(사업명: 예외 상황 합성데이터 생성 및 인공지능 예측 모델 고도화 기술 개발)

**시스템 설계서**

[3.2.3 에이전트 행동 모델링 서브시스템 17](#에이전트-행동-모델링-서브시스템)

[4.3 에이전트 행동 모델링 서브시스템 22](#에이전트-행동-모델링-서브시스템-1)

[4.3.1 구조 22](#구조-2)

[4.3.2 주요 기능 23](#주요-기능-2)

[4.3.3 인터페이스 23](#인터페이스-2)

# 시스템 기능 및 구조

## 시스템 기능

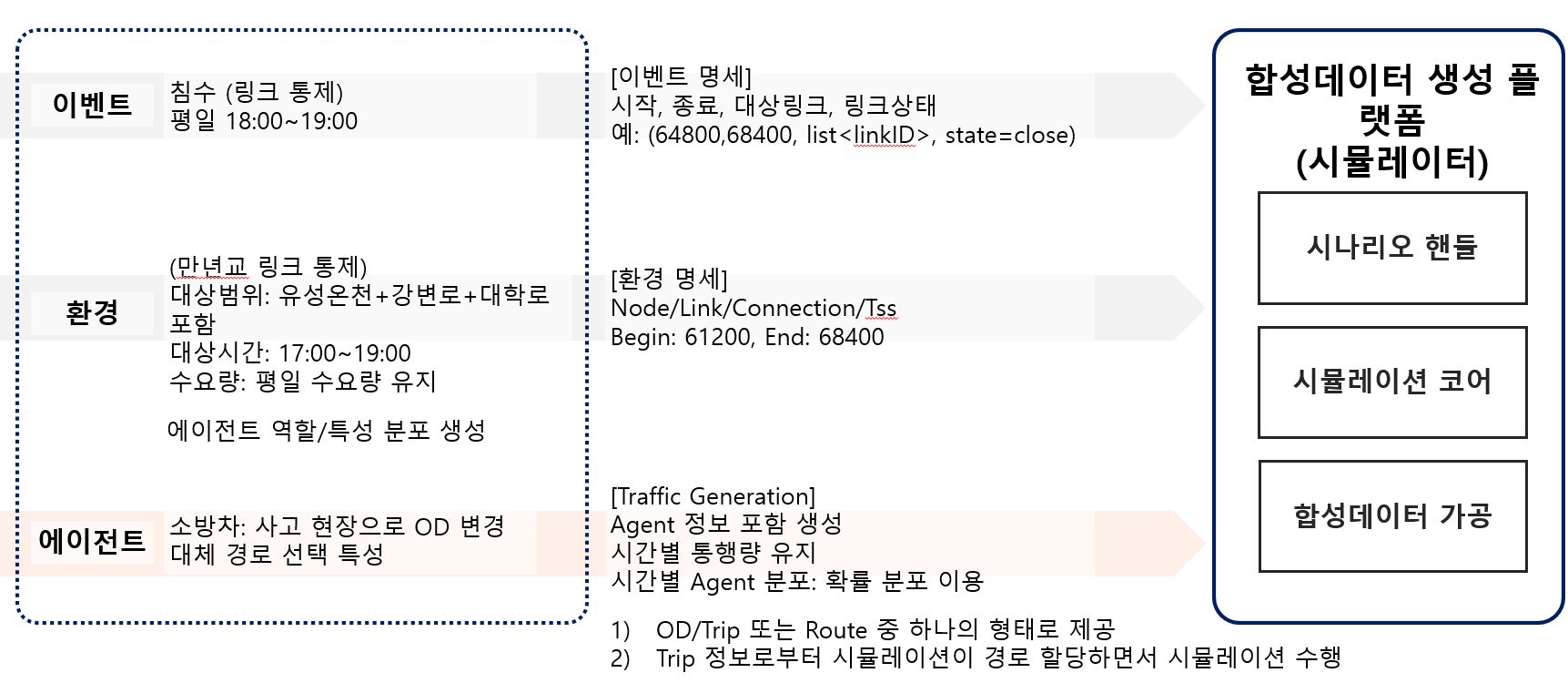
#### 에이전트 행동 모델링

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

시나리오 별로 정의된 에이전트 정적 속성에 따라 에이전트의 역할과 에이전트의 출발지, 목적지, 경로 선택 성향을 반영한 특성를 정의한다. 정의된 에이전트의 역할/특성에 대한 분포의 변화를 시간에 따라 정의한다. 정의된 에이전트의 정적속성과 시간에 따른 유형 분포를 반영하여 정의된 수 만큼의 에이전트를 생성하고 이를 하나의 에이전트 정책으로 묶는다. 여러가지 변수를 적용하여 에이전트 정책에 대한 1만가지의 변위를 생성한다.

