DEVELOPMENT OF A MANUFACTURING SIMULATION MODEL USING OBJECT ORIENTED PRINCIPLES

Shahab Khanian and Cathal Heavey*

**Department of Manufacturing and Operations Engineering, University of Limerick, Limerick, Ireland. Tel: +353-61-202891, Fax: +353-61-202913, Email: cathal.heavey@ul.ie*

Abstract:
This paper discusses the development of a simulation model for a manufacturing company developed using Object Oriented Programming (OOP) principles. In creating the simulation model emphasis was placed on developing a tool that could be used by production personnel and be easy to maintain, thereby facilitating continuous use of the model by the company. This paper describes issues relating to simulation model development that highlight the advantages of using Object Oriented principles and tools when developing large-scale simulation models. The package used was eM-Plant, which is an Object Oriented simulation package.

Keywords: *Object Oriented Simulation.*

1.0 Introduction

Simulation is a powerful tool and has been around for the past several decades, however it has several shortcomings. Some of the difficulties associated with simulation are:

- Development of a simulation model is expensive in time and money.
- For successful implementation it is a technology that requires personnel that are highly skilled in a number of topics such as:
  - Probability,
  - Statistics,
  - Design of experiments,
  - Modelling,
  - Computer programming.
- Simulation Models developed are usually used in once off projects and not on a continuous basis. This is due to many factors including the difficulties and skills required to use and maintain simulation models.

The above difficulties have resulted in a low take up of simulation by industry. This paper highlights issues relating to model development, model maintenance and ease-of-use by describing a simulation project case study where an object oriented simulation package was used.

2.0 Case Study

The company, a leading health care company, had carried out a previous simulation project. This project involved the development of a simulation model for one of the manufacturing areas of the company using a simulation package called ProModel (Promodel 2001). In general, the model developed had the difficulties listed in the last section. More specifically the company experienced the following difficulties:

1. ProModel uses high level modelling objects. These modelling objects facilitate quick and easy modelling when they map directly onto the system being modelled. However, these high level-modelling objects cannot be easily applied to systems that do not fall within their modelling domain. This situation usually results in a convoluted and cumbersome model that is not easy to understand or modify.
2. There is no encapsulation in the ProModel package. As a result, a change applied to one part of the model could affect the other sections of the model. This leads to difficulties in maintaining and applying changes to the model.
3. In ProModel there is no hierarchical structure and therefore all parts of the model are at the same level where as the real system has a hierarchical structure. This resulted in difficulties in mapping the real system into the model.
4. In ProModel both the user and the developer use the same interface. As the manufacturing system of the company is complex, the ProModel modelling objects had to be combined in an unstructured manner in order to simulate the system. This resulted in an increase in model complexity. As the user and developer interfaces could not be separated, any end-user of the model required a high degree of expertise in ProModel to use or to apply even minor changes to the model.

These difficulties led to a short lifetime of the model. The model could not be reused unless the developer spent a great amount of effort to apply changes for every application. In an attempt to overcome some of these difficulties an object-oriented simulation package eM-Plant (Tecnomatix 2001) was used to develop the model described in this paper.

2.1 System Description

The company produces many different types of contact lenses. A section of the manufacturing system dedicated to manufacturing one particular type of contact lens was modelled. The section consisted of three modules. The structures of these modules are similar. Each module contains two injection-moulding machines and four casting lines. A Casting line contains an epsilon casting machine, inspection unit, blister (Dial) packager machine, and an auditing unit, as shown in Figure 1.

