

Volume

1

LITHP SYSTEMS BV

Advanced Information processing

Park Simulator

General Information Manual

INTERACTIVE PARK SIMULATION SYSTEM

Park Simulator

© 1999 Lithp Systems bv
Waterlandlaan 120 • 1441RW Purmerend • the Netherlands
Phone +31-299-471617 • Fax +31-299-438877

Version 0.0 - 17-Sep-99

Confidential

This document is for evaluation purposes only. Its contents is protected by copyright laws and should not be distributed or used for any other purpose without prior written approval of Lithp Systems bv.

Table of Contents

Status	2	2. GetDragInfo DockID,X,Y	14
Analysis Goals	3	3. GetDockPos DockID,X1,Y1,X2,Y2	14
Proposed modifications	3	4. DockObj Obj	15
Questions to be answered	4	5. UnDockObj Obj	15
To Do	4	Agents	16
Components	5	Walking Speed	16
The Environment	6	Agent Size	17
Attractions	6	1. EnterPos Obj,Pos,AgentID	17
Attraction Servers	7	2. MovePos Route, AgentID,Incr,Pos	17
Agents	7	3. ClearPos Route	17
Routes	7	4. GetConnections Obj	18
Boards	7	5. CreateAgent	18
The Environment	8	6. SplitAgent	18
Tick	8	7. MergeAgent	18
Timer	8	8. SetGoals Agent	18
Monitor	8	9. SelectGoal	18
Agents Monitor	9	Collected Data:	19
Attractions Monitor	9	Agent Decision Making	20
Agents	9	Boards	22
Agent Generator	9	Collected Data:	22
Collected Data:	9	Setting the Parameters	25
Planned action codes	9	Reporting the Results	25
Attractions	10	Viewing Progress	26
Collected Data	10		
Infill Attractions	11		
Attraction Servers	12		
Routes	13		
Collected Data:	13		
Component docking	14		
1. GetDockInfo DockID,X1,Y1,X2,Y2	14		

Introduction

What is the Park Simulator

The Park simulator is a system that allows the interactive definition of a number of elements that together form a Theme Park simulation system. The main purpose of such a simulation is to study the effects of altering attractions configuration and the introduction of novel ways of managing queues in the park.

The environment is based in the Acquaint ® toolkit which contains facilities for intelligent agent creation, the use of neural networks, genetic algorithms and an inference engine. Although several generalized simulation systems exist, the approach described here realizes a much more dedicated environment that also forms the basis for a real-time control system, once the simulation stage has been completed. It allows for a discrete simulation or a hybrid of model-based and discrete simulation.

With these facilities a complete environment is provided that allows the definition of all relevant elements that constitute the influencing factors in the entire process of running a theme park.

Setting up a simulation for a certain park is usually started with a map of the park. After that all attractions are put in the right place. Attractions are then connected through routes that represent the roads in the park. At certain places smaller attractions may be placed along with information providers, called boards. The visitors are represented as groups, called agents. Based on the defined characteristics of a group, the agents make certain decisions that determine the behavior of the members of the group. Things like the wait time or the fact that an attraction is closed may influence this behavior. Also the weather may play a role in the modification of behavior. The various factors that may influence this behavior are described further in this manual.

The system is implemented as a number of components that are used as controls in a Visual Basic environment. Once the entire Park environment has been defined, the resulting system is compiled and results in an efficient simulation module. The requirements for a system to run the simulation on is a 400 mHz system with 256 Mb memory and an XGA screen.

Status

The current version of the system is continually under development. Several functions in this document are for planning purposes only. When a described feature is not commented with a note that it is not supported yet, this is planned to become part of the first version.

Many new features are continually added to the system. This document reflects the current status and thinking about short-term versions.

The current version is intended to be a working version only that is mostly used by the developers and is not intended to be a production version. When the system proves successful a production version will be developed later on.

Analysis Goals

Running the simulation has the following goals:

1. To find out what the effect of proposed measures could be.
2. To gain insight in the range of influence that certain changes in park management could be.
3. To study the effects of new attractions or modifications to existing attractions.

Proposed modifications

To influence the wait times a number of changes are suggested. The simulations need to indicate the effect of these changes and might also suggest other suggestions.

1. Publishing wait times at strategic places in the park. This may stimulate people to first select the attractions with the least wait time.
2. Allow booking for single attractions, effectively distributing the waiting queue throughout the park.
3. Allow a full-day plan ahead for visitors, making sure that an optimal distribution of visitors is achieved.
4. Allow single party seating to quickly fill unused places.