다음은 시나리오에 따라 생성되는 입력값의 예시이다.



## 시스템의 구조

### 에이전트 행동 모델링 서브시스템

텍스트, 스크린샷, 도표, 번호이(가) 표시된 사진

자동 생성된 설명

# 시스템 상세 구조 및 기능

## 에이전트 행동 모델링 서브시스템

### 구조

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

### 주요 기능

도메인 지식 기반 예외상황 에이전트 정적 모델링에 따라 에이전트 유형 집합을 생성한다.

에이전트 유형 집합은 역할, 특성, Trip정보를 갖는다.

각 유형 집합은 시간에 따른 유형 분포를 갖는다.

시간에 따라 유형 분포를 갖는 에이전트 유형 집합을 생성한다.

인스턴스로 생성된 에이전트는 출발지, 목적지, 경로 선택에 대하여 여러 변위를 생성한다.

여러 변위는 참고 통행량 정보의 확률에 따라 생성된다.

### 인터페이스

행동모델에서 사용한 SUMO 시뮬레이션에서 주요 입력 파일은 ,net.xml과 routes.xml입니다. 시뮬레이션 출력파일은 tripinfos.xml과 fcd-export.xml입니다.

해당 입출력 파일의 주요 요소와, XSD 스키마 정의에 대해 각각 살펴 보겠습니다.

아래는 **SUMO .net.xml 파일**의 주요 요소인 location, edge, tlLogic, junction, connection, roundabout 각각에 대한 **XSD 정의**와 **설명**입니다.

#### <location>: 네트워크 위치

##### **XSD 정의**

xml

<xs:element name="location">  
<xs:complexType>  
<xs:attribute name="netOffset" type="xs:string" use="required"/>  
<xs:attribute name="convBoundary" type="xs:string" use="required"/>  
<xs:attribute name="origBoundary" type="xs:string" use="required"/>  
<xs:attribute name="projParameter" type="xs:string" use="optional"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **netOffset**: 네트워크 원점 좌표 (x,y 형식, 미터 단위).
* **convBoundary**: 네트워크의 변환된 경계 (x1,y1,x2,y2 형식).
* **origBoundary**: 원래 지리적 경계 (longitude1,latitude1,longitude2,latitude2 형식).
* **projParameter**: 좌표 변환 파라미터 (예: UTM 또는 WGS84).

#### <edge>: 도로 정의

##### **XSD 정의**

xml

<xs:element name="edge">  
<xs:complexType>  
<xs:sequence>  
<xs:element name="lane" maxOccurs="unbounded"/>  
</xs:sequence>  
<xs:attribute name="id" type="xs:string" use="required"/>  
<xs:attribute name="from" type="xs:string" use="optional"/>  
<xs:attribute name="to" type="xs:string" use="optional"/>  
<xs:attribute name="priority" type="xs:integer" use="optional"/>  
<xs:attribute name="spreadType" type="xs:string" use="optional"/>  
<xs:attribute name="function" type="xs:string" use="optional"/>  
<xs:attribute name="shape" type="xs:string" use="optional"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **id**: 엣지의 고유 식별자.
* **from**, **to**: 시작과 끝 노드.
* **priority**: 엣지의 우선순위.
* **spreadType**: 차선 배치 방식 (center, left, right).
* **function**: 엣지의 역할 (normal, internal, connector 등).
* **shape**: 엣지의 기하학적 좌표 경로 (x1,y1 x2,y2 ... 형식).

#### <tlLogic>: 신호등 로직 정의

##### **XSD 정의**

xml

<xs:element name="tlLogic">  
<xs:complexType>  
<xs:sequence>  
<xs:element name="phase" maxOccurs="unbounded">  
<xs:complexType>  
<xs:attribute name="duration" type="xs:integer" use="required"/>  
<xs:attribute name="state" type="xs:string" use="required"/>  
</xs:complexType>  
</xs:element>  
</xs:sequence>  
<xs:attribute name="id" type="xs:string" use="required"/>  
<xs:attribute name="type" type="xs:string" use="optional"/>  
<xs:attribute name="programID" type="xs:string" use="optional"/>  
<xs:attribute name="offset" type="xs:integer" use="optional"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **id**: 신호등 로직의 고유 ID.
* **type**: 신호등 로직 유형 (static, actuated 등).
* **programID**: 프로그램 ID.
* **phase**:
  + **duration**: 각 단계의 지속 시간(초).
  + **state**: 신호등 상태 (G: 녹색, r: 적색, y: 황색).

