3rd International Industrial simulation Conference ISC'2005 June 9-11 Berlin Germany
Eurosis-ETI. ISBN 90-77381-18-X

GREENHOUSE AUTOMATION
A Model for Design and Evaluation

Jaap A. Ottjes, Joan C. Rijsenbrij and Auke Groenenboom
Delft University of Technology, the Netherlands, Faculty of
Mechanical, Maritime and Materials Engineering
j.a.ottjes@wbmt.tudelft.nl

KEYWORDS

Transport, Greenhouse-automation, mobile robot, process oriented simulation

ABSTRACT

A model structure is proposed for the design and evaluation of greenhouse factories equipped with an automated system for transportation of half products. The model is set up according the process interaction method. The system control decisions are made decentralized by a number of system elements and are modeled and clearly identifiable at several levels in the model. A series of experiments with a prototype model is carried out and presented. The possibilities of re-using the control part of the model in a real system as well as the potential of the design tool are discussed.

INTRODUCTION

Mass production of plants and flowers is performed in two steps. The first step concerns the growth from seed or cutting to relatively small plants in specialized greenhouse factories and the second step is the full development of these half products in greenhouses specialized in a small number of product types, (F+H 2004 and Florensis 2005). This paper concerns the first step, the growing of half products. In this branch only a few enterprises take care of the total market demand. Some of these enterprises have a broad assortment of products. Each product may have its own ideal grow path in terms of time, temperature and moisture rate.

There are two ways to approximate the ideal grow path of a plant. The first way is to keep the plant at a fixed location and to adapt the temperature and moisture rate as a function of time according the ideal grow path. The second way is to establish a limited number of climate zones with a stabilized temperature and moisture rate and to move the plant from zone to zone in such a way that the climate fits best to the plant ideal grow path. The first way has the advantage that the ideal grow path is approximated well, but in case of a large assortment with many different grow paths this will demand a lot of space. The second way requires less space but puts high demands on the timely transportation of plants from one zone to another. Moreover the fit of the realized grow path on the ideal grow path is worse. In the paper we focus on the second way of production, using climate zones with transportation of products between the zones. Next we will describe a generic model of a greenhouse factory where the transportation of products between climate zones is automated using mobile robots. The focus in this paper lies on the modeling. First the modeling technique used is explained, then the model is presented in pseudo code and finally a series of experiments is shown.

MODELING APPROACH

We use the process interaction method for modeling, (Zeigler 2000). The modeling approach can be summarized in three steps:

- Decompose the system to be studied into relevant classes of elements or functions, preferably patterned on the real-world elements of the system.
- Identify the properties and methods of each class. A class is characterized by its properties and methods. An instance of a class is called an element.
- Distinguish the "living" element classes and provide their process descriptions. A process governs the dynamic behavior of each element. A process is a method and has special features with respect to so called 'time consuming commands' that may occur in its body

Advantages of process interaction modeling are that it allows communication with non simulation experts and it offers a natural opportunity to model agent based systems because active elements are autonomous and capable of interacting.
We apply a simple vocabulary expressing the typical commands needed for process interaction modeling as well as a simple pseudo code (Ottjes and Veeke , 2002). All commands of our vocabulary will be written in *italic* style. We will use the common object oriented 'dot' notation for qualifying methods and properties
The clock time in the modeled system is called *now.*

A distinction can be made between time_consuming commands that govern the time consumption in a process and interaction commands that affect a state change of elements.

Time consuming commands

We have three forms of time-consuming commands to be used in a process description of an element class.

– *advance* (t) where t is a time interval generating a scheduled element state lasting that interval. t is determined by the element itself.

- *advance* (*while* condition) where the element state becomes scheduled as long as the condition applies. The condition is determined by the element itself.
- *advance* without any argument. Here the element state becomes suspended. A command (*resume*) originating from the process of another element is needed to continue the process.
- *finish* has the same effect as *advance* but skips to the end of the process.

Aliases for *advance* are used to increase the readability of the model. Here we use *wait* and *drive.*

Interaction commands

Interaction commands can be given by some element to influence the process of another element. We distinguish *interrupt* causing the targeted element to change from scheduled into suspended state and *resume* causing the targeted element to continue its process.

Use of sets

A set is used as a collection of elements. A set may represent a physical waiting queue but is also used to establish a 'one to many' relationship. The latter concept is often used control functions. An element may reside in more than one set at a time but not twice in the same set. Both an element and a set own methods for set operations. An element itself may *enter* and *leave* a set and a set may *add* and *remove* an element to/from its content.