In order to make a lens, two plastic cavities (moulds) called Anterior and Posterior are made by two separate injection-moulding machines as shown in the above diagram Figure 1. These disposable moulds are then used on the casting lines to manufacture lenses. Depending on the characteristics of these Anterior and Posterior moulds different lenses can be made. To create these Anterior and Posterior moulds inserts (i.e. Tools) are placed in the cavities of the injection moulding machines. Therefore, it is by changing these tools that different moulds are produced. The main characteristics of the manufacturing facility from a modelling perspective are:

- Tightly coupled manufacturing system

- Batch production and large product range
- Variable product quantities (i.e. Variable batch sizes)
- Complex set up procedures including line clearance
- Complicated control structure
- Extensive quality control system

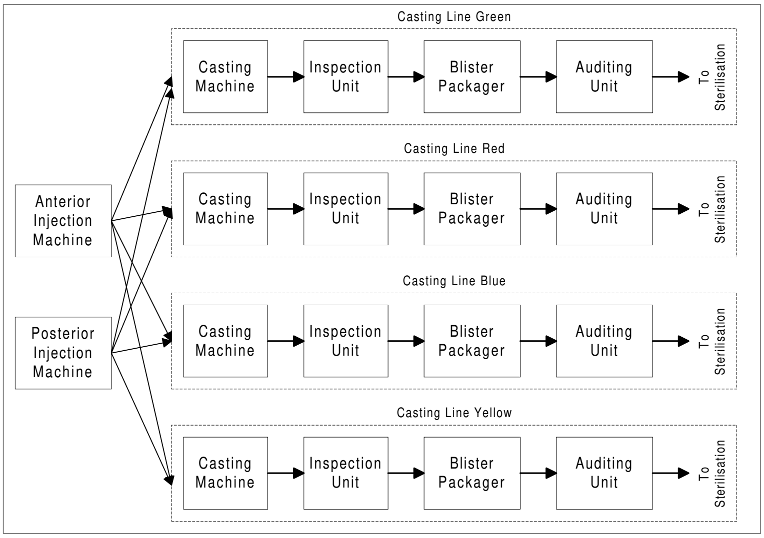

Figure 1 Structure of a module

3.0 Object Oriented Simulation

One way of categorising simulation software is by differentiating between Object Oriented and Non Object Oriented packages. Most of the simulation packages available in the market are Non Object Oriented, e.g. ProModel and Witness (Lanner 2001). Examples of Object Oriented packages are eM-Plant and MODSIM III. Ambler (1995) defines the object-oriented paradigm as "A development strategy based on the concept of building systems from (one hopes) reusable components called objects."

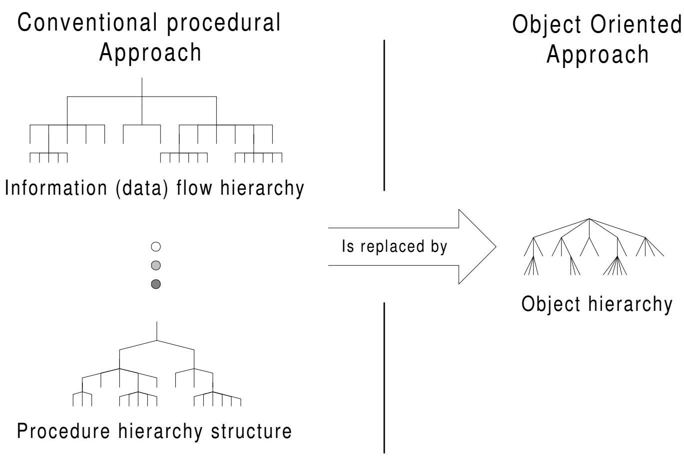

Figure 2 Unified hierarchy of Object Oriented approach

Dessouky and Roberts (1997), referencing other papers, make the point that in the last few years, replacing the procedural paradigm in the development of simulation models has become an important issue. The object-oriented paradigm has received the greatest attention as a substitute method. Object-oriented evolved from the idea of developing a computer representation that is close to the way people think. Thus, Object-oriented attempts to narrow the bridge between the model and the system modelled. Narayanan et al (1998) state, ″There is significant compatibility between the OOP (Object Oriented Programming) paradigm and the discrete-event world view formalisation. To be truly useful, though, research in OOP applied to simulation modelling must have an impact on one or more of the key aspects of simulation modelling and analysis: abstraction, implementation, application, verification, and validation″.

