A Scipt Language – supported Approach to ARGESIM Benchmark C14 'Supply Chain' in Enterprise Dynamics

Philipp Gutwenger, Do Duc Hoa, Wolfgang Puffitsch, Shabnam Tauböck [*]

[1] Inst. for Analysis and Scientific Computing, Vienna University of Technology, Wiedner Haupstraße 8-10, 1040 Vienna, *Austria; shabnam.tauboeck@tuwien.ac.at

Simulator. Enterprise Dynamics is a simulation environment for discrete systems. It employs an event-oriented approach and supports 2D and 3D animations. Models consist of 'atoms' which can be configured with the 4D Script language. Atoms can inherit attributes from 'mother' atoms and can thus form a hierarchy. Communication between atoms takes place via channels.

Modelling: Figure 1 shows how a *factory* (including delays to distributors) was modeled. Figure 2 shows a *distributor*. The complete model consists of 4 factories, 4 distributors and a group of wholesalers.

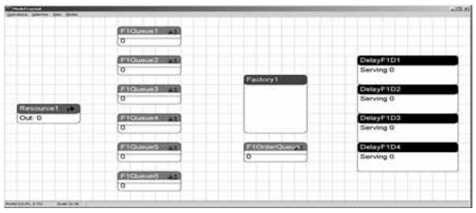

Figure 1. Factory Modelling.

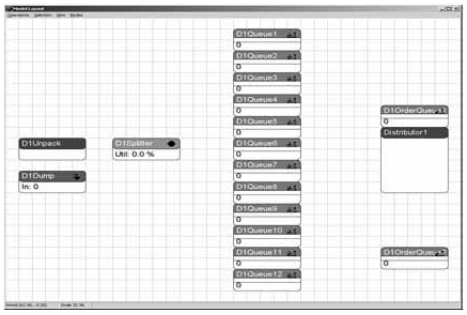

Figure 2. Distributor Modelling.

Factory. Products are spawned in a generic source and are distributed to a random product queue of the factory. An order is a special type of atom, containing information about the type and number of the ordered product and the distributor ordering it.
If such an order enters through the respective queue, the factory checks the content of the queue for that type of product and delivers the products (packed into the order) through the appropriate delay element or returns the order to the distributor if unavailable.

The following code (written in 4D script) is located in the *onEntered handler* and builds the core of the factory. Checking for available products and possibly postponing them is done here. Routing to the appropriate distributor is determined by the attribute *DistributorNumber* of the order automatically. The function *moveatom* was for sending the atoms.

```
if ((att([ProductCount],i) >
    content(atomByName(concat([F1Queue],
    string(att([ProductNumber],i))),
    Model))),
 do ( {* postpone order *}
  case (att([DistributorNumber],i),
   moveatom(i,atombyName(
     [D1OrderQueue2], Model)),
   moveatom(i,atombyName(
     [D2OrderQueue2], Model)),
   moveatom(i,atombyName(
     [D3OrderQueue2], Model)),
   moveatom(i,atombyName(
     [D4OrderQueue2], Model))
  ),
  return
 ),
 do ( {* set table for packing *}
  cell(att([ProductNumber],i)+1-6,
       att([ProductNumber],i),c) :=
    att([ProductCount],i),
  finishquant :=
    +(content(i),
      sum(nrows(c),cell(count,curcolref,c))),
  setloc(0,0,zsize(c),i)
```

Distributor. In the distributors, orders are unpacked and products are stored in the appropriate queue. Upon request from the group of wholesalers, the distributor checks its queues and delivers the product if available. Every 24 hours, a special order is sent to the distributors. This order causes the distributor to pass its orders to the factories along the implemented strategy.

The 4D-language code describing the handling of the atoms in distributor follows same principles as the code for actions in the factories. Attributes record and control unfulfilled requests; two different attributes are needed in order to implement the different control strategies properly.

Group of wholesalers. The group of wholesalers is a source which produces orders for random products and sends it to a random distributor.

Orders. All information is passed inside the orders, which are derived from the standard Container atom with additional attributes for holding the appropriate information. This has the advantage, that for the factories and the distributors, standard atoms could be used too (albeit extended to some degree).

