DEVELOPING AN OBJECT-ORIENTED REFERENCE MODEL FOR MANUFACTURING

Ricardo Chalmeta*; Theodore J. Williams**; Francisco Lario***; Lorenzo Ros***

*Grupo IRIS. Departamento de Informatica. Universidad Jaume I. Campus Penyeta Roja s/n. Castellón. SPAIN. E-mail: rchalmet@inf.uji.es Fax: INT + 34-64-34 58 48
** Institute for Interdisciplinary Engineering Studies. Schools of Engineering, Purdue University. West Lafayette. Indiana. USA. E-mail: tjwill@ecn.purdue.edu
*** Grupo GIP. Departamento de Organizacion de Empresas. Universidad Politecnica de Valencia. Valencia. SPAIN. Tel.: INT + 34-6-387 76 80. Fax: INT + 34-6-387 76 89

Abstract: A fundamental early step in the design process for a Manufacturing System is to develop a model which describes adequately the proposed system, in order to be able to study and to evaluate the impact of the design decisions on the system performance, before its construction. Among the different modeling methodologies applicable, the Object-Oriented Methodology is very useful due to its capability to represent all aspects involved in a complex Manufacturing System, including its dynamic dependencies. However, no such models exist at present in the literature. This paper shows such a detailed Object-Oriented Reference Model for Manufacturing System. This new model can be used to develop and to simulate particular manufacturing object-oriented models.

Keywords: Design, Manufacturing System, Object Models, Dynamic Models, Simulation.

1. INTRODUCTION1

The Design Process for a Manufacturing System requires that the designer have a clear knowledge of the overall system in order to be able to evaluate and to decide among different design alternatives involved in developing the system. However, the overall complexity of a Manufacturing System, especially when its design decisions are taken by multidiscipline groups with different needs and methods, requires the use of a model that describes the proposed system, with the required detail level, in a formal, concise and suitable way, in order to: (1) make the system understanding easier, (2) direct the analysis, design and construction of the system, integrating the four views (functional, informational, decisional and physical) that compose it (Amice, 1988) and (3) to be able to study the future impact of the design decisions on the system performance, before its construction, from the changes produced in the model.

2. MODEL REQUIREMENTS

In order to use a model in the design of a Manufacturing System, the model requires the following features:

1. Static features that describes the Manufacturing System from its four views:
   - physical view: represents the resources to carry out the activities
   - informational view: models all information needed to take and to execute decisions.
   - organizational view: describes who and how the decision of the system are taken.
   - functional view: describes the system activities.

2. Dynamic features that represent the process behavior.

3. Execution features, that is the possibility to bind the dynamic model with a computer language, in order to create a simulation computer program, to analyze

---

1 This research has been supported by the CICYT inside the TAP Project Number 95-0880.
the system performance.

A criterion which has been applied for many years is to use different modeling techniques for the different views. Therefore, a variety of modeling techniques exist and have been applied in the industrial enterprise field. For instance, general methods have been adapted to model manufacturing systems (i.e. structured analysis and design techniques) and specific tools have been developed for functional modeling (i.e. IDEFO), for informational modeling (data and process oriented methods) and for dynamic modeling (i.e. PetriNets).

The connections between the different views are essential to understand and to help in the Manufacturing System design. However, the above techniques cover different manufacturing views but none of them can model the manufacturing system as a whole. Also no one method is able to group all the partial models generated using these different methodologies in a coherent way. Therefore, the techniques used up to now are not yet suitable to generate a Manufacturing System Integrated Model.

3. OBJECT-ORIENTED METHODOLOGY

To fulfill the above need, the Object-Oriented Methodology (OOM) offers the features to properly carry out this process not only for its general applicability but also because there are many software tools supporting this methodology (Taylor, 1995). The OOM develops models that contain all the necessities features to assist in the design process of a manufacturing system. The resulting model can represent all informational, organizational, technological and functional aspects involved in a complex production system. It allows us to simulate them in order to analyze the dynamic behavior of the manufacturing system and it enables to develop informational computer systems to support the functions and decisions. That is to say, the OOM integrates static system features (considering all system views), dynamic modeling and simulation.

The application of this modeling focus based on objects to manufacturing systems has been recent. Thus, it is difficult to find detailed reports of projects. For instance, (Mize and Prat, 1991; Gaspart, 1991; Quellenberg, 1994) can be mentioned. However, none of these references present the detailed level of the Model that is proposed in this paper.

4. DEVELOPING OBJECT-ORIENTED MODELS

Although different object-oriented modeling techniques (Yourdon, 1979; Booch, 1996; Rumbaugh, 1991) have been developed, no one method is widely accepted as a standard and none is able to solve all problems appearing in its industry application. For that reason, a new methodology, named RCOO, is used.

The procedure to apply RCOO (Chalmeta, 1997), is structured into four stages, (domain, structural, functional and dynamic) that are shown in detail in the section five of this paper. For those who have a background in OOM, the last three phases are very close to the Rumbaugh proposal but have some difference like the graphic tools or the use of message flow diagrams in the functional model, instead of the data flow diagrams (that is because the Rumbaugh methodology is more orientated to software design and RCOO is oriented to model general systems).