Each element in a set knows its *predecessor* and its *successor* and each set owns the up to date values of its *mean length, maximum length*, *mean waiting time* and *maximum waiting time.* If there is no *predecessor* or *successor* the value *none* is assigned.

Other Set related commands are:
copy (Set1,Set2) All elements of Set1 are copied into Set2
move (Set1,Set2) All elements of Set1 are copied into Set2 and removed from Set1
empty (Set1) All elements in Set1 are removed from Set1

In order to look up elements with special characteristics we use compound commands like:
anElement = *first element* in aSet (*with* condition)
Here aSet is an arbitrary Set

The *loop* command is used for repeating actions.. There are three ways to use *loop*:
- *loop*: repeat indefinitely
- *loop* (n): repeat n times
- *loop* (*while* condition): repeat while condition

If a command acts on a command block, the commands in that block should be indented

All identifiers referring to a set will contain the word 'Set'. If it is necessary to add any explanation to a command this is preceded by //
All method names will be written in capitals.

THE MODEL

Now we will apply the process interaction modeling method on the greenhouse system

We distinguish three sub systems:
- The products
- The infrastructure with climate zones
- The mobile robots

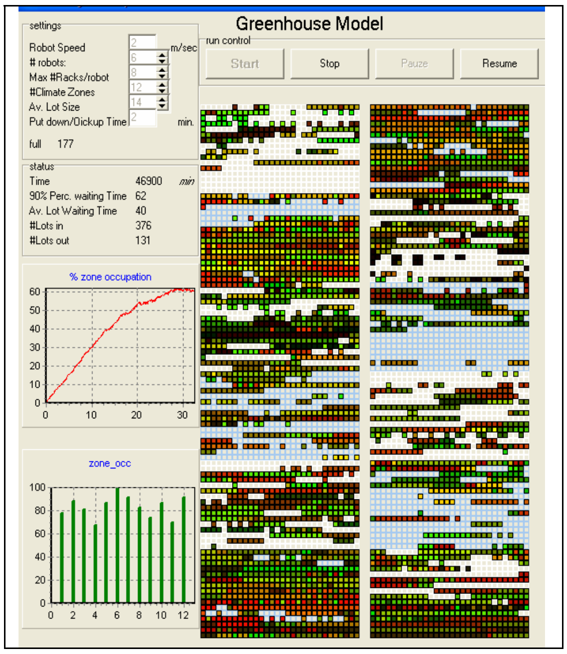
Applying the modeling recipe we first decide on the element classes and relevant properties and methods.

The products

A product (plant or flower) is defined by its name and its set of production steps. Each production step owns a reference to the climate it has to be grown in and the duration of this step: the GrowTime.
The basic product carrier is a rack. A rack is a plate, carrying a number of products of the same product type. A rack is the standard transportation unit.
A lot is the production unit. It consists of one or more racks. A lot contains aggregated individual client orders but is produced as one single production order. After production completion a lot has to be split up into the originating client orders and distributed to the final customer. We restrict ourselves to the lot production part of the production process.

In our model a lot acts as an 'agent' taking care of timely presenting itself for transportation. Consequently a lot, in process interaction terms, is an active element and owns a process. This process will be explained later in Table 2. The product related element classes are defined as follows:

```
Product
–   Name
–   ProductionStepSet //contains all specific steps
ProductionStep
–   ClimateZone
–   GrowTime
Rack
–   MyLot
–   CurrentPosition // position in climate zone
–   NewPosition
Lot
–   Product
–   RackSet // contains all racks of the Lot
–   RacksLeftToMoveSet
–   CurrentClimateZone
–   CurrentStep
–   NextClimateZone
–   NextStep
–   PROCESS
```


Greenhouse Model

settings
Robot Speed 2 m/sec
# robots: 6
Max #Racks/robot 8
#Climate Zones 12
Av. Lot Size 14
Put down/Oickup Time 2 min.
full 177

status
Time 46900 min
90% Perc. waiting Time 62
Av. Lot Waiting Time 40
#Lots in 376
#Lots out 131

Figure 1. Snapshot of the simulation screen. In the left part the overall occupation of all zones is shown as a function of time and the maximum occupation for each zone.

The infrastructure with climate zones

In Figure 1 a snapshot of the model screen shows an arbitrary configuration of 2 x 6 climate zones, separated by a middle path.