A-Task: Simple Order Strategy. As Figure 3 shows, the stock of Distributor 1 grows steadily (backwards history timescale) with the simple order strategy. This is caused by the fact that on average, less than 2 products are ordered each day from the group of wholesalers.

Table 1 shows the number of delivered products *D*, the total costs *T* and the costs per product *R*. 100 runs with the experimentation tool in Enterprise Dynamics were made to compute the statistics.

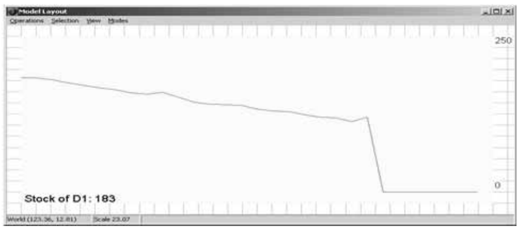
Model Layout
Operations Selection View Modes
250
0
Stock of D1: 183
World (123.36, 12.81) Scale 23.07

Figure **3.** Stock with simple order strategy (backwards time scale)**.**

	Average	Deviation	Minimum	Maximum
D	225,50	13,79	194,00	256,00
T	€ 34.759	€ 1.496	€ 30.748	€ 37.766
R	€ 154,41	€ 4,69	€ 138,72	€ 170,56

Table 1. Simple order strategy costs**.**

B-Task: On Demand Order Strategy. Figure 4 shows that the on demand strategy yields a constant stock for Distributor 1 – the expected result, if the same number of products is ordered and sold. The costs for the on demand order strategy are only slightly lower than for the simple order strategy, as seen in Table 2, which might be explained with the lower costs for stock keeping.

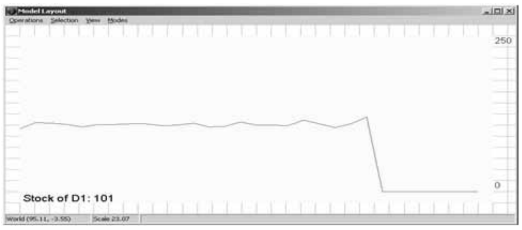
Model Layout
Operations Selection View Modes
250
0
Stock of D1: 101
World (95.11, -3.55) Scale 23.07

Figure 4. Stock with on demand order strategy, (backwards time scale)**.**

	Average	Deviation	Minimum	Maximum
D	228,24	13,36	195,00	257,00
T	€ 33.653	€ 1.501	€ 29.302	€ 36.790
R	€ 147,66	€ 5,65	€ 135,65	€ 161,60

Table 2. On demand order strategy costs.

C-Task: Minimal Supply Time Strategy. For this strategy it was necessary to extend the model, because the distributors can order from any factory with this strategy, and they have to check the stock of a factory before they send the orders.

The stock stays constant with this strategy, as with the on demand order strategy, as seen in Figure 5. Due to the lower cost for transportation, the costs with this strategy are significantly lower than with the other strategies, as Table 3 shows.

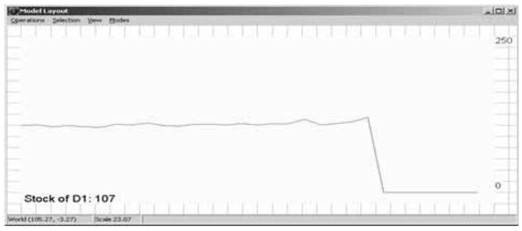
Model Layout
Operations Selection View Modes
250
0
Stock of D1: 107
World (105.27, -3.27) Scale 23.07

Figure 5. Stock with minimal supply time strategy, (backwards time scale)**.**

	Average	Deviation	Minimum	Maximum
D	229,17	12,88	201,00	262,00
T	€ 27.237	€ 1.201	€ 23.978	€ 30.362
R	€ 119,00	€ 4,36	€ 109,16	€ 132,36

Table 3. Minimal delay order strategy costs**.**

Summary. Enterprise Dynamics is an object-oriented simulation system well suited for process flow models and similar applications. Supply chain models require additionally a flow of orders, usually in the opposite direction than the classical physical entity flow. Because of its programming capabilities with 4D script language, various approaches for control of the process flow can be used. The approach used in this solution makes use of order objects which queue in the various object stations to be handled different.

Received: November 2009
Revised: July 1, 2010
Accepted: November 30, 2010