経路再計算トリガ(次の via ポイントに対しての遅延に対するトリガ)	60*10
RecalcIntervalForDestinationDelay	実数	経路再計算トリガ(目的地への到着予定時刻に対する現在の経路の遅延に対するトリガ)	60*10
RecalcIntervalForVehicleDelay	実数	経路再計算トリガ(乗車予定の路線の遅れに対するトリガ)	60*10
RecalcThresholdForCongestion	実数	経路再計算トリガ(混雑度に対するトリガ)	0.1
RecalcThresholdForUtility1	実数	経路再計算トリガ(Utility1に対するトリガ)	0
RecalcThresholdForUtility2	実数	経路再計算トリガ(Utility2に対するトリガ)	0
MinRouteRecalcInterval	実数	経路再計算トリガの最小間隔	0
RouteRecalcProbability	実数	経路再計算トリガを満たした場合の経路検索確率	0
RecalcProbWhenMissingAVehicle	実数	公共交通車両(電車、バス)に乗れなかった場合の経路検索確率	0
FuelConsumption	実数	ガソリンコスト	0
BoardingDelay	実数	バスの乗り込み時間[s]	0
PedestrianRecognitionProbability	実数	ドライバーが歩行者を認識する確率	1
TimeHeadway	実数	ドライバーの前方車両との時間間隔(ヘッドウェイ)[s]	1.5
MinVehicleGap	実数	ドライバーの最小車間距離[m]	3.0
MaxAcceleration	実数	ドライバーの最大加速度[m/s^2]	1.0

MaxDeceleration	実数	ドライバの最大減速度[m/s^2] (※減速度は負の値で指定する)	-3.0
MaxBrakingDeceleration	実数	車両の最大減速度[m/s^2]	-20
AccelerationExponent	実数	ドライバの加速度指数	4
SaveDeceleration	実数	車線変更による後方車両の減速度の閾値 [m/s^2]	-12
MinTurnSpeed	実数	カーブでの最小速度[m/s]	10
LaneChangeAccelerationThreshold	実数	ドライバの車線変更加速度閾値[m/s^2]	0.2
VelocityRatioGapDistance	実数	IDM モデル[1]における係数 s1 (通常の IDM モデルでは s1 の項は 0)	0
OtherVehicleEntranceTime	実数	交差点において、相手車両が交差点に進入しようとしていることを判断する時間距離 [s]	1
PassiveYieldTime	実数	交差点での譲り合いの際に他車両に対して消極的に譲歩する場合の速度差の閾値、最大の時間[s]	3.0
ActiveYieldTime	実数	交差点での譲り合いの際に他車両に対して積極的に譲歩する場合の速度差の閾値、最大の時間[s]	-3.0
YieldWaitingTime	実数	譲歩時間[s]	1
TravelDistance	実数	総移動距離[m]	0
TravelTime	実数	総移動時間[s]	0
RoadWidthWeight	実数	経路計算時の道路幅の重み	0
Disaster	実数	災害モード 災害モードが0の場合、歩行者は沿道移動し、1 の場合車両の有無に関係なく路上の任意の領域を移動する。	0
RouteRecalcInterval	実数	経路再計算間隔 0 以上を指定した場合、指定間隔毎に経路の再計算を行う。	0
WalkableDistanceToPrivateCar	実数	許容可能な自家用車までの歩行距離[m]	1000
WalkableDistanceToBusStopOrStation	実数	許容可能な駅・バス停までの歩行距離[m]	1000

MinDistanceToUseVehicle	実数	車・タクシーを利用する場合の最小の距離 [m]	0
MaxRouteCandidates	整数	料金、移動時間、乗り換え回数の各メトリックの経路について、検索する経路候補の最大数	3
NumberOfPeople	整数	人数[人]	1
MaxWaitingTimeAtEntrance	実数	入り口での最大待ち時間[s]	0
MaxWaitingTimeAtVehicleEntrance	実数	自家用車ドライバの入り口での最大待ち時間[s]	0
PrivateCarOwnership	実数	車の所有率	0
BicycleOwnership	実数	自転車の所有率	0
WalkSpeedAtTransfer	Normal Slow VerySlow WheelChair	公共交通機関の乗り換えの移動に要する時間 Normal: 3[m/s] Slow: 1[m/s] VerySlow, WheelChair: 0.5[m/s]	Normal
MaxWaitingTimeForTaxiAssignment	実数	タクシーの割り当てを待つ最大の時間	600
任意のパラメータ (※計算式で利用可能)	実数		0

予約された以外のパラメータについては、「_ (※アンダーバー)」から始まらないものであれば、自由に定義可能である。

車両の移動に関する TimeHeadway、MinVehicleGap、MaxAcceleration、MaxDeceleration、LaneChangeAccelerationThreshold、PassiveYieldTime、ActiveYieldTime については車両への乗車から降車まで固定値とする。これらの値を車両の乗車中に変更した場合、値の変更は次の車両に乗車するタイミングから値の反映が行われる。

式中で利用可能な演算子

演算子	返り値
+	和
-	差
*	積
/	商

%	Mod
---	-----

式中で利用可能な関数

関数名	返り値	補足
LOG10	常用対数	
LOGN または LOG	自然対数	
POW	乗数	第一引数:底 第二引数:乗数
MIN	最小値	引数二つ
MAX	最大値	引数二つ
SQRT	平方根	
SIN	正弦	
COS	余弦	
TAN	正接	
ABS	絶対値	
CEIL	切り上げ	
FLOOR	切り捨て	
PI	円周率	
EXP	指数関数	
UNI	一様分布(整数)	[第一引数, 第二引数)の整数値
UNID	一様分布(実数)	[第一引数, 第二引数)の字数値
NORMAL	正規分布	第一引数:平均 第二引数:分散
EXPDIST	指数分布	第一引数: λ
POISSON	ポワソン分布	第一引数: λ
EARLANG	アーラン分布	第一引数: λ 第二引数:位相

7.3.3. 計算式の設定

算出式の定義は以下の形式で記述する。

- Utility の算出式

$\text{Utility1Function} = \langle \text{値 or 計算式} \rangle$

$\text{Utility2Function} = \langle \text{値 or 計算式} \rangle$

Utility1Function の計算結果は Utility1 、 Utility2Function の計算結果は Utility2 に格納される。

Utility の計算は、「経路決定時」、「行動開始時」、「行動終了時」に、 Utility1 と Utility2 を独立して計算を行う。(※計算は Utility1 を先に行うため、 Utility1 を Utility2 の計算式で利用する場合や、 Utility2 を Utility1 の計算式で利用する場合は注意が必要である。)

移動途中に行動が切り替わる場合は、行動が変化する度に「行動開始」、「行動終了」による Utility の計算を行う。例えば、目的地まで、徒歩->電車->徒歩の経路で移動を行う場合、経路決定、徒歩(行動開始、行動終了)、電車(行動開始、行動終了)、徒歩(行動開始、行動終了)で計 7 回 Utility の計算を行う。

- 経路コストの算出式

$\text{RoutePriority} = \langle \text{値 or 計算式} \rangle$

経路コストは経路検索アルゴリズムで取得した経路について、経路コストを算出する際に利用する。

計算式ではパラメータで設定した値と、計算式の予約値を利用可能である。

予約値	パラメータ名
最後に訪れた via ポイント[*1]での、予定到着時刻に対する遅延 (E.g. 最後の via ポイントに 5 時の到着を予定し via ポイントに 5 時 30 分に到着した場合、30 分の遅延)	_DelayForLastViaPoint
次の via ポイントに対しての、予定到着時刻に対する遅延 (E.g. 次の via ポイントに 6 時の到着を予定し現在時刻が 6 時 5 分の場合、5 分の遅延。現在時刻が 6 時前である場合、遅延は常に 0)	_DelayForNextViaPoint
予定していた目的地への到着予定時刻に対する、経路検索した結果選択した経路での予定到着時刻の遅延 (E.g. 7 時の到着を予定して経路を検索した結果、7 時 10 分に到着する経路を選択した場合、10 分の遅延)	_DelayForExpectedArrivalTime
予定していた目的地への到着予定時刻に対する遅延	_DelayForSpecfiedArrivalTi

(E.g. 7 時の到着を予定し、現在時刻が 7 時 20 分である場合 20 分の遅延)	me
Utility1 の計算回数	_Utility1UpdateCount
Utility2 の計算回数	_Utility2UpdateCount

経路に関する予約値	パラメータ名
優先移動手段 ※経路が行動定義の PreferredMobilityMeans で指定された行動を利用して目的地へ向かう経路の場合は 1、それ以外は 0	_MoveByPreferredMobilityMeans
目的地への到着時刻	_ArrivalTime
目的地までの移動に要する時間	_TotalTravelTime
目的地までの総移動距離 ※道路の移動は経由する交差点の中心間を結んだ距離の総和、公共交通機関を利用する場合は、乗り降りする駅・バス停を直線で結んだ距離の総和	_TotalTravelDistance
利用する経路での公共交通機関の遅れの総和[s] ※電車の遅延は常に 0、バスの遅延は予定していた出発時刻に対する遅れ	_TotalPublicTransportationDelay
目的地までの移動にかかる料金の総和	_Price
公共交通機関の乗り換え回数	_TotalTransferCount
公共交通機関の乗り換え時間[s]	_TotalTransferDuration
経路検索時点での経路上の歩行者数の総和	_NumberOfPeopleOnRoute
経路検索時点での経路上のレーンあたりの車両数の総和	_NumberOfVehiclesOnRoute

*1 via ポイント: 行動の切り替わりの地点

Utility1、Utility2 の算出式で経路に関する予約値を利用する場合、適用される値は現在選択している経路に対する値となる。パラメータ設定と同様に、演算子、関数の利用が可能である。

プロファイルタイプの記述例を示す。

```
# >>>> Agent Profiles <<<<
#
#
#---+ Rule
#
# (Agent Definition)
#
# ProfileType : <User Definition Name>
# : --- Reserved = Taxi, Bus
#
# Age :
# WalkSpeed : (m/s)
# BicycleSpeed : (m/s)
# VehicleSpeed : (m/s)
# Utility1 :
# Utility2 :
# RecalcIntervalForLastViaPointDelay :
# RecalcIntervalForNextViaPointDelay :
# RecalcIntervalForDestinationDelay :
# RecalcIntervalForVehicleDelay :
# RecalcThresholdForCongestion :
# RecalcThresholdForUtility1 :
# RecalcThresholdForUtility2 :
# MinRouteRecalcInterval :
# RouteRecalcProbability :
# RecalcProbWhenMissingAVehicle :
# FuelConsumption : (money/m)
# BoardingDelay : (s)
# PedestrianRecognitionProbability :
# TimeHeadway : (s)
# MinVehicleGap : (m)
# MaxAcceleration : (m/s^2)
# MaxDeceleration : (m/s^2)
```

```

# LaneChangeAccelerationThreshold :: (m/s^2)
# PassiveYieldTime      : (s)
# ActiveYieldTime       : (s)
# TotalTravelDistance   : (m)
# TotalTravelTime        : (s)
# RoadWidthWeight       :
# Disaster              : (Disaster Mode:1)
# RoutecalculationTime  : (s)
# PrivateCarOwnership    : (YES:1, NO:0) [0-1]
# BicycleOwnership       : (YES:1, NO:0) [0-1]
# WalkSpeedAtTransfer   : (Normal, Slow, VerySlow)
# Utility1Function       :
# Utility2Function       :
#
#
# (Route Search)
#
# RoutePriority          : <Route Priority Formula>
#
# Predefined variables for the formula
# _MoveByPreferredMobilityMeans      : 0 or 1
# _ArrivalTime                 : arrival time [s]
# _TotalTravelTime             : total moved time [s]
# _TotalTravelDistance         : total moved distance [m]
# _TotalPublicTransportationDelay : total transportation delay time [s]
# _Price                      : total used price
# _TotalTransferCuont          : total number of connection times
# _TotalTransferDuration       : total connection time [s]
# _NumberOfPeopleOnRoute       : congestion of people on the road [0-1]
# _NumberOfVehiclesOnRoute     : congestion of vehicle on the road [0-1]
#
#
# (Other Rules)
#
# LOG10(value)           : Decadic Logarithm
# LOG(value)             : Natural Logarithm

```

```

# POW(value, multiplier)      : Power
# MIN(value1, value2)        : Minimum
# MAX(value, value2)        : Maximum
# SQRT(value)                : Square Root
# SIN(value)                 : Sine
# COS(value)                 : Cosine
# TAN(value)                 : Tangent
# ABS(value)                  : Absolute
# CEIL(value)                 : Ceiling
# FLOOR(value)                : Floor
# PI()                      : PI
# EXP(value)                  : Exponential function
# UNI(min, max)              : Uniform integer distribution
# UNID(min, max)              : Uniform double distribution
# NORMAL(average, deviation) : Normal distribution
# EXPDIST(lambda)              : Exponential distribution
# POISSON(lambda)              : Poisson distribution
# ERLANG(lambda, phase)       : Erlang distribution
#
#-----
# Agent: Student (example)
#-----


ProfileType : Student
WalkSpeed = UNID(4.0, 5.0)
BicycleSpeed = 10
VehicleSpeed = 60*1000 / 3600
RecalcThresholdForCongestion = 0.5
MinRouteRecalcInterval = 60*10
RouteRecalcProbability = 0.8
PrivateCarOwnership = 0
BicycleOwnership = 0.3
FuelConsumption = 0

RoutePriority = (1 + _MoveByPreferredMobilityMeans) * (_TotalTravelDistance /

```

```

MAX(1,      _TotalTravelTime)) / LOG10(MAX(10,      (_Price      /      MAX(1,
_TotalTravelDistance))*1000))

#-----
# Agent: Worker (example)
#-----

ProfileType : OfficeWorker
WalkSpeed = UNID(4.0, 6.0)
BicycleSpeed = 10
VehicleSpeed = 60*1000 / 3600
RecalcIntervalForLastViaPointDelay = 120
RecalcIntervalForNextViaPointDelay = 120
RecalcIntervalForDestinationDelay = 120
RecalcIntervalForVehicleDelay = 120
RecalcThresholdForCongestion = 0.9
RecalcThresholdForUtility1 = 0.2
RecalcThresholdForUtility2 = 0.2
RouteRecalcProbability = 0.1
MissedVehicleRouteRecalcProbability = 0.001
RecalcThresholdForCongestion = 0.5
MinRouteRecalcInterval = 60*10
PrivateCarOwnership = 0.1
BicycleOwnership = 0
FuelConsumption = 0
BoardingDelay = 1.0
PedestrianRecognitionProbability = 1.0
TimeHeadway = 1.5
MinVehicleGap = 3.0
MaxAcceleration = 1.0
MaxDeceleration = -3.0
LaneChangeAccelerationThreshold = 0.2
PassiveYieldTime = 3.0
ActiveYieldTime = -3.0
Utility1 = 0
Utility2 = 0

```

```

Utility1Function = UNID(0.0, 1.0)
Utility2Function = UNID(0.0, 1.0)
RoutePriority = (1 + _MoveByPreferredMobilityMeans) * (_TotalTravelDistance /
MAX(1,      _TotalTravelTime))      /      LOG10(MAX(10,      (_Price      /      MAX(1,
_TotalTravelDistance))*1000))

#-----
# Agent: VisitorA (example)
#-----

ProfileType : VisitorA
WalkSpeed = UNID(3.0, 5.0)
BicycleSpeed = 10
VehicleSpeed = 60*1000 / 3600
PrivateCarOwnership = 0
BicycleOwnership = 0
FuelConsumption = 0

RoutePriority = (1 + _MoveByPreferredMobilityMeans) * (_TotalTravelDistance /
MAX(1,      _TotalTravelTime))      /      LOG10(MAX(10,      (_Price      /      MAX(1,
_TotalTravelDistance))*1000))

#-----
# Agent: VisitorB (example)
#-----


ProfileType : VisitorB
WalkSpeed = UNID(3.0, 3.5)
BicycleSpeed = 10
VehicleSpeed = 50*1000 / 3600
PrivateCarOwnership = 0
BicycleOwnership = 0
FuelConsumption = 0

RoutePriority = (1 + _MoveByPreferredMobilityMeans) * (_TotalTravelDistance /

```

```

MAX(1,      _TotalTravelTime)) / LOG10(MAX(10,      (_Price      /      MAX(1,
_TotalTravelDistance))*1000))

#-----
# Reserved Agents
#-----

ProfileType : Taxi
VehicleSpeed = 80*1000 / 3600
FuelConsumption = 5
RoutePriority = 0

ProfileType : Bus
VehicleSpeed = 60*1000 / 3600
FuelConsumption = 0.2
RoutePriority = 0

```

また指定のなかった予約済みパラメータについては、PathQuery 用のパラメータ (RecalcIntervalForxxx は値が無限大、それ以外のパラメータは値が 0 として扱われる。タクシー、バスの挙動を決定するために、ProfileType として Taxi と Bus を必ず定義する必要がある。Taxi、Bus は車両での移動となるため、必要に応じて次の項目を設定する。

- Taxi

VehicleSpeed: 最大速度(m/s)
FuelConsumption: 運賃(料金/走行距離[m])
PedestrianRecognitionProbability: ドライバが歩行者を認識する確率 1
TimeHeadway: 車両のヘッドウェイ[s]
MinVehicleGap : 車両の最小車間距離[m]
MaxAcceleration: 車両の最大加速度[m/s²]
MaxDeceleration: 車両の最大減速度[m/s²]
MaxBrakingDeceleration: 車両の最大減速度[m/s²]
AccelerationExponent: ドライバの加速度指数
SaveDeceleration: 車線変更による後方車両の減速度の閾値[m/s²]
MinTurnSpeed: カーブでの最小速度[m/s]
LaneChangeAccelerationThreshold: 車両の車線変更加速度閾値[m/s²]

VelocityRatioGapDistance: IDM モデル[1]における係数 s1

OtherVehicleEntranceEntranceTime: 交差点において、相手車両が交差点に進入しようとしていることを判断する時間距離[s]

PassiveYieldTime: 車両の消極的な譲歩時間[s]

ActiveYieldTime: 車両の積極的な譲歩時間[s]

- Bus

VehicleSpeed: 最大速度(m/s)

PedestrianRecognitionProbability: ドライバが歩行者を認識する確率 1

TimeHeadway: 車両のヘッドウェイ[s]

MinVehicleGap : 車両の最小車間距離[m]

MaxAcceleration: 車両の最大加速度[m/s^2]

MaxDeceleration: 車両の最大減速度[m/s^2]

MaxBrakingDeceleration: 車両の最大減速度[m/s^2]

AccelerationExponent: ドライバの加速度指数

SaveDeceleration: 車線変更による後方車両の減速度の閾値[m/s^2]

MinTurnSpeed: カーブでの最小速度[m/s]

LaneChangeAccelerationThreshold: 車両の車線変更加速度閾値[m/s^2]

VelocityRatioGapDistance: IDM モデル[1]における係数 s1

OtherVehicleEntranceEntranceTime: 交差点において、相手車両が交差点に進入しようとしていることを判断する時間距離[s]

PassiveYieldTime: 車両の消極的な譲歩時間[s]

ActiveYieldTime: 車両の積極的な譲歩時間[s]

VehicleSpeed: 最大速度(m/s)

※バスの運賃の設定は公共エージェントタイムテーブル設定ファイルで行う

7.4. エージェント行動定義ファイル

エージェント行動定義ファイルではエージェントの行動モデルを設定可能である。行動モデル定義では以下の項目を設定可能である。

- 場所のグループ定義
- 行動タイプ定義
- 行動リスト設定
- パラメータの変更

- ファイル構文
 - 一行につき一項目記述する
 - 文字コード・改行コードはシミュレーション実行環境に合わせる。
 - 空行はスキップされる
 - 「#」で開始する行はコメント行としてスキップされる

7.4.1. 場所のグループ定義

エージェントの初期配置、移動目的地として設定可能な場所をグループ化して指定を行いたい場合、事前に複数の場所をグループ化して名称を定義することで初期配置、移動目的地に利用可能である。

`LocationGroup <グループ名> = <場所名 1>,<場所名 2>,<場所名 3>,...`

`LocationIdGroup <グループ名> = <場所 ID1>,<場所 ID2>,<場所 ID3>,...`

グループ化可能な場所のタイプは、建物(Building)、公園(Park)、POI のいずれかである。同一のグループ内に複数のタイプが混在しても良い。

LocationGroup の場合、場所名は文字列で指定し、LocationIdGroup の場合、場所 ID はシナリオのシェープファイル(building.shp、park.shp、poi.shp)について"id"属性で指定されるオブジェクト毎のID(数字)を指定する。

一つのグループには何項目でも場所名を含めることができる。場所数がいくつ含まれていても必ず一行で記述する。LocationGroup は別名のグループであれば何項目でもグループを定義が可能である。

7.4.2. 行動タイプ定義

エージェントの行動タイプを定義する。各行動タイプに対して"BehaviorType:"の予約語から記述を始める。"BehaviorType"以降に記述された行動リストの設定およびパラメータの変更は、次に新しい"BehaviorType"を記述した行が表れるまで、その行動タイプの設定となる。

BehaviorType:<行動タイプ名>

7.4.3. 行動リスト設定

定義した行動モデルについて、シミュレーション内での行動を定義する。エージェントの行動予定は実行される順に一行ずつ記述する。時刻・時間に関する記述は全て秒単位で指定する。また、全ての行動リストを終了したエージェントはシミュレーションエリアから消滅する。

[<時刻>, <プロファイルタイプ条件>, <プロファイルパラメータ条件>] <初期配置>, <目的地>, <目的地選択種別>, <出発時刻>, <到着時刻>, <優先移動手段>, <行動指定>, <待機時間>, <最小行動継続時刻>, <アプリケーション生成>, <割り込み指定> <経由交差点>, <プロファイルパラメータ変更>

※省略可能な記述: [], <目的地選択種別>, <出発時刻>, <到着時刻>, <優先移動手段>, <行動指定>, <待機時間>, <最小行動継続時刻>, <アプリケーション生成>, <割り込み指定>, <プロファイル値の変更>, <交通機関予約>

[]を省略した場合、すぐにスケジュールが実行される。出発時刻、到着時刻を指定した場合、スケジュールが開始してからすぐに移動を開始する。

- 時間指定
 - 時刻: 行動を実行する時刻

時刻が未来の時刻である場合、その時刻まで待ってから行動を開始する。既に過ぎた過去の時刻である場合や、"-”で時刻が指定されている場合はすぐに行動を開始する。

- 条件指定
 - プロファイルタイプ条件: ProfileType = <プロファイルタイプ名>
 - パラメータ条件: 実数で指定したパラメータに対する値の条件比較式。

一致するパラメータ名のパラメータを自身のプロファイルに持たない場合、その値は 0 として条件の比較を行う。

条件指定を行う場合、条件内の全ての条件を満たした場合に実行される。(条件を指定しない場合は常に行動を実行する)条件を満たさない場合はその行動をスキップし、次の行動に移行する。スキップした行動については後で条件を満たしたとしても実行は行わない。

- 行動指定
 - 初期配置: InitialLocation/ InitialLocationId = <初期配置/初期配置 ID/PresentLocation>

エージェントの初期配置を指定する。初期配置は最初の行動の記述で指定しなければならない) 初期配置は公園、建物、エリアの名称(オブジェクトの ID でも可)、グループ名または"RandomBuilding"、"RandomPark"、"RandomPOI"のいずれかとする。

エリアを指定した場合、そのエリアに(一部でも)含まれる建物の中から建物の大きさに関わらずにランダムな建物を選択する。

グループ名で指定する場合は"InitialLocation"を利用する。グループ指定では、グループ内の建物、公園、エリアから一様にランダムに配置先を決定する。RandomBuilding、RandomPark、RandomPOI を指定した場合、シナリオ内の全ての建物または全ての公園から重みを一様としてランダムに建物または公園を選択する。