5. OBJECT-ORIENTED REFERENCE MODEL FOR MANUFACTURING

Although as it has been showed above, an Object-Oriented model of a manufacturing system can assist in the design process, detailed examples of how to apply the object methodology to develop models of manufacturing systems do not yet exist in the literature (Mertins, 1994).

For this reason, this paper presents a Object-Oriented Reference Model for a Manufacturing System, using the RCOO methodology. This model can be used by different users to develop a particular manufacturing object-oriented model of their system that can help to understand the manufacturing complexity and to analyze the system behavior and the impact of decision making through simulation of the model. The methodology used follows:

5.1 Domain

Before starting any modeling process it is necessary to define the model (system) limits showing the internal objects that are included in the model, the external objects and the linking objects which connect external and internal entities (Taylor, 1995). Domain definition is a first approximation to delimit what entities are going to be completely modeled (internal objects and interface objects) and what entities are only considered from the point of view of their influence on the system without internally analyzing them (external objects). Figure 1 shows such an example domain for the model.
In the following discussion, every one of sub-models which form the Manufacturing System Reference Model (Structural, Functional and Dynamic) are presented (see figure 2).

5.2. Structural model

Classes identification. Object-oriented modeling with the RCOO methodology allows one to structure a complex system in an hierarchical way through composition and heritage relations. Thus, in the maximum aggregation level only the main entities are shown. The selection of such entities must be done according to the system analysis focus and in the detail required. In a first approximation, the considered entities appear reflected in the following description of a manufacturing system:

"A manufacturing system is considered to produce a set of final products through a set of elemental transformations (operations) of raw material (initial products) which are carried out in one or several machines. To regulate production, storage areas are used. These are physical spaces where products are placed temporarily. The movement or product flow into the system is carried out by transports. Finally, to execute an operation, auxiliary elements are eventually required, named resources" (Proth, 1995).

Therefore, the fundamental entities used in this model are six object classes as follow:

- parts (raw materials, WIP and final products)
- stores (elements of temporary pieces storage)
- machines (facilities to produce parts)
- transports (elements to carry out part movement)
- resources (material and human ones)
- operations (transformation or assembly)

Stores, machines, transports and resources are classes which form the production system fundamental structure, that is, the physical structure.

Over this structure the parts flow is defined, and then, new classes are included in the model. Raw materials enter the system from the external class named part arrival which acts as a source and is connected to the system through entry parts. Pieces leave the system through exit parts. These classes allow the system interconnection with its environment of material suppliers, product customers, etc.

Operation is a class which acts as a link between the plant static structure and the material flows. A external class called process plan determines the operational sequence necessary to make a product and relates it with machines, pieces, resources and transports.

The Manufacturing System requires an element that carries out the system management, that is to say, that implement the decisional view. In the developed model, the decisions are represented by the decisional class. This class receives from its associated classes information about what kind of pieces are in the system, what operations can be made, the machines and resources available, etc. in order to take decisions according to some implemented rules.

Finally, it is necessary to consider an interruption class describing the system stops produced due to breakdowns, maintenance, personnel resting, etc.

In the OOM, the structural model has a different meaning than in a conventional modeling methodology, that would be more oriented to the physical architecture of the system. With the OOM methodology, the structural model collects all system classes, including those corresponding to the physical, decisional, functional and informational entities. This is an advance in order to develop a integrated model.

Once the main classes of the system are identified, successive activities will gradually give a more detailed model description.
Relationships among classes. Production system classes are not independent but a set of Relationships among them can be defined. These relationships can be classified in three categories: Heritage, Association and Composition.

Heritage. This is a feature of the OOM through which a class named child acquires the properties of other system classes named father. However, it is not necessary that father classes exist in the system. It can happen that several classes use similar attributes and methods. It is thus convenient to introduce a virtual father class defining the common features. In the model this circumstance arises with stores, transports and machines where the same procedure to assign and place pieces is used. It happens the same with transports, machines and resources where a similar protocol for interruption management is used.

Figure 3 shows a hierarchical tree of classes of the main manufacturing system model entities, which appear due to the heritage relationship. When a class is derived from another, an arrow is painted from the more specific class towards the more general one.

Class hierarchy also introduces additional classes to allow one to complete the model and, at the same time,

simplify it. The model is made more simple due to the common attributes and methods of the several classes which need to be specified only once in the superclass.

Association. The model classes are not isolated but they interact among themselves. This relationship is called association and is represented graphically by lines linking the related classes. Association adds much information to the model and offers a global view of it. Besides, it allows one to identify those attributes which link classes. Figure 4 shows the main associate relations between the classes.

The first relation appears due to the parts movement through the system. The generator class is an entity in charge of introducing parts objects into the system (it model the Material Requirements System), assigning them to the entry parts store class. Then, parts pass through a net composed of stores, transports and machines until they are final products. The flow of the parts in the system is managed by the decisional class, asking the process plans class and receiving information about the state in the plant of the different classes. The decision class take decisions about the pull and push parts rules for the stores and the machines, the parts inventory rule for the stores, the machine operations, and so on.