All of these measures need to be studied in detail to gain insight in the quantity of improvement this may realize. Also the simulations might indicate other factors that may be important that are currently not considered or are unknown.

Questions to be answered

The following points need investigation before we can start with the simulation:

1. Relatie weer/ bezoekers
2. Relatie weer/ tijd
3. Relatie datum/ bezoekers
4. Bezoekers samenstelling
5. Relatie bezoekers/ bezetting attracties
6. Bezoek, wachtrij en wachttijd informatie per 15 min
7. Storingsfrequentie, spreiding, reden
8. Ritduur + var
9. Loadtime + var
10. Aantal servers per attractie
11. Interesse profielen
12. Plattegrond in bitmap formaat
13. Bewegwijzerings-info, locatie, aanduiding
14. Straatnamen, lengte, looptijd, capaciteit wegen
15. Splitsen familie informatie
16. Aantal lege plekken in karretjes door verlies
17. Huidige bekabeling
18. Verzamelen teller informatie en verzamelfrequentie
19. Max lengte wachtrij per attractie
20. Criteria inzet servers binnen attracties
21. Restricties per attractie
22. Aantal rolstoelen en andere vertragende factoren en distributie over attracties.
23. Informatie over loopsnelheden per leeftijd
24. Informatie over eet-, drink-, snack- en toiletgedrag.

To Do

1. Beschrijven van parameters
2. Bij alle objecten de huidige properties opnemen en beschrijven
3. Verificatie loopsnelheden
4. Verificatie leeftijd/ lengte tabel

The Simulation Environment

Description of the backgrounds of the Park Simulator

The simulator consists of a number of components that represent real-world entities. These are the following:

Component	Description	Entity	Description
The Park		Environment	Consists of general parameters like Date, Weather etc.
Attractions		Attractions Infill	All attractions including Parking, Entrances, Restaurants, Shows and Rides.
Roads		Route	
Visitors		Agents	Are groups and subgroups that are used as a unit of measurement.
Information		Boards	Are all information boards that can be used to influence people's behavior

