Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

FAST CREATION OF REALISTIC AND EFFICIENT FREE PATH NETWORK WITHIN A SIMULATION MODEL OF A SHOP FLOOR AND A SUPPLY CHAIN SYSTEM

Michal Stec

Simulations- und Automations-Technologie AG,
Badenweilerstr 4, 79115 Freiburg, Germany

ABSTRACT

Good animation is invaluable to understanding and verifying the processes in a simulation model. In a shop floor and a supply chain simulation model, animation refers to animating movements of workers, forklifts or trucks entities between nodes. The large (thousands) number of possible connections in a complex model of at least 100 nodes requires a new approach to this undertaking. Instead of drawing separately connections between each and every node, a free path integrated network is needed that spans all nodes. This paper describes a technique developed for Arena (Rockwell Software) that builds automatically a realistic and integrated network among hundreds of nodes in a matter of minutes, giving the modeler practically unlimited field for further enhancement as far as the movement logic is concerned. Above that the created network "knows" the distances and shortest directions between each and every pair of nodes and navigates the free path object accordingly.

1 A NEED FOR A NEW MODELING MECHANISM

A typical Discrete Event Simulation (DES) model of a shop floor system consists of a number of nodes such as stations, manufacturing cells and buffers, where some production processes are modeled. Similarly, a model of a distribution system consists of a number of nodes representing suppliers, producers, wholesalers and retailers. All these nodes in the model have to be accessed by some type of transporters; that is, workers, forklifts, trucks or you name them, whose main task could be as simple as to move parts (load) between the nodes or just to perform some task at a node.

Unless there is a well regulated and organized transportation method like conveyors, these transporters move freely in an area utilizing allowed ways such as corridors, halls, roads etc. In such a case each of the transporters can be viewed as a separate unit, having one simple moving rule meaning to get to the destination point as quickly as possible. Simulating and animating just these movements is quite a challenge in a large simulation model, because of the large number of possible connections in the model. The decision which route to take, is left to the transporter. Thus, all possible movements have to be considered in designing the model, which boils down to creating a Free Path Network.

A great feature that is frequently used to enhance simulation models is animation. Animation, besides being eye-catching, also allows the user to understand what is going on in the model. The availability of animation has resulted in a greater understanding and use of simulation by engineering managers (see Law 1999). Thus, the movements between the nodes should be rather well animated.

The entity movement in a simulation model is based on the functionality of the used simulation package, i.e. "route" or "free path transporter" in Arena. These allow nodes to connect with each other schematically and separately. Animating such separate connections highlights the flow of single entities in the model from one node to another. However, it does not necessarily allow simulation of the dynamic material flow, i.e. where single entities occasionally create streams of entities.

Building a node to node network type manually is mundane work. The number of direct connections between *n* nodes increases exponentially (exactly n*(n-1)/2) with the increase of the nodes' number. For models with, say, 100 nodes, it results already with 4950 connections. Some modern simulation tools with object oriented structure and a programmatic interface can help to automate the creation of such a network. However, using even the most modern tools still does not change the fact that each and every connection in such a model still has to be shown separately. Figure 1 shows such connections from one node to 100 nodes. All the other nodes should be likewise connected with the rest as well in order to allow the "All with All" network, what creates 4950 lines.

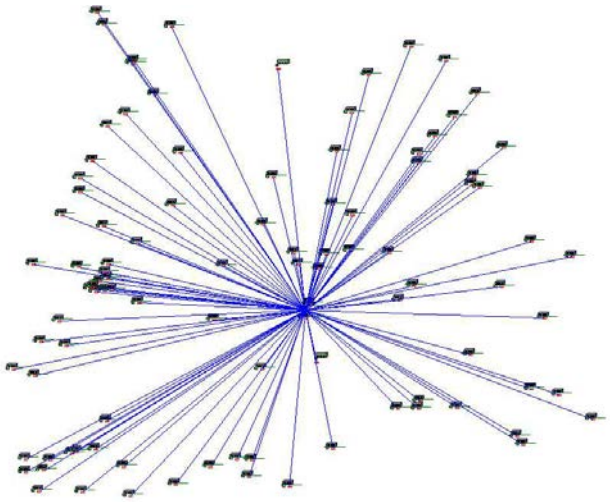

Figure 1. Star type - Straight Connections from One Node to 100.

An alternative solution, instead of linking the nodes with one another, would rather take into consideration the positions of the movement channels such as ways or corridors, where the main movement between areas (groups of nodes) takes place, after which it would connect the nodes to the channels. There could be a hall in a building, or a main road between two cities. All objects moving between nodes would take that road and not create their own links through the walls, fields etc. Figure 2 shows the same nodes as in figure 1, but the network utilizes the movement channels in the simulated area.