これらのランダム指定における初期座標は決定した建物、公園内のランダムな位置とする。

また、"PresentLocation"を指定した場合にはシナリオで設定したエージェントの位置が、シミュレーション実行時の初期位置として設定される。

- 目的地: MoveToDestination/MoveToDestinationId = <目的地名/目的地 ID/グループ名>
 "MoveToDestination"として初期配置と同様に公園、建物、エリアの名称、グループ名または"RandomBuilding"、"RandomPark"、"RandomPOI"を指定可能である。
 エージェントの初期配置を目的地として指定する場合は"InitialLocation"、移動を行わずに待機状態を設定する場合は"None"を目的地に指定する。目的地は指定した建物、公園内のランダムな位置となる。
 "MoveToDestination"に対してグループ名を指定した場合は、グループ内のランダムな場所が目的地となる。目的地の選択方法を変更する場合は、目的地選択種別の指定も行う。、
 目的地への移動が不可である場合および道路網の変更により目的地への移動が不可であることがわかった場合、行動をスキップし、行動リストの次の行動を行う。目的地をグループで指定している場合は、グループ内の他の場所を目的地として決定し移動を続行する。全ての目的地に対して到達不可である場合は行動をスキップする。
 また、予定していた目的地に入ろうとした際、その目的地のキャパシティがいっぱいである場合は、目的地の指定が单一の指定である場合は、入り口で Human に設定された人数分のキャパシティの空きが出来るのを待つ。また、入り口での待機時間が入り口での最大待ち時間 (MaxWaitingTimeAtEntrance)を超え、かつ目的地の指定がグループである場合は、その目的地へ入るのは諦め、グループ内の他の場所を目的地として再度移動を行なう。目的地のキャパシティは Human に設定した人数(NumberOfPeople)の数により計算される。
 さらに自家用車で目的地へ移動するする場合には、目的地に設定される Vehicle Capacity に空き出るまで待機する。自家用車での入り口での待機時間が入り口での最大待ち時間 (MaxWaitingTimeAtVehicleEntrance)を超え、かつ目的地の指定がグループである場合は、その目的地へ入るのは諦め、グループ内の他の場所を目的地として再度移動を行なう。

例 1:100 秒の時点で Building1 を目的地として移動を開始する。

[100] MoveToDestination = Building1

例 2:プロファイルタイプが Student のエージェントについて、100 秒の時点で Building1 を目的地として移動を開始する。

[100, ProfileType = Student] MoveToDestination = Building1

- 目的地選択種別:DestinationChoiceType = <Random/Nearest>
 "MoveToDestination"に対してグループ名が指定した場合に利用可能である。"Random"の場合、グループ内のランダムな場所が目的地となり、"Nearest"の場合は、行動開始時点で、目的地選択の基準場所である"DestinationChoiceBaseLocation"、に指定した場所に最も近い位置が目的地となる。また、"DestinationchoiceBaseLocation"の指定が無い場合には最も現在位置に近い場所が目的地となる。
- 目的地選択の基準場所:DestinationChoiceBaseLocation = <場所名/AgentLocation>
 "DestinationChoiceType"に対して、"Nearest"を指定した場合に利用可能である。最寄の目的地を選択する際の、基準の場所を指定する。場所名に"AgentLocation"を指定した場合には、目的地選択の際のエージェントの位置を基準となる。
- 待機時間:Wait = <時間>
 目的地に到着してから待機する時間を指定する。(目的地に到着できず、行動をスキップする場合、Wait の指定は無効となる。)

例 1:100 秒の時点で Building1 を目的地として移動し、目的地に到着してから 50 秒間待機する

[100] MoveToDestination = Building1, Wait = 50

- 最小行動継続時刻:WaitUntil = <時刻>
 行動が終了する最小の時刻を指定する。最小行動継続時刻以前に目的地に到着したとしても、指定した時刻までは次の行動に移行せずに待機する。最小行動継続時刻以降に目的地に到着した場合は、すぐに次の行動へ移行する。

例 1:100 秒の時点で Building1 を目的地として移動を開始し、目的地に到着したとしても少なくとも 200 秒までは待機を行う。

[100] MoveToDestination = Building1, WaitUntil = 200

- 経路検索に利用する出発時刻: $\text{DepartureTime} = <\text{時刻}>$, $\text{EarliestDepartureTime} = <\text{時刻}>$, $\text{LatestDepartureTime} = <\text{時刻}>$

※省略可能な記述: $\text{EarliestDepartureTime}$ および $\text{LatestDepartureTime}$

公共交通機関の検索に利用する出発時刻となる。(エージェントの行動開始時刻ではない) エージェントは、公共交通機関を用いた経路検索において次の条件の交通機関を対象とする。

検索対象

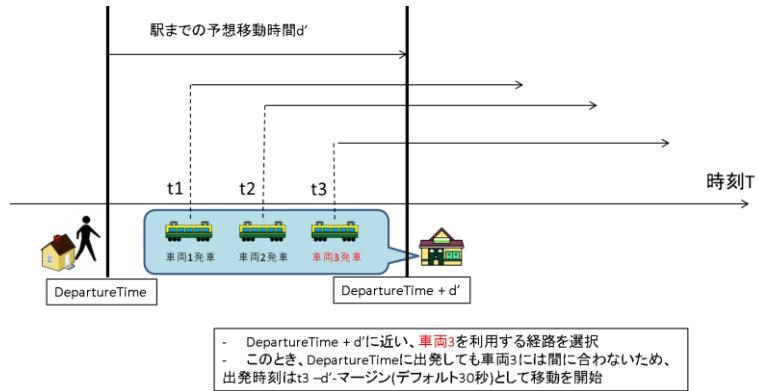
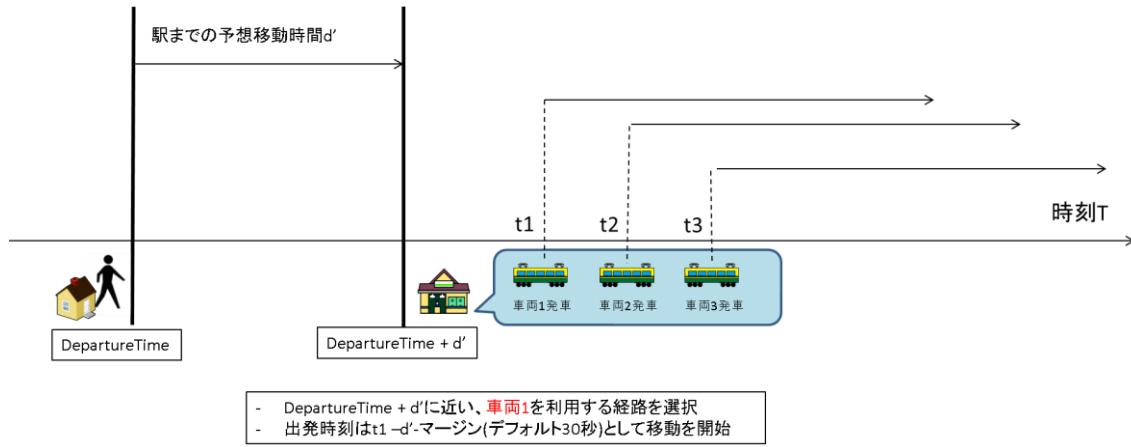
乗車駅で、 $\text{DepartureTime} + d'$ (=乗車駅到着時刻)にもっとも近い時刻に発車する車両

d' : 出発地から乗車駅までの、歩行による予想移動時間

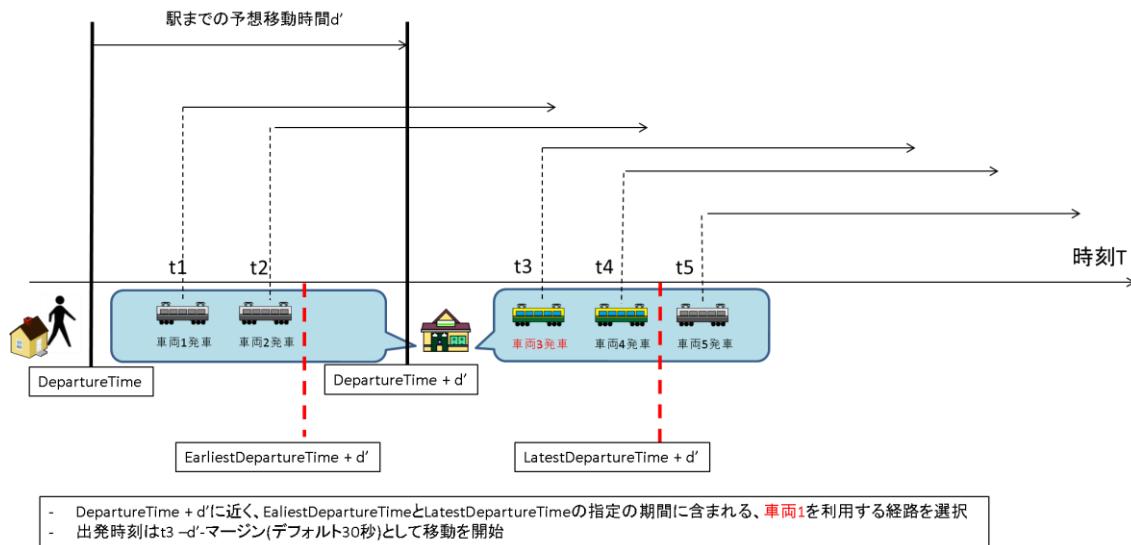
- 通常、エージェントは、エージェントの乗車駅への到着時刻より前に到着する車両を選択するが、該当する車両が無い場合は、エージェントの乗車駅への到着時刻より後に到着する車両を選択する。(いずれの場合も、エージェントの乗車駅への到着時刻に最も近い車両を選択する)
- 移動行動時、すぐに移動を開始しても乗車に間に合わない電車は検索の対象とならない。
- 乗車駅到着時刻より前の車両を選択しようとする際に、 $\text{EarliestDepartureTime}$ の指定がある場合は、 $\text{EarliestDepartureTime} + d'$ より前の車両は選択しない
- 乗車駅到着時刻より後の車両を選択しようとする際(DepartureTime の前に該当する車両はない)に、 $\text{LatestDepartureTime}$ の指定がある場合は、 $\text{LatestDepartureTime} + d'$ より後の車両は選択しない。
- DepartureTime より大きな $\text{EarliestDepartureTime}$ を指定した場合 $\text{EarliestDepartureTime}$ は無効になる。同様に、 DepartureTime より小さい $\text{LatestDepartureTime}$ を指定した場合 $\text{LatestDepartureTime}$ は無効となる。

“ $\text{EarliestDepartureTime}$ ”、“ $\text{LatestDepartureTime}$ ”を指定する場合は、必ず” DepartureTime ”を指定しなければならない。

$\text{DepartureTime} + d'$ 前に到着する電車がある場合

DepartureTime + d' 前に到着する電車がない場合

EarliestDepartureTime、LatestDepartureTime を指定した場合



例 1: DepartureTime = 20 分 (出発地から出発駅(バス停)から目的地までの予測移動時間 = 5 分)
 25 分以前で目的地に向かって最も遅い時間に出発する電車(バス)を選択する。見つからなければ
 25 分以降で最も早い時間に出発する電車(バス)を選択する。

例 2: DepartureTime = 20 分、EarliestDepartureTime = 19 分 (出発地から出発駅(バス停)から目的地までの予測移動時間 = 5 分)
 24 分から 25 分の間で、25 分に近い時間に出発する電車(バス)を選択する。見つからなければ 25
 分以降で最も早い時間に出発する電車(バス)を選択する。

例 3: DepartureTime = 20 分、LatestDepartureTime = 21 分 (出発地から出発駅(バス停)から目的地までの予測移動時間 = 5 分)
 25 分以前で、25 分に近い時間に出発する電車(バス)を選択する。見つからなければ 25 分から 26
 分の間で目的地に向かって最も早い時間に出発する電車(バス)を選択する。

例 4: DepartureTime = 20 分、EarliestDepartureTime = 19 分、LatestDepartureTime = 21 分 (出
 発地から出発駅(バス停)から目的地までの予測移動時間 = 5 分)
 24 分から 25 分の間で、25 分に近い時間に出発する電車(バス)を選択する。見つからなければ 25
 分から 26 分の間で目的地に向かって最も早い時間に出発する電車(バス)を選択する。

- 経路検索を利用する到着時刻: ArrivalTime = <時刻>, EarliestArrivalTime = <時刻>, LatestArrivalTime = <時刻>

※省略可能な記述: EarliestArrivalTime および LatestArrivalTime

公共交通機関の検索を利用する到着時刻となる。(エージェントの到着時刻ではない) エージェントは、公共交通機関を用いた経路検索において次の条件の交通機関を対象とする。

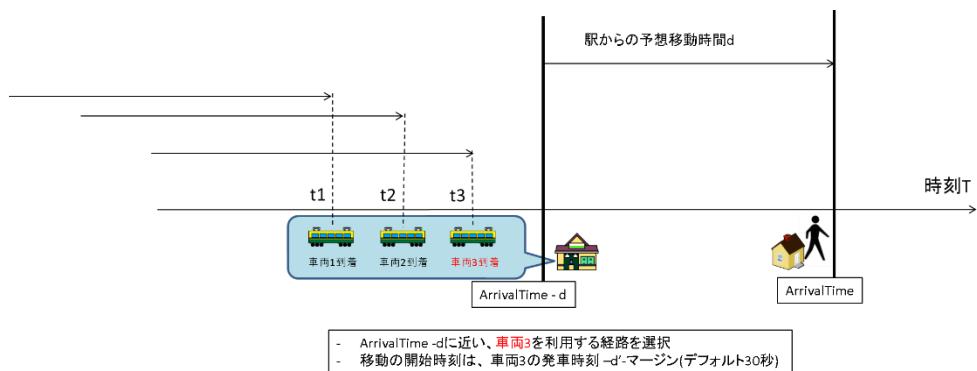
検索対象

降車駅に、ArrivalTime - d (= 降車駅到着時刻) にもっとも近い時刻に到着する車両
 d : 降車駅から目的地までの、歩行による予想移動時間

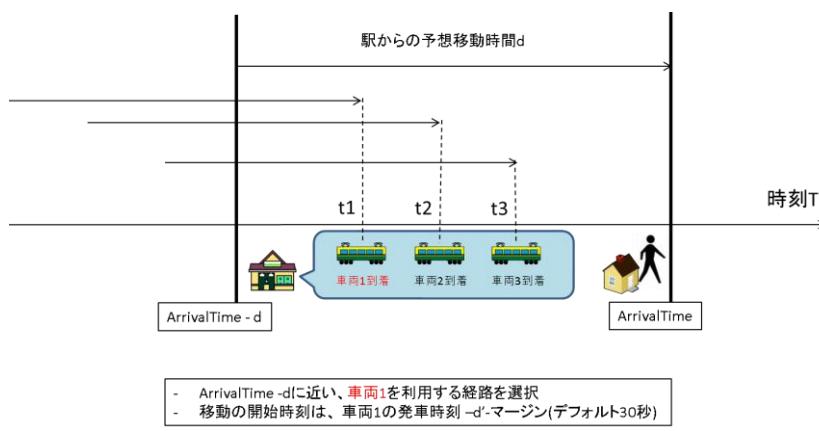
- 通常、エージェントは、エージェントの降車駅への到着時刻より前に到着する車両を選択するが、該当する車両が無い場合は、エージェントの降車駅への到着時刻より後に到着する車両を選択する。(いずれの場合も、エージェントの降車駅への到着時刻に最も近い車両を選択する)

- 移動行動時、すぐに移動を開始しても乗車に間に合わない電車は検索の対象とならない。
- 降車駅到着時刻より前の車両を選択しようとする際に、EarliestArrivalTime の指定がある場合は、EarliestArrivalTime - d より前に到着する車両は選択しない
- 降車駅到着時刻より後の車両を選択しようとする際(ArrivalTime の前に該当する車両はない)に、LatestArrivalTime の指定がある場合は、LatestArrivalTime - d より後に到着する車両は選択しない。
- ArrivalTime より大きな EarliestArrivalTime を指定した場合 EarliestArrivalTime は無効になる。同様に、ArrivalTime より小さい LatestArrivalTime を指定した場合 LatestArrivalTime は無効となる。

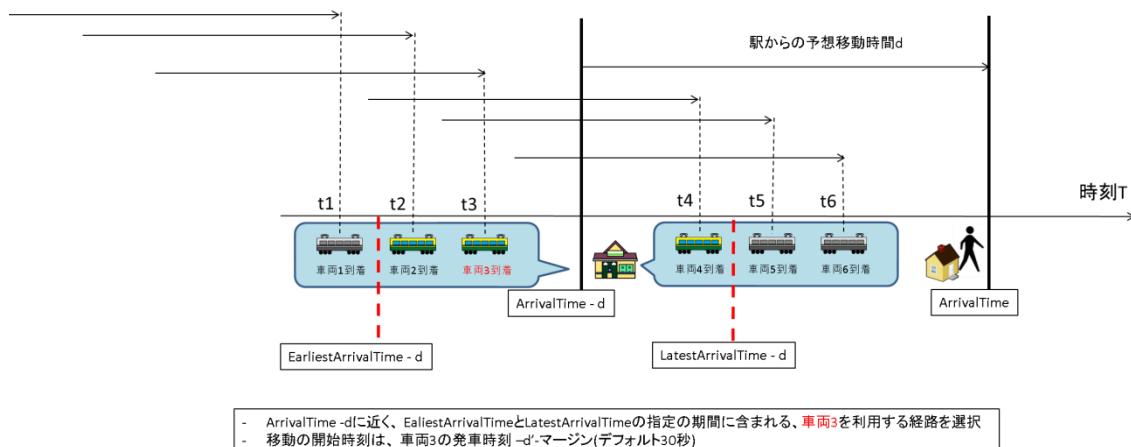
ArrivalTime - d 前に到着する電車がある場合



ArrivalTime - d 前に到着する電車がない場合



EarliestArrivalTime、LatestArrivalTime を指定した場合



例 1: ArrivalTime = 10 分 (目的駅(バス停)から目的地までの予測歩行時間 = 2 分)

8 分に最も近い時間に到着する電車を選択。見つからなければ 8 分以降で最も速い電車(バス)を選択する。

例 2: ArrivalTime = 10 分、EarliestArrivalTime = 9 分 (目的駅(バス停)から目的地までの予測歩行時間 = 2 分)

7 分から 8 分の間で、8 分に近い時間に到着する電車(バス)を選択。見つからなければ 8 分以降で最も速い電車(バス)を選択する。

例 3: ArrivalTime = 10 分、LatestArrivalTime = 20 分 (目的駅(バス停)から目的地までの予測歩行時間 = 2 分)

8 分に最も近い時間に到着する電車(バス)を選択する。見つからなければ 8 分から 19 分の間に到着する最も速い電車(バス)を選択する。

例 4: ArrivalTime = 10 分、EarliestArrivalTime = 9 分、LatestArrivalTime = 20 分 (目的駅(バス停)から目的地までの予測歩行時間 = 2 分)

7 分から 8 分の間で、8 分に近い時間に到着する電車(バス)を選択する。見つからなければ 8 分から 19 分の間に到着する最も速い電車(バス)を選択する。

- 優先移動手段: PreferredMobilityMeans = <Pedestrian/Vehicle/Taxi/Bus/Train>

経路候補計算時に、各経路候補に対してプロファイルで定義される"RoutePriority"の計算式が適用されるが、経路候補が"PreferredMobilityMeans"で指定した移動行動と一致する場合、その経路候補の経路計算式において"_MoveByPreferredMobilityMeans"変数の値が 1 となる。一致しない経路候補の場合は"_MoveByPreferredMobilityMeans"変数は 0 となる。

優先移動手段名	“_MoveByPreferredMobilityMeans”変数が1となる経路候補条件
Pedestrian(または”Walk”でも可)	徒歩のみを利用した経路である。
Vehicle	自家用車のみを利用した経路である。
Taxi	タクシーのみを利用した経路である。
Bus	バスのみを利用した経路である(バス間での乗り換えがあっても良いが、途中で電車への乗り換えがある場合は一致しないとみなして”_MoveByPreferredMobilityMeans”変数を0とする)。
Train	電車のみを利用した経路である(電車間での乗り換えがあっても良いが、途中でバスへの乗り換えがある場合は一致しないとみなして”_MoveByPreferredMobilityMeans”変数を0とする)。

- 行動指定 : MobilityMeans = <(Pedestrian/Vehicle/Taxi/Bus/Train)>

経路候補計算時に、行動指定と一致する経路候補がある場合、必ずその経路候補を利用する。一致する経路が複数見つかった場合は経路候補の中から一様にランダムに一つの経路を選択する。一致する経路が一つも見つからなかった場合は、通常の経路選択アルゴリズムに従って経路を選択する。

行動指定	行動指定と一致する経路候補条件
Pedestrian(または”Walk”でも可)	徒歩のみを利用した経路である。
Vehicle	自家用車のみを利用した経路である。
Taxi	タクシーのみを利用した経路である。
Bus	バスのみを利用した経路である(バス間での乗り換えがあっても良いが、途中で電車への乗り換えがある場合は一致しないとみなす)。
Train	電車のみを利用した経路である(電車間での乗り換えがあっても良いが、途中でバスへの乗り換えがある場合は一致しないとみなす)。

- 経由交差点 : IntersectionsToGoThrough = <交差点名 1:交差点名 2:...>

道路歩行、自転車、自家用車、タクシーを利用した経路において、途中で通過する交差点を指定する。交差点名は経由順”:(コロン)で区切って指定することで、複数指定可能。交差点名に”RandomIntersection”を指定した場合、任意の交差点を経由する。

- アプリケーション生成:GenerateApplication = “cbr-destination = 1, cbr-start-time = 10, ...”
 値には、生成するアプリケーションに関するシミュレーションパラメータのセットをコンマ区切りで直接に指定する。
 <目的地>の前に<アプリケーション生成>を記述することで、行動の開始時刻に指定の通信アプリケーションを生成する。<目的地>の後、かつ<待機時間>の前に記述することで、目的地に到着後、指定の通信アプリケーションを生成する。<目的地>の後、かつ<待機時間>の後に記述することで、目的地に到着し待機時間待った後、通信アプリケーションを生成する。
 通信アプリケーションは、生成時刻を基準として開始する。例えば、<目的地>到着後にアプリケーション生成を行い、cbr-start-time = 10 の指定がある場合、目的地到着後 10 秒後に最初の CBR パケットが送信される。
- 割り込み指定:InterruptCurrentAction = <ResumeAfterInterruption/TerminateNow>
 指定時間に、割り込みの行動を行う。”ResumeAfterInterruption”を指定した場合には、割り込み行動を終了した後、割り込み前の行動を再度実施する。”TerminateNow”を指定した場合には、割り込み行動終了後、割り込み前の行動に戻らず、その次に予定していた行動を実施する。
 なお、割り込み指定は、必ず時刻とセットで指定なければならない。
- プロファイルパラメータ変更:パラメータ名 = 値 or 計算式
 <目的地>の後に<プロファイル値の変更>を記述することで、目的地に到着した際にプロファイル値の変更が行われる。<目的地>の前に<プロファイル値の変更>を記述した場合は、目的地への経路計算の直前にプロファイル値の変更が行われる。