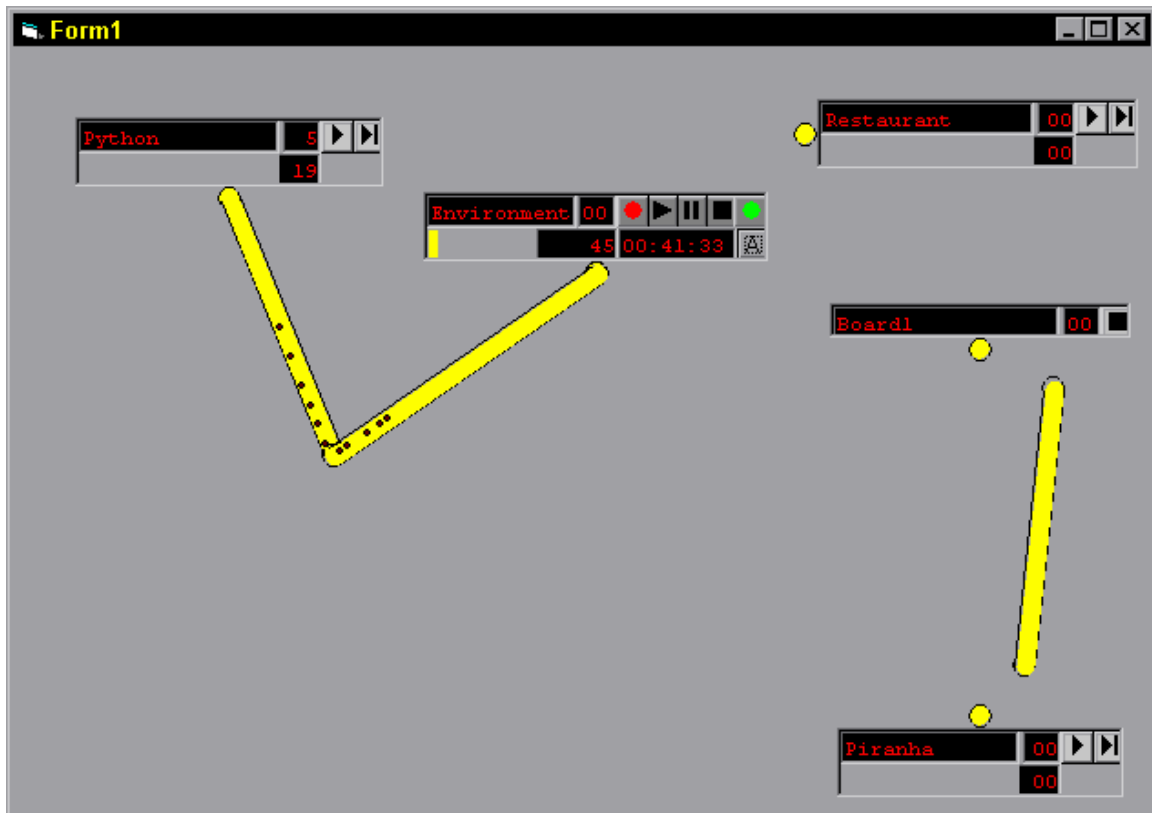
Components

The various components are first created and provided with detailed information. These details are used to define the behavior of the various components and are depending on the type of component. Details are described elsewhere in this manual.

Usually the first step is to create the various attractions. Then the routes are created to connect the components. Connecting components is done by dragging them on the screen and dropping them on another object. All components have one or two Docking connectors. When two docking connectors overlap, the components will connect.

A connection means that agents may enter another component when walking in there. The decision to walk into another component is based on the plan of an agent, described in an agent's goals.

Small attractions like infills or toilets are only entered based on current interest. Routes are only entered if the agent wants to reach another location. Attractions are only entered, when it is an agent's goal or if it raises an agent's interest. When one or more members of an agent may not enter an attraction, because of restrictions, the agent will be split and all other members will form a new so-called split group that may travel independently of the remainder of the group.



The Environment

The environment component is used to define general information about the Park and the environment. Here we define the Tick frequency and the weather definition. It is also an information center that show the total number of visitors and other statistics.

Attractions

Attractions are the definitions of all units that may be used as an attraction. These include shows and restaurants. A separate type of attraction is defined for small and infill attractions that do not have queues. Examples of these are toilets and infill attractions. In general only attractions that may cause a visitor to stop and wait for a small amount of time are represented as these small attractions.

Attraction Servers

Within an attraction a number of servers may be present. A roller coaster may have a number of trains that run simultaneously. At the entrance, many cash registers may be present. Depending on the way the attraction is organized, different loading patterns may exist. Within an attraction a mini-simulator may represent the way these systems operate.

Agents

Agents are representations of groups of people. In general they are families or groups like schools. A group is treated as a whole, but depending on the members of a group, they may become split and merged again, depending on whether members of a group may enter certain attractions.

Routes

Visitors may only walk on roads in the model that are used to connect all attractions. They represent the only way that agents may move through the park and may enter attractions. Routes may connect to all other components, including other routes.

Boards

Boards are information providers that publish information like directions and wait times. They are used by agents to make decisions about places to go to. Decisions like these are usually based on the properties of the agents and the nature of the information provided.

Technical Component Reference

This section defines the details of the various components of a simulation system.

The Environment

The environment component is the nerve center of the simulation. It contains control information that sets the process in motion and determines the conditions of the simulation. It contains a number of variables that influence the behavior of the simulation.

Tick

Indicates the duration in minutes of every simulation tick. During a tick all objects update their status. This is done by the Monitor function.

Timer

The timer sets the amount of CPU time every tick may take. This determines the speed of the simulation and the maximum number of agents that can be processed during a tick.

When the tick monitor starts and stops it calculates the total time taken to complete all updates. The amount of time is an indication of the processing load of the simulation and is used to determine the way the simulation is run.

If the load becomes too high, either the update is done randomly, but then all agents need to check how many steps have been skipped before they can update. Another mode is to create larger agents that contain more people with similar characteristics. Experience with the speed of simulation is first needed to be able to determine its nominal capacity in number of agents simulated.

Monitor

The Tick Monitor receives control after each tick and starts updating all components of the simulator. First all agents are updated, then the attractions are updated.

Agents Monitor

Updates the position of all agents. Agents communicate their position to Routes and Attractions. Based on the returned information the Agent then needs to decide on new routes, on entering an attraction or on setting a new goal. When updating its position it communicates this to the route or the attraction. The route or attraction then takes care of actually drawing the agent on the screen.

Attractions Monitor

Based on the current status of the attraction, its new status is determined and people are entering from the queue. Agents that completed are placed in the output route and the statistics are updated. The queue length and wait times are calculated and communicated to all boards. Statistics gathered are Q-size, load %, ride counts, and wait time.

Agents

Collects info about the number of agents that need to be created and their average profiles along with arrival times at the gate. It also shows the number of splits, average wait time and average ride count. All averages are complemented with standard deviation information.

Agent Generator

In the simplest implementation a single agent is generated each time the environment is activated. The composition of the agent may be done at random, but this has to adhere to a certain distribution, that represents the actual situation.