In the greenhouse model a climate zone has its own typical climate and its set of rack positions. Each rack position has a reference to its Rack and knows its own location. Each climate zone has a DestinationSet with Destination elements each representing one of the other climate zones. The idea is that all racks to be moved from some zone A to zone B are sorted in the RacksToMoveSet of the Destination representing zone B.
The classes and Sets in this subsystem are:

```
ZonesToServeSet   //contains the zones with transport
                  //demand sorted according criteria
ClimateZone
–   Climate
–   RackPositionSet
–   FreeRackPositionSet
–   DestinationSet
–   Location        // coordinates of entrance
Destination
–   Zone
–   RacksToMoveSet
RackPosition
–   ClimateZone
–   Location       // coordinates
–   MyRack          // reference to a Rack or None
```

The Mobile Robot system

Figure 2a shows the principle of the mobile robot and figure 2b a real one. A robot can one by one load racks up to a maximum number and after transportation put them down one by one according the rule 'first in last out'. Robots take care of the transportation of lots between and inside climate zones. The task of the robots is to assure that all racks arrive at the right time in the right climate zone with a minimum of transport delay. In the configuration of Figure 1 the middle path is used by the robots to drive from one zone to another. Inside a zone a robot drives over the racks both in x and y direction. It is necessary that there is enough room between the racks for the wheels of the robot. A robot can load racks one by one up to a maximum number of MaxRacks and transport these racks to another zone. The class robot owns a Speed and a RackSet in which all racks that physically are picked up by the robot (are onboard) are listed. All control decisions are taken locally and are explained in the robot process description given in Table 1

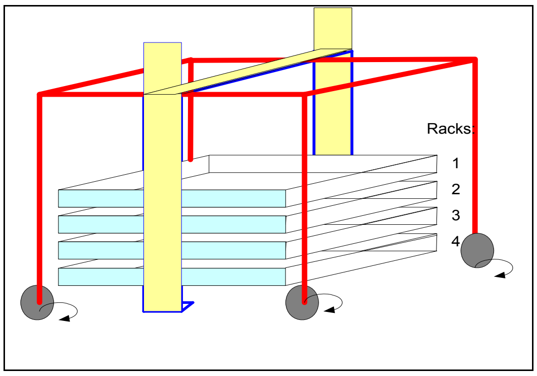


Figure 2a. Sketch of the automated (un)loading mobile robot.

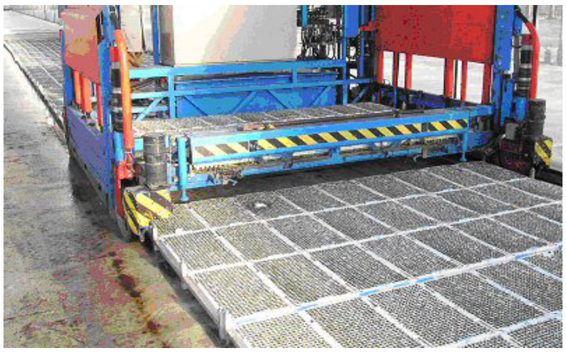


Figure 2b. A real (un)loading mobile robot, driving over a lane with racks containing trays with plants. (Florensis 2005)

The classes and sets in the robot subsystem are:

```
RobotIdleSet   //this set contains idle robots.

Robot
–    Speed
–    MyLot
–    MyRack
–    RacksToPickSet // racks to be loaded
```

- RackSet // loaded racks ´on board´
- MyLocation
- MyZone
- MaxRacks // maximum #racks possible
- PutDownTime
- PickUpTime
- SELECTDESTINATION
- PROCESS

The Processes

The robots act like agents, autonomously following their process description. The processes of the robot and lot classes are interrelated. The pseudo code of the robot process is given in Table 1 and the code of the lot in Table 2. Lines preceded by '//' contain additional comment. A short explanation will be given of both processes.
The Robot stays idle as long as there are no racks to be transported (ZonesToServeSet is empty). If there is work to do the robot selects a new destination (=climate zone) to go to. The method SELECTDESTINATION is used for that. A simple implementation of that method could be to just take the first climate zone in the ZonesToServeSet. If the destination is known the robot drives to the entrance of that zone and reserves the racks to be picked.