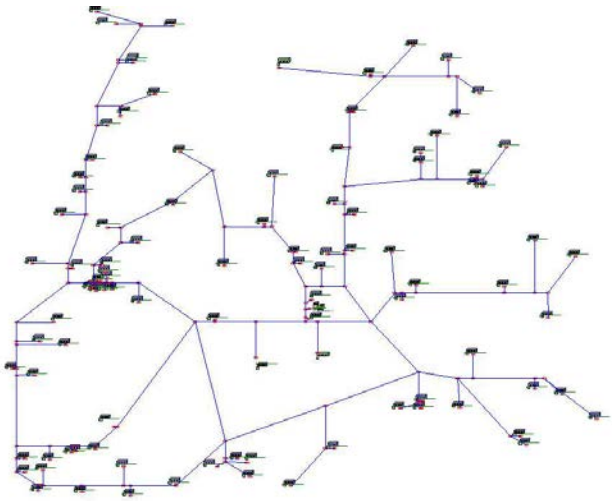

Figure 2. Network Type Utilizes the Movement Channels

Additionally, it would be great just to position the work-station nodes in the model, draw the main paths (corridors, roads, etc.) and finally click a button what would automatically connect all the node modules with the main paths, thus generating the network. Recreating the network after altering the quantity and/or the positions of the nodes would be equally easy. How much of the modeling time could be saved!

Such a type of efficient network would:

- allow building up streams of entities and additionally, what has not been yet mentioned
- allow for freely assignable and of required complexity transporters (having complex schedules and responding to failures) and most importantly,
- not place a big burden on the modeler and on the CPU

Our experience, however, is that such concepts usually stay on the "wish list".

In this paper the author presents a solution for automatic creation of such a network in a simulation model, based on Arena Professional Edition (Rockwell Software). The next paragraph discusses the possible animated movement types in Arena. This will be followed by a description of an approach to this undertaking and later by the design of the solution. Finally, the author presents examples where this solution has been successfully applied.

2 FOUNDATIONS OF A FREE PATH NETWORK

The author used Arena simulation software in order to develop this network solution. Arena provides a few possible alternative building blocks that animate movement. According to Arena 10 Help (2005):

- **Routes** are the simplest form of entity movement between stations. The travel time associated with a route is a simple delay time specified in the model. A route does not consider collision avoidance or synchronized movement.
- **Distances** define the path of a free-path transporter as it travels between stations. The data characteristics of the transporter and its path—speed, distance, and connecting stations—are defined in the model. Free-path transporters do not consider collision avoidance requirements.
- **Networks** define the path of a guided transporter as it travels between intersections. The data characteristics of the transporter—speed, distance, alternate paths, and connecting intersections—are defined in the model. Guided transporters have logic that considers collision avoidance, uni- or bi-directional paths, acceleration and deceleration, and turning velocities.

Valentin and Verbraeck (2002) mentioned that the sets of elements offered by commercial simulation tools solve some problems, but not all. Commercial tools are good as long as the simulation experiments are nothing more than adjusting the values of some parameters. They have been

developed for a small number of questions and the design of these elements does not contain the rich set of solutions needed by the successive simulation studies.

From the three movement methods described above (Route, Distance, Network) only the "Route" method seems to satisfy the requirement for free and open future development of the model. Basically, it is the simplest method, and does not carry with itself unnecessary transporter requirements. The other methods in Arena require significant bending as far as their main purpose or the required functionality is concerned. "Distances" move the free path transporter only (not just any entity), also this transporter is not even as flexible as the standard resource element offered within Arena (i.e. lack of failure or schedule feature). "Networks" move guided transporters in a very detailed way (CPU intensive), which is not in the scope of a supply chain or generally seen in a shop floor system.

The author used the route element together with added control logic to design and develop the Free Path Movement Network. The goal was to use a more systematic approach to map the complex network of paths that are commonly found in multiple simulation models.

3 DESIGN ASSUMPTIONS

Generating the Free Path Network first involves the important step of reading positions and dimensions of the channels automatically. All the spatial data is actually already in the design of the simulated area. The dimensions of buildings, walkways and areas are in the design file (i.e. AutoCAD). The design can be inserted into the model window and used as a background drawing. On this the Arena lines, from the drawing toolbar, can be simply but manually drawn. The lines reflect the directions and also the proportional lengths of the channels. The distance unit has to be adjusted afterwards. The gain is the following: coordinates of the Arena lines are freely accessible through the Object Arena Model.