A more advanced method is to define a number of profiles that describe the range of properties of groups of agents that depend on the time of year and maybe the weather. These profiles are defined as a number of simple files and during a simulation such a profile is selected.

The Agent Generator then creates groups of Agents, based on the distribution pattern of the profile and presents the Agents to the Park entrance.

Collected Data:

Date:

Opening:

Weather: Warm, Hot, Cloudy, Rain, Cold

Period: Vacation, Weekend, Weekday, Midweek, Holiday

Display: Show/ Hide

Planned action codes

Plan Goal

Choose Goal

Move to Goal

Rest/ Wander

Choose Route

Enter Wait

Enter Attraction

Quit Wait

Check Wait Time

Exit Attraction
 Set Environment
 Log actions
 Snapshot of simulation
 Report data
 Set Tick size
 Start:

Set Date
 Set Environment
 Automatic
 Manual:

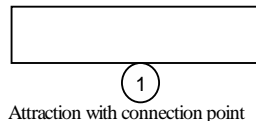
Set Agent Profiles
 Create Agents

Run
 Pause
 Stop

Attractions



Attractions are defined as a component that contains a number of indicators. Like all other components an attraction has a docking connector that is used to connect it to routes, through which the agents may enter and exit an attraction.



Attraction with connection point

Attractions have several indicators that specify the queue length, current waiting time etc. Its status indicates what is currently happening inside the attraction.

Collected Data

Name:
 Location:
 RideTime: Short , Long, Avg, Dev
 LoadTime: Avg, Dev
 LostSpace: Avg, Dev
 TotalCap: Avg, Dev, Time Schedule
 Popularity: Per group profile / Target audience
 Restrictions:
 DownTime: Avg, Dev, Freq, Occurrences, Reason
 Ride Type: Dark Ride
 Ride
 Walk-Thru
 Infill
 Shop

Restaurant
SnackBar
Show – Scheduled/ Unscheduled
Entrance
Ticket Booth
Lost/ Found
Services
Ext. Services
Parking
Booking
Toilet

Display:Show/ Hide
Picture: Bitmap with repress of attraction
Info: Q-Length
Expected Q-Length
Wait Time
Expected Wait Time

Infill Attractions
Name:
Location:
RideTime:
Target:
Picture:
Display:Show/ Hide

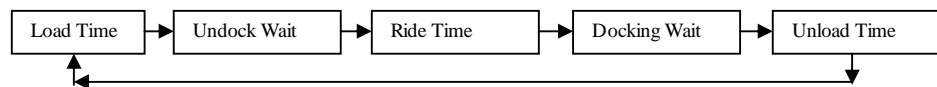
Attraction Servers

A number of different attraction configurations may exist, depending on the way the attraction uses its servers. A server is either a piece of equipment or a person that serves the agents that present themselves at an attraction. Different server types exist:

1. **Statistical Servers.** Here no attempt is done to represent the handling mechanism of the attraction, but a simple average ride time is used, with a random deviation, but based on ride capacity.
2. **Serial Servers.** This is when a certain capacity is handled as a batch and only one batch may be handled at a time, like a train with a certain capacity.
3. **Parallel Servers.** This is when a number of groups may be handled by multiple batches simultaneously, for instance a roller coaster with a number of trains.
4. **PaterNoster Servers.** This is when a loading facility is present that has a fixed speed and capacity, for instance a rotation loading dock for boats.

Depending on the type of server, an internal model for an attraction may be built that represents that handling characteristics of the system. In the first version of this system, only the statistical server is implemented. Later on the more sophisticated servers will be implemented as well.

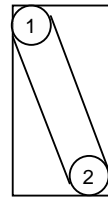
The internal model of an attraction consists of the following stages, that may be present or absent.



When an attraction has a number of stages, like a pre-show of a number of pre-shows, each stage may have the same basic model. Every server in an attraction may be coupled in a way, similar to the way attractions are coupled to the park routes. In the current version no Attraction Servers have been implemented.

Routes

Routes define the connections between components on which the agents may move themselves between locations in the park. A route is a road segment and is defined as a rectangle that connects two opposite corners. The following configurations are possible:



Left-Right



Right-Left

In each configuration the orientation determines how the route flows in the rectangle. The circles are the docking connections and are always numbered 1 and 2 from left to right. The direction of flow is determined by the agent, depending on where the route is entered. The point of entry is always the from point.

The direction of travel is defaulted to 1 to 2. The actual direction in which an agent travels depends on the point of entry. When entering at point 2, the travel direction is toward 1.