#### <junction>: 교차점 정의

##### **XSD 정의**

xml

<xs:element name="junction">  
<xs:complexType>  
<xs:attribute name="id" type="xs:string" use="required"/>  
<xs:attribute name="type" type="xs:string" use="required"/>  
<xs:attribute name="x" type="xs:double" use="required"/>  
<xs:attribute name="y" type="xs:double" use="required"/>  
<xs:attribute name="incLanes" type="xs:string" use="optional"/>  
<xs:attribute name="intLanes" type="xs:string" use="optional"/>  
<xs:attribute name="shape" type="xs:string" use="optional"/>  
<xs:attribute name="radius" type="xs:double" use="optional"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **id**: 교차점 ID.
* **type**: 교차점 유형 (priority, traffic\_light, internal).
* **x, y**: 교차점의 좌표 (미터).
* **incLanes**: 교차점으로 진입하는 차선.
* **intLanes**: 교차점 내부의 차선.
* **shape**: 교차점의 기하학적 형태.
* **radius**: 교차점의 반경.

#### <connection>: 도로 간 연결 정의

##### **XSD 정의**

xml

<xs:element name="connection">  
<xs:complexType>  
<xs:attribute name="from" type="xs:string" use="required"/>  
<xs:attribute name="to" type="xs:string" use="required"/>  
<xs:attribute name="fromLane" type="xs:integer" use="optional"/>  
<xs:attribute name="toLane" type="xs:integer" use="optional"/>  
<xs:attribute name="via" type="xs:string" use="optional"/>  
<xs:attribute name="dir" type="xs:string" use="optional"/>  
<xs:attribute name="state" type="xs:string" use="optional"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **from**, **to**: 연결 시작 및 종료 엣지.
* **fromLane**, **toLane**: 시작 및 종료 차선.
* **via**: 중간 노드 또는 내부 엣지.
* **dir**: 연결 방향 (l, r, s).
* **state**: 연결 상태 (M: 허용, o: 금지).

#### <roundabout>: 로터리 정의

##### **XSD 정의**

xml

<xs:element name="roundabout">  
<xs:complexType>  
<xs:attribute name="nodes" type="xs:string" use="required"/>  
<xs:attribute name="edges" type="xs:string" use="required"/>  
</xs:complexType>  
</xs:element>

##### **설명**

* **nodes**: 로터리에 포함된 노드 ID들 (공백으로 구분).
* **edges**: 로터리를 구성하는 엣지 ID들 (공백으로 구분).

SUMO의 .net.xml 파일에서 특정 **Edge**에서 도달 가능한 모든 **Edge**를 찾으려면, 네트워크를 그래프 데이터로 모델링한 후 그래프 탐색 알고리즘을 사용하면 됩니다. 일반적으로 다음과 같은 과정이 필요합니다:

#### .net.xml 파일을 파싱

.net.xml 파일에서 **Edge**, **Connection** 데이터를 추출하여 그래프를 생성합니다.

##### **Edge와 Connection의 관계**

* **<edge>**: 도로를 나타냅니다.
* **<connection>**: 특정 Edge의 Lane 간의 연결을 정의합니다.

xml

<edge id="A" from="n1" to="n2" ...>  
<lane id="A\_0" ... />  
</edge>  
<connection from="A" to="B" fromLane="0" toLane="0" ... />

* from="A", to="B"인 **connection**은 Edge A에서 B로 이동할 수 있음을 나타냅니다.

#### Connection의 state 속성의 의미

* **state="M"**:
  + 차량 이동이 \*\*허용(Movable)\*\*된 상태를 나타냅니다.
  + 경로 탐색 및 시뮬레이션에서 유효한 연결로 간주됩니다.
* **state="o"**:
  + 차량 이동이 \*\*금지(Obstacle)\*\*된 상태를 나타냅니다.
  + 해당 Connection은 경로 탐색에서 무시해야 합니다.
* **state="-"**:
  + 연결 상태가 정의되지 않았음을 나타냅니다.
  + 기본적으로 이동이 허용되지 않는 것으로 간주됩니다.

#### 

#### 주의사항

1. **state가 없는 경우 기본값 처리**:
   * state 속성이 없는 Connection은 이동 불가능한 것으로 간주됩니다.
   * 필요 시 기본값을 "-"로 설정하고 제외 처리합니다.
2. **특정 방향 연결 제한**:
   * dir 속성(예: l, r, s)으로 이동 방향을 추가적으로 필터링할 수 있습니다.
   * 특정 시뮬레이션 목적에 따라 state와 함께 조건을 확장할 수 있습니다.