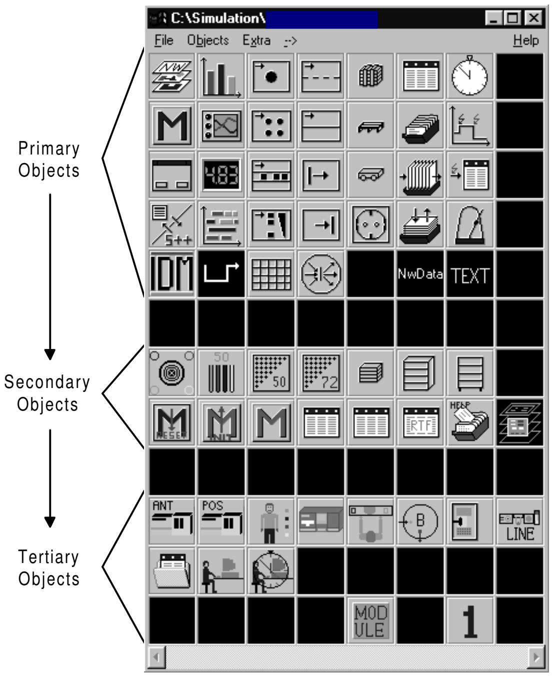

Figure 3 The object library developed for the company

3.1 Basic concept of Object Oriented Programming

The main characteristics of OOP that are of major benefit to the development of simulator models are polymorphism, inheritance, encapsulation and hierarchical structure.

- ***Polymorphism***: The same operation/function may apply to more than one object or class. It may apply to many classes. When this operation takes different forms in different classes it is called polymorphic. In other words a polymorphic operation is an operation that assumes different forms and whose forms can change dynamically. This characteristic of OOP is called Polymorphism.
- ***Inheritance***: is sharing of attributes and characteristics among objects based on a hierarchical relationship. Booch (1994) gives the following definition for inheritance: “A relationship among classes, wherein one class shares the structure or behaviour defined in one (single inheritance) or more (multiple inheritance) other classes. Inheritance defines an "is-a" hierarchy among

classes in which a subclass inherits from one or more generalised superclasses; a subclass typically specialises its superclasses by augmenting or redefining existing structure and behaviour."

- ***Encapsulation***: is most often achieved through information binding, which is the process of hiding all the details of an object that do not contribute to its essential characteristics. Rumbaugh, Blaha et al. (1991) define the encapsulation as consisting "… of separating the external aspects of an object, which are accessible to other objects, from the internal implementation details of the object, which are hidden from other objects".
- ***Hierarchical structure***: Booch (1994) defines hierarchical structures as: "A ranking or ordering of abstraction. The two most common hierarchies in a complex system include its class structure (including "kind of" hierarchies) and its object structure (including "part of" and collaboration hierarchies); hierarchies may also be found in the module and process architecture of a complex system". As stated by Booch complex systems may have several different hierarchies (Figure 2). However the Object Oriented approach has one unified hierarchy. Figure 2 illustrates this schematically.

Object **O**riented **S**imulation (OOS) provides direct benefits in many ways. These benefits arise from the basic characteristics of Object Oriented Programming (OOP). As Lin, Yeh and Shen. (1996) state ″Virtues introduced by OOP, in the context of simulation, include the concept that it is intuitive to view the real world as being composed of objects and that inheritance features of OOP greatly enhances the extensibility and reusability of simulation models″.