Agents determine their direction on 3 sources of information:

1. Map. This only occurs if the agent has obtained a map.
2. Signs. Each sign is checked with the agent's goals. If no goals have been set the agent is wandering and decides when an attraction is reached.
3. Boards. These are indicators of the wait time of attractions. Based on this an agent may set a new goal. He then has to follow the signs. Boards contain signs for all indicated attractions.

Collected Data:

Name:

Location:

ConnectionList:

Capacity: Avg, Dev, Current

Distance in M:

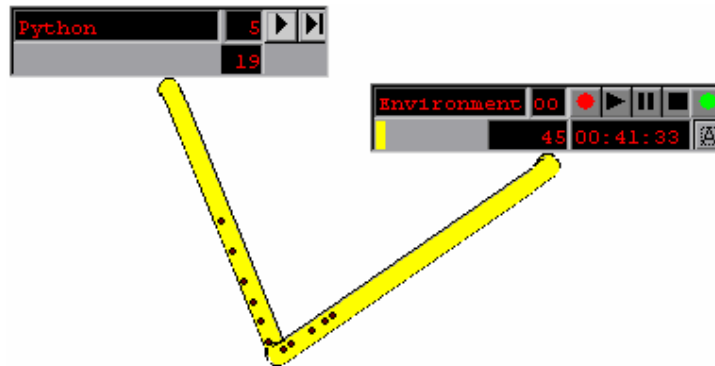
Type: Scenic, Main, Shady/ Sunny/ Covered

Picture:

Display: Show/ Hide

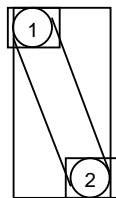
Component docking

Routes are needed to connect other components. Docking is done by dragging a component with its docking connector onto the docking connector of another component. When connected, both docking connectors are snapped and a connection is stored.



To be able to dock, a number of functions are defined that provide the information to support the docking.

1. GetDockInfo DockID,X1,Y1,X2,Y2

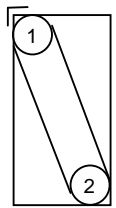


GetDockInfo provides information about the docking point coordinates of a component. Depending on the type it may have 1 or 2 docking connectors.

DockID is 1 or 2, indicating which docking point's info is requested.

X1,Y1,X2,Y2 are the coordinates of the Upper and Lower corners of the connection point itself. These coordinates are used to determine if connection points overlap.

2. GetDragInfo DockID,X,Y



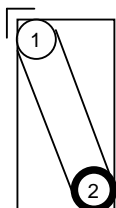
GetDragInfo gets information about the dragging points of an object. This is used to determine if the dragged object is over a docking connector. For this the corner coordinates of the component are needed.

DockID is 1 or 2, indicating which corner is needed. Depending on the orientation of the component, either the top or corner coordinates are returned.

X,Y returns the coordinates of the indicated dragging corner.

3. GetDockPos DockID,X1,Y1,X2,Y2

GetDockPos returns the left upper coordinates for a component, based on the indicated docking point. For instance if point 2 is the docking point, then the top left coordinates are calculated such that the docking point is exactly over the indicated points.



DockID is 1 or 2, indicating which is the docking point.

X1,Y1 is the top left corner of the docking point where the component needs to be docked.

X2,Y2 returns the top left corner coordinates of the entire block to align it over the indicated docking point.

4. DockObj Obj

Performs the actual docking operation of an object. The current object is docked onto the indicated object.

Obj is the object receiving the connection.

Both objects update their connection list.

5. UnDockObj Obj

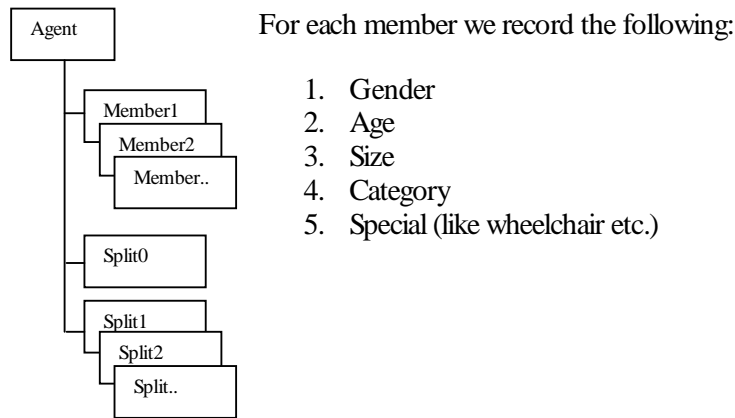
UnDockObj is the reverse of the DockObj operation. It removes all connections from both object's connection lists.