#### 확장 기능

* **특정 Lane 필터링**: fromLane 또는 toLane 값을 기준으로 필터링하여 특정 차선의 연결만 포함할 수 있습니다.
* **Edge 속성 통합**: Edge의 속성(예: 속도, 차선 수)을 기준으로 유효한 경로만 포함하는 그래프를 생성할 수 있습니다.

#### 결론

* \*\*state="M"\*\*인 Connection만 경로 탐색에 포함해야 합니다.
* 불필요한 Connection(예: state="o" 또는 state="-")은 무시하여 경로 유효성을 유지합니다

#### 그래프 데이터 생성

XML 파싱 라이브러리(xml.etree.ElementTree, lxml 등)를 사용하여 **Edge**와 **Connection** 데이터를 그래프 구조로 변환합니다.

#### 그래프 탐색

특정 Edge에서 도달 가능한 모든 Edge를 찾기 위해 **DFS**(깊이 우선 탐색) 또는 **BFS**(너비 우선 탐색) 알고리즘을 사용합니다.

SUMO의 **Routes** 파일은 시뮬레이션에 등장하는 차량과 그 차량의 경로(Edges)를 정의합니다. 위의 예제는 여러 차량의 **id**, **출발 시간**, 그리고 **경로**에 대한 정보를 XML 형식으로 보여줍니다.

#### SUMO Routes 파일 요소 설명

##### **1. <routes>: 루트 파일의 루트 요소**

* <routes>는 모든 차량, 경로, 흐름 등을 정의하는 컨테이너입니다.
* 주요 속성:
  + xsi: XML Schema의 네임스페이스 URI.
  + noNamespaceSchemaLocation: 경로 파일의 XSD 스키마 정의 위치를 지정.

##### **2. <vehicle>: 차량 정의**

* 각 <vehicle> 태그는 하나의 차량을 나타냅니다.
* 주요 속성:

|  |  |
| --- | --- |
| 속성 | 설명 |
| **id** | 차량의 고유 식별자. |
| **depart** | 차량의 출발 시간 (초 단위). |
| **departLane** | 차량이 출발할 차선 선택 기준 (best, random, free 등). |

##### **3. <route>: 차량의 경로**

* 각 <route> 태그는 차량이 이동할 **Edge ID**의 순서를 나타냅니다.
* 주요 속성:

|  |  |
| --- | --- |
| 속성 | 설명 |
| **edges** | 차량이 이동할 도로(Edge)의 ID를 공백으로 구분한 목록. |

#### 

#### 예제 분석

xml

<routes xsi="http://www.w3.org/2001/XMLSchema-instance" noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/routes\_file.xsd">

`<vehicle id="3" depart="4.00" departLane="best">`  
  
 `<route edges="563100096 563113060 -563113059 563100081 -563100066 563100065 -563101184 -563101155 563101963 -563104129" />`  
  
`</vehicle>`  
  
`<vehicle id="429517" depart="86388.00" departLane="best">`  
  
 `<route edges="563112313 -563112308 -563112310 -563103201 563101166 563101187" />`  
  
`</vehicle>`  
  
`<vehicle id="429518" depart="86389.00" departLane="best">`  
  
 `<route edges="563103895 -563103894 563101247 563107678 -563107679 -563101163 563107677 -563107676 -563101129 563112968 -563112967 -563101104 -563101091 -563100809 563100808 -563100992" />`  
  
`</vehicle>`  
  
`<vehicle id="429520" depart="86396.00" departLane="best">`  
  
 `<route edges="-563107369 -563100532 -563100523 -563100119 563107377 -563107378 563100527 563107382 -563107383 -563106510 563106626 563111381 -563111380 563101799 563107280 -563107279 -563102383" />`  
  
`</vehicle>`

</routes>

##### **요소별 설명**

1. **첫 번째 차량 (<vehicle id="3">)**:
   * 출발 시간: **4.00초**.

경로:

563100096 → 563113060 → -563113059 → 563100081 → -563100066 → 563100065 ...

* -로 시작하는 Edge는 **역방향 Edge**를 의미.

1. **두 번째 차량 (<vehicle id="429517">)**:
   * 출발 시간: **86388.00초**.

경로:

563112313 → -563112308 → -563112310 → -563103201 → 563101166 → 563101187

1. **세 번째 차량 (<vehicle id="429518">)**:
   * 출발 시간: **86389.00초**.