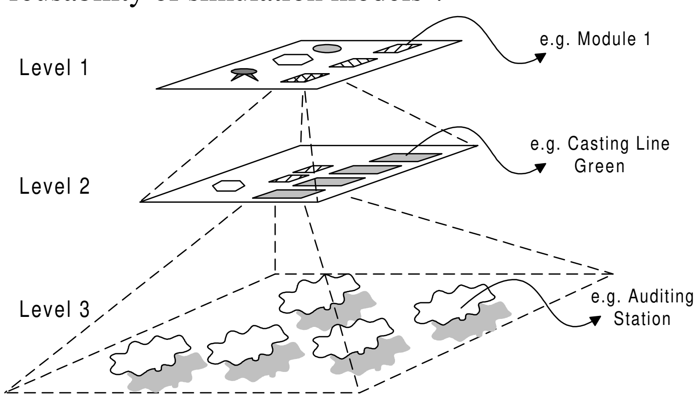

Figure 4 Model hierarchical structure

4.0 Model Description

In eM-Plant, in order to create a model, a library of objects is first designed and developed. This is achieved by using the primary objects in the initial library of eM-Plant. The object library designed for this project simulation model is divided into three sections.

- Section one, which is the primary section, contains all the basic objects that are initially used to create other objects.
- Section two is the secondary library created by using the primary objects.

- Section three contains objects that are created using both the primary and secondary objects. These objects are located at higher levels of the hierarchy. They may be called tertiary.

Figure 3 shows a screen print of the object library developed. The three sections of the library can be seen in this image.

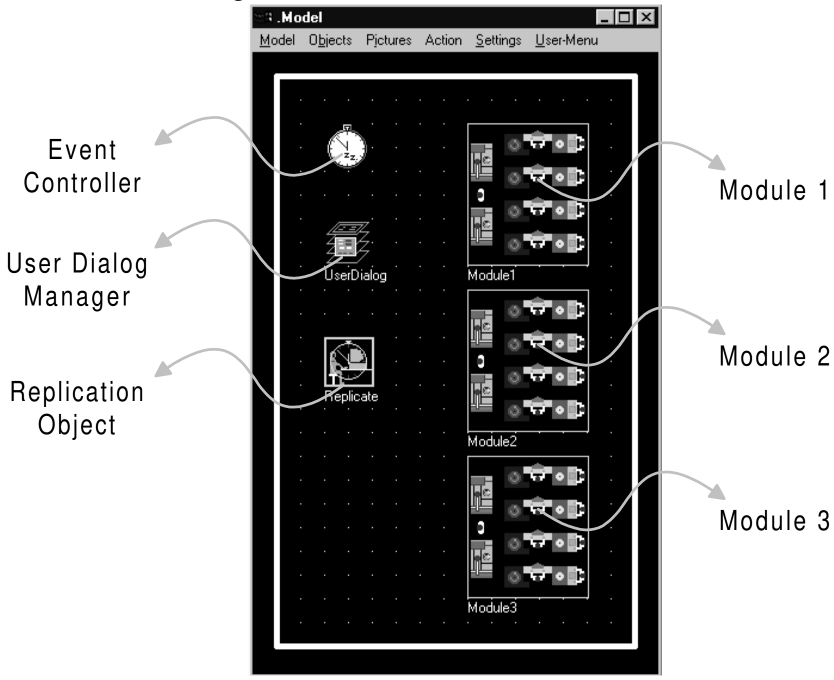

Figure 5 Level 1 in the model

The model was constructed using hierarchical principles. In order to describe the model it is best to introduce this structure. The model tries to map and copy the physical existing structure of the manufacturing environment. Figure 4 shows the hierarchy of the model developed.