Table 1. Process of the Robot class

```
Robot PROCESS
Loop // this process is repeated infinitely
  enter(RobotIdleSet)
  wait while ZonesToServeSet Empty  //wait for work
  leave(RobotIdleSet)
  //Determine and drive to the right zone
  MyDestination =  SELECTDESTINATION
  MyZone = MyDestination.Zone
  Drive (Time to MyZone.Location)
  n=MAX(MyDestination.RacksToMoveSet.Length,
    MaxRacks)
  Move first n Racks from MyDestination. RacksToMoveSet
    into RacksToPickSet
  //Collect the racks to be transported
  Loop while(RacksToPickSet.NotEmpty)
    MyRack = RacksToPickSet.First
    MyRack.Leave(RacksToPickSet)
    Drive(Time to MyRack.Currentposition.Location)
    Wait(PickUpTime)
    MyRack.Currentposition.Enter(
      MyZone.FreeRackLocationSet)
    MyRack.Enter(Rackset)
  //Drive to next Climate Zone and distribute
  MyZone = MyDestionation.MyZone
  Drive (Time to MyZone.Location)
  Loop while(RackSet Not Empty
    MyRack=RackSet.Last
    MyRack.NewPosition = MyZone.FreeLocationSet.First
    MyRack.NewPosition.Leave(MyZone.FreeLocationSet)
    Drive (time to MyRack.NewPosition.Location)
    MyRack.leave(RackSet)
    Wait(Put_down_time)
    MyRack.Leave(MyRack.MyLot.RacksLeftToMoveSet)
    If MyRack.MyLot.RacksLeftToMoveSet Empty
       MyRack.MyLot.Resume
```

After that the robot loads these racks in a sequence of driving and picking and then drives to the entrance of the destination zone. After arrival it distributes the loaded racks in the zone by occupying free locations from the FreeLocationSet of the zone. These free locations are sorted in a proper way, for example fitting together. After each put down action the robot tests whether or not a lot-transportation-job is completed. If that is the case the lot is resumed and will continue its own process

The lot process is described in Table 2. After having determined the next climate zone to go to, the lot identifies the right destination in the destination set of its current climate zone. Then it makes its racks available for transportation by copying them into the right RacksToMoveSet according a sorting criterion. Sorting criteria are free to be chosen, for example according the physical location of the racks in the zone in order to minimize robot pick-up times. The lot further copies all racks into its own RacksToMoveSet, enabling the robots to keep track of the progress of transportation of the lot. Finally it sorts its CurrentClimateZone into ZonesToServeSet thus indicating that there is work to do in that zone. After having done so the lot *waits:* It becomes *suspended.* Only after all its racks are transported to the proper zone, the lot process is resumed by the robot that has placed the last rack. The delay that is caused in that way is the key performance indicator of the model. After having been resumed the lot waits during the grow time of its next step and after that repeats its process until all steps are done.

Table 2. Process of the Lot class

```
Lot PROCESS
NextStep =  Product.ProductionStepSet.First
Loop While (NextStep Not None)
    NextClimateZone = NextStep.ClimateZone
    NewDestination = First Destination in
      CurrentClimateZone.DestinationSet with
      DestinationZone = NextClimateZone
    Copy (RackSet, NewDestination.RacksToMoveSet)
    Copy (RackSet, RacksLeftToMoveSet)
    (re)sort CurrentClimateZone in ZonesToServeS
    Wait  // wait until the all racks are transported
    Wait NextStep.GrowTime
    CurrentClimateZone= NextClimateZone
    NextStep = NextStep.Successor
    (Product.ProductionStepSet)
Finish
```

The assignment of robots to climate zones is controlled via the ZonesToServeSet. The zones that need robots are sorted according criteria that are also free to choose. Examples of criteria are the distances to the robots, the total number of racks to be moved from or to the zone or the cumulative delay of the waiting lots or combinations of these criteria. In this case a robot only needs to choose the first zone in the ZonesToServeSet.

Apart from the mobile robot class and the lot class we need an element to take care of the generation of lots to be produced by the system: The lot generator. This generator may use historical input, for example all lots produced in the past year, or just generate input with the help of statistical distributions. If the model is detailed to a certain extend it would be possible to feed the model with real time data and run it real time, for example to use it as a shadow system for planning and control.
The lot generator will not be worked out further here.

Using the model in practice may demand that some additional functionality have to be added.

- Equipment failures
- Using other transport equipment, such as overhead cranes
- Reshuffling of free rack positions, comparable with the compression function for a computer hard disk
- Flexible input generator
- A distributed implementation of the model (Duinkerken et al. 2002)

The model is implemented in the simulation package Tomas (Veeke and Ottjes, 2000 and 2002)

EXPERIMENTS

In this section some test results are shown as an example how to use the model for design and evaluation purpose. The main parameter settings are listed in Table 3. The lot-stream input used is characterized in Figure 3 and Figure 4, showing the distribution of total growing times and the distribution of the lot sizes in numbers of racks respectively.