경로:

563103895 → -563103894 → 563101247 → 563107678 → -563107679 ...

1. **네 번째 차량 (<vehicle id="429520">)**:
   * 출발 시간: **86396.00초**.

경로:  
diff

-563107369 → -563100532 → -563100523 → -563100119 → 563107377 → -563107378 ...

#### 파일의 구조화된 XSD 정의

##### **XSD 스키마**

xml

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

`<xs:element name="routes">`  
  
 `<xs:complexType>`  
  
 `<xs:sequence>`  
  
 `<xs:element name="vehicle" maxOccurs="unbounded">`  
  
 `<xs:complexType>`  
  
 `<xs:sequence>`  
  
 `<xs:element name="route" maxOccurs="1">`  
  
 `<xs:complexType>`  
  
 `<xs:attribute name="edges" type="xs:string" use="required"/>`  
  
 `</xs:complexType>`  
  
 `</xs:element>`  
  
 `</xs:sequence>`  
  
 `<xs:attribute name="id" type="xs:string" use="required"/>`  
  
 `<xs:attribute name="depart" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="departLane" type="xs:string" use="optional"/>`  
  
 `</xs:complexType>`  
  
 `</xs:element>`  
  
 `</xs:sequence>`  
  
 `<xs:attribute name="xsi" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="noNamespaceSchemaLocation" type="xs:string" use="optional"/>`  
  
 `</xs:complexType>`  
  
`</xs:element>`

</xs:schema>

##### **XSD 구조 요약**

1. **<routes>**:
   * 자식 요소로 <vehicle> 포함.
   * 스키마 위치 정보 속성 포함 (xsi, noNamespaceSchemaLocation).
2. **<vehicle>**:
   * 속성: id, depart, departLane.
   * 자식 요소로 <route> 포함.
3. **<route>**:
   * edges: 차량의 Edge ID 목록을 공백으로 구분한 문자열.

#### 특별한 조건 처리

1. **역방향 Edge (-로 시작)**:
   * 경로에서 -가 붙은 Edge는 도로의 **역방향**을 의미.
   * 시뮬레이션에서는 Edge의 양방향 여부에 따라 차량이 이동 가능 여부가 결정됩니다.
2. **departLane="best"**:
   * 차량이 가장 적합한 차선에서 출발하도록 설정.
   * departLane 속성은 best, random, 또는 특정 차선 번호(예: 0, 1)로 설정 가능.

SUMO의 **tripinfos** 파일은 시뮬레이션에서 각 차량의 이동 정보를 포함하며, XML 형식으로 작성됩니다. 이 파일은 주로 차량의 출발, 도착, 이동 시간, 대기 시간, 경로 길이 등을 분석하는 데 사용됩니다.

#### 구조 및 주요 요소 설명

##### **파일 헤더**

xml

<?xml version="1.0" encoding="UTF-8"?>

<tripinfos xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

`xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/tripinfo\_file.xsd">`

* **xmlns:xsi**: XML Schema 네임스페이스 정의.
* **xsi:noNamespaceSchemaLocation**: tripinfo\_file.xsd 파일로 XML의 유효성을 검증할 수 있도록 경로를 정의.

##### **<tripinfo> 요소**

* 각 <tripinfo>는 차량 한 대의 이동 정보를 나타냅니다.

###### **속성**

|  |  |
| --- | --- |
| 속성명 | 설명 |
| **id** | 차량의 고유 식별자. |
| **depart** | 차량이 시뮬레이션에서 출발한 시간(초). |
| **departLane** | 차량이 출발한 차선의 ID. |
| **departPos** | 차량의 차선 내 출발 위치(미터). |
| **departSpeed** | 차량의 출발 속도(미터/초). |
| **departDelay** | 차량의 출발 지연 시간(초). |
| **arrival** | 차량이 도착한 시간(초). |
| **arrivalLane** | 차량이 도착한 차선의 ID. |
| **arrivalPos** | 차량의 차선 내 도착 위치(미터). |
| **arrivalSpeed** | 차량의 도착 속도(미터/초). |
| **duration** | 차량의 총 이동 시간(초). |
| **routeLength** | 차량의 경로 길이(미터). |
| **waitingTime** | 차량이 대기한 총 시간(초). |
| **waitingCount** | 차량이 대기한 횟수. |
| **stopTime** | 차량이 정지한 총 시간(초). |
| **timeLoss** | 차량이 최적 조건보다 손실된 시간(초). |
| **rerouteNo** | 차량이 재경로를 설정한 횟수. |
| **devices** | 차량에 적용된 SUMO 장치 정보(예: FCD, TripInfo 장치 등). |
| **vType** | 차량의 유형(기본값: DEFAULT\_VEHTYPE). |
| **speedFactor** | 차량의 속도 계수(차량마다 속도가 무작위로 조정된 비율). |
| **vaporized** | 차량이 증발되었는지(시뮬레이션 내 삭제된 차량). |