※パラメータを変更する場合、エージェントは必ずそのパラメータ名と一致するパラメータを自身のプロファイルに持つていなければならない。

7.4.4. パラメータの変更

エージェントの行動に関係なく絶対時刻でプロファイルのパラメータを変更する場合、行動リスト記述以降の行で、時刻と共にパラメータの変更を記述する。

[<時刻>] <プロファイル/パラメータ名> = <値 or 計算式>

※パラメータを変更する場合、エージェントは必ずそのパラメータ名と一致するパラメータを自身のプロファイルに持つていなければならない。

例 1: 60 秒になつたら歩行速度を 3[m/s]に変更する。

[60] WalkSpeed = 3

例 2:60 秒になつたら歩行速度を 1[m/s]、Utility1 を計算式により算出する。

[60] WalkSpeed = 1, Utility1 = 0.5*Age

行動タイプの記述例を示す。

```
# >>>> Agent Behaviors <<<<
#
#---+ Rule
#
# <Agent behavior definition format>
# BehaviorType:{Behavior name}
# [Time] Behavior
# [Time, Condition] Behavior
# ...
#
# <Time format>
# HH:MM or HH:MM:SS or seconds in double or "-".
#
# Example:
# [08:32] InitialLocation = Building1
# [08:32:00] InitialLocation = Building1
# [30720] InitialLocation = Building1
# [-] InitialLocation = Building1
#     - "-" means the time when previous action/behavior completed.
#
#
# <Condition format>
# {status/profile in agent profile} => {value in double}
#
# Example:
# [120, Utility1 > 0.5] MoveToDestination = Building1
#     - On 120 seconds, only agents which have more than 0.5 for Utility1 move
#       to Building1.
#
#
```

```

# <Behavior format>
#
# {event} = {value}, {event} = {value}, ...
#
# Predefined event name and value:
#
# InitialLocation = {initial location name}
#   - Agent will show up at a specified location.
#     - {initial location name} must be a name of building/park/POI,
#       "RandomBuilding", "RandomPark",
#       "RandomPoi", "PresentLocation" or a user-defined location group name.
#       - RandomBuilding/RandomPark/RandomPoi is a randomly picked
#         building/park/POI from all buildings/parks/POIs.
#   - PresentLocation means a current position of the agent.
#
# InitialLocationId = {initial location ID}
#   - Agent will show up at a specified location.
#     - {initial location ID} must be an ID of building, park or POI.
#
# MoveToDestination = {destination}
#   - Agent will move to a specified destination.
#     - {destination} must be a name of building/park/POI, "RandomBuilding",
#       "RandomPark", "RandomPoi",
#       "InitialLocation" or a user-defined location group name.
#       - RandomBuilding/RandomPark/RandomPoi is a randomly picked
#         Building/Park/POI from all buildings/parks/POIs.
#   - InitialLocation means agent initial position defined in "InitialLocation"
#     or "InitialLocationId".
#
# MoveToDestinationId = {destination ID}
#   - Agent will move to a specified destination.
#     - {destination ID} must be a name of building/park/POI.
#
# DestinationChoiceType = "Random" or "Nearest"
#   - This is used when value of MoveToDestination is a user-defined location
#     group.
#   - "Random" means a randomly picked destination from a user-defined location

```

```

group.

#      - "Nearest" means a nearest destination of a user-defined location group
from agent current location or
#      a location specified in DestinationChoiceBaseLocation.
#
# DestinationChoiceBaseLocation = {base location to pick a destination}
#      - This is used when DestinationChoiceType = Nearest.
#      - {base location to pick a destination} must be "AgentLocation" or a name
of building/park/POI.
#      - "AgentLocation" means a location nearest to agent location at this event
execution will be picked
#      from a user-defined location group.
#      - When a name of building/park/POI is specified, a location nearest to the
specified location
#      will be picked from a user-defined location group.

#
# MobilityMeans = "Walk", "Bicycle", "PrivateCar", "Taxi", "Bus", "Train".
#      - Agent will use the specified mobility means if available.
#
# Wait = {duration of waiting time}
#      - Agent will stay at current location for specified duration.
#
# WaitUntil = {waiting time}
#      - Agent will stay at current location until specified time.
#
#
# <user-defined Location group format>
# LocationGroup {location group name} = {comma-delimited location names}
# LocationIdGroup {location ID group name} = {comma-delimited location IDs}
#
# Example:
# LocationGroup Parkings = building1, building2, building3
# LocationIdGroup StructureIds = 102000001, 102000002, 102000003
#
#Locations

```

```

LocationGroup           RandomHouse          =
house1,house2,house3,house4,house5,house6,house7,house8,house9,house10,house1
1,house12,house13,house14,house15,house16,house17,house18,house19,house20,hou
se21,house22,house23,house24,house25,house26,house27,house28,house29,house30,
house31,house32,house33,house34,house35,house36,house37,house38,house39,house
40,house41,house42,house43,house44,house45,house46,house47,house48,house49,ho
use50,house51,house52,house53,house54,house55,house56,house57,house58,house59
,house60,house61,house62,house63,house64,house65,house66,house67,house68,hous
e69,house70,house71,house72,house73,house74,house75,house76,house77,house78,h
ouse79,house80,house81,house82,house83,house84,house85,house86,house87,house8
8,house89,house90,house91,house92,house93,house94,house95,house96,house97,hou
se98,house99,house100,house101,house102,house103,house104,house105,house106,h
ouse107,house108

LocationGroup Stationbuildings = StationBuilding1,StationBuilding2

LocationGroup           RandomOffice         =
Office1,Office2,Office3,Office4,Office5,Office6,Office7,Office8,Office9,Offic
e10,Office11,Office12,Office13,Office14,Office15,Office16,Office17,Office18,O
ffice19,Office20,Office21,Office22,Office23,Office24,Office25,Office26,Office
27,Office28

LocationGroup ElementarySchool = School1,School2,School3,School4,School5

LocationGroup           RandomShop          =
Shop1,Shop2,Shop3,Shop4,Shop5,Shop6,Shop7,Shop8,Shop9,Shop10,Shop11,Shop12,Sh
op13,Shop14,Shop15,Shop16,Shop17,Shop18

LocationGroup           Parks              =
Park15,Park16,Park17,Park18,Park19,Park20,Park21,Park22,Park23,Park24,Park25,
Park26,Park27,Park28,Park29,Park30,Park31,Park32,Park33,Park34,Park35,Park36,
Park37,Park38,Park39,Park40,Park41

BehaviorType:OfficeWorkerBehavior1

[-] InitialLocation = RandomHouse
[UNI(1, 20)] MoveToDestination = RandomOffice, DestinationChoiceType = Random,
PreferredMobilityMeans = bus, WaitUntil = "UNI(270,290)"
[-] MoveToDestination = RandomShop, DestinationChoiceType = Random,

```

```
PreferredMobilityMeans = Walk
[-] MoveToDestination = RandomHouse, DestinationChoiceType = Random, WaitUntil
= 600

BehaviorType:OfficeWorkerBehavior2

[UNI(1, 100)] InitialLocation = Stationbuildings
[-] MoveToDestination = RandomOffice, DestinationChoiceType = Random,
PreferredMobilityMeans = Train, WaitUntil = "UNI(300,320)"
[-] MoveToDestination = Stationbuildings, DestinationChoiceType = Random,
PreferredMobilityMeans = Train

BehaviorType:StudentBehavior

[-] InitialLocation = RandomHouse
[UNI(1,10)] MoveToDestination = ElementarySchool, DestinationChoiceType =
Nearest, PreferredMobilityMeans = Walk, WaitUntil = "UNI(200,210)"
[-] MoveToDestination = Parks, DestinationChoiceType = Nearest
[-] MoveToDestination = RandomHouse, DestinationChoiceType = Random,
PreferredMobilityMeans = bus, Wait = 900
```

7.5. エージェントタイムテーブル設定ファイル

エージェントタイムテーブル設定ファイルでは電車・バスのスケジュールを設定可能である。スケジュールの定義は以下の項目を設定可能である。

- 車両定義
- 運行ラインの定義
- 運賃設定
- 駅/バス停設定
- 経由交差点設定
- 運行スケジュール設定
- ファイル構文
 - 一行につき一項目記述する
 - 文字コード・改行コードはシミュレーション実行環境に合わせる。
 - 空行はスキップされる
 - 「#」で開始する行はコメント行としてスキップされる

7.5.1.車両定義

電車・バスの車両を定義する。車両定義は"Family"から開始し、車両名、大きさ、許容人数を定義する。名前が重複しない限りは何項目でも車両の定義が可能である。

Family,<車両名(文字列)>,<幅[m](実数)>,<長さ[m](実数)>,<許容人数[人](整数)>

電車のスケジュール定義内で、定義した車両名を利用するため、電車のスケジュール定義よりも先に記述を行う必要がある。幅と長さを指定した長方形が車両の領域となり、乗車するエージェントは車両内の任意の位置まで移動して降車するまで待機する。許容人数を超えるエージェントはその車両に乗り込むことが出来ない。駅(バス停)での待ち時間が長いエージェントから順に乗車を行う。
RoutePriority

7.5.2.運行ラインの定義

運行ラインを定義する。ラインの定義は"Line"から開始しライン名とラインタイプで指定する。"Line"以降に記述された運賃設定、駅/バス停設定および運行スケジュール設定は、次に新しい"Line"を記述した行が表れるまでその運行ラインの設定となる。複数のラインを定義する場合、ライン名の重複は不可である。同一路線の上り下りについては別の運行ラインで定義する。

Line,<ライン名>,<タイプ(Train/Bus)>

運行ラインが電車の場合は"Train"、バスの場合は"Bus"をタイプに指定する。RoutePriority

7.5.3. 運賃設定

電車・バスの運賃設定を行う。運賃設定を行う場合、必ず"Line"の定義が行われていなければならない。運賃定義は"Price"から開始し、一般料金について、距離あたりの金額を<金額>/<距離>の形式で記述する。

"Price"以降に記述された駅/バス停設定は、次に新しい"Price"を記述した行が表れるまでその運賃設定となる。一つの運行ラインの定義内で複数回運賃設定を行っても良い。

Price, <一般料金>

記述を省略した運賃を利用する場合、デフォルト値として無料の運賃を適用する。

例 1: 100m ごとに 20 の料金を設定する場合

Price, 20/100

※100/10 と記述した場合 10m までが一律 100 円、10m から 20m までが一律で 200 円となる。同様に 100/100 のように記述した場合は 100m までが 100 円、200m までが 200 円となる。0/1 は無料を示す。距離は乗車駅(バス停)と降車駅(バス停)を直線で結んだ距離を用いる。

7.5.4. 駅/バス停設定

電車・バスが停車する駅(バス停)を指定する。駅/バス停設定を行う場合、必ず"Line"の定義が行われていなければならない。また駅/バス停設定の前に"Price"の定義が行われていない場合は、運賃を無料とする。設定は"Stop"から開始し、停車する駅(バス停)を停車する順に指定する。

普通電車、特急電車等で同じ路線を利用するが停車する駅、停車しない駅が分かれている場合、一つの運行ライン内で複数の駅/バス停設定が可能である。"Stop"以降に記述された運行スケジュール設定は、次に新しい"Stop"を記述した行が表れるまでその駅/バス停設定となる。

Stop,<駅(バス停)>,<駅(バス停)>,<駅(バス停)>,<駅(バス停)>

運行ラインで電車が指定される場合は必ず駅を指定し、バスが指定されている場合はバス停を指定する。

停車する駅(バス停)は何個でも続けて記述して良い。連続で同一名の駅(バス停)を続けることは不可であるが、連続していなければ同一名の駅(バス停)を続けても良い。同一の運行ライン内に停車順が全く同一である駅/バス設定を記述してはならない。

運行ラインがバスの場合、バスは各バス停を最短経路つないだ道路上を移動する。

電車の場合、停車する駅を順に繋いた線路上を移動する。線路の端点と駅が交差していない場合、各駅から駅半径の2倍までの距離にある線路は駅への接続があると仮定し、なるべく誤差の小さい線路を選択する。線路が駅半径の2倍以上距離が離れている場合、駅への移動・停車が不可であるとして設定エラーとなる。

7.5.5. 経由交差点設定

バスが経由する交差点を指定する。経由交差点の設定を行う場合、必ず"Line"の定義が行われていなければならない。経由する交差点は停車するバス停の順に、次のバス停に向かう際に経由すべき交差点名を指定(コロン区切りで複数指定可)する。経由交差点を指定しない場合、バスの経路は最短の経路が自動で設定される。

Route,< 交差点名 1:交差点名 2:>,<交差点名 3:交差点名 4:>,<交差点名 5:交差点名 6:>

7.5.6. 運行スケジュール設定

駅/バス停設定した停車駅(バス停)に対して、停車時刻(時刻表)を設定する。運行スケジュール設定を行う場合、必ず"Stop"の定義が行われていなければならない。運行スケジュールは車両名("Family"で定義した文字列)から開始し、その車両が駅に停車している時刻(到着時刻-発射時刻)を駅/バス停設定と対応させて記述する。0は00:00 AMを指す。

<車両名>,<到着時刻>-<発車時刻>,<到着時刻>-<発車時刻>,<到着時刻>-<発車時刻>,<到着時刻>-<発車時刻>

※到着してすぐに発車する場合、<到着時刻>のみの記述としても良い。

到着時刻、発射時刻は時系列順に指定する。運行スケジュールの項目は車両の本数分、時系列順に記述する。シミュレーション終了時刻以降の交通機関のスケジュールはスキップされる。スキップされた車両を利用する経路はエージェントの経路計算では考慮されない。

エージェントタイムテーブル設定ファイルの記述例を示す。

```
# >>>> Agent Time Table <<<<
#
#---+ Rule
#
# (Vehicle Size)
```

```

#
# "Family", <vehicle family name>, <width>, <length>, <capacity>
#
# (Timetable)
#
# "Line", <line name>, <"Train" or "Bus">
# "Price", <Price/Distance(meters)>
# "Stop", <stop station name1>, <stop station name2>,...
# <vehicle family name>, <arrival time>"-<departure time>,...
#
# Family,Vehicle0,4,24,250,0

Line,TrainLine1,Train
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,station1,station2,station3,station4,station5
0,Vehicle0,20-30,60-70,100-110,140-150,180-190
1,Vehicle0,50-60,90-100,130-140,170-180,210-220
2,Vehicle0,80-90,120-130,160-170,200-210,240-250
3,Vehicle0,110-120,150-160,190-200,230-240,270-280
4,Vehicle0,150-160,190-200,230-240,270-280,310-320
5,Vehicle0,200-210,240-250,280-290,320-330,360-370
6,Vehicle0,230-240,270-280,310-320,350-360,390-400
7,Vehicle0,260-270,300-310,340-350,380-390,420-430
8,Vehicle0,290-300,330-340,370-380,390-400,430-440
9,Vehicle0,320-330,360-370,400-410,440-450,480-490
10,Vehicle0,350-360,390-400,430-440,470-480,510-520
11,Vehicle0,380-390,420-430,460-470,500-510,540-550
12,Vehicle0,410-420,450-460,490-500,530-540,570-580
13,Vehicle0,440-450,480-490,520-530,560-570,600-610
14,Vehicle0,470-480,510-520,550-560,590-600,630-640
15,Vehicle0,500-510,540-550,580-590,620-630,660-670
16,Vehicle0,550-560,600-610,650-660,700-710,750-760

Family,Vehicle1,4,24,250,0

```

```

Line,TrainLine2,Train
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,station5,station4,station3,station2,station1
0,Vehicle1,20-30,60-70,100-110,140-150,180-210
1,Vehicle1,50-60,90-100,130-140,170-180,210-240
2,Vehicle1,80-90,120-130,160-170,200-210,240-270
3,Vehicle1,110-120,150-160,190-200,230-240,270-300
4,Vehicle1,150-160,190-200,230-240,270-280,310-340
5,Vehicle1,200-210,240-250,280-290,320-330,360-390
6,Vehicle1,230-240,270-280,310-320,350-360,390-420
7,Vehicle1,260-270,300-310,340-350,380-390,420-450
8,Vehicle1,290-300,330-340,370-380,390-400,430-460
9,Vehicle1,320-330,360-370,400-410,440-450,480-510
10,Vehicle1,350-360,390-400,430-440,470-480,510-540
11,Vehicle1,380-390,420-430,460-470,500-510,540-570
12,Vehicle1,410-420,450-460,490-500,530-540,570-600
13,Vehicle1,440-450,480-490,520-530,560-570,600-630
14,Vehicle1,470-480,510-520,550-560,590-600,630-660
15,Vehicle1,500-510,540-550,580-590,620-630,660-690
16,Vehicle1,550-560,600-610,650-660,700-710,750-780

Family,Vehicle2,3,9,50,0

Line,BusLine1,Bus
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop1,busstop2,busstop3,busstop4,busstop5,busstop6,busstop7
Vehicle2,10-20,50-60,120-130,190-200,240-250,310-320,340-350
Vehicle2,40-50,80-90,150-160,220-230,270-280,340-350,370-380
Vehicle2,70-80,110-120,180-190,250-260,300-310,370-380,400-410
Vehicle2,110-120,150-160,220-230,290-300,340-350,410-420,440-450
Vehicle2,160-170,200-210,270-280,340-350,390-400,460-470,490-500
Vehicle2,210-220,250-260,320-330,390-400,440-450,510-520,540-550
Vehicle2,260-270,300-310,370-380,440-450,490-500,560-570,590-600
Vehicle2,310-320,350-360,420-430,490-500,540-550,610-620,640-650
Vehicle2,360-370,400-410,470-480,540-550,590-600,660-670,690-700

```

```

Vehicle2,410-420,450-460,520-530,590-600,640-650,710-720,740-750
Vehicle2,460-470,500-510,570-580,640-650,690-700,760-770,790-800
Vehicle2,510-520,550-560,620-630,690-700,740-750,810-820,840-850
Vehicle2,560-570,600-610,670-680,740-750,790-800,860-870,890-900
Vehicle2,610-620,650-660,720-730,790-800,840-850,910-920,940-950
Vehicle2,660-670,700-710,770-780,840-850,890-900,960-970,990-1000
Vehicle2,710-720,750-760,820-830,890-900,940-950,1010-1020,1040-1050
Vehicle2,760-770,800-810,870-880,940-950,990-1000,1060-1070,1090-1100
Vehicle2,810-820,850-860,920-930,990-1000,1040-1050,1110-1120,1140-1150

```

Family,Vehicle3,3,9,50,0

```

Line,BusLine2,Bus
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop7,busstop6,busstop5,busstop4,busstop3,busstop2,busstop1
Vehicle3,10-20,50-60,130-140,190-200,240-250,290-300,320-330
Vehicle3,60-70,100-110,180-190,240-250,290-300,340-350,370-380
Vehicle3,110-120,150-160,230-240,290-300,340-350,390-400,420-430
Vehicle3,160-170,200-210,280-290,340-350,390-400,440-450,470-480
Vehicle3,210-220,250-260,330-340,390-400,440-450,490-500,520-530
Vehicle3,260-270,300-310,380-390,440-450,490-500,540-550,570-580
Vehicle3,310-320,350-360,430-440,490-500,540-550,590-600,620-630
Vehicle3,360-370,400-410,480-490,540-550,590-600,640-650,670-680
Vehicle3,410-420,450-460,530-540,590-600,640-650,690-700,720-730
Vehicle3,460-470,500-510,580-590,640-650,690-700,740-750,770-780
Vehicle3,510-520,550-560,630-640,690-700,740-750,790-800,820-830
Vehicle3,560-570,600-610,680-690,740-750,790-800,840-850,870-880
Vehicle3,610-620,650-660,730-740,790-800,840-850,890-900,920-930
Vehicle3,660-670,700-710,780-790,840-850,890-900,940-950,970-980
Vehicle3,710-720,750-760,830-840,890-900,940-950,990-1000,1020-1030
Vehicle3,760-770,800-810,880-890,940-950,990-1000,1040-1050,1070-1080
Vehicle3,810-820,850-860,930-940,990-1000,1040-1050,1090-1100,1120-1130

```

Family,Vehicle4,3,9,50,0

Line,BusLine3,Bus

```

Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop8,busstop9,busstop10,busstop11,busstop12,busstop13
Vehicle4,10-20,60-70,110-120,190-200,260-270,320-330
Vehicle4,60-70,100-110,160-170,240-250,310-320,370-380
Vehicle4,110-120,160-170,210-220,290-300,360-370,420-430
Vehicle4,160-170,200-210,260-270,340-350,410-420,470-480
Vehicle4,210-220,260-270,310-320,390-400,460-470,520-530
Vehicle4,260-270,300-310,360-370,440-450,510-520,570-580
Vehicle4,310-320,360-370,410-420,490-500,560-570,620-630
Vehicle4,360-370,400-410,460-470,540-550,610-620,670-680
Vehicle4,410-420,460-470,510-520,590-600,660-670,720-730
Vehicle4,460-470,500-510,560-570,640-650,710-720,770-780
Vehicle4,510-520,560-570,610-620,690-700,760-770,820-830
Vehicle4,560-570,600-610,660-670,740-750,810-820,870-880
Vehicle4,610-620,660-670,710-720,790-800,860-870,920-930
Vehicle4,660-670,700-710,760-770,840-850,910-920,970-980
Vehicle4,710-720,760-770,810-820,890-900,960-970,1020-1030
Vehicle4,760-770,800-810,860-870,940-950,1010-1020,1070-1080
Vehicle4,810-820,860-870,910-920,990-1000,1060-1070,1120-1130

```

Family,Vehicle5,3,9,50,0

```

Line,BusLine4,Bus
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop13,busstop12,busstop11,busstop10,busstop9,busstop8
Vehicle5,10-20,70-80,130-140,180-190,230-240,280-290
Vehicle5,60-70,120-130,180-190,230-240,280-290,330-340
Vehicle5,110-120,170-180,230-240,280-290,330-340,380-390
Vehicle5,160-170,220-230,280-290,330-340,380-390,430-440
Vehicle5,210-220,270-280,330-340,380-390,430-440,480-490
Vehicle5,260-270,320-330,380-390,430-440,480-490,530-540
Vehicle5,310-320,370-380,430-440,480-490,530-540,580-590
Vehicle5,360-370,420-430,480-490,530-540,580-590,630-640
Vehicle5,410-420,470-480,530-540,580-590,630-640,680-690
Vehicle5,460-470,520-530,580-590,630-640,680-690,730-740
Vehicle5,510-520,570-580,630-640,680-690,730-740,780-790

```

```

Vehicle5,560-570,620-630,680-690,730-740,780-790,830-840
Vehicle5,610-620,670-680,730-740,780-790,830-840,880-890
Vehicle5,660-670,720-730,780-790,830-840,880-890,930-940
Vehicle5,710-720,770-780,830-840,880-890,930-940,980-990
Vehicle5,760-770,820-830,880-890,930-940,980-990,1030-1040
Vehicle5,810-820,870-880,930-940,980-990,1030-1040,1080-1090

```

```
Family,Vehicle6,3,9,50,0
```

```

Line,BusLine5,Bus
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop14,busstop15,busstop16,busstop17
Vehicle6,10-20,60-70,120-130,160-170
Vehicle6,60-70,110-120,170-180,210-220
Vehicle6,110-120,160-170,220-230,260-270
Vehicle6,160-170,210-220,270-280,310-320
Vehicle6,210-220,260-270,320-330,360-370
Vehicle6,260-270,310-320,370-380,410-420
Vehicle6,310-320,360-370,420-430,460-470
Vehicle6,360-370,410-420,470-480,510-520
Vehicle6,410-420,460-470,520-530,560-570
Vehicle6,460-470,510-520,570-580,610-620
Vehicle6,510-520,560-570,620-630,660-670
Vehicle6,560-570,610-620,670-680,710-720
Vehicle6,610-620,660-670,720-730,760-770
Vehicle6,660-670,710-720,770-780,810-820
Vehicle6,710-720,760-770,820-830,860-870
Vehicle6,760-770,810-820,870-880,910-920
Vehicle6,810-820,860-870,920-930,960-970