The underneath picture shows the Environment docked into a route and this one connected to an attraction though another route.

Agents

Agents are the active components in the simulation. They make decisions about which attractions to enter and decide on their route through the park. Each Agent has a position indicator which shows where is currently is (an attraction or a route) and the position and direction it is taking there. It also has an ID and will get assigned a new ID every time it enters a component. This ID is the representation of the agent on the screen. Drawing agents is not the responsibility of the agent but of the routes or attractions.

Agents consist of a number of members.



For each split we set the Goals and current location.

Split0 always contains all common goals that all members of the Agent have.

All members of a split share the same goals. We keep a count of the number of people in a split but not who they are. When creating a split the system must make sure that the goals set may be executed.

When arriving at an attraction and the agent decides to enter, a split is created for all members that do not enter. The two splits then continue independently as subagents. If both splits meet later on at the same location, they will be joined again.

Walking Speed

To gain a good insight in the way agents move through the park, every agent has a speed indicator. We assume that the agent's speed is determined by the slowest individual in the party and that the party will try to stay together unless a split is activated.

For this speed the following assumptions are made:

Age	m/Sec	Km/u	rem
<3			Below this age they do not walk
4-6	1	3.5	
6-15	1.5	5.5	
>15	1.9	7.0	Adults walk at faster pace
>50	1.5	5.5	At this age walking is slower

Agent Size

Depending on the agent's gender and age an assumption is made about the average height. This information is needed to determine if restrictions apply for an attraction.

Age	Boy cm	Girl cm	Rem
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			

The following Agent functions exist:

1. EnterPos Obj,Pos,AgentID

This actually enters a new object. It checks if the object is one of the goals of the agent and if so, it removes the goal from the goal list.

In case of a route, it sets the direction, creates a new AgentID and sets the initial position. This AgentID must be used in all subsequent commands

2. MovePos Route, AgentID,Incr,Pos

Changes the current position of the agent in the route. It returns the new position. If the position is -1 the end of the route has been reached. The agent must then decide if a connecting object is entered or if it reverses direction. If a new position is given after this, the direction is reversed automatically.

3. ClearPos Route

This removes the agent instance from a route, so it is no longer displayed on the given route.

4. GetConnections Obj

Gets a list of all connections that the given object has. An agent needs to decide what to do with these connections:

1. Stay on current position
2. Enter the connection (CheckGoal)
3. Change direction

5. CreateAgent

Creates a new Agent. See the description for all details that are defined for a new agent.

6. SplitAgent

When an Agent enters an attraction that has age/ size restrictions or that has members that do not have the attraction as their goal, the Agent is split. Every split Agent may travel on its own and is a temporary copy of the main agent. A color code is used to indicate the Agent type on the screen:

1. Red is a family
2. Green is a group
3. Blue is a split agent
4. White is a single person, may be a split as well

7. MergeAgent

When several splits that belong to the same agent meet at exactly the same spot, they will automatically be merged again. The splits are removed and replaced by a new AgentID.

8. SetGoals Agent

Set a number of goals for an agent or a member. It needs to indicate the object and possibly a time slot when it wants to enter. Note that an attraction may not be entered if outside the time slot, unless the time slot is then removed.

Also note that when an attraction is located, a check is made if this is on the goal list and if it may be entered. Planning of the route to an attraction is mainly done based on information on boards. This way also the sign system of a park may be tested for its validity. Agents may also plan a route based on a map. In this case the agent must have retrieved a map.

9. SelectGoal

When an agent has no current goal, it needs to select one. This may be one of the following:

1. Select a random goal
2. Activate a decision procedure. Here many things might happen.
3. Select a goal based on time
4. Wander, this means that no goal is needed and decisions are made based on information presented on route.

When a decision needs to be made to enter an attraction, the system checks if it may be entered because restrictions may apply. Based on that either the agent selects another goal, enters or splits. This may also depend on the status of the attraction. The agent also has a status code, that indicates what the agent is doing inside the attraction. This is like the position in a route.

1. Waiting
2. Exit (prematurely)
3. PreShow
4. Loading
5. Riding
6. Ready and exiting

Collected Data:

Name:

Location:

Type: Group, Family, Individual

Profile: Age Group, Avg Age Group

Members:

Arrival:

Departure:

PlanCode:

NrOfRides:

AvgWait:

Status: Waiting, Walking to Goal, In Attraction, Wandering

DistToGoal:

Picture:

Walking speed:

Display: Show/ Hide/ Filter

Agent Decision Making

Every agent has a behavior pattern, called a genetic code. These codes are assigned when agents are created. This process is based on a profile definition which in itself is driven by the time of year, the type of day, weather etc.