#### 예제 분석

##### **첫 번째 <tripinfo>**

xml

<tripinfo id="26" depart="29.00" departLane="563100809\_0" departPos="5.10" departSpeed="0.00" departDelay="0.00"

`arrival="72.50" arrivalLane="-563101241\_0" arrivalPos="63.98" arrivalSpeed="28.14" duration="43.50"`   
  
 `routeLength="507.76" waitingTime="0.00" waitingCount="0" stopTime="0.00" timeLoss="28.52" rerouteNo="0"`   
  
 `devices="tripinfo\_26 fcd\_26" vType="DEFAULT\_VEHTYPE" speedFactor="0.83" vaporized=""/>`

|  |  |  |
| --- | --- | --- |
| 속성명 | 값 | 설명 |
| **id** | 26 | 차량 ID. |
| **depart** | 29.00 | 29초에 출발. |
| **departLane** | 563100809\_0 | 563100809\_0 차선에서 출발. |
| **arrival** | 72.50 | 72.5초에 도착. |
| **arrivalLane** | -563101241\_0 | 역방향 차선 -563101241\_0에서 도착. |
| **duration** | 43.50 | 총 이동 시간 43.5초. |
| **routeLength** | 507.76 | 이동한 경로의 총 길이는 507.76m. |
| **waitingTime** | 0.00 | 대기 시간이 없음. |
| **timeLoss** | 28.52 | 최적 이동 조건보다 28.52초 손실. |

##### **두 번째 <tripinfo>**

xml

<tripinfo id="6973" depart="3337.00" departLane="-563114172\_0" departPos="5.10" departSpeed="0.00" departDelay="0.00"

`arrival="3525.50" arrivalLane="563105332\_0" arrivalPos="63.60" arrivalSpeed="31.24" duration="188.50"`   
  
 `routeLength="1580.07" waitingTime="79.50" waitingCount="3" stopTime="0.00" timeLoss="152.25" rerouteNo="0"`   
  
 `devices="tripinfo\_6973 fcd\_6973" vType="DEFAULT\_VEHTYPE" speedFactor="1.08" vaporized=""/>`

|  |  |  |
| --- | --- | --- |
| 속성명 | 값 | 설명 |
| **id** | 6973 | 차량 ID. |
| **depart** | 3337.00 | 3337초에 출발. |
| **duration** | 188.50 | 이동 시간은 188.5초. |
| **routeLength** | 1580.07 | 경로의 길이는 1580.07m. |
| **waitingTime** | 79.50 | 총 대기 시간은 79.5초. |
| **waitingCount** | 3 | 차량이 대기한 횟수는 3번. |
| **timeLoss** | 152.25 | 최적 조건보다 152.25초 손실. |

#### XSD 스키마

##### **XSD 정의**

xml

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

`<xs:element name="tripinfos">`  
  
 `<xs:complexType>`  
  
 `<xs:sequence>`  
  
 `<xs:element name="tripinfo" maxOccurs="unbounded">`  
  
 `<xs:complexType>`  
  
 `<xs:attribute name="id" type="xs:string" use="required"/>`  
  
 `<xs:attribute name="depart" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="departLane" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="departPos" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="departSpeed" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="departDelay" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="arrival" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="arrivalLane" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="arrivalPos" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="arrivalSpeed" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="duration" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="routeLength" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="waitingTime" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="waitingCount" type="xs:integer" use="optional"/>`  
  
 `<xs:attribute name="stopTime" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="timeLoss" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="rerouteNo" type="xs:integer" use="optional"/>`  
  
 `<xs:attribute name="devices" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="vType" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="speedFactor" type="`

SUMO의 \*\*timeLoss\*\*는 차량이 이동 중 경험한 시간 손실을 나타내며, 차량이 최적 조건에서 이동했을 경우와 실제 시뮬레이션에서 이동한 시간 간의 차이로 계산됩니다.

최적 조건은 차량이 도로의 제한 속도에서 장애물(정지, 대기, 신호등, 교통 혼잡 등)에 영향을 받지 않고 자유롭게 이동했을 경우의 시간을 기준으로 합니다.