All the pathways, door to door, along main and smaller corridors are under consideration. The "Arena lines" mark all the passages as it is shown with the orange lines in figure 3. When these are ready, the rest can be automated.

Let us assume for now that walking time depends only on the distance and the transporter speed, e.g. the office personnel walk with the same speed along a corridor as well as within a small room from one desk to another. This assumption on one hand does not disturb the general correctness of the constructed solution, because the majority of distances are covered with the average speed for a given transporter. This might however not be true for the distribution networks, where the road class significantly affects the driving time. This assumption will be discussed once again in the conclusion of this paper where some alternatives are given.

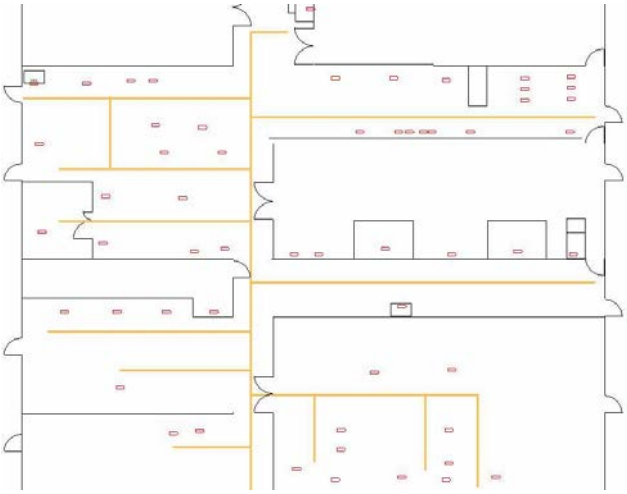

Figure 3. A Layout with Orange Lines

4 CONSTRUCTION METHODOLOGY

The Free Path Network consists of the following components: drawing lines, database tables ("Line", "Node" and "Junction"), VBA macros, Junction templates, and Distance and Direction matrices. Table 1 presents the components together with the implementation method.

Table 1. Solution Components

Solution component	**Where**	**How**
Drawing lines	Arena	Orange draw line
Transformation database	Access	Tables: "Line", "Node", "Junction"
Program	Access	VBA macro
Junction logic	Arena	Template
Direction and distance data	Arena	Variable arrays for Direction and Distance matrices
Free Path Network	Arena	Routes and stations

The author has used Microsoft Access, as it is a part of popular Microsoft Office package and it avails of VBA for writing user macros. The author has programmed such a macro to do the whole work of generating the Free Path Network. Other options would include using VB or C++ with proprietary databases. The remaining of this section describes the step by step actions that conclude with the ready to use Free Path Network.

As described already, the simulation consultant starts creating the network manually by drawing the lines along the corridors, paths and walkways in the model. These lines should be of particular characteristics (e.g. orange), so the macro can distinguish them automatically. This is all

what he/she is asked to deliver, before handing the remaining work over to the macro. On pressing a button, the Access macro adds the Free Path Network along the lines and connects it with all the nodes in the model.

The macro starts from reading the position coordinates of the drawing lines into the "Line" table and also positions the nodes into the "Node" table. Next, with a few rules in hand, the connections (new lines) between the nodes and the network are automatically decided upon and inserted into the same "Line" table.

The author chose the following rules to decide on connection to each and every node:

- If the nearest line in the network to the node is horizontal or vertical in the model, then the macro adds a new perpendicular line to the network line reaching the node, which means that the network line is divided into two parts in the very point where the perpendicular connection crosses the network line.
- Otherwise, the nearest network line end is chosen as the connection point from the network to the node.

After resolving how the nodes connect with the network, the macro populates the "Junction" table with coordinates and parameters of all the network junctions. These are the network lines ends defined in the "Line" table, except for the ones connecting with the nodes. The junctions get an Id from 1 to n (n junctions), which a moving transporter will later use to position itself in the direction/distance matrices.

Each and every node has only a single junction, through which the connection with the network exists. Thus there is a 1 to 1 relation between a node and a junction (or rather n to 1 as many nodes can have the same junction, but not vice versa). One can build a function returning the junction number for the given node in the model. This function can be used during the model run to check if the current junction is already the link to the required node or not. Thus, the number of junctions is kept to the minimum.