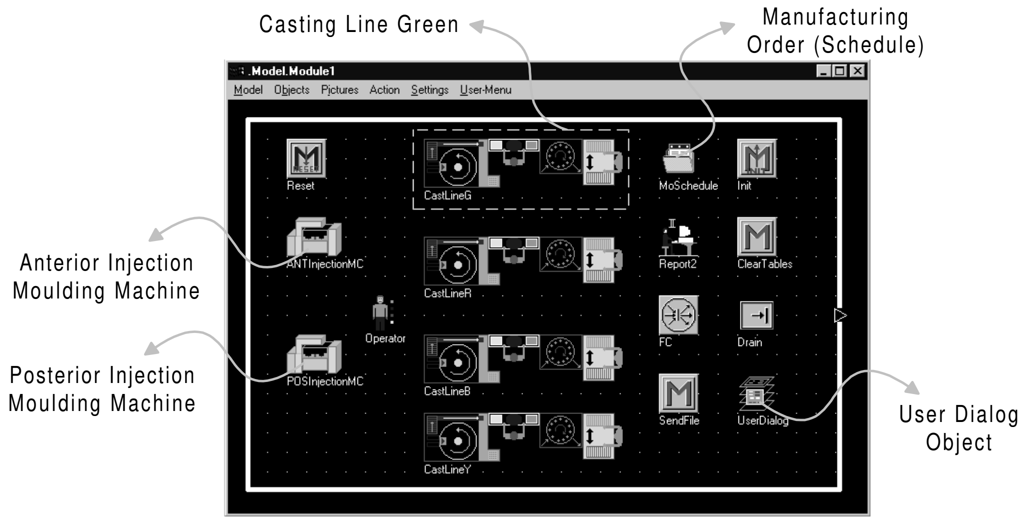

Figure 6 Level 2 - Individual Module

Level 1

This is the top level of the model developed. The end users access this level. It includes the 3 manufacturing modules as shown in Figure 5.

Level 2

Level 2 models the individual modules as shown in Figure 6. A module contains the following modelling objects:

- Anterior Injection Moulding Machine
- Posterior Injection Moulding Machine
- Four Casting Lines
- Manufacturing Order (Schedule) object

Level 3

Level 3 contains the individual objects that model a Casting Line (see), i.e.,

- Casting Machine
- Inspection Unit
- Blister Packer
- Auditing Station

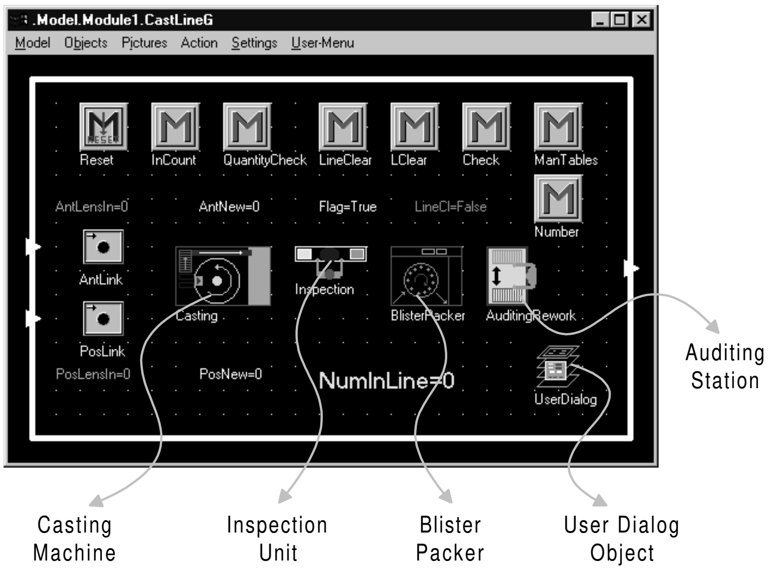

Figure 7 Level 3 - Casting Line

4.1 User Dialogs

Numerous dialogs were developed within eM-Plant to separate the User's interface from the Developer's interface. A sample dialog for an inspection mould machine is shown in Figure 8. The dialog allows full access to all data that parameterise an Inspection Machine.

- Machine Cycle Time.
- Tool Change Time.
- Tool Adjustment Time.
- Time Between Failures (TBF).

- Failure Duration.
- Shift_Auditing test Intervals (time).
- The pass chance of the Shift_Auditing test.
- Number of lenses taken for the shift test from every tube.
- The pass chance of the new lot test (e.g. 90) %.
- Sample size for the New Lot test.