#### timeLoss 계산 과정

1. **최적 이동 시간 (Optimal Time) 계산**: 최적 이동 시간은 차량이 **경로의 총 길이**를 각 Edge의 **최대 허용 속도**로 이동했을 경우의 시간을 합산하여 계산됩니다.
   * Edge 의 길이 (미터 단위).
   * Edge 의 제한 속도 (미터/초).
2. **실제 이동 시간 (Actual Time)**: 실제 이동 시간은 차량이 시뮬레이션에서 출발부터 도착까지 걸린 총 시간입니다.
   * arrival​: 차량의 도착 시간.
   * depart​: 차량의 출발 시간.
3. **시간 손실 계산**: 시간 손실은 실제 이동 시간과 최적 이동 시간 간의 차이로 정의됩니다.  
   timeLoss=actual​−optimal​

#### 시간 손실 계산

시간 손실은 실제 이동 시간과 최적 이동 시간의 차이로 계산됩니다:

timeLoss=40−55=−15초 (음수일 경우, 결과 재검증 필요)

#### 특별한 주의사항

* **Edge의 이동 방향**:
  + Edge의 방향(정방향 또는 역방향, -edge)이 경로에 정확히 반영되어야 합니다.
  + 잘못된 방향을 사용하면 최적 이동 시간과 실제 이동 시간이 일치하지 않을 수 있습니다.
* **교통 흐름 조건**:
  + 최적 조건에서는 차량이 교차로나 신호등에서 대기하거나 감속하지 않는다고 가정합니다.
  + 실제 이동 시간은 이러한 조건을 포함하므로 손실 값에 영향을 미칩니다.
* **timeLoss가 음수**:
  + 이는 최적 경로 계산에서 입력 데이터가 잘못되었음을 나타낼 수 있습니다.
  + 다시 계산해야 합니다.

SUMO의 **FCD(Full Control Data) Export**는 시뮬레이션 중 각 시간 단위(timestep)에서 차량의 상태를 기록한 결과 파일입니다. **FCD-export**는 차량의 위치, 속도, 차선 등 다양한 정보를 포함하며, 시뮬레이션 데이터를 분석하거나 시각화하는 데 사용됩니다.

#### 구조 및 주요 요소 설명

##### **파일 헤더**

xml

<fcd-export xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

`xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/fcd\_file.xsd">`

* **xmlns:xsi**: XML Schema 네임스페이스 정의.
* **xsi:noNamespaceSchemaLocation**: FCD 파일의 XSD 스키마 정의 위치.

##### **1. <timestep>: 시뮬레이션 시간 단위**

* 각 <timestep>은 특정 시간에서의 차량 상태를 나타냅니다.
* **속성**:
  + **time**: 현재 시간(초 단위).

##### **2. <vehicle>: 차량 상태**

* <timestep> 내의 <vehicle> 요소는 해당 시간에서 차량의 정보를 제공합니다.
* **속성**:

|  |  |
| --- | --- |
| 속성명 | 설명 |
| **id** | 차량의 고유 식별자. |
| **x** | 차량의 X 좌표 (지도상의 위치, 미터 단위). |
| **y** | 차량의 Y 좌표 (지도상의 위치, 미터 단위). |
| **angle** | 차량의 방향 (도로 축 기준, 도 단위). |
| **type** | 차량의 유형 (기본값: DEFAULT\_VEHTYPE). |
| **speed** | 차량의 속도 (m/s). |
| **pos** | 차량이 차선 내에서 이동한 거리 (미터). |
| **lane** | 차량이 있는 차선 ID. |
| **slope** | 차량이 위치한 도로의 경사도 (기울기). |

#### 예제 분석

xml

<fcd-export xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/fcd\_file.xsd">

`<timestep time="0.00"/>`  
  
`<timestep time="0.50"/>`  
  
`<timestep time="1.00"/>`  
  
`<!-- ... -->`  
  
`<timestep time="4.00">`  
  
 `<vehicle id="3" x="2735.73" y="3704.03" angle="100.85" type="DEFAULT\_VEHTYPE" speed="0.00" pos="5.10" lane="563100096\_0" slope="0.00"/>`  
  
`</timestep>`  
  
`<timestep time="3372.50">`  
  
 `<vehicle id="6585" x="1520.57" y="1949.03" angle="270.47" type="DEFAULT\_VEHTYPE" speed="0.00" pos="169.35" lane="-563113843\_0" slope="0.00"/>`  
  