Table 3. main run parameters	
Run length	90 days
Mobile robot speed	0.5 m/s
#Mobile robots	4
Pick-up & put down times	2 min.
Max. # racks per robot	8
# climate zones	12
Dimensions of climate zones	30 x 16 racks
Rack dimensions	2 x 2 m. incl paths
#product types	1000
#steps per product type	Uniform[3,4,5]
#Racks per lot	Exp[8] with max. 24
Input location	31,1
Output location	32, 96

The green house configuration applied, is according Figure 1: there are 12 climate zones, each having 16 x 30 = 480 rack positions. The rack dimension is taken 2 x 2 m. including the robot tracks. The robot speed was set to 0.5 m/sec for all runs. The first series of experiments concern the number of robots needed. Figure 5 shows the results. The transport delay is the performance criterion. We conclude that 5 robots are sufficient. Next the influence of the maximum number of racks per robot is investigated. From Figure 6 we conclude that the lot delay tends to dramatically increase below 3 racks per robot. Finally the role of pick up and put down times of racks is studied. The results are put in Figure 7. We conclude that in this configuration the rack handling times have a very strong influence on systems performance. In practice numerous experiments varying much more variables are necessary.

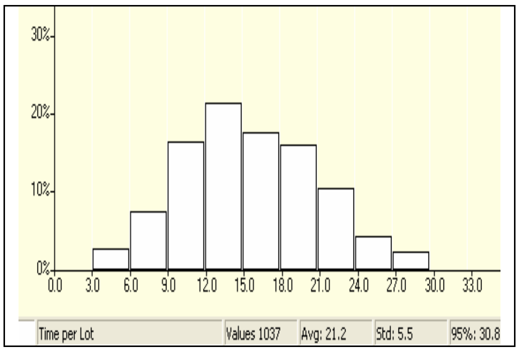


Figure 3. Distribution of total grow times per lot (days)

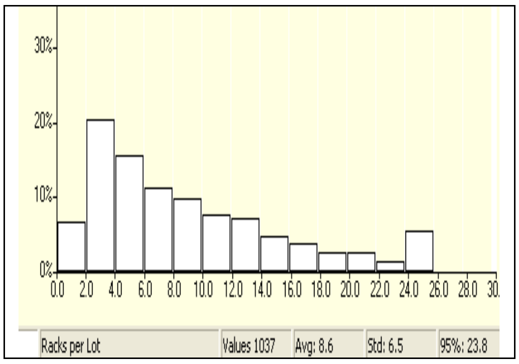


Figure 4. Distribution of the number of racks per lot

Next to the data incorporated in the graphs, all kinds of other information is available in the model such as robot travel distances, robot trip distances, occupation rates, rack filling rates, climate zone occupation etc.

The question arises if it would be possible to reuse a control system, developed and tuned with simulation, to a real system. To answer that question we should realize the practical consequences of the approach. The key issues here are to keep the necessary information up-to-date and to make the control distributed. The information part relates to all data about the lots and the racks. The type of information needed can be derived from the properties of the element classes and from the processes. For example the free rack position status should be updated every time a rack is handled and each robot should have access to a correct set of destination zones. Technologies that are very promising in that respect are radio frequency identification (Ni L.M et al 2004 and Stefansson G. and Tilanus B, 2000) and multi agent control (Bussmann et al., 2004)