Next the macro updates the Direction and Distance matrices (n x n junctions). The data in these matrices will navigate the moving transporter along the network junctions (providing the direction and distance). The macro finds the distances between the adjacent junctions directly in the "Line" table. The other positions in the matrices are filled based on calculations of the shortest distances and preferred directions within the network through the use of Dijkstra's algorithm (see Dijkstra (2006)).

This is all what the macro needs in order to build the complete network in the Arena model. It places the junction stations of the network according to the "Junction" table and draws the routes according to the "Line" table. Additionally, each junction station receives a logic module that is an instance of a Junction template. The direction and distance matrices are copied to the Variable arrays.

The logic in the Junction template has the following flow. An arriving entity has an attribute with the destination node name. After arriving at a junction module, the junction id corresponding to the node stored in the attribute is found (with the function mentioned earlier) and is compared against the current junction Id. If the names match, then the entity is sent to the required destination node. Otherwise, the next junction to visit is found in the Direction matrix for the current junction Id and the junction of the destination node. With the new direction in hand, the distance to that next junction is read from the Distance matrix. The travel time needed to reach the next junction is calculated with the speed parameter.

An entity entering the network must have only 1 attribute with the destination node defined (there must exist a corresponding junction to the network for that destination node). The network accepts any entity that has this attribute set. It makes no difference here if this is a transporter-like entity or any other entity.

5 IMPLEMENTATION EXAMPLES

Having the automatic solution for the stations' connections, it is easy to apply it to a project. The author will describe two such projects. One is a shop floor system, and the second is a distribution network.

Let us see how this can be applied to build a simulation model of a shop floor system. There are over 50 machines in the building. The number of operators according to skills and the number of walked kilometers is under investigation. The number of resources (machines and operators) is limited, they have to resolve a number of problems or issues a day, and they are skilled or prioritized to serve in particular roles. The question is how the processes should be relocated in order to reduce the number of walked kilometers. Or differently, what set of skills should all the operators posses in order to operate all the machines without delays. Surely, simulating it without the solution described in this paper would be a difficult task.

The simulation consultant places simply the machines' modules in the model, draws the lines (in orange), starts the macro and after a minute sees the outcome similar to the one in figure 4. The model is ready to run. After a while, he redesigns the pathways, he/she moves machines to another location, the model is redrawn, and generated once again. After a minute, another version is ready to run (figure 5).

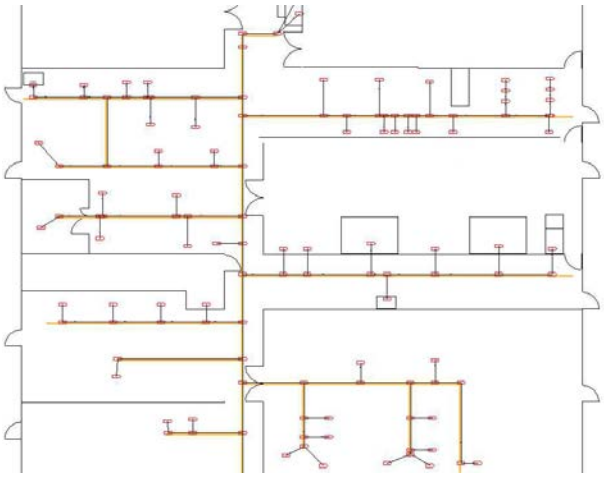

Figure 4. Free Path Network in a Shop Floor, Design 1

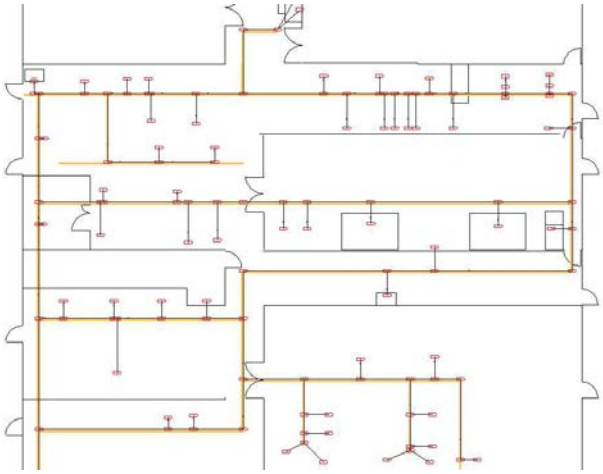

Figure 5. Free Path Network in a Shop Floor, Design 2

Surely, the operator logic has to be defined separately in the model, and the machines' logic modules have to "know" how to use these operators. An operator is modeled by binding the operator entity with a resource (thus creating a transporter). Such an approach requires defining a pool of resources, each of which comes with a predefined schedule together with a Variable array, where all the particulars about each and every operator will be stored and seen globally in the model.