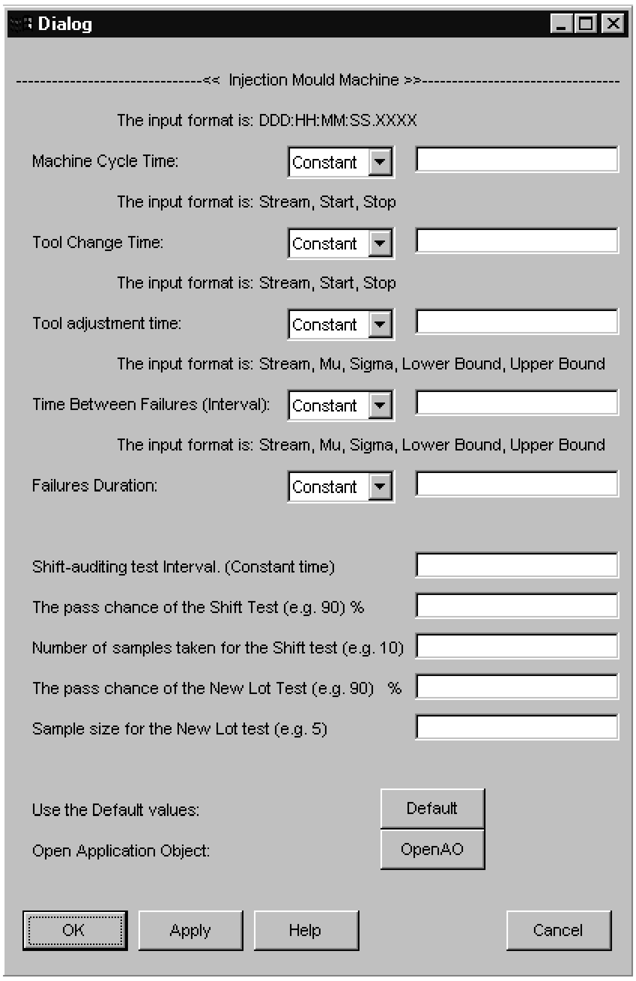

Figure 8 The dialog box for the injection moulding machines

5.0 Conclusions

This paper describes the development of a simulation model for a manufacturing company developed using Object Oriented Programming (OOP) principles. In brief the following general conclusions are highlighted as a result of this work:

I. This work highlights the advantages of using Object Oriented principles and tools when developing large-scale simulation models.
II. OO and eM-Plant properties were used successfully to separate the end-user's interface from the developer's interface, which resulted in lower skills requirements from the end user.

III. OO principles such as inheritance, encapsulation and polymorphism were used in developing objects, thereby reducing the model maintenance task and enhancing object reuse.
IV. OO principles facilitated the development of a model with a hierarchical structure that resembles the real system structure, which increased considerably the useability of the model.
V. eM-Plant objects such as tables were used in managing and controlling data and information transfer between objects within the model or external files. This characteristic of eM-Plant was found to be very useful for modelling the information flow within the system.

REFERENCES

Ambler, S. W. (1995). The Object Primer (The application Developer's Guide to Object-Orientation). New York, SIGS Publications, Inc.

Booch, G. (1994). Object-Oriented Analysis and Design With Applications. California, The Benjamin/Cummings Publishing Company, Inc.

Dessouky, Y. and C. A. Roberts (1997). “A Review and Classification of Combined Simulation.” Computers in Industrial Engineering Vol. 32(No. 2): pp. 251-264.

Lanner (2001), lanner Group web site: http://www.lanner.com

Narayanan, S., D. A. Bodner, U. Sreekanth, T. Govindaraj, L. F. McGinnis, C. M. Mithchell. (1998). “Research in Object-Oriented Manufacturing Simulations: An Assessment of the State of the Art.” IIE Transactions (Institute of Industrial Engineers) 30: 795-810.

ProModel (2001), ProModel web site: http://www.promodel.com

Rumbaugh, J., M. Blaha, William Premerlani, Frederick Eddy, William Lorensen. (1991). Object-Oriented Modelling and Design. New Jersey, Prentice-Hall, Inc.

Tecnomatix (2001), Tecnomatix web site: http://www.tecnomatix.com/