```

```
Family,Vehicle7,3,9,50,0
```

```

Line,BusLine6,Bus
Price,100/1000,100/1000,0/1000,100/1000,100/1000
Stop,busstop18,busstop19,busstop20,busstop21
Vehicle7,10-20,60-70,120-130,170-180

```

```
Vehicle7,60-70,110-120,170-180,220-230
Vehicle7,110-120,160-170,220-230,270-280
Vehicle7,160-170,210-220,270-280,320-330
Vehicle7,210-220,260-270,320-330,370-380
Vehicle7,260-270,310-320,370-380,420-430
Vehicle7,310-320,360-370,420-430,470-480
Vehicle7,360-370,410-420,470-480,520-530
Vehicle7,410-420,460-470,520-530,570-580
Vehicle7,460-470,510-520,570-580,620-630
Vehicle7,510-520,560-570,620-630,670-680
Vehicle7,560-570,610-620,670-680,720-730
Vehicle7,610-620,660-670,720-730,770-780
Vehicle7,660-670,710-720,770-780,820-830
Vehicle7,710-720,760-770,820-830,870-880
Vehicle7,760-770,810-820,870-880,920-930
Vehicle7,810-820,860-870,920-930,970-980
```

#で始まる行はコメント行として扱う。

7.6. 信号パタン定義ファイル

信号パタン定義ファイルにより、信号機単位で指定可能な信号の制御パタンを定義可能である。制御パタン定義には次の二つの定義方法がある。

- 間隔指定
- スケジュール指定

- ファイル構文
 - 一行につき一項目記述する
 - 文字コード・改行コードはシミュレーション実行環境に合わせる。
 - 空行はスキップされる
 - 「#」で開始する行はコメント行としてスキップされる

7.6.1. 間隔指定

青、赤、黄を 1 サイクルとして周期ごとに同一の信号制御を繰り返す場合、間隔指定での制御パタンを設定する。

<制御パタン名> <青色時間> <黄色時間> <赤色時間>

制御パタンは何項目定義しても良い。制御パタン名の重複は不可である。

黄色時間の定義は将来の拡張のために予約された定義である。車両、歩行の移動モデルでは黄色は青色と同じ挙動で行動する。

例 1:0 秒から 30 秒まで青、30 秒から 40 秒まで黄、40 秒から 80 秒まで赤を一サイクルとする信号制御パタンを定義する。(80 秒から 110 秒まで青、110 秒から 120 秒まで黄、120 秒から 160 秒まで赤)

TrafficLightPatternA 30 10 40

```
#PatternName BlueDuration YellowDuration RedDuration
Default 30 10 40
Default1 30 11 41
```

7.6.2. スケジュール指定

時刻ごとに制御パタンが変化するような複雑な信号制御パタンを定義する場合、スケジュール指定で設定を行う。スケジュール指定では時刻ごとに信号が何色になるかを指定する。

<制御パタン名> <時刻>:<信号色> <時刻>:<信号色> <時刻>:<信号色> <時刻>:<信号色>

信号色は青(Blue)、黄(Yellow)、赤(Red)の三種類で指定可能で、時刻ごとの色の指定はいくつ設定しても良い。最初の時刻が 0 から開始していない場合、最初の時刻指定までの間は青色の信号として扱う。

制御パタンは何項目定義しても良いが、制御パタン名の重複は不可である。

信号の切り替わるタイミングは GIS 情報の更新の間隔で行うため、GIS 情報の同期間隔に依存する。信号の切り替わりのタイミングを正確に行う場合、時刻の指定を GIS 情報の同期間隔の倍数で指定することが望ましい。

Yellow の定義は将来の拡張のために予約された定義である。車両、歩行の移動モデルでは Yellow は Blue と同じ挙動で行動する。

指定によって交差点の信号機の色が矛盾する場合もあるが、その場合人や車は自身の対象としている信号だけを見て移動する。

例 1:0 秒から 30 秒まで青、30 秒から 40 秒まで黄、40 秒から 80 秒まで赤、80 秒から 110 秒まで青、110 秒から 120 秒まで黄、120 秒以降シミュレーションの終了まで赤の信号制御パタンを定義する。

```
TrafficLightPatternB 0:Blue 30:Yellow 40:Red 80:Blue 110:Yellow 120:Red
```

```
#PatternName Time1:LightType(Blue/Yellow/Red) Time2:LightType Time3:LightType...
Default2 0:Blue 70:Red 90:Blue 100:Red
```

7.7. シープファイル

マルチエージェントシミュレーションで利用可能なシェープファイルには次の種類がある。シェープファイルの詳細に関しては「Base Simulator ModelReference」を参照。

ファイル種別	内容
道路ファイル(road.shp)	道路情報
交差点ファイル(intersection.shp)	交差点情報
信号ファイル(trafficlight.shp)	信号情報
バス停ファイル(busstop.shp)	バス停情報
建物ファイル(building.shp)	建物情報
公園ファイル(park.shp)	公園情報
線路ファイル(rail.shp)	線路情報
駅ファイル(station.shp)	駅情報
入り口ファイル(entrance.shp)	入り口情報
エリアファイル(area.shp)	エリア情報
POI ファイル(poi.shp)	POI 情報

7.7.1. 道路ファイル

歩行者/車両共用の道路の場合、歩行者は道路の端を歩行し、車両は車線の中央を走行する。歩行者専用の道路の場合、車両は走行不可であり歩行者は道路上の任意の位置を移動する。車両専用の道路の場合、歩行者は移動不可であり、車両は車線の中央を走行する。

キャパシティを超える歩行者は道路に進入することが出来ない。キャパシティを超えた状態で道路に進入しようとする歩行者は道路端で停止しキャパシティに空きが生じるのを待つ。複数人の歩行者が進入待ちの状態である場合、キャパシティに空きが出た時点で待ち時間が長い歩行者から優先的に進入を行う。

7.7.2. バス停ファイル

バス停は、バス停に指定した点から最も近い道路上の点に自動で作成される。バス停の名前に指定した文字列は駅/バス停設定でバスが停車する場所として利用可能である。

キャパシティを超える歩行者は バス停で待つことが出来ない。キャパシティを超えた状態でバス停に進入しようとする歩行者は停止しキャパシティに空きが生じるのを待つ。複数人の歩行者が待ち状態である場合、キャパシティに空きが出た時点で待ち時間が長い歩行者から優先的にバス停待ちを行う。バスに乗車可能なエージェントはバス停で待っている人のみである。

7.7.3. 建物ファイル

建物の名前に指定した文字列はエージェントの目的地として利用可能である。

キャパシティを超える人数は建物に入ることが出来ない。キャパシティを超えた状態でその建物を唯一の目的地としている人は入り口で停止しキャパシティに空きが生じるのを待つ。複数人が入り口で待っている場合、キャパシティに空きが出た時点で待ち時間が長い人から優先的に建物に入る。

建物の入り口は、入り口ファイルで建物内に設定された入り口に対して作成される。また、入り口数がコンフィグレーションファイルの GIS-NUMBER-ENTRANCES-TO-BUILDING で指定された数を満たしていないければ、足りない入り口数分、建物の基準となる頂点から順番に頂点をたどり、壁の中点に対して入り口を作成する。

入り口が道路と接続していない場合、入り口の点から最寄りの道路の点に対して接続を作成する。

7.7.4. 公園ファイル

公園の名前に指定した文字列はエージェントの目的地として利用可能である。

キャパシティを超える人数は公園に入ることが出来ない。キャパシティを超えた状態でその公園を唯一の目的地としている人は入り口で停止しキャパシティに空きが生じるのを待つ。複数人が入り口で待っている場合、キャパシティに空きが出た時点で待ち時間が長い人から優先的に公園に入る。

公園の入り口は、入り口ファイルで公園内に設定された入り口に対して作成される。また、入り口数がコンフィグレーションファイルの GIS-NUMBER-ENTRANCES-TO-PARK で指定された数を満たしていないければ、足りない入り口数分、公園の基準となる頂点から順番に頂点をたどり、辺の中点に対して入り口を作成する。

入り口が道路と接続していない場合、入り口の点から最寄りの道路の点に対して接続を作成する。

7.7.5. 線路ファイル

線により電車の移動する線路を定義する。エージェントタイムテーブル設定ファイルの駅/バス停設定で設定される駅が順番に接続するよう、駅と線路の接続を行う必要がある。

7.7.6. 駅ファイル

駅の名前に指定した文字列はエージェントの目的地として利用可能である。

キャパシティを超える人数は駅に入ることが出来ない。キャパシティを超えた状態で駅に入ろうとする人は入り口で停止しキャパシティに空きが生じるのを待つ。複数人が入り口で待っている場合、キャパシティに空きが出た時点で待ち時間が長い人から優先的に駅に入る。電車に乗車可能なエージェントは駅に入って待機している人のみである。

7.7.7. 線路ファイル

点により建物または公園の入り口を定義する。入り口では、指定した流入レートまでの人の流入が可能となる。それ以上の人人が入り口を利用する場合には、入り口で整列して待機する。

7.7.8. エリアファイル

エリアの名前に指定した文字列はエージェントの目的地として利用可能である。

7.7.9. POI ファイル

POI の名前に指定した文字列はエージェントの目的地として利用可能である。

キャパシティを超える人数は POI に入ることが出来ない。キャパシティを超えた状態でその POI を唯一の目的地としている人は入り口で停止しキャパシティに空きが生じるのを待つ。複数人が入り口で待っている場合、キャパシティに空きが出た時点で待ち時間が長い人から優先的に POI に入る。

7.7.10. シェーブの作成について

qgis を利用した shp を利用する場合、shape ファイルの出力が m 単位系となっているのに対して、出力される prj ファイルが緯度経度単位の指定となっている場合がある。

編集した shp が原点,0 の直交座標系で m 単位となっている場合、projection の選択で[Mercator] -> WGS 84/World Mercator 等を選択することで読み込みが可能である。

7.8. OSM(OpenStreetMap)ファイル

Visual Lab からシナリオを作成する際に、OpenStreetMap の地図を利用可能である。

7.8.1. ファイルの入手

利用可能な地図は、OpenStreetMap から入手可能な OSM 形式(xml 形式)のファイルである。

OpenStreetMap URL: <http://www.openstreetmap.org/>

OSM 形式のファイル OpenStreetMap ウェブサイトのエクスポートオプションより地図の範囲を指定し、OpenStreetMap XML データを選択することで OSM 形式のファイルをダウンロードが可能である。ウェブサイトよりダウンロード可能な OSM 形式のファイルは 5 万オブジェクト程度のエリアに制限されるため、広範囲の地図を含んだ OSM 形式のファイルを利用する場合、PlanetOSM より全ての地図(日本全国等)をダウンロードした後、Osmosis(コマンドラインツール)や JOSM(GUI ツール)等のソフトを利 用し、必要なエリアの地図を OSM 形式のファイルとして切り出す。

PlanetOSM URL: <http://planet.openstreetmap.org/>

(データは、.osm またはバイナリ形式の.pbf でダウンロードが可能)

OSM の情報の切り出し例

OSM の情報の切り出しには Osmosis を利用するのが便利である。Osmosis は以下よりダウンロー ドが可能である。

Osmosis: <http://wiki.openstreetmap.org/wiki/Osmosis>

Windows での切り出し例 (入力には OSM のバイナリ形式である pbf を利用)

```
- 関東・首都圏の切り出し
.¥osmosis-latest¥bin¥osmosis --rbf japan-latest.osm.pbf --bounding-box
top=35.1363 left=139.7000 bottom=34.5323 right=140.9903 --wb tokyo.pbf

- 主要道路(secondry まで)を抽出
.¥osmosis-latest¥bin¥osmosis --rbf tokyo.pbf --tf accept-ways highway=* --tf
reject-ways
highway=pedestrian,footway,steps,cycleway,bridleway,residential,path,living_s
treet,track,service,tertiary,tertiary_link,road,elevator,proposed,constructio
n,raceway,bus_guideway,unclassified --tf reject-relations --used-node --wb
```

```
tokyo-road.pbf
```

(さらに細かい道路を含める場合には、--tf reject-ways highway=...の項目から tertiary,tertiary_link を外して実行する。)

- 切り出した主要道路(secondaryまで)を OSM に変換

```
.¥osmosis-latest¥bin¥osmosis --rbf tokyo-road.pbf --wx tokyo-road.osm
```

- 名前付き建物を抽出

```
.¥osmosis-latest¥bin¥osmosis --rbf tokyo.pbf --tf accept-ways building=* --tf accept-ways name=* --tf reject-relations --used-node --wb tokyo-building-name.pbf
```

- 高さ情報の付与された建物を抽出

```
.¥osmosis-latest¥bin¥osmosis --rbf tokyo.pbf --tf accept-ways building=* --tf accept-ways height=* --tf reject-relations --used-node --wb tokyo-building-height.pbf
```

- 抽出した道路、建物情報を結合して一つの OSM に変換

```
.¥osmosis-latest¥bin¥osmosis --rbf tokyo-building-name.pbf --rbf tokyo-building-height.pbf --merge --wb tokyo-building-name-height.pbf .¥osmosis-latest¥bin¥osmosis --rx tokyo-road.pbf --rbf tokyo-building-name-height.pbf --merge --wx tokyo-road_building.osm
```

※本例で作成した OSM を Visual Lab で読み込む場合、3G 程度のメモリ領域が必要となるので注意。また、本例のような広域エリアで多数のオブジェクトを含んだ OSM の場合、事前に Visual Lab のメニュー > Tools > Options > Display より、デフォルトの読み込みモードを Display に変更してから Import を行うことで高速に読み込みが可能となる。

Visual Lab での OSM ファイルの読み込みは一つファイルで行うため、必要なエリアのデータは一つの OSM ファイルにまとまっている必要があるため、複数の OSM を利用する場合は Osmosis、JOSM 等であらかじめマージを行っておく。