Now let us examine how the macro works within a distribution network, including a section of the south west German road network. Let us have a look once again on figure 1. It presents a hub, from which the goods are sent around to retailers in the area. A star approach to the connections between the nodes is not very realistic. It is very schematic and too simplistic. Applying the Free Path Network design, however, produces a simulation that resembles real-world conditions much more closely. The simulation consultant positions the nodes of the suppliers, hubs and retailers. He/she draws the main roads on the map and eventually starts the macro. The result is shown in figure 6. Adding another node, or changing its location is no longer any major task at all. In figure 6, it is easy to see the streams of trucks driving along the busiest road in the area.

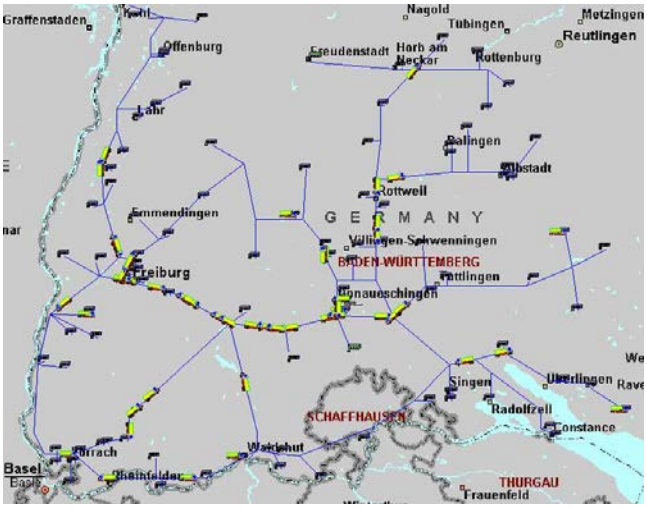

Figure 6. Free Path Network in a Supply Chain, in the Background a Map of SW Germany. There are Visible Trucks in Yellow (see the streams of trucks along the main road).

6 CONCLUSIONS AND FURTHER WORK

The Free Path Network solution really makes a difference when applied to complex simulation models. It automatically generates a network that is complex enough to resemble real-world conditions. It also makes the job of a simulation consultant easier. This macro not only speeds up drawing and defining all the network sections, but also enforces the correct principles in the model, bringing each project closer to a success. The principles are to clearly separate each node in the model (best using building blocks such as templates) and to let the network do the whole transporting activities in the model.

Further development of the automatic Free Path Network includes the following areas:

- Controlling the traffic on chosen sections of the network: reducing or closing the sections of the network according to a schedule,
- Allowing an alternative route when a planned travel section is blocked or not available or rerouting to the sections with lower traffic,
- Differentiating between the road classes in the model. A car driving on windy roads in mountains has a different traveling speed than a car moving on a straight motorway. This can be achieved by assigning parameters to each section of the net-

work (additional columns in the "Line" table for the road type parameters),

- Incorporating Geographical Information System (GIS) data in building the network. Instead of drawing the lines on a map, one could use the data saved already in GIS database, thus generating the main roads between locations represented by nodes in the model.

REFERENCES

Arena 10 Help. 2005. Rockwell Software.

Banks, J. 1998. *Handbook of Simulation*. John Wiley & Sons, Inc.

Dijkstra, 2006. Dijkstra's algorithm: web page <www.wikipedia.org>, [accessed March 10, 2006]

Law, A.; D. W. Kelton. 1999. *Simulation Modeling & Analysis. 3rd ed*. McGraw-Hill.

Valentin, E. C. and A.. Verbraeck. 2002. Guidelines for designing simulation building blocks. *In Proceedings of the 2002 Winter Simulation Conference*. 563-571. <http://www.wintersim.org/prog02.htm> [accessed March 10, 2006]

AUTHOR BIOGRAPHY

Michal Stec is a simulation consultant at Simulations- und Automations-Technologie (SAT) AG in Freiburg, Germany. He received Mgr. Eng in Information Technology (1998) from Technical University in Wroclaw, Poland and M.Sc.Eng. (1999) in Industrial Engineering from National University of Ireland, Galway, where his final thesis was on simulating supply chain in the food industry. He worked as a process consultant in Accenture and also in Microsoft, Ireland. Since joining SAT in 2003, he has successfully delivered multiple simulation projects including shop floor systems (car manufacturing), supply chains, airports, waste treatment plant and laboratory work. His email addresses are: <mistec@gmail.com>, <stec@sat-ag.com>