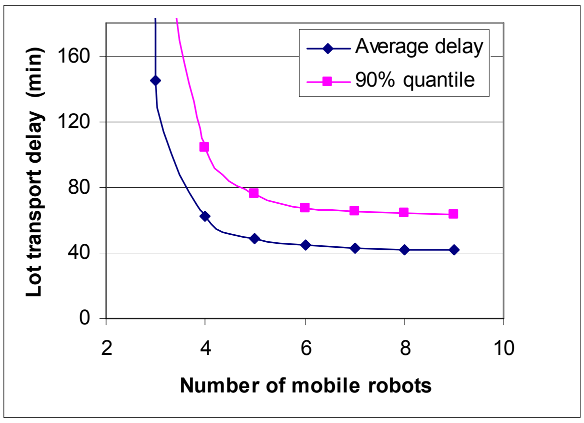


Figure 5. Transport delay as a function of the number of mobile robots

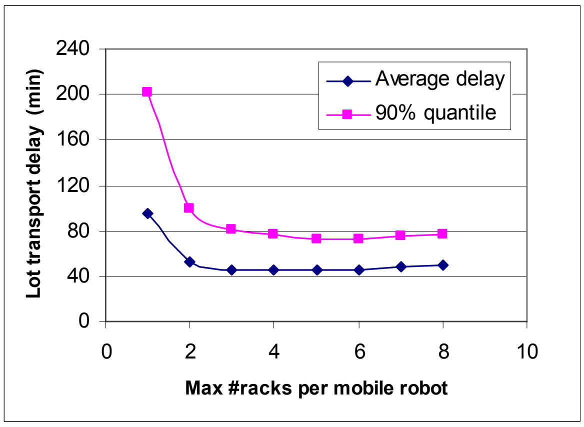


Figure 6. Transport delay as a function of the rack load capacity of a robot with 5 robots.

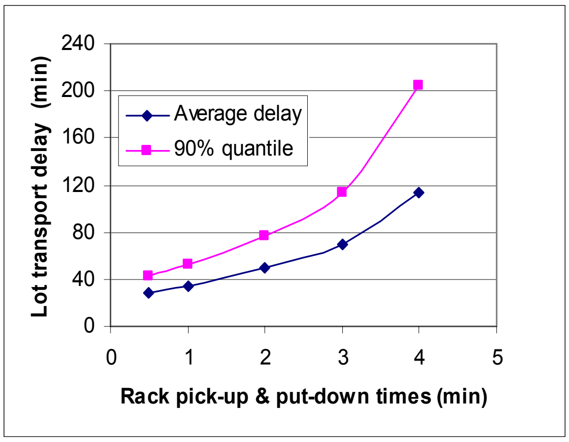


Figure 7. Transport delay as a function pick-up and put-down time per the rack with 5 robots

CONCLUSIONS

A model is developed for the design and evaluation of facilities for mass production of half products for the horticulture industry.. The model focuses on a production method in which production lots have to be transported frequently between different locations having different temperature and humidity settings. The focus was put on the modeling. The model is described in pseudo code and experiments have been carried out to illustrate the use of the model. Future model extensions will relate to the concept evaluation and analysis of other rack handling systems in the greenhouse such as overhead handling cranes and the analysis of climate zone arrangements and interchange buffers.

REFERENCES

Bussmann S, N.R. Jennings and M. Wooldridge,2004. "*Multiagent Systems for Manufacturing Control"*, Sringer-Verlag Berlin Heidelberg 2004

Duinkerken, M.B, J.A. Ottjes, G. Lodewijks, 2002; *The Application of Distributed Simulation in TOMAS: Redesigning a Complex Transportation Model.* Proceedings of the 2002 Winter Simulation Conference (WSC 2002). December 2002. San Diego. ISBN 0-7803-7615-3

F+H , 2004 Jungpflanzen automatisch auf Tour. (Fordern und Heben) 54 (2004) Nr5. pp 280-281.

Florensis 2005. http://www.florensis.nl/

Fishman, G.S. 2001. *Discrete Event Simulation. Modeling, Programming, and Analysis.* @001 Springer-Verlag New York, Inc. ISBN: 0-387-95160-1, 52-59

Ottjes J. A. , Veeke, H. P.M., , 2002 "*Prototyping In Process Oriented Modeling And Simulation"*,
Proceedings 16th European Simulation Multiconference ESM 2002, pp 20-26, ISBN 90-77039-07-04, Darmstadt, Germany

Ni L.M., Liu Y, Lau Y.C. and Zhang X, 2004. LANDMARC: Indoor Location sensing Using Active RFID. Wireless Networks, Vol 10 Issue 6, Nov 2004, pp 701-710

Stefansson G. and Tilanus B, 2000. *Tracking and tracing:principles and practice.* Int. J. Technology Management, Vol 20, Nos ¾, 2000

Veeke, Hans P.M., Jaap A. Ottjes, 2000. Tomas: Tool for Object-oriented Modeling And Simulation. *In proceedings of Advanced Simulation Technology Conference (ASTC2000).* April 16-20, 2000, Washington, D.C. pp. 76-81. The Society for Computer Simulation International (SCS), ISBN: 1-56555-199-0

Veeke, Hans P.M., Jaap A. Ottjes, 2002. TomasWeb: web site: www.tomasweb.com

Zeigler B.P., Praehofer H and Kim T.G., 2000. "*Theory of Modeling and Simulation* 2nd Ed. Academic Press, San Diego.