7.8.2. ファイルの読み込み

OSM ファイルでシナリオに反映されるデータを解説する。

エリア情報

OSM ファイル(XML 形式)の 3 行目に記述されるエリア情報を利用する。

OSM ファイル例

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="CGImap 0.0.2">
  <bounds minlat="-33.9126000" minlon="151.1681000" maxlat="-33.8532000"
  maxlon="151.2692000"/>
  <node id="324883" lat="-33.9176762" lon="151.1888395" user="Ebenezer"
  uid="20949" visible="true" version="3" changeset="4228056"
  timestamp="2010-03-25T10:25:53Z"/>
  <node id="324884" lat="-33.9108977" lon="151.1949969" user="Elaena"
  uid="407085" visible="true" version="3" changeset="7252463"
  timestamp="2011-02-11T07:36:24Z">
    <tag k="highway" v="traffic_signals"/>
  </node>
  <node id="624529" lat="-33.8976541" lon="151.2276855" user="Franc"
  uid="7617" visible="true" version="2" changeset="6704320"
  timestamp="2010-12-19T10:50:32Z"/>
  <node id="624532" lat="-33.8972163" lon="151.2288961"
  user="MichaelCollinson" uid="308" visible="true" version="1"
  changeset="2769" timestamp="2006-03-14T09:29:00Z"/>
```

- Point オブジェクト

Point オブジェクトで読み込み可能な項目を示す。

Key	Value	補足
highway	traffic_signals	オブジェクトを信号として読み込む
	bus_stop	オブジェクトをバス停として読み込む
railway	station	オブジェクトを駅として読み込む
name	文字列	オブジェクトの名前
amenity	bus_station	オブジェクトをバス停として読み込む

- Way オブジェクト

Way オブジェクトで読み込み可能な項目を示す。

Key	Value	補足
highway	motorway trunk trunk_link primary primary_link secondary secondary_link tertiary residential unclassified road lining_street service track pedestrian raceway services bus_guideway path cycleway footway bridleway byway steps	オブジェクトを道路として読み込む
	bus_stop	オブジェクトをバス停として読み込む
railway	subway rail monorail	オブジェクトを線路として読み込む
	station	オブジェクトを駅として読み込む
building		オブジェクトを建物として読み込む
leisure		オブジェクトを公園として読み込む
natural	coastline	オブジェクトを海岸線(Line)として読み込む
	wood	オブジェクトを森林(Polygohn)として読み

		込む
landuse	forest	オブジェクトを森林(Polygohn)として読み込む
lanes	整数	道路のレーン数
name	文字列	オブジェクトの名前

8. 行動アルゴリズム

エージェントの行動アルゴリズムの詳細について解説する。

8.1. 状態遷移

エージェントの状態遷移を示す

エージェントの状態	トリガ	エージェントの状態遷移先
共通		
初期状態	行動の決定	待機状態
待機状態	行動開始時刻	移動開始状態
移動開始状態	経路計算待ち	経路問い合わせ状態
経路問い合わせ状態	経路候補決定	経路決定状態
経路決定状態	移動開始時刻	移動状態
移動状態	移動手段決定(自由歩行)	移動状態(自由歩行)
移動状態	移動手段決定(車)	移動状態(自家用車ドライバ: 車への歩行移動)
移動状態	移動手段決定(公共交通機関)	移動状態(電車・バスゲスト: 停留所への歩行移動)
移動状態	移動手段決定(タクシーゲスト)	移動状態(タクシーゲスト: 待機)
移動状態	移動手段決定(自転車)	移動状態(自転車)
移動状態	移動手段決定(歩行)	移動状態(道路歩行)
移動状態	移動手段決定(バスドライバ)	移動状態(バスドライバ: 待機)
移動状態	移動手段決定(タクシードライバ)	移動状態(タクシードライバ: 待機)
移動状態	移動手段決定(電車ドライバ)	移動状態(電車ドライバ: 待機)
移動完了状態 (※via-pointに到着)	経路計算待ち	経路問い合わせ状態
移動完了状態	移動開始時刻	移動開始状態
移動完了状態	目的地へ到着	待機状態
停止状態	-	-
自由歩行		
移動状態(自由歩行)	目的地に到着	移動完了状態
入場待ち状態	建物または公園に空きがある	移動状態(自由歩行)
移動状態(自家用車ドライバ: 目的地への移動)	経路再計算タイミング	経路問い合わせ状態
自家用車ドライバ		

移動状態(自家用車ドライバ:目的地への移動)	目的地へ到着	移動完了状態
移動状態(自家用車ドライバ:車への歩行移動)	車の止まっている頂点に到達	移動状態(自家用車ドライバ:目的地への移動)
移動状態(自家用車ドライバ:目的地への移動)	交差点において経路再計算トリガのいずれかを満たした - DelayForLastViaPoint - DelayForNextViaPoint - DelayForExpectedArrivalTime - 道路の混雑度 - Utility1 - Utility2	自家用車経路再計算状態
移動状態(自家用車ドライバ:目的地への移動)	利用中の道路が無効状態となった	停止状態
移動状態(自家用車ドライバ:目的地への移動)	経路再計算タイミング	自家用車経路再計算状態
自家用車経路再計算状態	経路決定	移動状態(自家用車ドライバ:目的地への移動)
電車・バスゲスト		
移動状態(電車・バスゲスト:降車)	via-point へ到着	移動完了状態
移動状態(電車・バスゲスト:停留所への歩行移動)	停留所へ到着	移動状態(電車・バスゲスト:乗車待ち)
移動状態(電車・バスゲスト:乗車待ち)	交差点において経路再計算トリガのいずれかを満たした - DelayForLastViaPoint - DelayForNextViaPoint - DelayForExpectedArrivalTime - Utility1 - Utility2	経路問い合わせ状態
移動状態(電車・バスゲスト:乗車待ち)	経路再計算タイミング	経路問い合わせ状態
移動状態(電車・バスゲスト:乗車待ち)	車両に乗れなかった	経路問い合わせ状態
移動状態(電車・バスゲスト:乗車待ち)	交通機関到着	移動状態(電車・バスゲスト:乗車)

移動状態(電車・バスゲスト:乗車)	降車地点に到着	移動状態(電車・バスゲスト:降車)
移動状態(電車・バスゲスト:降車)	乗り換え経路あり	移動状態(電車・バスゲスト:停留所への歩行移動)
タクシーゲスト		
移動状態(タクシーゲスト:乗車)	目的地へ到着	移動完了状態
移動状態(タクシーゲスト:待機)	タクシー到着	移動状態(タクシーゲスト:乗車)
移動状態(タクシーゲスト:乗車)	経路再計算タイミング	経路問い合わせ状態
移動状態(タクシーゲスト:待機)		
自転車		
移動状態(自転車)	目的地へ到着	移動完了状態
移動状態(自転車)	交差点において経路再計算トリガのいずれかを満たした - DelayForLastViaPoint - DelayForNextViaPoint - DelayForExpectedArrivalTime - 道路の混雑度 - Utility1 - Utility2	経路問い合わせ状態
移動状態(自転車)	経路が行き止まりだった	経路問い合わせ状態
移動状態(自転車)	経路再計算タイミング	経路問い合わせ状態
移動状態(自転車)	利用中の道路が無効状態となった	停止状態
道路歩行		
移動状態(道路歩行)	目的地へ到着	移動完了状態
移動状態(道路歩行)	交差点において経路再計算トリガのいずれかを満たした - DelayForLastViaPoint - DelayForNextViaPoint - DelayForExpectedArrivalTime - 道路の混雑度 - Utility1 - Utility2	経路問い合わせ状態

移動状態(道路歩行)	経路が行き止まりだった	経路問い合わせ状態
移動状態(道路歩行)	経路再計算タイミング	経路問い合わせ状態
移動状態(道路歩行)	利用中の道路が無効状態となった	停止状態
移動状態(道路歩行)	目的地の建物または公園に到着した	入場待ち状態(自由歩行)
移動状態(道路歩行)	次の道路へ移動	入場待ち状態(道路歩行)
入場待ち状態(道路歩行)	次の道路に空きがある	移動状態(道路歩行)
バスドライバ		
移動状態(バスドライバ:待機)	発車時刻かつ、全員の乗車が完了	移動状態(バスドライバ:次のバス停まで移動)
移動状態(バスドライバ:次の停留所まで移動)	停留所に到着	移動状態(バスドライバ:待機)
移動状態(バスドライバ:次の停留所まで移動)	経路が行き止まりだった	バスドライバ経路再計算状態
移動状態(バスドライバ:次の停留所まで移動)	利用中の道路が無効状態となった	停止状態
バスドライバ経路再計算状態	経路決定	移動状態(バスドライバ:次の停留所まで移動)
タクシードライバ		
移動状態(タクシードライバ:待機)	予約確定	移動状態(タクシードライバ:予約のあったエージェントまで移動)
移動状態(タクシードライバ:予約のあったエージェントまで移動)	予約のあったエージェントの地点に到着	移動状態(タクシードライバ:乗車待機)
移動状態(タクシードライバ:乗車待機)	エージェントが乗車	移動状態(タクシードライバ:目的地へ移動)
移動状態(タクシードライバ:目的地へ移動)	乗車しているエージェントの目的地に到着	移動状態(タクシードライバ:降車待機)
移動状態(タクシードライバ:降車待機)	エージェントが降車	移動状態(タクシードライバ:タクシーの登録された駅まで移動)
移動状態(タクシードライバ:タクシーの登録された駅まで移動)	タクシーの登録された駅に到着	移動状態(タクシードライバ:待機)
移動状態(タクシードライバ:予約のあったエージェントまで移動)	交差点において経路再計算トリガのいずれかを満たした	タクシー経路再計算状態

エントまで移動) 移動状態(タクシードライバ: 目的地へ移動) 移動状態(タクシードライバ: タクシーの登録された駅まで移動)	- DelayForLastViaPoint - DelayForNextViaPoint - DelayForExpectedArrivalTime - 道路の混雑度 - Utility1 - Utility2	
移動状態(タクシードライバ: 予約のあったエージェントまで移動) 移動状態(タクシードライバ: 目的地へ移動) 移動状態(タクシードライバ: タクシーの登録された駅まで移動)	経路が行き止まりだった	タクシー経路再計算状態
移動状態(タクシードライバ: 予約のあったエージェントまで移動) 移動状態(タクシードライバ: 目的地へ移動) 移動状態(タクシードライバ: タクシーの登録された駅まで移動)	経路再計算タイミング	タクシー経路再計算状態
移動状態(タクシードライバ: 予約のあったエージェントまで移動) 移動状態(タクシードライバ: 目的地へ移動) 移動状態(タクシードライバ: タクシーの登録された駅まで移動)	利用中の道路が無効状態となった	停止状態
タクシー経路再計算状態	経路決定	移動状態(タクシードライバ)
電車ドライバ		
移動状態(電車ドライバ: 待機)	発車時刻	移動状態(電車ドライバ: 次の駅まで移動)
移動状態(電車: 次の駅まで移動)	駅に到着	移動状態(電車ドライバ: 待機)

移動状態(電車ドライバ:待機)	利用している線路が無効状態となつた	停止状態
移動状態(電車:次の駅まで移動)		

8.2. 自由歩行

自由歩行には建物または公園に進入する場合と外に出る行動がある。

- 建物、公園内に進入する行動

建物・公園の入り口の座標から、目的地に指定されている点に対して直線的に歩行移動を行う。目的地点は建物・公園ポリゴン内のランダムな点に決定する。

歩行速度 = エージェントのプロファイルにおいて WalkSpeed で設定した値

目的地に到達した時点で、「目的地に到達の通知」を行う。

- 建物または公園内から外に出る行動

建物・公園内から、入り口に対して直線的に歩行移動を行う。

歩行速度 = エージェントのプロファイルにおいて WalkSpeed で設定した値

8.3. 道路歩行

道路歩行には道路の移動行動と、交差点を渡る行動がある。

- 道路の移動行動

道路端を歩行行動する。

歩行速度 = エージェントのプロファイルにおいて WalkSpeed で設定した値 × 道路の混雑度(※
1m/s 以下の値となる場合は 1m/s)

- 交差点を渡る行動

交差点を跨いで道路を移動する場合は、最も交差点を横切る回数が少なくなる方向に交差点の横断を行う。交差点のエリアは隣り合う道路の外線の交点部分までとなり、交差点エリアを沿うようにして横断を行う。

信号がある場合は、横断する道路の(車両用の)信号が赤の場合に横断を行う。歩行速度より逆算して交差点を横断中に信号が赤になる場合は横断を行わずに、次に信号が赤になるタイミングで横断を開始する。信号が無い交差点については車の有無に関係なく横断を行う。

歩行速度 = エージェントのプロファイルにおいて WalkSpeed で設定した値

- 経路変更のタイミング

経路変更の要求があった場合、次に到達した交差点において経路の変更が適用される。

8.4. 自転車

歩行者と同一の行動を行う。速度は WalkSpeed の代わりに BicycleSpeed を利用する。

8.5. 自家用車ドライバ

自家用車ドライバには移動行動と、交差点を渡る行動がある。

- 道路の移動行動

道路の車線の中心を移動する。道路を移動する際にはドライバーモデルの一つとして知られている Intelligent Drivers Model(IDM)[1]を用いた車両の挙動を行う。

前方車両はこれから通過する道路について、100m 前方の距離まで確認を行う。一つの道路に複数車線存在する場合、左折は左レーン、右折は右レーンから行う。エージェントは次に曲がる方向を考慮し交差点の手間で必ず所望のレーンに移動する。所望のレーンに移れない場合、停止して車線変更が可能になるのを待つ。直進する場合はなるべく右車線を走行し、追い抜きが可能である場合は追い抜きを行う。

1) 追従走行モデル

前方の車両より次の式を用いて加速度を決定して移動する。

$$\text{加速度} \left(\frac{dv}{dt} \right) = a \left[1 - \left(\frac{v}{v_0} \right)^{\delta} - \left(\frac{s^*(v, \Delta v)}{s} \right)^2 \right]$$

$$s^*(v, \Delta v) = s_0 + \max \left[0, \left(vT + s1 \sqrt{\frac{v}{v_0}} + \frac{v\Delta v}{2\sqrt{ab}} \right) \right]$$

パラメータ	値
a	最大加速度

v_0	所望速度 (プロファイルの最大速度と、道路の制限速度で小さい方の値)
δ	加速度指数 (所望速度に対して、どの程度積極的に加速度を大きくしていくか)
v	現在の速度
Δv	前方車両との速度差
s_0	最小車間距離
s_1	所望速度に対する速度差の係数(通常の IDM モデル[1]では、 $s_1=0$)
s	前方車両との車間
T	ヘッドウェイ

2) 車線変更モデル

次の条件を満たした場合に車線変更を行う。

$$\begin{aligned} acc'(B') &> -b_{\text{save}} \\ acc'(M') - acc(M) &> \rho(acc'(B') - acc'(B)) + a_{\text{thr}} \end{aligned}$$

パラメータ	値
$acc(M)$	レーン変更前の車両の加速度
b_{save}	車線変更による(後方車両の)最大減速度 (車線変更によって、変更側の後方車両に減速が生じる場合の、後方車両の最大の減速度)
$acc'(M')$	レーン変更後の車両の加速度
$acc(B)$	レーン変更前の後方の車両の加速度
$acc'(B')$	レーン変更後の後方の車両の加速度
a_{thr}	所望速度
ρ	係数($= 1$)

- 交差点を渡る行動

交差点を通過する場合、次の項目を全て満たした場合に通過を行う。

- 1) 歩行者が横断していない。または歩行者が横断しているがエージェントのプロファイルで指定された歩行者認識確率(PedestrianRecognitionProbability)を満たさなかった。(※歩行者認識確率の判定は同期間隔の度に行う。)
- 2) 信号が青色または黄色。
- 3) 次に移動する道路に自身の車両の大きさ分の空きがある。
- 4) 他に交差点を移動する車両がない。または他に交差点を移動する車両がいるが、お互いが交錯しない移動であるか自身の方が優先である。

- 経路変更のタイミング

経路変更の要求があった場合、現在移動を予定している道路(100m 前方)の交差点まで移動した後、経路の切り替えを行う。

8.6. バスドライバ

バスドライバは出来る限りスケジュールに従ってバス停間の移動を行う。道路上での車両の挙動は自家用車ドライバと同様に IDM モデル[1]に従う。

- バス停間の移動

運行ラインで指定した最初のバス停から、スケジュールに従って移動を行う。予定時刻よりも早くバス停に到着した場合でも出発時刻まではバス停で待機する。遅延してバス停に到着した場合、バス停で待っている人を定員まで乗せ終わったらすぐに出発する。

- 経路変更のタイミング

経路変更の要求があった場合、現在移動を予定している道路(100m 前方)の交差点まで移動した後、経路の切り替えを行う。。

8.7. バスゲスト

バスゲストはバスに乗車して移動する。

- 乗車

道路の中心線状のバス停で乗車予定のバスが到着するのを待つ。バスが到着した場合、先にバス停に到着して待っていた人から乗車を行う。乗車後は車両内の任意の位置に歩行移動し停止する。定員となり予定していたバスに乗れなかった場合やバス停に到着したがバスに乗り遅れてしまった場合、エージェントプロファイルの乗車失敗の経路検索確率(RecalcProbWhenMissingAVehicle)を満たしているか計算する。確率を満たしていれば経路の再計算を行う。(※経路の再計算の確率の判定は同期間隔のたびに行う。)

バスに乗車する際には、エージェントプロファイルで指定した BoardingDelay の時間を要する。

乗り換えの必要がある場合は、歩行移動により乗り換えの電車またはバスへと乗り換える。

- 降車

降車予定のバス停に到着した際に乗車した点まで移動して降車する。乗車中に経路の再計算が発生し、別の経路・移動手段が良いと判断した場合には途中のバス停で降車する。エージェントタイムテーブル設定ファイルの運賃設定に従い運賃を支払う。

8.8. 電車ドライバ

電車ドライバはスケジュールに従って駅間の移動を行う。

8.9. 電車ゲスト

電車ゲストは電車に乗車して移動する。

- 乗車

駅内の所望の路線への乗車位置で乗車予定の電車が到着するのを待つ。電車が到着した場合、先に駅に到着して待っていた人から乗車を行う。乗車後は車両内の任意の位置に移動し停止する。定員となり予定していた電車に乗れなかった場合や駅に到着したが電車に乗り遅れてしまった場合、エージェントプロファイルの乗車失敗の経路検索確率(RecalcProbWhenMissingAVehicle)を満たしているか計算する。確率を満たしていれば経路の再計算を行う。(※経路の再計算の確率の判定は同期間隔のたびに行う。)

乗り換えの必要がある場合は、歩行移動により乗り換えの電車またはバスへと乗り換える。

- 降車

降車予定の駅に到着した際に乗車した点まで移動して降車する。乗車中に経路の再計算が発生し、別の経路・移動手段が良いと判断した場合には途中の駅で降車する。エージェントタイムテーブル設定ファイルの運賃設定に従い運賃を支払う。

8.10. タクシードライバ

タクシードライバは、タクシーゲストへの移動、目的地への移動、駅への移動の行動を行う。道路上での車両の挙動は自家用車ドライバと同様に IDM モデル[1]に従う。

- タクシーゲストへの移動

タクシーの呼び出しがあった場合、最寄りの駅よりタクシーが発生する。タクシーは呼び出しを行ったタクシーゲストの場所まで移動を行う。タクシーゲストの呼び出しがあった場所に移動したがタクシーゲストが居ない場合は駅へ戻る。

- 目的地への移動

タクシーゲストの保持している目的地までの経路に従って、目的まで移動する。経路の再計算が発生した場合は、タクシードライバの判断で目的地までの経路を選択する。(※タクシーゲスト側経路の再計算を行った場合は、タクシーゲストの指定した経路に従う。)。

タクシーゲストを目的地まで乗せた場合、タクシーゲストが途中で降車した場合は所得する駅への移動を行う。

- 駅への移動

所属する駅への移動が完了したら、次のタクシーの呼び出しがあるまで待機する。

8.11. タクシーゲスト

タクシーゲストはタクシーに乗車して移動する。

- タクシー待ち

道路上でタクシーの到着を停止して待つ。待機中に経路の再計算が発生しタクシー以外の経路が選択された場合、タクシーを無視して行動を開始する。

- 乗車

道路上でタクシーの到着を待つ。タクシーが到着したらタクシーに乗車する。タクシーが目的地に着いたら降車する。同時に乗車することが可能なエージェント数は一人である。

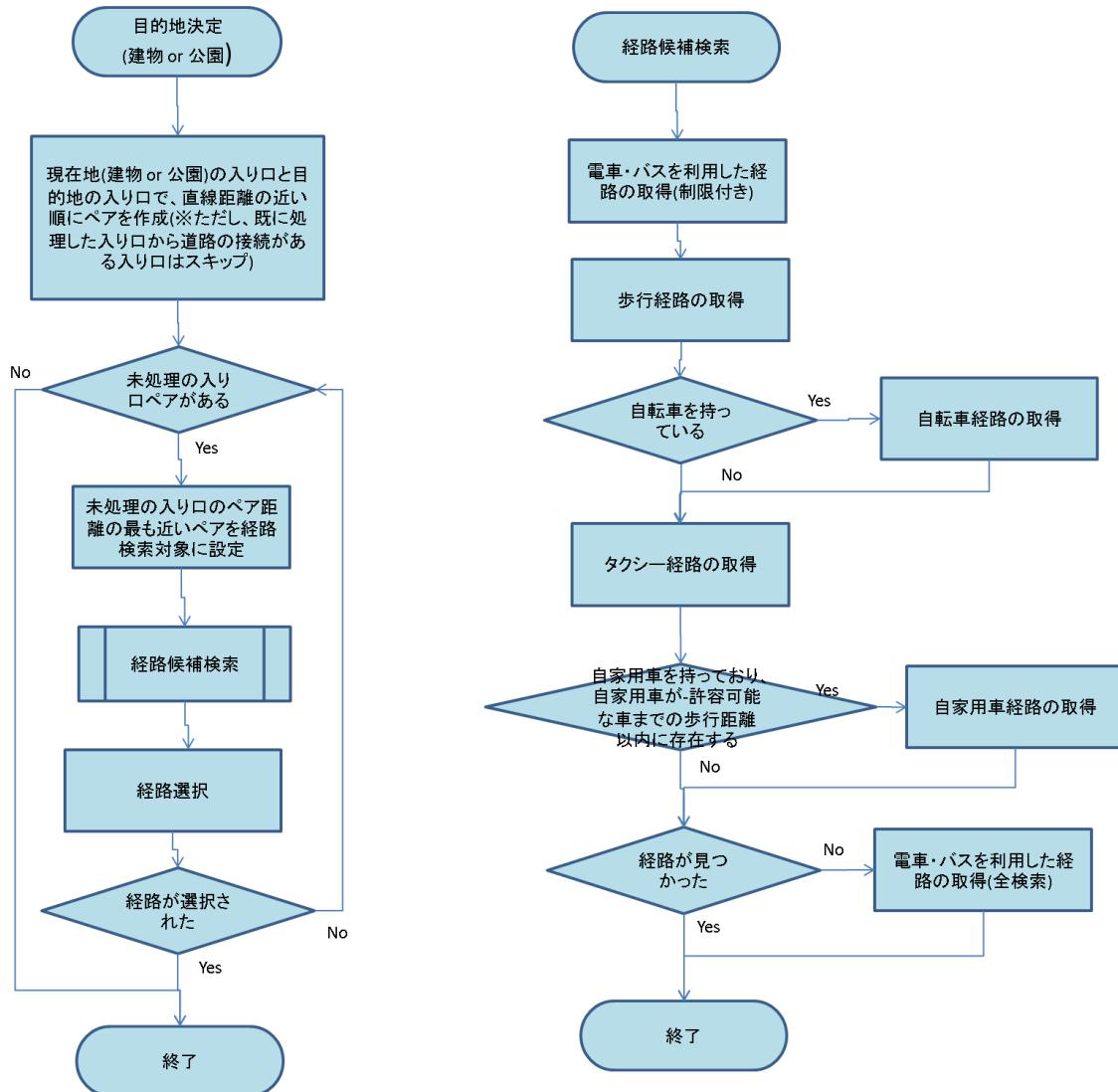
タクシーに乗車中に経路の再計算を行い、タクシー以外の経路を選択した場合はタクシーから降車し行動を開始する。タクシーの経路を選択した場合は、タクシードライバに経路の変更を伝える。

- 降車

降車予定の駅に到着した際に乗車した点まで移動して降車する。

9. 経路決定

経路決定のフローチャートを示す。



9.1. 目的地の決定

目的地の決定方法には二通りの方法がある。

- エージェント定義ファイルによる決定

エージェント行動定義ファイルで指定された目的地を利用する。通常のシミュレーションの場合はこちらの方法で目的地を決定する。目的地はエージェント行動定義ファイルの"MoveToDestination"で指定された場所である。

"MoveToDestination"値には単一の目的地、またはグループ化した目的地名を指定可能である。

単一の目的地を指定した場合は、その地点を目的地とし、"MoveToDestination"に対してグループ

化した目的地名を指定した場合は、グループ内のランダムな場所を目的地とする。“DestinationChoiceType”に“Nearest”を指定している場合、目的地の候補の中から、“DestinationBaseLocation”に指定された場所に最も近い目的地を選択する。“DestinationBaseLocation”的指定が無い場合には、現在位置に最も近い目的地を選択する。どの目的地にも経路が見つからない場合、行動をスキップする。

- エージェント定義ファイルによる決定

通信機インターフェースの目的地変更の API を利用することで、ソースコードレベルで任意の建物または公園に目的地変更可能である。

9.2. 電車・バスを利用した経路の取得(制限付き)

前提条件

点:駅、バス停

計算可能な最大の移動時間:172800 秒 (2 日)

9.2.1. 経路候補の作成

経路候補の作成手順を示す。

- 次の条件を満たす始点と終点の候補を、点の種類ごとに近いものから 3 つまで選択する。(始点、終点のそれぞれについて最大で駅 3 つ、バス停 3 つ)

始点:出発地からの距離が、始点から終点までの距離の半分かつ最大歩行距離(1km)以下の点

終点:目的地からの距離が、始点から終点までの距離の半分かつ最大歩行距離(1km)以下の点

- 1)で検索した始点から終点の全ペアに対して以下の条件を満たす経路候補(始点-終点のペア)を作成する。

始点:目的地よりも出発地が近い

終点:出発地よりも目的地が近い

始点から終点までの距離が、出発地から始点までの距離と終点から目的地までの距離を足したものより長い。

- 2)で作成した全てのペアに対して経路検索を行う。到着時刻の指定がある場合には終点側から経路の検索を行い、指定がない場合は出発地点から検索を開始する。

9.2.2. 終点からの経路検索

電車・バスの経路を終点側から検索する手順を示す。

- 経路検索

- 1) 終点から目的地まで徒歩で移動する時間を計算する。到着時刻には歩行の誤差時間(30秒)を加える。
- 2) 終点に接続する全ての路線について、EarliestArrivalTime から ArrivalTime の間に終点に到着する乗り物で最も ArrivalTime に近い乗り物を検索する。EarliestArrivalTime から ArrivalTime までに到着するものがなければ、代わりに ArrivalTime から LatestArrivalTime の間に到着する乗り物で最も ArrivalTime に近いものを検索する。EarliestArrivalTime、ArrivalTime、LatestArrivalTime は終点から目的地までの移動時間を考慮してオフセットしたものを利用する。EarliestArrivalTime を指定しない場合 EarliestArrivalTime は 0、LatestArrivalTime を指定しない場合、LatestArrivalTime はシミュレーションの終了時刻となる。
- 3) 2)で取得した全ての乗り物について、経路コストとして乗り換え回数、移動時間、運賃の条件でそれぞれ「終点からの乗り換え検索」を行う。

- 乗り換え検索

電車・バスの経路を終点側からの乗り換えの検索手順を示す。

- 4) 終点までの移動コストを 0 として、移動コストが小さい順に始点までの経路検索を行う。検索アルゴリズムは A*アルゴリズムを拡張した検索方法を用いる。移動コストは検索条件毎に異なり、それぞれ移動時間、運賃、乗り換え回数をコストとして用いる。(同一の運賃、同一の乗り換え時間となる場合には、移動時間が短い方を移動コストの小さい点とみなす)
- 5) 移動コストの最も小さい点(降車点)を取得する。(最初の検索時には終点を取得する。)
- 6) 5)で取得した点を降車点として、その点に接続する全ての路線について、最も待ち時間の少ない車両を検索する。この時、この点に移動するまでに既に一度選択している路線は検索対象から除外する(同じ路線は 2 度使用しない)

※降車点が終点の場合、「終点からの経路検索」で計算済みの乗り物が対象となる。。

7) 6)で取得した全ての乗り物について、これまで一度も訪れていない全ての乗車点を取得し、各乗車点から降車点まで移動コストを計算する。

- 移動コスト

移動時間の場合：乗車点での待ち時間と乗車点から降車点までの移動時間の和

運賃の場合：AgentTimeTable.txt で指定した運賃

乗り換え回数の場合：1

8) 7)で取得した全ての乗車点(乗り物)について以下の全ての条件を満たすものを、降車点から移動候補点とする。

- 始点までの予測移動コストが、これまでに分かっている始点への最小の移動コストよりも小さい。予測移動コストは予測用の最大車両速度(300m/s) を用いて始点への直線距離で計算する。
- 乗車点での移動コストが、これまでに分かっているその点までの最小の移動コストよりも小さい
- 乗車点までの移動時間が、終点から始点まで乗り換え回数を最も少なくしたときにかかる移動時間以下
- 乗り換え回数が最大の乗り換え回数(10 回)を超えていない

9) 5)で取得した点に対して、前の移動が乗り換えの移動でなければ、徒歩で移動可能な全ての点を取得する。徒歩で移動可能な点は以下の条件を満たす点である。

- 点同士の距離が最大乗り換え距離(200m)以下
- 同じ路線の点ではない。

10) 9)で取得した全ての乗り換え候補点について、これまで一度も訪れていない全ての乗り換え点について移動コストを計算する。

- 移動コスト

移動時間の場合：徒歩での移動時間(移動時間の計算には WalkSpeedAtTransfer を利用する。デフォルト設定での移動速度 Normal:3m/s, Slow:1m/s, VerySlow:0.5m/s, WheelChair:0.5m/s)

11) 10)で取得した全ての乗り換え点について以下の全ての条件を満たすものを移動候補点とする。

- 始点までの予測移動コストが、これまでに分かっている始点への最小の移動コストよりも小さい。予測移動コストは予測用の最大車両速度(300m/s)を用いて始点への直線距離で計算する。
- 移動コストが、これまでに分かっているその点までの最小の移動コストよりも小さい
- 移動時間が、終点から始点まで乗り換え回数を最も少なくしたときにかかる移動時間以下

12) 8)、11)で計算した移動候補点を、移動コストとともに登録して 5)に戻る。

13) 始点までのコストが最も小さかったものを経路候補とする。

9.2.3. 始点からの経路検索

電車・バスの経路を始点側から検索する手順を示す。

- 経路検索

- 1) 出発地から始点まで徒歩で移動する時間を計算する。到着時刻には歩行の誤差時間(30秒)を加える。
- 2) 始点から接続する全ての路線について、EarliestDepartureTime から DepartureTime の間に始点を出発する乗り物で最も DepartureTime に近い乗り物を検索する。EarliestDepartureTime から DepartureTime までに出発するものがなければ、代わりに DepartureTime から LatestDepartureTime の間に出来する乗り物で最も DepartureTime に近いものを検索する。EarliestDepartureTime、DepartureTime、LatestDepartureTime は出発地から始点までの移動時間を考慮してオフセットしたものを利用する。EarliestDepartureTime を指定しない場合 EarliestDepartureTime は 0、LatestDepartureTime を指定しない場合、LatestDepartureTime はシミュレーションの終了時刻となる。
- 3) 1)で取得した全ての乗り物について、経路コストとして乗り換え回数、移動時間、運賃の条件でそれぞれ「始点からの乗り換え検索」を行う。

- 乗り換え検索

- 4) 始点からの移動コストを 0 として、移動コストが小さい順に終点までの経路検索を行う。検索アルゴリズムは「終点からの乗り換え検索」を逆順に行ったものとなる。
- 5) 移動コストの最も小さい点(乗車点)を取得する。(最初の検索時には始点を取得する)
- 6) 3)で取得した点を乗車点として、その点に接続する全ての路線について、最も待ち時間の小さい車両を検索する。この時、この点に移動するまでに既に一度選択している路線は検索対象から除外する('同じ路線は 2 度使用しない')

※乗車点が始点の場合、「始点からの経路検索」で計算済みの乗り物が対象となる。

- 7) 6)で取得した全ての乗り物について、これまで一度も訪れていない全ての降車点を取得し、乗車点から各降車点まで移動コストを計算する。

- 移動コスト

移動時間の場合:乗車点での待ち時間と乗車点から降車点までの移動時間の和

運賃の場合:AgentTimeTable.txt で指定した運賃

乗り換え回数の場合:1

8) 7)で取得した全ての降車点(乗り物)について以下の全ての条件を満たすものを、乗車点から移動候補点とする。

- 終点までの予測移動コストが、これまでに分かっている終点への最小の移動コストよりも小さい。予測移動コストは予測用の最大車両速度(300m/s)を用いて終点への直線距離で計算する
- 降車点での移動コストが、これまでに分かっているその点までの最小の移動コストよりも小さい
- 降車点までの移動時間が、始点から終点までの乗り換え回数を最も少なくしたときにかかる移動時間以下
- 乗り換え回数が最大の乗り換え回数(10 回)を超えていない

9) 5)で取得した点に対して、前の移動が乗り換えの移動でなければ、徒歩で移動可能な全ての点を取得する。徒歩で移動可能な点は以下の条件を満たす点である。

- 点同士の距離が最大乗り換え距離(200m)以下
- 同じ路線の点ではない。

10) 9)で取得した全ての乗り換え候補点について、これまで一度も訪れていない全ての乗り換え点について移動コストを計算する。

- 移動コスト

移動時間の場合:徒歩での移動時間(移動時間の計算には WalkSpeedAtTransfer を利用する。デフォルト設定での移動速度 Normal:3m/s、Slow:1m/s、VerySlow:0.5m/s、WheelChair:0.5m/s)

11) 10)で取得した全ての乗り換え点について以下の全ての条件を満たすものを移動候補点とする。

- 終点までの予測移動コストが、これまでに分かっている終点への最小の移動コストよりも小さい。予測移動コストは予測用の最大車両速度(300m/s)を用いて終点への直線距離で計算する。
- 移動コストが、これまでに分かっているその点までの最小の移動コストよりも小さい

- 移動時間が、始点から終点まで乗り換え回数を最も少なくしたときにかかる移動時間以下

12) 8)、11)で計算した移動候補点を、移動コストとともに登録して 5)に戻る。

13) 始点までのコストが最も小さかったものを経路候補とする。

9.2.4. 料金計算

バスや電車の料金設定に関しては、AgentTimeTable.txt 内で路線(Line)毎に Price で設定された値を利用する。

料金設定は [料金/(乗車)距離] の形式で定義され、距離に対する関数となる。

※1000/10 と記述される場合 0~10m までが一律 1000 円、11m から 20m までが一律 2000 円となる。

同様に 300/100 のように記述した場合は 0~100m までが 300 円、101~200m までが 600 円となる。

0/1 は無料。距離は乗車と降車の駅、バス停を直線で結んだ距離となる。

9.2.5.経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	全ての乗り換えが Train なら PreferredMobilityMeans=Train のとき に _MoveByPreferredMobilityMeans=1 全ての乗り換えが Bus なら PreferredMobilityMeans=Bus のとき に _MoveByPreferredMobilityMeans=1 それ以外は 0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	
総乗り換え時間(秒)	_TotalTransferDuration	
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	乗り換え点を直線で結んだ距離の和
交通網の遅れの和	_TotalPublicTransportationDelay	
運賃の和	_Price	
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	常に 0
経路上のレーンあたりの車両数	_NumberOfVehiclesOnRoute/Vehicle CongestionOnRoute	常に 0

9.3. 歩行経路

道路を利用した経路検索では A*アルゴリズムに基づいて道路ネットワーク上で最も短い所要時間で到達可能な経路を検索する。歩行経路の検索は常に実行。

9.3.1. 歩行者経路の計算方法

歩行可能な道路を経路候補として、歩行者と自転車の混雑度を利用して各道路の所要時間の計算を行う。

予測最大速度: AgentProfiles.txt で指定された歩行速度

予測最小速度: 1m/s

道路幅の重み: $1 + (\text{シナリオ内での最大の道路幅に対する道路幅の割合} - 1) \times \text{道路幅の重み}$

予測速度: 予測最大速度 $\times (1 - \text{歩行者混雑度}) \times \text{道路幅の重み}$ 、と予測最小速度で大きい方の値を利用する

歩行者混雑度: 道路上の歩行者および自転車を利用している人の人数 / 道路長[m]

所要時間: 道路長 / 予測速度

9.3.2. 経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	PreferredMobilityMeans = Pedestrian(Walk) のとき に _MoveByPreferredMobilityMeans=1 それ以外は 0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	常に 0
総乗り換え時間(秒)	_TotalTransferDuration	常に 0
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	
交通網の遅れの和	_TotalPublicTransportationDelay	常に 0
運賃の和	_Price	常に 0
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	
経路上のレーンあたりの車両数	_NumberOfVehiclesOnRoute/VehicleCongestionOnRoute	

※_TotalPublicTransportationDelay はバス停に停車しているバスの出発時刻に対する遅れ(秒)となる。

9.4. 自転車経路

自転車を所有している場合に計算を行う。

9.4.1. 自転車経路の計算

歩行者と自転車の混雑度を利用して各道路の所要時間の計算を行う。

予測最大速度:AgentProfiles.txt で指定された自転車速度

予測最小速度:1m/s

道路幅の重み:1 + (シナリオ内での最大の道路幅に対する道路幅の割合 - 1) × 道路幅の重み

予測速度:予測最大速度 × (1 - 歩行者混雑度) × 道路幅の重みと予測最小速度で大きい方の値を利用する

歩行者混雑度:道路上の歩行者および自転車を利用している人の人数 / 道路長[m]

所要時間:道路長/予測速度

9.4.2. 経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	PreferredMobilityMeans =Bicycle のときには _MoveByPreferredMobilityMeans=1 それ以外は 0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	常に 0
総乗り換え時間(秒)	_TotalTransferDuration	常に 0
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	
交通網の遅れの和	_TotalPublicTransportationDelay	常に 0
運賃の和	_Price	常に 0
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	
経路上のレーンあたりの車両数	_NumberOfVehiclesOnRoute/VehicleCongestionOnRoute	

9.5. タクシー経路

経路検索は常にに行うが、経路の距離が車を利用する場合の最小の移動距離 (MinDistanceToUseVehicle)以下であれば経路を破棄する。

9.5.1. タクシー経路の計算

自動車で通行可能な道路を経路候補として、自動車の混雑度を利用して各道路の所要時間の計算を行う。

予測最大速度: AgentProfiles.txt で指定されたタクシーの速度

予測最小速度: 1m/s

予測速度: 予測最大速度 × (1 - 車両混雑度)と予測最小速度で大きい方の値を利用する

車両混雑度: 車長(5m) × 車両台数 / 道路長[m]

所要時間: 道路長/予測速度

9.5.2. 経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	PreferredMobilityMean s=Taxi のときには _MoveByPreferredMobilityMeans=1 それ以外は 0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	常に 0
総乗り換え時間(秒)	_TotalTransferDuration	常に 0
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	
交通網の遅れの和	_TotalPublicTransportationDelay	常に 0
運賃の和	_Price	AgentProfiles.txt の Taxi のプロファイルで指定した FuelConsumption × 乗車距離(m)
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	

経路上のレーンあたり の車両数	_NumberOfVehiclesOnRoute/VehicleCongesti onOnRoute	
--------------------	---	--

9.6. 自家用車経路

自家用車を保持しており、自家用車までの距離が自家用車までの最大の移動距離(WalkableDistanceToPrivateCar)以内なら計算を行う。自家用車を利用した経路の距離が車を利用する場合の最小の移動距離(MinDistanceToUseVehicle)以下であれば経路を破棄する。

自家用車の経路を利用する場合、エージェントは自家用車まで歩行で移動し、自家用車に乗車して車での移動を行なう。

9.6.1.自家用車経路の計算

自動車で通行可能な道路を経路候補として、自動車の混雑度を利用して各道路の所要時間の計算を行う。

予測最大速度:AgentProfiles.txt で指定された自動車の速度

予測最小速度:1m/s

予測速度:予測最大速度 × (1 – 車両混雑度)と予測最小速度で大きい方の値を利用する

車両混雑度:車長(5m) × 車両台数 / 道路長[m]

所要時間:道路長/予測速度

9.6.2.経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	PreferredMobilityMeans =Vehicle のときには _MoveByPreferredMobilityMeans=1 それ以外は0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	常に 0
総乗り換え時間(秒)	_TotalTransferDuration	常に 0
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	
交通網の遅れの和	_TotalPublicTransportationDelay	常に 0
運賃の和	_Price	AgentProfiles.txt の(エー

		ジェントの)プロファイルで指 定 し た FuelConsumption(通常 0)×移動距離(m)
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	
経路上のレーンあたりの車両数	_NumberOfVehiclesOnRoute/VehicleCongestionOnRoute	

9.6.3.経路コストの計算

取得した全ての経路に対して、経路内で発生するコストの総和を計算する。

コスト	RoutePriority 変数	補足
優先移動手段	_MoveByPreferredMobilityMeans	PreferredMobilityMeans=OnDemandBus のときには _MoveByPreferredMobilityMeans=1 それ以外は 0
到着時刻(秒)	_ArrivalTime	
乗り換え回数(回)	_TotalTransferCount	乗車点までの乗り換え回数+降車点からの乗り換え回数+1
総乗り換え時間(秒)	_TotalTransferDuration	乗車点までの乗り換え時間+降車点からの乗り換え時間
総移動時間(秒)	_TotalTravelTime	
総移動距離(m)	_TotalTravelDistance	乗車点までの距離+乗降車点からの距離
交通網の遅れの和	_TotalPublicTransportationDelay	
運賃の和	_Price	乗車点までの運賃+降車点からの運賃
経路上の歩行者数	_NumberOfPeopleOnRoute /PedestrianCongestionOnRoute	
経路上のレーンあたりの車両数	_NumberOfVehiclesOnRoute/VehicleCongestionOnRoute	

9.7. 電車・バスを利用した経路の取得(全検索)

「9.2 電車・バスを利用した経路の取得(制限付き)」について次の制限を無くして検索を行う。

- 駅・バス停の始点と終点の候補は全てとする。
- 乗り換え回数に制限を与えない。

9.8. 経路候補の選択

計算された数十から数百の経路について以下の各メトリックで値の小さいものから順に、最大経路候補数(MaxRouteCandidates: デフォルトでは 3 つ)取得して経路候補とする。

- 移動時間
- 運賃
- 乗り換え回数

※各メトリックについて 3 つずつ取得した場合、9 つの経路が候補として計算される。また同一経路の複数メトリック間での選択を許す。

9.9. 経路選択アルゴリズム

経路検索アルゴリズムで検索した経路候補に対して経路ユーティリティを計算し、ユーティリティの値を元に経路を選択する。

- 1) 経路候補が見つからない場合、経路選択を終了する。
- 2) 経路検索により取得した経路候補について経路ユーティリティを計算する。経路ユーティリティの値はエージェントプロファイルの RoutePriority で指定される計算式から算出した値となる。RouteCost は 0 以上の値でなければならない。値の大きい経路ほどエージェントに選択されやすい。
- 3) 2)で計算した経路ユーティリティの総和を計算し、各経路に対して経路の選択確率を算出する。

経路の選択確率 = 経路ユーティリティ / 経路ユーティリティの総和

※全ての経路の選択確率の和 = 1

- 4) 乱数を生成し、経路の選択確率を元に経路を選択する。全ての経路候補の RoutePriority が 0 である場合、各経路候補の重みを一様としてランダムに経路を選択する。

10. 経路再計算と目的地の変更

目的地の変更を行う場合、

10.1. 経路の再計算

シミュレーション中に経路再計算の API(Agent::RecalculateRoute())が呼び出され経路の再計算が発生した場合は、再度経路決定のアルゴリズムを適用する。

通常、目的地への移動中に経路の再計算を行った場合、目的地までの経路が存在しているが、道路の無効化等の理由により目的地へ到達できなくなってしまうことがある。この場合、单一の目的地("MoveToDestination")を指定していた場合は次の行動に移行する。グループを目的地としている場合は、グループ内からランダムに選択した他の場所、"DestinationChoiceType"に"Nearest"を指定している場合は、"DestinationBaseLocation"で指定される場所に対して、次に近い場所への移動を行う。"DestinationBaseLocation"で指定が無い場合には、現在位置に対して次に近い場所への移動を行う。

ただし、スキップする行動に"WaitUntil"が指定してある場合は WaitUntil まで待ってから行動の移行を行う。

10.2. 目的地の変更

目的地変更の API を利用することで、目的地が変化する。目的地の変更を行った場合、経路の再計算(経路変更)を行う。

目的地の変化は現在の行動に適用され、現在向かっていた目的地がある場合は、その目的地への移動は行わなくなる。"WaitUntil"、"Wait"、パラメータの変更といった目的地以外の行動条件は、目的地変更後の行動にも引き継がれる。

10.3. 経路変更のタイミング

経路再計算後の経路変更のタイミングは、その時のエージェントの行動状態によって異なる。

行動	経路変更タイミング
自由歩行	入り口到着時
道路歩行	次の交差点到着時
自転車	次の交差点到着時
自家用車ドライバ	現在目視ずみの最終の交差点
バスドライバ	現在目視ずみの最終の交差点
バスゲスト	<ul style="list-style-type: none"> - バス停でバスを待っている場合: そのバス停から経路変更 - バスに乗車している場合: バスが停車している場合は現在停車中のバス

	停、バスが動いている場合は次のバス停
電車ドライバ	経路変更不可
電車ゲスト	<ul style="list-style-type: none"> - 駅で電車を待っている場合 その駅から経路変更 - 電車が停車している場合 <p>現在停車中の電車、電車が動いている場合は次の駅</p>
タクシードライバ	現在目視ずみの最終の交差点
タクシーゲスト	現在タクシードライバが目視ずみの最終の交差点

※タクシードライバに対して経路再計算を行う場合、タクシードライバのプロファイル値を利用して経路計算を行うが、タクシーゲストに対して経路再計算を行う場合、タクシーゲスト側のプロファイルを利用して経路計算を行う。

10.4. 経路変更後の移動手段

経路変更後にとりうることが可能な行動は、経路変更時のエージェントの行動状態によって変化する。

変更可能な行動 \\ 変更前の行動	自由歩行	道路歩行	自転車	自家用車ドライバ	バスゲスト	電車ゲスト	タクシーゲスト	バスドライバ	電車ドライバ	タクシードライバ
自由歩行	○ ○ ○ ○ ○ ○ ○ ○ - - -									
道路歩行	- ○ ○ ○ ○ ○ ○ ○ - - -									
自転車	- ○ ○ ○ ○ ○ ○ ○ - - -									
自家用車ドライバ	- - - ○ - - - - - -									
バスゲスト	- ○ ○ ○ ○ ○ ○ ○ - - -									
電車ゲスト	- ○ ○ ○ ○ ○ ○ ○ - - -									
タクシーゲスト	- ○ ○ ○ ○ ○ ○ ○ - - -									
バスドライバ	- - - - - - - - ○ - -									
電車ドライバ	- - - - - - - - - - -									
タクシードライバ	- - - - - - - - - - ○									

11. シミュレーション実行時に初期化される変数の確認

シミュレーションを実行する際に自動生成される GIS オブジェクトと、実行時に算出されるエージェントの初期位置およびプロファイル初期値の確認方法を紹介する

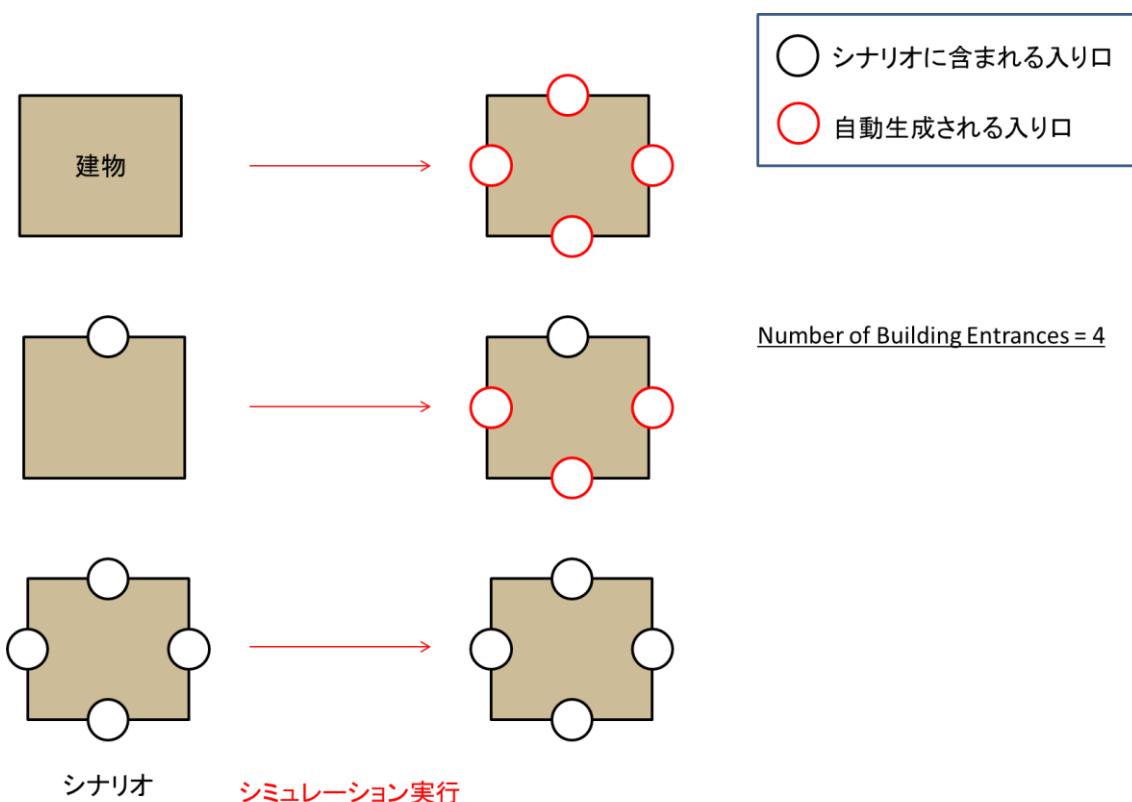
11.1. シミュレーション実行時に自動生成される GIS オブジェクト

シナリオ内に配置された GIS オブジェクトのトポロジーと、パラメータ設定に基づいてシミュレーション実行時に入り口または道路オブジェクトが追加で配置される。

- 入り口

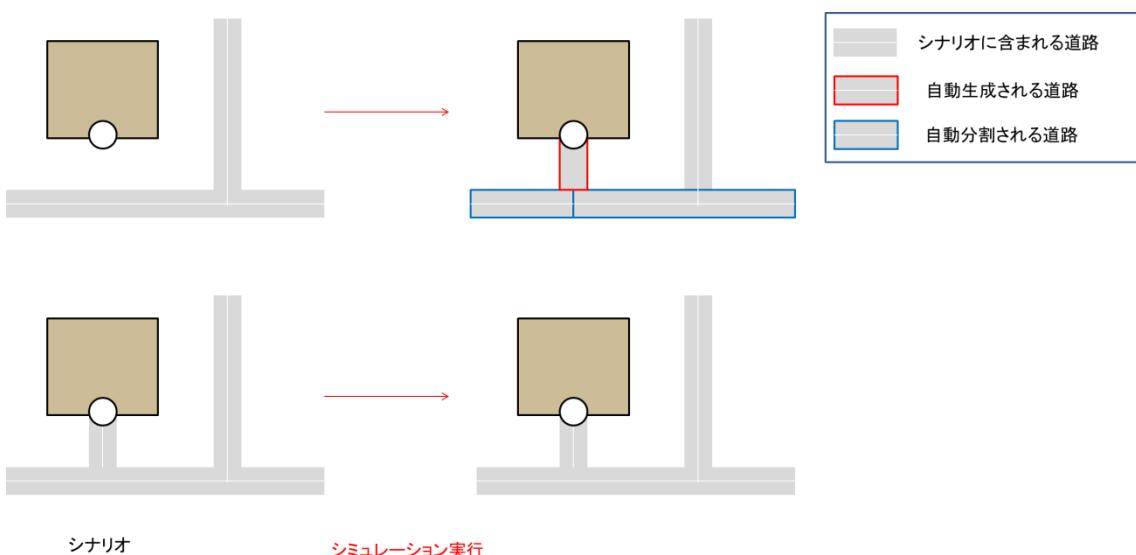
GIS グローバルのパラメータで、最小の入り口数を指定することで、対象の GIS オブジェクト全てに、最小数を満たす分の入り口が追加される。最小数分の以上の入り口が既にシナリオに含まれている場合には、その GIS オブジェクトには新たな入り口の追加は行われない。

パラメータ名	対象 GIS オブジェクト
Number of Building Entrances	建物
Number of Station Entrances	駅
Number of Bus Stop Entrances	バス停
Number of Park Entrances	公園



- 道路

建物や公園などの入り口を持つ GIS オブジェクトまたは POI に対して、周囲との道路接続が無い場合には、最寄りの道路に対して自動で道路を生成し、道路経由で GIS オブジェクトに移動可能となるよう補間処理を実施する。自動で生成される道路幅等の設定は、接続先の道路と同様の値が利用される。

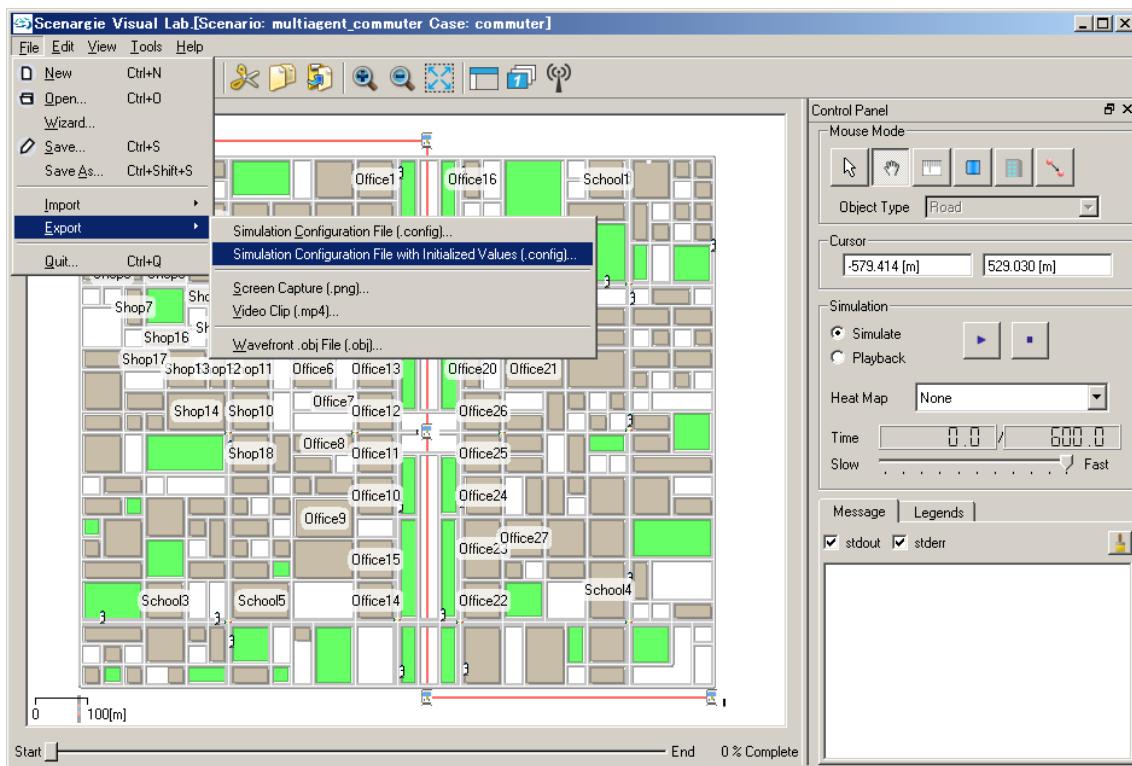


11.2. 自動生成される GIS オブジェクトの確認

シミュレーション実行時に自動生成される GIS オブジェクトは、シナリオ生成モードで”コンフィギュレーションファイルを出力することで確認が可能である。

- Visual Lab からのコンフィグレーションファイルの生成

メニューの[File] -> [Export] -> [Simulation Configuration File with Initialized Values (.config)]より出力する。



- コマンドラインからのコンフィグレーションファイルの生成
シミュレーション実行の引数として、"-scenario <出力ファイル名>"を追加することで、入力としたコンフィグレーションファイルをベースとして、シミュレーション実行時に生成された GIS オブジェクトを含むコンフィグレーションファイルが生成される。

実行例