The profiler, which is part of the environment will then start generating agents, who appear first at the park gates. Based on the form of transportation they will enter through the parking lot or another gate.

The genetic code consists of a number of genes and every gene drives a number of rules that determine the agent's behavior.

During the simulation the agent will be placed in a position where decisions need to be made continually. Basically there are 2 sources of decision making:

1. **Time.** Based on the time and the agent's plans goals will be set to reach a certain attraction, to exit or to visit a restaurant or toilet.
2. **Routes.** When a route ends, it connects to another route or an attraction of some type. The attraction contains information on which a decision can be made.

These decisions basically are to enter the attraction, if it meets the goals and conditions of the agent, or to select another goal and associated route. Wait time information at an attraction determines such a decision. A board or a sign provides information about where to go to reach a certain goal. Wait time information at an attraction or at a board may set new goals or change the priority of the goals already set.

Agents make decisions based on Rules, that are defined in a separate editor. All Rules define the Genes and the context within which they operate. For instance a rule may be defined:

Context: A route needs to be selected

If the Agent has a goal and
the current route mentions the goal
Then select the route

If the Agent has a goal and
the current route does not mention the goal and
the Agent has a Map and
the current route matches with the agent's goals
Then Select the route

If the Agent has no goal and
the Agent has random-behavior
Then Select a route at random

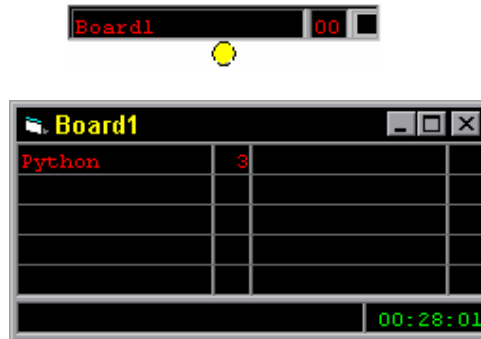
If the Agent has no goal and
the Agent has no random-behavior
Then Select the most quiet route

A so-called inference engine that selects the most appropriate rule executes these rules. In addition to the simple rules as mentioned here, also so-called neural rules

exist, that will actually learn their behavior from showing them examples. The decision rules will be extracted from these examples automatically. A Separate training session is required for these rules to learn their behavior.

So by altering the Genes in an Agent, the behavior of Agents may be controlled. Future versions of the system may allow the generation of various Agent types, using a genetic algorithm, this finding the most optimal configuration. This is less important for Agents, but is a more appropriate feature for the Servers of an Attraction. With this same mechanism, various alternatives for servers may be studied by automatically generating the most desirable configuration.

Boards



Boards to some extent also act as a kind of attraction. There are different types of boards. A Sign is a simple direction information source, a full board contains both directions and wait time information.

When an Agent encounters a Board it inspects the board and checks it with its goals. If it is a full board, it checks the wait times and selects the one with the most attractive wait time. This decision is based on the behavior pattern of the agent. The direction information is used to determine which route to select next. Since attractions and boards are always placed at the end of a route, every agent always needs to decide which route to take. Boards and Signs form a source of decisions that influence where the agent will go next.

Agents may also get instantaneous goals, for instance hunger, thirst or toilet need may set priority goals that influence the decision to change the routing. Complex decision rules will drive the behavior of an agent.

Boards are continually updated by all attractions and provide this information to the agents and the simulation system.

When the expand button is clicked during a run, the full board for the selected board is shown with all published wait times. A board shows 10 attractions at a time. When more attractions publish their times, the board will start scrolling to show the remainder.

In addition the board shows the current time and may display a message to give general information other than the wait times. These may also downtime notifications like Droomvlucht expected to open in 1 hour.

Collected Data:

Name:

Location:

Display Data:

Type:

Full Board

Small Board

Qtime Display
Direction
Directory
Interactive
Screen
PC
Internet/ Agent

Display: Show/ Hide

Creating a Simulation

Actions to take to set up a simulation

Running a Simulation

Actions to take on a simulation

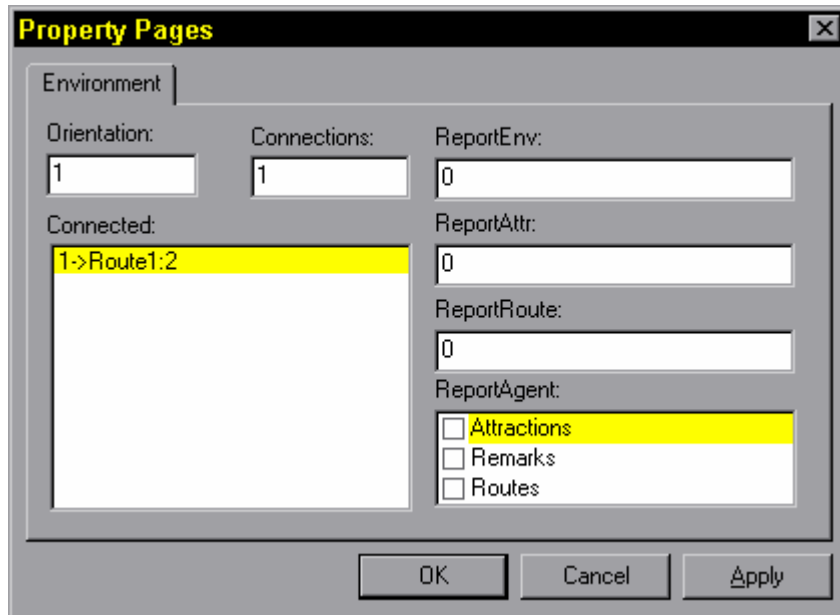
Setting the Parameters

Most of the parameters are set using property sheets. No details are known yet.

Reporting the Results

A number of reporting features are available that allow the creation of a log file, that contains details, required for later analysis. Depending on the analysis several things may be defined. These are set using the property screen of the environment.

The files are written as comma-delimited files, that may be read by Excel and used for further analysis.



The Environment, Attractions and Routes all may set the number of minutes after which statistics need to be written to the reporting file.

Agents report much more detailed information. They may collect data about attractions, routes or remarks. Remarks are made by the simulator to give feedback about things that might be important like when an agent cannot find a route to a destination and makes a guess, this is reported, since it may be an indication that there is an error in the model or that the actual situation is confusing for people as well.

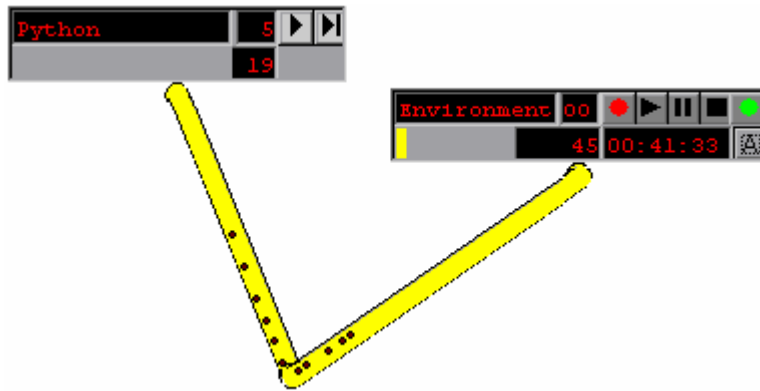
The following table lists the information, recorded for each type:

Object						
Agent	Attraction	Time-in	Time-out	Publ Wait	Act wait	
	Route	Time-in	Time-out		Act travel	Distance
	Remark					
Environment	Time	Agents	People	Cum Agents	Cum People	Avg Wait
Attraction	Time	QLen	QSize	Publ Wait	Avg Act Wait	Fill %
Route	Time	LR-Agents	LR-People	RL-Agents	RL-People	

Viewing Progress

During a simulation, crowd movement is shown in the simulation window in the form of small dots. The color of the dot indicates if this is a family, a group, a split group or a single individual. All dots are referred to as agents. The updating of moving agents takes a considerable amount of time, so attempts are made to minimize the overhead involved.

Two agents that are at the same location and the same speed are drawn as one dot. So dots do not overlap each other, except when they have different walking speeds.



During a simulation the green button may be used to freeze the screen. This will stop the agents update but continues the simulation. Pressing the button again will update the screen. Another mode of operation is to update randomly. Here not all agents are updated on the screen but only a portion of them. The number can be set as a parameter. For a normal simulation updating is not required, the update feature is only to give the user some feedback and a graphical impression of the number of people in certain locations.

Index

A

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
 Index 3, 3
Index 1, 1
Index 1, 1

B

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2

C

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1

D

Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1

E

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1

G

Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1

H

Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1

K

Index 1, 1

L

Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1

M

Index 1, 1
Index 1, 1

Index 1, 1
 Index 2, 2

N

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1

R

Index 1, 1
Index 1, 1

S

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1

T

Index 1, 1
Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2

W

Index 1, 1
Index 1, 1
Index 1, 1
 Index 2, 2
Index 1, 1
Index 1, 1
Index 1, 1

Index 1, 1