Another important link is due to the interruptions, that can affect the transports, machines and resources.

Composition. The composition relationship, also named aggregation, describes those objects that contain other objects. This kind of relationship is given many times in enterprises. For instance, a final product is made with several components, or a department is formed by employees. Formally, in the Object-Oriented Methodology, the composed object does not physically contain the component objects but refers to them by means of attributes. This allows the same object to be a component of more than one other object. Figure 5 shows an example of a composition relationship in the manufacturing system.
**Attributes.** From the relations among classes, attributes of every entity are obtained. An attribute is any property or feature that all the objects of a same class have. Table 1 shows some characteristic attributes of the main classes.

<table>
<thead>
<tr>
<th>Table 1. Some classes attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>break time</td>
</tr>
<tr>
<td>duration</td>
</tr>
<tr>
<td>state back</td>
</tr>
<tr>
<td>resources</td>
</tr>
</tbody>
</table>

5.3. Functional model

Model objects interact among themselves creating a **communication flow** in which data transfer, control flow and service requirements are combined. The functional modeling phase consists of defining different **scenarios** that reflect the communication among classes. A scenario describes, in terms of **message flow diagrams** (MFD), interactions produced among system objects due to the execution of a specific process associated with a predetermined event collection.

Figure 6 presents a scenario corresponding to a PULL example, using a DFM. It is described below.

Figure 6 shows the message sequence taking place. The machine asks the **decisional class** about entities that can send it Type 1 parts, through the message **entities_up?.** As a result, it receives the answer message **entities_up** saying that it must receive type 1 parts from store A and store B. First, it tries to obtain a part from store A (because this store is the higher priority one) through message **parts_available?.** In return, it receives the message **available_parts (0)** saying that store A has no available pieces. Then, the machine sends store B the message **release_parts** indicating for it to transfer the part. Then, the machine receives the message **parts** with the mentioned piece. Finally, when the machine has processed the Type 1 part, it sends the message **parts** to the exit store, sending it one Type 2 part.

**Messages and Methods.** Once the existing relations among objects in a system is understood, by means of the **scenarios analysis**, the **messages** (elements used to communicate among objects) are determined and the **methods** which deal with them are described. Some of the methods of the machine class are:

- **Obtain entities up.** Send the message **entities_up?,** asking for the entities that have the parts that are needed, and receive the message **entities_up** with these entities.

- **Choose entities up.** Following the pull rule, send to the entities the message **parts_available?** And receive the message **parts_available** with the number of parts.

- **Assign parts.** Send the message **release_parts,** and receive the message **parts,** with the part.

If the objective were to model an information system, these attributes will be the data to introduce in the data base, and the methods will be the computer programs that transform them into information.

The importance of a graphic notation to understand the final model has led us to develop a diagram called **EAMM** (Entity-Attribute-Method-Message). It represents, in a graphical way, the fundamental mechanisms of a class. Examples of this diagram appears in (Chalmeta, 1997).

5.4. Dynamic model

Since the manufacturing system is not static but changes with time, it is necessary to represent its
temporal behavior in order to have a total understanding of the system. In the RCOO Methodology, this task is carried out by means of modeling the events that happen in the system and the different states that produce in a class. A class state is defined as the value of its attributes at a specific time. Thus, state evolution describes the temporal behavior of every class.

Table 2 shows the main events that can happen in a manufacturing system, and figure 7 represents a state diagram for the machine class. This diagram shows the different states of a machine, the events that produce the state changes and the value of some attributes. The dynamic behavior of all the entities of the model can be modeling following the same approach.

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>Assigning a part</td>
</tr>
<tr>
<td>e2</td>
<td>Releasing a part</td>
</tr>
<tr>
<td>e3</td>
<td>Notifying parts to transport</td>
</tr>
<tr>
<td>e4</td>
<td>A transport starts</td>
</tr>
<tr>
<td>e5</td>
<td>A transport ends</td>
</tr>
<tr>
<td>e6</td>
<td>Return to origin</td>
</tr>
<tr>
<td>e7</td>
<td>Notifying operation</td>
</tr>
<tr>
<td>e8</td>
<td>Assigning resources</td>
</tr>
<tr>
<td>e9</td>
<td>An operation starts</td>
</tr>
<tr>
<td>e10</td>
<td>An operation ends</td>
</tr>
<tr>
<td>e11</td>
<td>Interruption starts</td>
</tr>
<tr>
<td>e12</td>
<td>Interruption ends</td>
</tr>
<tr>
<td>e13</td>
<td>Final product</td>
</tr>
</tbody>
</table>

In this paper, a Reference Model of a Manufacturing System has been presented. This model is useful for an enterprise to develop a Particular Model (Amice, 1988) of its Manufacturing System. Then this particular model can be used for two applications. On the one hand, to help in the development of a Manufacturing Information System to support the management of the system. From this point of view, attributes are the data to incorporate to the data base and methods constitute the software program algorithm. On the other hand, the model incorporates the system dynamic features. So, with Object-Oriented Computer Technologies, it is possible to directly transform the model into an executable model, that can be simulated by a computer.

REFERENCES