```
$ sim simulation.config –scenario out.config
```

Visual Lab または、コマンドラインにより出力されたコンフィグレーションファイルを Visual Lab の[File] -> [Import] -> [Simulation Configuration File]でインポートすることにより、自動生成された GIS オブジェクトを視覚的に確認することが可能である。

自動生成された GIS オブジェクトのコンフィグレーションファイルへの出力は入り口オブジェクトのみ対応している。

GIS オブジェクト	自動生成された GIS オブジェクトのコンフィグレーションファイル出力
入り口	○
道路	×

11.3. エージェント初期位置の確認

「11.2 自動生成される GIS オブジェクトの確認」と同様の手順で出力される位置情報ファイル”.pos”を参照することで確認が可能である。

GIS オブジェクトの確認と同様に、出力されたコンフィグレーションファイルを Visual Lab でインポートすることにより初期位置を視覚的に確認することが可能である。また位置情報ファイルに含まれる位置は、エージェントの行動設定の初期位置(“InitialLocation”)の設定値を”PresentLocation”とすることにより、シミュレーション実行時に利用することが可能である。

11.4. プロファイル初期値の確認

マルチエージェントグローバル設定の Agent Profile Value Output File を設定してシミュレーションを実行することで、各エージェントの初期化時のプロファイル値がファイルに出力される。

出力例

