

# Programming with Java

## Events

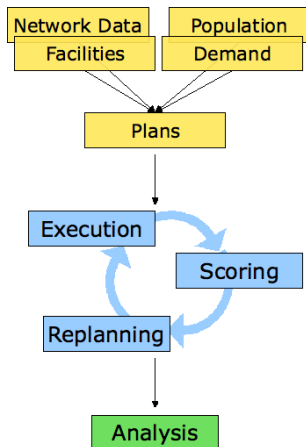
Dominik Grether

Transport System Planning and Transport Telematics  
Berlin Institute of Technology

19.05.2010

# Introduction

- Event based programming key concept in MATSim
- Several event mechanisms in framework
- This session: Events of Microsimulation
- Events describe what happens in physical world
- Events useful replanning, analysis



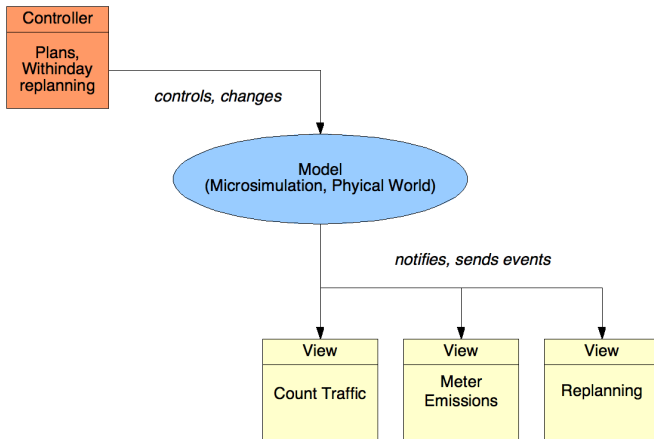
# Introduction

## What you'll learn

- Model View Controller (Observer) pattern - event based programming
- How to analyse simulation results
  - How to write an event handler
  - How to produce graphical output

# Model View Controller

The pattern at a quick glance



# Handling simulation events

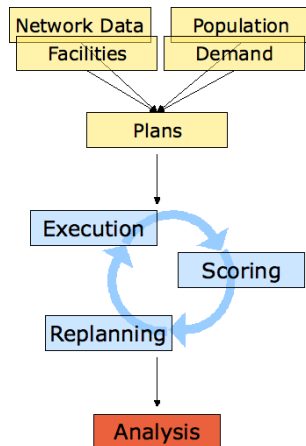
## Analyse results

- Controller generates some analysis
  - Score statistics
  - Plans (per 10th iteration)
  - Snapshots (per 10th iteration)
  - Leg histograms (per iteration)

- Task 1: Extend config for equil by

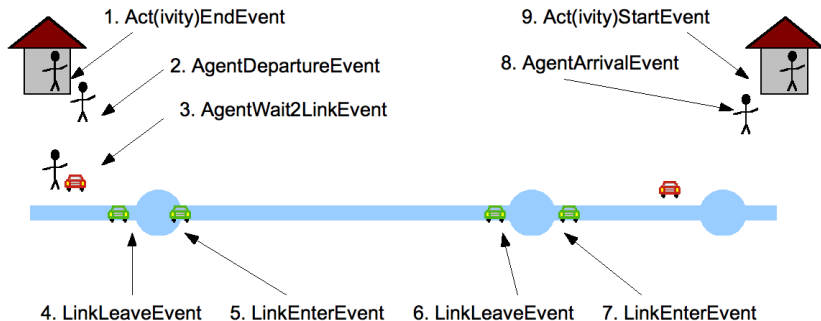
```
<module name="controller">  
  ...  
  <param name="eventsFileFormat"  
    value="txt, xml" />  
</module>
```

- Task 2: Take a look at your output directory, look at graphs and event files



# Handling simulation events

## Event structure



# Handling simulation events

## Technical information

- Events packages:
  - `org.matsim.core.api.experimental.events`
  - `org.matsim.core.events`
- Handler interfaces packages:
  - `org.matsim.core.api.experimental.events.handler`
  - `org.matsim.core.events.handler`
- EventsManager at
  - `org.matsim.core.api.experimental.events.EventsManager`
  - `org.matsim.core.events.EventsManagerImpl`
- Access to EventsManager via  
**public final** `EventsManager` `getEvents()` in  
`org.matsim.core.controller.Controller`

# Handling simulation events

## Reading events

```
public static void main(String [] args){
...
    //path to events file
    String inputFile = "output/ITERS/it.10/10.events.xml.gz";
    //create an EventsManager object
    EventsManager events = new EventsManagerImpl();
    //create the reader and read the file
    MatsimEventsReader reader =
        new MatsimEventsReader(events);
    reader.readFile(inputFile);
    System.out.println("Events file read!");
...
}
```

# Handling simulation events

## Implementing an EventHandler

```
//create a class implementing the interface(s)
public MyEventHandler implements LinkEnterEventHandler ,
LinkLeaveEventHandler {
    //implement the required methods...
    public void handleEvent(LinkEnterEvent event) {
        ...do something...
    }

    public void handleEvent(LinkLeaveEvent event) {
        ...do something...
    }

    public void reset(int iteration) {
        ...reset something...
    }
}
```

# Handling simulation events

## Register the EventHandler

- Create an instance: `MyHandler handler = new MyHandler();`
- Register at EventsManager instance: `events.addHandler(handler);`

# Handling simulation events

## Putting it all together

```
public static void main(String [] args){
...
    //path to events file
    String inputFile = "output/ITERS/it.10/10.events.txt.gz";
    //create an event object
    EventsManager events = new EventsManagerImpl();
    MyHandler handler = new MyHandler();
    events.addHandler(handler);
    //create the reader and read the file
    MatsimEventsReader reader =
        new MatsimEventsReader(events);
    reader.readFile(inputFile);
    System.out.println("Events file read!");
...
}
```

# Handling simulation events

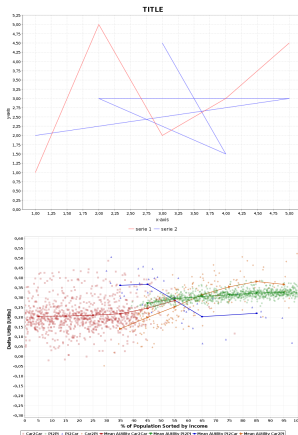
## Exercises

- Write the attributes of the LinkEnter- and LinkLeaveEvents to the console (use the command `System.out.println (String output)`)
- Write an event handler that calculates total time agents spent on the road and the average travel time per agent under the assumption that the size of the population will be given as parameter.

# Handling simulation events

## Produce charts

- MATSim uses JFreeChart library for chart generation
- JFreeChart complicated
- Useful tools: `org.matsim.core.util.charts`



# Handling simulation events

## Produce charts – Example

```
//create the chart object with Title and axis descriptions
XYLineChart chart = new XYLineChart("TITLE",
    "x-axis", "y-axis");
//add the data
chart.addSeries("serie 1",
    new double [] {1.0, 2.0, 3.0, 4.0, 5.0},
    new double [] {1.0, 5.0, 2.0, 3.0, 4.5});
chart.addSeries("serie 2",
    new double [] {1.0, 5.0, 2.0, 4.0, 3.0},
    new double [] {2.0, 3.0, 3.0, 1.5, 4.5});
//write it to a file
chart.saveAsPng(filename, 800, 600);
```

See also: `org.matsim.core.util.charts.Demo`

# Handling simulation events

## Exercise

- Collect the number of vehicles on link 6 per hour and write it to a chart.