 `<vehicle id="7022" x="3431.11" y="3502.44" angle="319.58" type="DEFAULT\_VEHTYPE" speed="3.62" pos="3.52" lane=":563102858\_2\_1" slope="0.00"/>`  
  
`</timestep>`

</fcd-export>

##### **첫 번째 <timestep> (time="4.00")**

xml

<timestep time="4.00">

`<vehicle id="3" x="2735.73" y="3704.03" angle="100.85" type="DEFAULT\_VEHTYPE" speed="0.00" pos="5.10" lane="563100096\_0" slope="0.00"/>`

</timestep>

|  |  |  |
| --- | --- | --- |
| 속성 | 값 | 설명 |
| time | 4.00 | 시뮬레이션 시간 4초. |
| id | 3 | 차량 ID는 3. |
| x, y | 2735.73, 3704.03 | 차량의 위치는 (2735.73, 3704.03)(미터 단위). |
| angle | 100.85 | 차량이 100.85도의 각도로 이동 중. |
| type | DEFAULT\_VEHTYPE | 기본 차량 유형. |
| speed | 0.00 | 차량은 정지 상태(속도 0). |
| pos | 5.10 | 차량이 차선 내에서 5.10m 이동. |
| lane | 563100096\_0 | 차량이 위치한 차선 ID는 563100096\_0. |
| slope | 0.00 | 도로 경사도는 0. |

##### **두 번째 <timestep> (time="3372.50")**

xml

<timestep time="3372.50">

`<vehicle id="6585" x="1520.57" y="1949.03" angle="270.47" type="DEFAULT\_VEHTYPE" speed="0.00" pos="169.35" lane="-563113843\_0" slope="0.00"/>`  
  
`<vehicle id="7022" x="3431.11" y="3502.44" angle="319.58" type="DEFAULT\_VEHTYPE" speed="3.62" pos="3.52" lane=":563102858\_2\_1" slope="0.00"/>`

</timestep>

1. **차량 6585**:
   * 시간: 3372.50초.
   * 위치: (1520.57, 1949.03).
   * 속도: 0.00 m/s (정지 상태).
   * 차선: -563113843\_0 (역방향).
2. **차량 7022**:
   * 시간: 3372.50초.
   * 위치: (3431.11, 3502.44).
   * 속도: 3.62 m/s.
   * 차선: :563102858\_2\_1 (내부 차선).

#### XSD 스키마

##### **FCD-export XSD**

xml

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

`<xs:element name="fcd-export">`  
  
 `<xs:complexType>`  
  
 `<xs:sequence>`  
  
 `<xs:element name="timestep" maxOccurs="unbounded">`  
  
 `<xs:complexType>`  
  
 `<xs:sequence>`  
  
 `<xs:element name="vehicle" maxOccurs="unbounded" minOccurs="0">`  
  
 `<xs:complexType>`  
  
 `<xs:attribute name="id" type="xs:string" use="required"/>`  
  
 `<xs:attribute name="x" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="y" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="angle" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="type" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="speed" type="xs:double" use="required"/>`  
  
 `<xs:attribute name="pos" type="xs:double" use="optional"/>`  
  
 `<xs:attribute name="lane" type="xs:string" use="optional"/>`  
  
 `<xs:attribute name="slope" type="xs:double" use="optional"/>`  
  
 `</xs:complexType>`  
  
 `</xs:element>`  
  
 `</xs:sequence>`  
  
 `<xs:attribute name="time" type="xs:double" use="required"/>`  
  
 `</xs:complexType>`  
  
 `</xs:element>`  
  
 `</xs:sequence>`  
  
 `</xs:complexType>`  
  
`</xs:element>`

</xs:schema>

#### 데이터 활용

1. **위치 및 속도 분석**:
   * 특정 시간에 차량의 위치와 속도를 분석해 교통 흐름을 평가할 수 있습니다.
2. **경로 추적**:
   * 개별 차량의 이동 경로와 차선 변화를 추적하여 시뮬레이션 검증 및 분석.
3. **교통 시각화**:
   * SUMO GUI 또는 외부 도구(Matplotlib, Paraview 등)로 데이터를 시각화.
4. **대기 및 정지 시간 계산**:
   * 속도가 0인 경우를 추적해 차량의 대기 시간을 분석.

#### Agent.xml 인터페이스 정의

1. **Routes의 확장**:
   * Routes의 vehicle 태그에 해당 Agent의 정적속성(역할, 특성)으로 구분된 에이전트 유형과 에이전트 유형 비율을 포함한다.
2. **에이전트 정책**:
   * Agent.xml은 에이전트 정책을 정의한다.