```
...
ProfileType:1
accelerationexponent = 4
activeyieldtime = -3
age = 0
bicyclespeed = 10
boardingdelay = 0
disaster = 0
fuelconsumption = 0
lanechangeaccelerationthreshold = 0.200000000000000011102230246252
maxacceleration = 1
maxbrakingdecceleration = -20
maxdeceleration = -3
maxroutecandidates = 3
maxwaitingtimeatentrance = 3600
maxwaitingtimeatvehicleentrance = 1
maxwaitingtimefortaxiassignment = 600
mindistancetousevehicle = 0
minrouterecalcinterval = 600
minturnspeed = 2.7777777777777767909128669999
minvehiclegap = 3
```

```

numberofpeople = 1
othervehicleentrancetime = 1
passiveyieldtime = 3
pedestrianrecognitionprobability = 1
recalcintervalfordestinationdelay = 1.79769313486231570814527423732e+308
recalcintervalforlastviapointdelay = 1.79769313486231570814527423732e+308
recalcintervalfornextviapointdelay = 1.79769313486231570814527423732e+308
recalcintervalforvehicledelay = 1.79769313486231570814527423732e+308
recalcprobwhenmissingavehicle = 0
recalcthresholdforcongestion = 0.5
recalcthresholdforutility1 = 1.79769313486231570814527423732e+308
recalcthresholdforutility2 = 1.79769313486231570814527423732e+308
roadwidthweight = 0
routerecalcprobability = 0.800000000000000044408920985006
routerecalculationtime = 0
savedeceleration = -12
timeheadway = 1.5
totaltraveldistance = 0
totaltraveltime = 0
utility1 = 0
utility2 = 0
vehiclespeed = 16.666666666666678509045596002
velocityratiodgapdistance = 0
walkabledistancetobusstoporstation = 1000
walkabledistancetoprivatecar = 1000
walkspeed = 4.2812471576035022735595703125
yieldwaitingtime = 2
privatecarownership = 0
bicycleownership = 0
walkspeedattransfer = normal
routepriority = (1 + _movebypreferedmobilitymeans) * (_totaltraveldistance / max(1, _totaltraveltime)) / log10(max(10, (_price / max(1, _totaltraveldistance))*1000))

ProfileType:2
accelerationexponent = 4
activeyieldtime = -3

```

```

age = 0
bicyclespeed = 10
boardingdelay = 0
disaster = 0
fuelconsumption = 0
lanechangeaccelerationthreshold = 0.200000000000000011102230246252
maxacceleration = 1
maxbrakingdecceleration = -20
maxdeceleration = -3
maxroutecandidates = 3
maxwaitingtimeatentrance = 3600
maxwaitingtimeatvehicleentrance = 1
maxwaitingtimefortaxiassignment = 600
mindistancetousevehicle = 0
minrouterecalcinterval = 600
minturnspeed = 2.77777777777777767909128669999
minvehiclegap = 3
numberofpeople = 1
othervehicleentrancetime = 1
passiveyieldtime = 3
...

```

11.5. プロファイル初期値の再利用

出力したファイルは、マルチエージェントグローバル設定のエージェントプロファイル定義ファイルに指定し、エージェントの Profile Type を{SIMULATION_NODE_ID}とすることで、シミュレーション時の初期値として再利用が可能である。

12. API 一覧

マルチエージェントシミュレーションで利用する主な API を紹介する。

12.1. マルチエージェントシミュレータ

クラス名 :MultiAgentSimulator

機能 :マルチエージェントシミュレータ

関数名	機能
RunSimulationUntil	指定時刻までシミュレーションを実行
CurrentTime	現在時刻
NextTime	次タイムステップの時刻
TimeStep	タイムステップ
GetTaskStartTime	タスクの開始時刻
GetRunSeed	シミュレーションシード
GetParameterDatabaseReader	パラメータのデータベース
GetPublicVehicleTable	時刻表の取得
AttachCommunicationNode	通信機の追加
DetachCommunicationNode	通信機の取り外し
OutputTraceEvent	トレースイベント出力
OutputTrace	デバッグ情報出力
IncrementTimeStep	シミュレーションのステップ実行

12.2. エージェントモデル

ファイル名 :scensim_multiagent.h

クラス名 :Agent

機能 :エージェント

関数名	機能
CalculateWakeupTime	行動開始時刻の計算
IncrementTime	同期時間分時間を進める
IsDeletedAfterEndOfTimeStep	エージェントの削除判定
GetAgentId	エージェント ID
OutputTrace	デバッグ情報の出力
AddCommunicationNode	通信機の追加
DeleteCommunicationNode	通信機の取り外し
RecalculateRoute	経路の再計算
OutputTraceEvent	トレースイベントの出力

クラス名 :AgentResource

機能 :エージェントのリソースへのアクセス

関数名	機能
IsAvailable	有効なエージェントかの判定
ThreadNumber	スレッド番号
CurrentTime	時刻
CurrentTravelTime	移動時間
CurrentDelay	遅れ
AgentId	エージェント ID
ProfileType	プロファイルタイプ
GetProfileName	ユーザタイプ
WalkSpeedAtTransfer	移動クラス
TicketType	チケットタイプ
DestPositionId	目的地の場所 ID
HomePositionId	開始点の場所 ID
CurrentRoute	現在の経路
NextRouteIsStop	次の経路が停留所かの判定
GetNextRouteStopId	次の経路の停留所を取得
UnreachableDestinationIds	到達不可な場所 ID

LastVertexId	最後に訪れた点
PositionId	現在の場所 ID
DestVertexId	目的地の点
Status	状態
Position	位置(x,y,z)
Age	年齢
WalkSpeedMetersPerSec	歩行速度
BicycleSpeedMetersPerSec	自転車速度
MaxVehicleSpeedMetersPerSec	自動車速度
RecalcIntervalForLastViaPointDelay	Last Delay の経路検索トリガ
RecalcIntervalForNextViaPointDelay	Next Delay の経路検索トリガ
RecalcIntervalForDestinationDelay	Trip Delay の経路検索トリガ
RecalcIntervalForVehicleDelay	Vehicle Delay の経路検索トリガ
RecalcThresholdForCongestion	混雑度の経路検索トリガ
RecalcThresholdForUtility1	ユーティリティの経路検索トリガ
RecalcThresholdForUtility2	ユーティリティ 2 の経路検索トリガ
MinRouteRecalcInterval	最小の経路検索間隔
RouteRecalcProbability	経路検索確率
RecalcProbWhenMissingAVehicle	乗り遅れの場合の経路検索確率
FuelConsumptionPerMeters	ガソリンコスト[/m]
Utility1	ユーティリティ 1
Utility2	ユーティリティ 2
BoardingDelay	乗り込み時間
PedestrianRecognitionProbability	歩行者認識確率
TimeHeadway	ヘッドウェイ
MinVehicleGap	最小車間距離
MaxAcceleration	最大加速度
MaxDeceleration	最大減速度
LaneChangeAccelerationThreshold	車線変更加速度閾値
PassiveYieldTime	車両の消極的な譲歩時間
ActiveYieldTime	車両の積極的な譲歩時間
TotalTravelDistance	総移動距離
TotalTravelTime	総移動時間
RoadWidthWeight	経路計算時の道路幅の重み
IsDisasterMode	災害モード

RouteRecalcInterval	経路再計算間隔
DelayForLastViaPoint	最後の via ポイントでの遅れ
DelayForNextViaPoint _TripDelay	次の via ポイントに対する遅れ
DelayForExpectedArrivalTime	到着予定時刻に対する遅れ
Congestion	混雑度
PrivateCarOwnership	車の所持判定
BicycleOwnership	自転車量の所持判定
CanUseCar	車の使用可否
CanUseBicycle	自転車の使用可否
PathQueryTriggerIsAvailable	経路検索トリガを利用可能かの判定
SetLastPathQueryTriggerTime	経路検索トリガ時刻の保存
ExceededRouteRecalcInterval	経路再計算間隔を超えた
SetPosition	位置の設定
SetVertexId	頂点の設定
SetPositionId	位置 ID の設定
SetDirectionRadians	向きの設定
SetCongestion	混雑度の設定
RecalculateRoute	経路の再計算
ArrivedAtDeadEndNotification	行き止まりの通知
UnreachableDestinationNotification	到達不可の通知
ArrivedAtDestinationNotification	目的地到着の通知
EnteredToDestinationNotification	目的地進入の通知
SetDestination	目的地の設定
AddUnreachablePositions	到達不可の場所の追加
SetOwnerAgent	親エージェントの設定
RemoveOwnerAgent	親エージェントの設定解除
WaitEntrance	入り口待機
AllowedEntrance	進入許可

クラス名 : AgentProfile

機能 : プロファイル

関数名	機能
GetProfileName	プロファイル名
GetProfileType	プロファイルタイプ

GetTypicalWalkSpeed	公共交通機関の乗り換えの移動速度
GetTicketType	チケットタイプ
GetPrivateCarOwnershipRatio	車の所持率
GetBicycleOwnershipRatio	自転車の所持率
GetParameters	全パラメータの取得
CalculateRouteUtility	経路ユーティリティの計算
CalculateUtility1	ユーティリティ 1 の計算
CalculateUtility2	ユーティリティ 2 の計算

12.3. 行動モデル

ファイル名 :scensim_multiagent_behavior.h

クラス名 :AgentBehavior

機能:共通の行動モデル

関数名	機能
HasFinished	行動終了の判定
IncrementTimeStep	行動の実行
GetBehaviorType	行動種別
GetBehaviorName	行動名
MakePositionTraceString	トレース情報の作成
GetRoute	経路
EndBehaviorAtViaPoint	次の via-point で行動を終了
GetViaPointVertexId	次の via-point
IsAcceptableRouteChange	許容可能な経路の変更かの判定
ChangeRoute	経路の変更

クラス名 :FreeWalkBehavior (public AgentBehavior)

機能:自由歩行

関数名	機能
-	-

クラス名 :PedestrianBehavior (public AgentBehavior)

機能:道路歩行

関数名	機能
-	-

UpdateNextWalkPath	次の移動経路を更新
ResetDestinationPosition	目的地の再設定
ForceStopWalk	歩行の停止

クラス名 : BicycleBehavior(public AgentBehavior)

機能 : 自転車

関数名	機能
UpdateNextWalkPath	次の移動経路を更新
ResetDestinationPosition	目的地の再設定
ForceStopWalk	歩行の停止

クラス名 : VehicleDriverBehavior(public AgentBehavior)

機能 : 自家用車ドライバ

関数名	機能
RideOnVehicle	乗車
UpdateVelocity	速度更新
GoToVehicle	車両へ移動
UpdateVisibleRouteIfNecessary	移動予定経路の更新
SetNextWaypoint	次の移動経路を更新
ViewFrontStatus	前方を確認
ChangeLaneIfNecessary	可能であれば車線変更
CanChangeLane	車線変更の可否
ChangeLane	車線変更
ParkingVehicleIfPossible	可能であれば駐車
RestartRun	再発進
ForceStopVehicle	強制的に停車

クラス名 : BusDriverBehavior(public VehicleDriverBehavior)

機能 : バスドライバ

関数名	機能
IsStopPoint	バス停かの判定

クラス名 : TrainDriverBehavior(public AgentBehavior)

機能 : バスドライバ

関数名	機能

-	-
---	---

クラス名 : TaxiDriverBehavior(public VehicleDriverBehavior)

機能 : タクシードライバ

関数名	機能
IsStopPoint	停車するポイントかの判定
RestartRun	再発進
DecideRoute	経路の決定

クラス名 : TaxiGuestBehavior(public AgentBehavior)

機能 : タクシーゲスト

関数名	機能
TakeoffTaxi	降車

クラス名 : PublicVehicleBehavior (public AgentBehavior)

機能 : バスゲスト・電車ゲスト

関数名	機能
IsOnPublicVehicle	乗車しているかの判定
RideOnVehicleIfPossible	可能であれば乗車
GetDownVehicle	降車
IsGetDownStop	降車駅またはバス停かの判定
DecidePositionInVehicle	車内での待機位置の決定
SetPedestrianBehaviorTowardNextStop	次の乗り換え駅またはバス停までの歩行設定
MissedPublicVehicle	予定していた車両に乗れなかつたかの判定
ForceStopRiding	強制的に乗車を中止

12.4. マルチエージェント GIS 情報システム

ファイル名 : scensim_multiagent_gis.h

クラス名 : MultiAgentGis

機能 : マルチエージェント GIS 情報

関数名	機能
GetAgentRoad	道路の取得
GetAgentIntersection	交差点の取得
GetAgentBuilding	建物の取得
GetAgentPark	公園の取得
GetAgentArea	エリアの取得
GetAgentStation	駅の取得
GetAgentBusStop	バス停の取得
CanReach	経路があるかの判定
SearchRoadRoute	道路の経路取得
CallTaxi	タクシーの呼び出し
RideOnPublicVehicle	公共交通車両への乗車
GetDownPublicVehicle	公共交通車両からの降車
PeopleGiveUpEntrance	進入をあきらめる

ファイル名 : scensim_gis.h

クラス名 : GisSubsystem

機能 : GIS 情報

関数名	機能
OutputVertexInformation	頂点情報の出力

12.5. 経路検索システム

ファイル名 :scensim_multiagent_routearch.h

クラス名 :AgentRouteSearchSubsystem

機能 :経路検索機能

関数名	機能
SearchRouteCandidates	経路候補の取得
SearchPublicVehicleRoutes	電車・バスを利用した経路の取得(制限付き)
SearchAllPublicVehicleRoutes	電車・バスを利用した経路の取得(全検索)

12.6. 公共交通機関情報システム

ファイル名 :scensim_multiagent_publicvehicle.h

クラス名 :Vehicle

機能 :車両

関数名	機能
GetPosition	位置
GetDirectionRadians	方向
GetVelocityMetersPerSec	速度
GetDistanceToIntersection	交差点までの距離
GetRoadId	道路 ID
GetTurnType	交差点で曲がる方向
GetLaneNumber	レーン番号
GetDriverAgentId	ドライバのエージェント ID
GetVehicleConstant	車両の設定

クラス名 :Taxi

機能 :タクシー

関数名	機能
PopReservedCustomer	先頭の予約を消去
PopCustomer	エージェントの降車
AddReservedCustomer	予約の追加
ContainsCustomer	エージェントの有無
HasReservation	予約の有無
IsTookOnCustomer	乗車済みのエージェントの判定
IsReservedCustomer	予約済みのエージェントの判定
TakeOn	エージェントの乗車
Contains	指定エージェントの乗車判定
AtHomeVertex	ホームの場所にいるかの判定
GetHomeVertexId	ホームの頂点
GetHomeStationId	ホームの駅

クラス名 :PublicVehicle

機能 :公共交通車両

関数名	機能

GetPublicVehicleId	公共交通機関の車両 ID
GetLineId	路線 ID
GetRouteId	経路 ID
GetVehicleNumber	経路内の車両番号
GetVehicleFamily	車両のファミリ
GetPosition	位置
GetDirectionRadians	方向
GetVehicleType	車両種別
GetDepartureTime	出発時刻
AtStop	駅・バス停にいるかの判定
GetStopId	駅・バス停の ID
GetViaPointVertexId	次の via-point の頂点
GetVehicleName	車両名
Contains	乗車済みのエージェントの判定
CanTakeOn	乗車可否の判定
IsFull	満員判定
Pop	エージェントの降車
HasAnAgent	乗車しているエージェントが一人でもいるかの判定
ResetRejectedAgentids	乗車出来なかったエージェントをリセット
AddRejectedAgent	乗車出来なかったエージェントを登録
IsAlreadyRejectedAgent	乗車できなかったエージェントの判定
GetCurrentDelay	遅延

クラス名 : Train (public PublicVehicle)

機能 : 電車

関数名	機能
SetTimeTo	指定の時刻まで移動
GetStationId	駅 ID

クラス名 : Bus (public PublicVehicle)

機能 : バス

関数名	機能
ArrivedAtBusStop	指定の時刻まで移動
ArrivedAtTheLastBusStop	駅 ID
GetStartBusStopId	始発バス停

GetDestBusStopId	終点バス停
GetNextBusStopId	次のバス停
GetRoute	経路
ContainsDestinationStopId	該当の行先を含むかの判定
GetStopNumber	停車したバス停の番号

クラス名 : PublicVehicleTable

機能 : 公共交通機関の経路・時刻表

関数名	機能
CreatePublicVehicles	公共交通車両の作成
SearchPublicVehicleRoute	公共交通車両の経路取得
FindStartAndStopIds	始点と終点の取得
GetStopName	駅またはバス停名
GetOrigVertexId	出発点
GetDestVertexId	到着点
GetLineVertexId	路線の乗車点
GetNearestEntranceVertexId	近傍の入り口
GetEntranceVertexIds	全ての入り口
GetStationId	駅 ID の取得
GetBusStopId	バス停 ID の取得
GetPositionId	場所 ID
CalculateCongestion	混雑度の計算
GetStopDelay	停留所での遅れ
GetVehicleDelay	車両の遅れ
GetScheduledArrivalTime	到着予定時刻
SetStopDelay	停留所での遅れの更新
SetVehicleDelay	車両の遅れの更新
GetLineId	路線 ID の取得
GetStopId	停留所 ID の取得
GetShortestRouteId	最短経路の取得
LineIsAvailable	運行ラインの有効/無効判定
CalculateGetDownPrice	降車運賃の計算

12.7. 通信機用インターフェース

ファイル名 : scensim_networkinginterface.h

クラス名 : AgentCommunicationNode

機能 : 通信機インターフェース

関数名	機能
IsAttached	通信機がエージェントと関連づいているかの判定
IsAvailable	エージェントが有効であるかの判定
GetPosition	エージェントの位置
GetProfileName	エージェントのプロファイル名
SetDestination	現在の行動の目的地を変更
AddUnreachablePositions	到達不可の場所を追加
RecalculateRoute	経路の再計算
ArrivedAtDeadEndNotification	行き止まり地点に到達の通知
ArrivedAtVertexNotification	頂点に到達の通知
ArrivedAtDestinationNotification	目的地に到達の通知
EnteredToDestinationNotification	目的地に進入の通知
ArrivedAtGisPositionNotification	場所の通知
UnreachableDestinationNotification	目的地に到達不可の通知

13. 通信とユーザ行動の連携

通信と行動の連携を行ってシミュレーションを行う場合、multisystemsにおいて MultiAgent Extension Module と任意の通信モジュールを有効にしたシミュレータを利用する。

シミュレータのビルド例

- MultiAgent Extension Module と Dot11 Module を利用

```
$ cd ~/scenargie_simulator/1.7/source/multisystems  
$ make -f makefile.linux MULTIAGENT_MODULE=on DOT11_MODULE=on
```

- MultiAgent Extension Module と Dot11 Module と ITS Extension Module を利用

```
$ cd ~/scenargie_simulator/1.7/source/multisystems  
$ make -f makefile.linux MULTIAGENT_MODULE=on DOT11_MODULE=on ITS_MODULE=on  
BASE_IPV6=on
```

- MultiAgent Extension Module と LTE Module を利用

```
$ cd ~/scenargie_simulator/1.7/source/multisystems  
$ make -f makefile.linux MULTIAGENT_MODULE=on LTE_MODULE=on
```

13.1. 通信結果に応じたユーザ行動の指定

通信結果に応じてエージェントの行動を指定する場合、利用用途に応じて、ソースコード中でエージェントの行動に関する関数を利用する。

利用用途

- 通信機からエージェントへの行動指定
- エージェントの行動結果を通信機で取得

クラス名 : AgentCommunicationNode

利用用途	関数名	機能
通信機からエージェントへの行動指定	SetDestination	現在の行動の目的地を変更
	AddUnreachablePositions	到達不可の場所を追加
	RecalculateRoute	経路の再計算
エージェントの行動結果を通信機で取得	ArrivedAtDeadEndNotification	行き止まり地点に到達の通知
	ArrivedAtVertexNotification	頂点に到達の通知
	ArrivedAtDestinationNotification	目的地に到達の通知
	EnteredToDestinationNotification	目的地に進入の通知
	ArrivedAtGisPositionNotification	場所の通知
	UnreachableDestinationNotification	目的地に到達不可の通知

通信結果に応じてエージェントの行動を指定する実装の例を示す。

実装例

アプリケーション: 1 秒間隔でブロードキャストの UDP パケットを送信
 ユーザ行動(通信の結果で移動行動を変更): パケットを 50 回受信する度に目的地を park1 に変更
 ユーザ行動(移動行動の結果に応じて通信): 目的到着時にパケットを送信
 対象エージェント: 通信機をもつ全エージェント

- アプリケーションクラス定義 (multisystems/sim.h)

```

...
map<Node IdType, shared_ptr<SimNode>> simNodePtrs;

};//MultiSystemsSimulator//
//--- Example Start-----

```

```

class TestApp : public ScenSim::Application {
public:
    TestApp(
        SimNode* initSimNodePtr,
        const shared_ptr<GisSubsystem>& initGisSubsystemPtr,
        const shared_ptr<SimulationEngineInterface>&
initSimEngineInterfacePtr,
        const NodeIdType& initNodeId)
        :
        Application(initSimEngineInterfacePtr, "TestApp"),
        nodePtr(initSimNodePtr),
        gisSubsystemPtr(initGisSubsystemPtr),
        nodeId(initNodeId),
        destinationPortId(1),
        numberReceivedPackets(0)
    {}

    void CompleteInitialization();

    void SendPacket();
    void ReceivePacket(unique_ptr<Packet>& packetPtr);

    void ArrivedAtDestinationNotification();

private:
    struct Payload {
        int value;

        Payload(const int initialValue) : value(initialValue) {}
    }; //Payload//

    SimNode* nodePtr;
    shared_ptr<GisSubsystem> gisSubsystemPtr;
    NodeIdType nodeId;
    unsigned short destinationPortId;
}

```

```

int numberReceivedPackets;

class BroadcastEvent: public SimulationEvent {
public:
    BroadcastEvent(TestApp* initTestAppPtr) : testAppPtr(initTestAppPtr) {}

    virtual void ExecuteEvent() { testAppPtr->SendPacket(); }
private:
    TestApp* testAppPtr;
}; //BroadcastEvent//


class UdpPacketHandler: public ScenSim::UdpProtocol::PacketForAppFromTransportLayerHandler {
public:
    UdpPacketHandler(TestApp* initTestAppPtr) : testAppPtr(initTestAppPtr)
{ }

    void ReceivePacket(
        unique_ptr<Packet>& packetPtr,
        const NetworkAddress& sourceAddress,
        const unsigned short int sourcePort,
        const NetworkAddress& destinationAddress,
        const PacketPriorityType& priority)
{ testAppPtr->ReceivePacket(packetPtr); }
private:
    TestApp* testAppPtr;
}; //UdpPacketHandler//


}; //TestApp//
//--- Example End-----


class SimNode : public AgentCommunicationNode {
public:
    SimNode(

```

- アプリケーションクラスの実体定義 (multisystems/multisystems_mac_and_phy.cpp)

TestApp::ComleteInitialization(): アプリケーションの初期化

TestApp::SendPacket(): 定期的なパケットの送信

TestApp::ReceivePacket(): パケットの受信

TestApp::ArrivedAtDestinationNotification(): 目的地に到達した場合の処理

```

...
using GeoNet::GeoNetworkingProtocol;
using GeoNet::BasicTransportProtocol;
using GeoNet::GeoNetMac;

using Its::ItsBroadcastApplication;

using ScenSim::InterfaceOutputQueue;

//--- Example Start-----
void TestApp::CompleteInitialization()
{
    assert(transportLayerPtr->udpPtr->PortIsAvailable(destinationPortId));

    transportLayerPtr->udpPtr->OpenSpecificUdpPort(
        NetworkAddress::anyAddress, destinationPortId,
        shared_ptr<UdpPacketHandler>(new UdpPacketHandler(this)));

    const double jitter = aRandomNumberGeneratorPtr->GenerateRandomDouble();

    simulationEngineInterfacePtr->ScheduleEvent(
        unique_ptr<SimulationEvent>(new BroadcastEvent(this)),
        simulationEngineInterfacePtr->CurrentTime() +
        ScenSim::ConvertDoubleSecsToTime(jitter));
}

//CompleteInitialization//


void TestApp::SendPacket()
{
    const Payload payload(nodeId);

    unique_ptr<Packet> packetPtr =

```

```

    Packet::CreatePacket(*simulationEngineInterfacePtr, payload);

    transportLayerPtr->udpPtr->SendPacket(
        packetPtr, 0, NetworkAddress::broadcastAddress, destinationPortId,
        0);

    cout << "node " << nodeId << " sent packet value " << payload.value << endl;

    simulationEngineInterfacePtr->ScheduleEvent(
        unique_ptr<SimulationEvent>(new BroadcastEvent(this)),
        simulationEngineInterfacePtr->CurrentTime() + 1*SECOND);
}

}//SendPacket//


void TestApp::ReceivePacket(unique_ptr<Packet>& packetPtr)
{
    const Payload payload = 
packetPtr->GetAndReinterpretPayloadData<Payload>();

    cout << "node " << nodeId << " received packet value = " << payload.value
    << endl;
    numberReceivedPackets++;

    if (numberReceivedPackets == 50) {
        numberReceivedPackets = 0;

        nodePtr->SetDestination(gisSubsystemPtr->GetPosition("park1"),
true);
    }

    DeleteDummyUniquePtrIfItIsDummy(packetPtr);
    packetPtr = nullptr;
}

}//ReceivePacket//


void TestApp::ArrivedAtDestinationNotification()
{
    const Payload payload(100000);

```

```

unique_ptr<Packet> packetPtr =
    Packet::CreatePacket(*simulationEngineInterfacePtr, payload);

transportLayerPtr->udpPtr->SendPacket(
    packetPtr, 0, NetworkAddress::broadcastAddress, destinationPortId,
0);

cout << "node " << nodeId << " arrived at destination" << endl;
cout << "node " << nodeId << " sent packet value " << payload.value << endl;
}//ArrivedAtDestinationNotification//
//--- Example End-----

class NoNetworkQueue : public InterfaceOutputQueue {
public:
    NoNetworkQueue(const shared_ptr<SimulationEngineInterface>&
initSimEngineInterfacePtr)
        : InterfaceOutputQueue(initSimEngineInterfacePtr) {}

...

```

- ユーザ行動に対する処理の定義 (multisystems/sim.h
SimNode で目的地到着時に呼び出される API をオーバーライド

```

...
void SetupInterfaces(
    const ParameterDatabaseReader& theParameterDatabaseReader,
    const shared_ptr<SimulationEngineInterface>&
simulationEngineInterfacePtr,
    const GlobalNetworkingObjectBag& theGlobalNetworkingObjectBag,
    const shared_ptr<GisSubsystem>& gisSubsystemPtr,
    const shared_ptr<ChannelModelSet>& channelModelSetPtr);

//--- Example Start-----
virtual void ArrivedAtDestinationNotification() {
    testAppPtr->ArrivedAtDestinationNotification();
}
```

```

    }

//--- Example End-----


private:
    virtual void Attach(const shared_ptr<ObjectMobilityModel>&
initNodeMobilityModelPtr);

```

- アプリケーションを端末に定義 (multisystems/sim.h)

SimNode のメンバ変数として TestApp を定義

```

...
struct InterfaceType {
    shared_ptr<MacLayer> macPtr;
    vector<shared_ptr<AntennaType> > antennaPtrs;
};

shared_ptr<AttachedAntennaMobilityModel> platformMobilityModelPtr;

vector<InterfaceType> interfaces;

//--- Example Start-----
shared_ptr<TestApp> testAppPtr;
//--- Example End-----


// For Gateway only.
shared_ptr<Lte::LteGatewayController> gatewayControllerPtr;

void SetupStationType(const ParameterDatabaseReader&
theParameterDatabaseReader);

```

- アプリケーションを端末に追加(multisystems/multisystems_mac_and_phy.cpp)

SimNode::SetupInterfaces()の最後でアプリケーションを作成。

```

...
} //for//

//--- Example Start-----
testAppPtr.reset(

```

```

new TestApp(
    this,
    gisSubsystemPtr,
    simulationEngineInterfacePtr,
    nodeId);

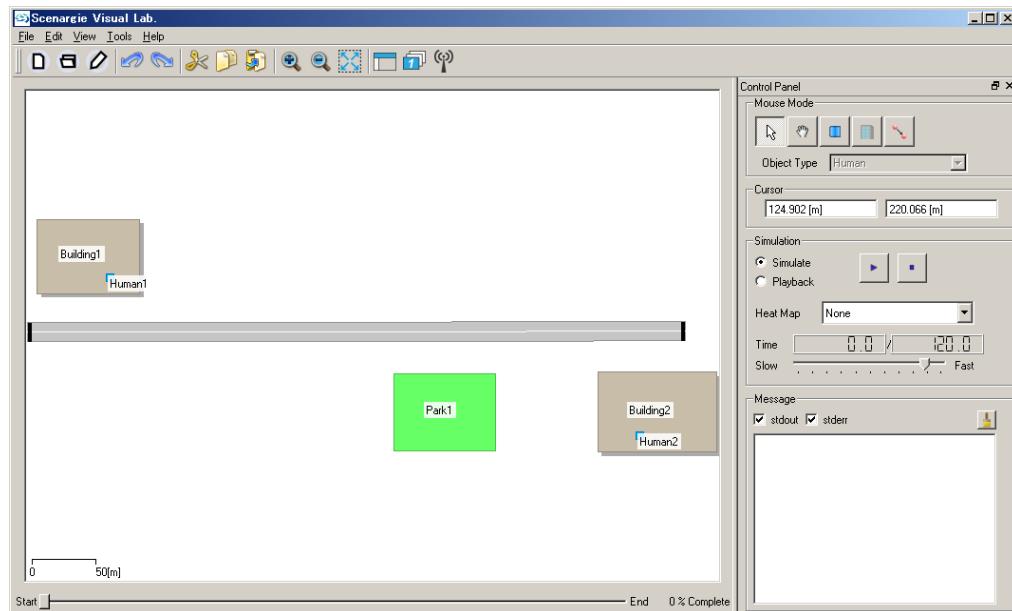
(*this).GetAppLayerPtr()->AddApp(testAppPtr);
testAppPtr->CompleteInitialization();
//--- Example End-----

(*this).CompleteAntennaNumberAssignment();

}//SetupInterfaces/

```

修正したモデルは例えば以下のように、建物を二つ、公園を一つ、通信機を持った人(Human-Dot11a等)を二人配置したシナリオで動作の確認が可能である。



13.2. 通信機の着脱

標準のソースコードでは通信機とエージェントは常に一対一の関係として扱われているが、通信機とエージェントが一対一の関係ではない場合やエージェントと通信機の移動が独立して行われる場合には、通信機を作成する際に次の API を利用する。

クラス名 :MultiAgentSimulator

関数名	機能
AttachCommunicationNode	エージェントに通信機をとりつける
DetachCommunicationNode	エージェントから通信機を外す

この API は例えば以下のような場合に利用する。

- エージェントが二つ以上の通信機を保持する
- シミュレーション中にエージェントへの通信機の着脱が行われる。
- 通信機がエージェント間で移動する。(あるエージェントの通信機を他のエージェントに渡す)

標準のソースコードでは通信機の作成がエージェントの生成と同期するようになっており、これはソースコード内で MultiAgentSimulator::AddCommunicationNode()を呼び出す位置に該当する。この API で追加された通信機は、通信機 ID と同一のエージェント ID をもつエージェントと同期が行われる。

multisystems/multisystems_netsim.cpp

```
void MultiSystemsSimulator::CreateNewNode(
    const ParameterDatabaseReader& theParameterDatabaseReader,
    const NodeIdType& nodeId,
    const shared_ptr<ObjectMobilityModel>& nodeMobilityModelPtr,
    const string& nodeTypeName)
{
    unsigned int partitionIndex = 0;

    if
(theParameterDatabaseReader.ParameterExists("parallelization-partition-index"
, nodeId)) {
        partitionIndex =
(theParameterDatabaseReader.ReadInt("parallelization-partition-index",
,
```

```

nodeId) %

    theSimulationEnginePtr->GetNumberPartitionThreads());
} //if//


const shared_ptr<SimulationEngineInterface> simulationEngineInterfacePtr =
theSimulationEnginePtr->GetSimulationEngineInterface(
    theParameterDatabaseReader, nodeId, partitionIndex);

const shared_ptr<AttachedAntennaMobilityModel>
attachedAntennaMobilityModelPtr(
    new AttachedAntennaMobilityModel(nodeMobilityModelPtr));

shared_ptr<SimNode> aSimNodePtr(
    new SimNode(
        theParameterDatabaseReader,
        theGlobalNetworkingObjectBag,
        simulationEngineInterfacePtr,
        nodeId,
        runSeed,
        theGisSubsystemPtr,
        attachedAntennaMobilityModelPtr,
        channelModelSetPtr));

(*this).AddCommunicationNode(aSimNodePtr);

simNodePtrs[nodeId] = aSimNodePtr;

}//CreateNewNode//

```

通信機の生成をエージェントと同期させない場合、MultiAgentSimulator::AddCommunicationNode() のかわりに NetworkSimulator::AddNode() と MultiAgentSimulator::AttachCommunicationNode() を利用する。この際、通信機とエージェントは別々のオブジェクトとして作成し、別々の通信機 ID とエージェント ID が割り当てられていなければならない。

multisystems/multisystems_netsim.cpp

```

void MultiSystemsSimulator::CreateNewNode(
    const ParameterDatabaseReader& theParameterDatabaseReader,
    const NodeIdType& nodeId,
    const shared_ptr<ObjectMobilityModel>& nodeMobilityModelPtr,
    const string& nodeTypeName)
{
    unsigned int partitionIndex = 0;

    if
(theParameterDatabaseReader.ParameterExists("parallelization-partition-index"
, nodeId)) {
        partitionIndex =
(theParameterDatabaseReader.ReadInt("parallelization-partition-index",
nodeId) %
theSimulationEnginePtr->GetNumberPartitionThreads());
} //if//


const shared_ptr<SimulationEngineInterface> simulationEngineInterfacePtr =
theSimulationEnginePtr->GetSimulationEngineInterface(
theParameterDatabaseReader, nodeId, partitionIndex);

const shared_ptr<AttachedAntennaMobilityModel>
attachedAntennaMobilityModelPtr(
new AttachedAntennaMobilityModel(nodeMobilityModelPtr));

shared_ptr<SimNode> aSimNodePtr(
new SimNode(
theParameterDatabaseReader,
theGlobalNetworkingObjectBag,
simulationEngineInterfacePtr,
nodeId,
runSeed,
theGisSubsystemPtr,
attachedAntennaMobilityModelPtr,
channelModelSetPtr));

```

```

(*this).AddNode(aSimNodePtr);

(*this).AttachCommunicationNode(agentId, aSimNodePtr);

simNodePtrs[nodeId] = aSimNodePtr;

}//CreateNewNode//

```

MultiAgentSimulator::AttachCommunicationNode()の第一引数には、通信機を取り付けたいエージェントの ID、第二引数には取り付ける端末を指定する。同一のエージェントに対して別の端末も取り付ける場合には、別の端末を第二引数として再度 API を実行する。

例では NetworkSimulator::AddNode()のすぐ後に通信機取り付けの API を呼び出しているが、この API は、エージェントに端末を取り付けたいタイミング(任意のシミュレーション時刻)で呼び出しても良い。エージェントに取り付けた通信機はエージェントと共に移動を行なう。

また、端末の取り外しを行う場合は MultiAgentSimulator::DetachCommunicationNode()を呼び出す。既にエージェントに取り付けずみの端末を別の端末に取り付ける場合には、一度取り外しの API を読んだ後、改めて別の端末に取り付ける API を呼び出す。

シミュレーション上でまだ生成されていないエージェントに対して MultiAgentSimulator::AttachCommunicationNode()を呼び出した場合は、その通信機はエージェントが生成されるのを待ってからエージェントに取り付けられる。

13.3. 通信アプリケーションでユーザ情報を取得

通信機を定義する SimNode の親クラスである AgentCommunicationNode クラスはエージェントのユーザ情報を保持する AgentResource クラスにアクセスすることが可能である。

エージェントが保持するユーザ情報を利用する場合には、AgentResource を利用する。AgentResource へのアクセスは、通信機がエージェントに取り付けられた後のマルチエージェントシミュレーションのタイムステップからアクセスが可能となる。

通信機がエージェントに取り付けられていない場合には、対象のエージェントが存在しないためエージェントのユーザ情報にアクセスすることが出来ない。)

13.4. 道路ネットワークに依存する伝搬モデルの利用

道路ネットワークに依存する伝搬モデルを用いてシミュレーションを行う場合、道路の区切りを交差点とするか、あるいは道路を構成する頂点とするかを適切に設定する必要がある。道路の区切りの指定は".config"ファイルの以下のパラメータで設定される。

パラメータ	プロパティ名	値
道路の分割	GIS-LOS-BREAK-DOWN-CURVED-ROAD-INTO-STRAIGHT-ROADS	true/false

true の場合は、道路を、道路を構成する頂点(道路においてカーブを表現している頂点)の単位で分割して道路ネットワークを構築する。一方で false の場合には、道路ネットワークの構成単位は、交差点単位で分割される道路とする。

MultiAgentExtensionModule を利用する場合、通常この値は false として設定を行うが道路ネットワークに遺贈する伝搬モデルとして併用してシミュレーションを行う場合、伝搬モデルの想定している形で道路ネットワークを設定する。

道路ネットワークに遺贈する伝搬モデルと、モデルが想定している道路ネットワークの区切りの単位を示す。

伝搬モデル	道路の区切り	GIS-LOS-BREAK-DOWN-CURVED-ROAD-INTO-STRAIGHT-ROADS の値
ITU-R P.1411	道路を構成する 頂点単位	true
Taga	道路を構成する 頂点単位	True

13.5. 通信と車両の状態の連携

車車間通信等を想定し、速度・車間距離等の状況から自車速度を変更するシミュレーションを実施する場合、ドライバーモデル、アプリケーションクラスを利用する。

ドライバーモデルでの自車速度(自車加速度)の計算は以下の API に含まれ、この計算で利用している各変数を変更することで、動的な速度の変更が可能となる。

ファイル名 :multiagent/multiagent_behavior.cpp

クラス名 :VehicleDriverBehavior

API:CalculateIntelligentDriversModelAcceleration()

API では加減速計算を実行する車両情報、前方車までの距離、前方車の速度、自車の最大速度を引数として加速度の計算を行っている。通信との連携を行う場合、この API で利用している変数に、通信で得られた情報を利用する。通信情報に関しては、以下の手順で通信アプリケーションのクラスを取得する。

1) Agent クラスのメンバ変数を介して端末情報を取得

クラス定義ファイル名 :multiagent/multiagent_agentsim.h

クラス名 :Agent

メンバ変数 :communicationNodePtrs

2) 端末情報からアプリケーションレイヤの情報を取得

クラス定義ファイル名 :simulator/scensim_netsim.h

クラス名 :NetworkNode

API 名 :GetAppLayerPtr()

3) アプリケーションレイヤからアプリケーション情報を取得

クラス定義ファイル名 :simulator/scensim_application.h

クラス名 :ApplicationLayer

API 名 :GetApplicationPtr<アプリケーションの型名>()

引数: アプリケーション名 (Application クラスのメンバ変数の instanceId で指定される文字列。例えば Basic Safety Message Application の場合は"BsmApp")

14. その他

その他の詳細に関して解説する。

14.1. 乱数シード

乱数生成器は各エージェントが保持し、マルチエージェントでは以下の処理で利用する。

エージェント	行動	乱数生成条件	生成タイミング	生成回数
Human	-	車の有無	シミュレーション開始時	1回
		自転車の有無	シミュレーション開始時	1回
		経路の決定	経路候補から経路決定時	1回
		経路計算式	計算式タイミング	各エージェントについて、計算式が利用される度に、乱数を利用した関数について UNI/1回 UNID/1回 NORMAL/12回 EXPDIST/1回 POISSON/N回 ERLANG/N回
		ユーティリティ 1 計算式		
		ユーティリティ 2 計算式		
	Pedestrian	初期配置、目的地の決定	行動リストに場所の指定 RandomBuilding、RandomPark、RandomPOI、Area を利用し、その行動を実行する際	場所の決定ごとに N 回
		歩行者用道路の移動オフセットの計算	歩行者行動開始時	1回
	VehicleDrive/ TaxiDriver/Bu	歩行者認識	シミュレーション同期間隔毎、交差点	前方の通過予定の交差点数

sDriver		状況確認時	
Pedestrian/ VehicleDrive/ TaxiDriver/Bu sDriver/ BusGuest/Trai nGuest	経路再計算トリガの判 定	via ポイント到着時	1 回
BusGuest/Trai nGuest	予定していた公共交通 機関に遅れた場合の 経路計算	シミュレーション同 期間隔毎、予定し ていた公共交通機 関に遅れて駅・バ ス停で待っている 場合	1 回
BusGuest/Trai nGuest	バス・電車内での位置 決定	車両に乗車する際	2 回

14.2. マルチエージェントデバッグ情報

コンパイル時に MULTIAGENT_DEBUG の定義を含めてシミュレータを作成することで、標準出力にデバッグ用の情報が出力される。

出力形式

時刻:[エージェント ID(最後に訪れた頂点番号)] デバッグ情報

各種デバッグ項目

デバッグ項目	デバッグ情報
エージェントの追加	Add Agent <エージェント番号>
経路無し	No Route
経路検索	SearchRoute v<始点頂点番号> to v<終点頂点番号>
経路候補	RouteCandidate<経路重み>:<経路リスト情報>
経路決定	Route:<経路情報>
目的地設定	SetDestination <目的地名>
行動終了	Finished Task
目的地が定員であった	Destination reached a limit capacity.
全ての行動が終了	Completed All Task
行動モデルの計算	Recalculate Behavior for <経路リスト情報>
行動モデルの割り当て	New Behavior:<行動名>
行動モデルの終了	End Behavior
ユーティリティ更新	Utility = <ユーティリティ 1>/<ユーティリティ 2>
PathQuery の判定	PathQuery = <有効/無効>, , DelayForLastViaPoint = <最後に訪れた via ポイントでの遅延>, DelayForNextViaPoint = <次の via ポイントに対する遅延>, DelayForExpectedArrivalTime = <目的地への到着予定期刻に対する現在の経路の遅延>, VehicleDelay = <乗車予定の路線の遅れ>, Congestion = <>, Utility = <ユーティリティ 1>/<ユーティリティ 2>
電車の生成	Create Train <エージェント ID>
バスの生成	Create Bus <エージェント ID>
タクシーの割り当て	Assign Taxi <タクシードライバエージェント ID> for <タクシーゲストエージェント ID>
位置の更新	<行動モデル名> (x 座標, y 座標)

車両の出発	Departed
車両の到着	Arrived
タクシードライバのタクシーゲストを乗せてからの経路	RouteToTransport
タクシードライバのタクシーゲストまでの経路	RouteToCustomer
タクシードライバの所属駅への経路	RouteToStation
定員オーバーによる経路の再計算	Route Recalculation for reaching a vehicle limit.
路線無効状態による経路の再計算	Route Recalculation for nonavailable line
車両に遅れた場合の経路の再計算	Route Recalculation for missed a vehicle
乗り換え経路	WalkToStop <経路情報>

<経路リスト情報>

行動モデルの変化の単位での<経路情報>の連続

<経路情報>

自由歩行:FreeWalk

道路の移動:Road(行動番号) <頂点番号>(r<道路番号>)

公共交通機関の移動:Public <公共交通機関の開始点番号>-<公共交通機関の終了点番号>(経路番号)

経路無し:Empty

行動番号	値
道路歩行	3
自転車	4
自家用車ドライバ	5
バスドライバ	11
タクシードライバ	12
タクシーゲスト	8

※頂点番号、道路番号はGISデバッグ情報の値となる。公共交通機関の点番号は公共交通機関情報システムの PublicVehicleTable の GetStopName より駅またはバス停名を取得可能である。

14.3. GIS デバッグ情報

scensim_gis.cpp/h の GisSubsystem クラスが保持している OutputVertexInformation() 関数により頂点情報と GIS オブジェクトとの関連を標準出力に出力可能である。

14.4. エージェントの挙動に関する注意点

エージェントの挙動に関する設定の注意点を開設する。

- 歩行者が車線道路の中心を歩く

4車線の道路ではなく2車線の道路が2つ並んでいるまたは重なっている場合、4車線道路の中心を歩いているように見える場合があるが、歩行者自体はそれぞれの道路の端を移動している。このような場合、シナリオの編集を行い当該の道路を一本の道路として車線数4(上り2、下り2)とすることで、歩行者は道路の端を移動するようになる。

- エージェントが駅の周りを回る

一斉に同じ駅に向かうようなシナリオでは道路から駅構内/駅に向かう経路が混雑し、別の入口から入って駅構内の乗車地点まで移動した方が早いと判断することがある。この場合、迂回する経路を移動するエージェントが発生する。

PathQueryによる経路の再検索は、最後に検索をした時点から最小経路検索間隔(MinRouteRecalcInterval)を経過していれば検索可能であり、混雑状況と最小経路検索間隔の指定によっては頻繁に経路の変更を行うエージェントが発生する。

頻繁に経路を変更して同じ個所を回るような挙動を行う場合、シナリオ設定において経路の再検索の最小間隔(MinRouteRecalcInterval)を十分な間隔に変更することで、何度も経路変更を行うようなエージェントは発生しなくなる。

- バスの台数の不足について

エージェント数が非常に多い場合に交通渋滞が発生する。この時、新しいバスは時刻通りに出現するがバスが目的地に到着出来ないという状況が発生し、同時刻内でのバスの利用数が上限に達しエラーとなる場合がある。このようなシナリオでシミュレーションを完了するためには、バスの台数を増やす必要がある。

15. 参考文献

1. M. Treiber, A. Hennecke and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E* 62, no 2, pp. 1805-1824, 2000.

