AN OBJECT-ORIENTED FRAMEWORK FOR PRODUCTION CONTROL

B.H.L. Betlem, R.M. van Aggele

University of Twente, The Netherlands

In the process industry the current computer architecture is based on the hierarchy of operational functions. At the lower control levels the functions are offered as configurable building blocks, but at the higher levels no fixed structure is used. Generally, data modelling is neglected and, if data models are present, they are poorly accessible for the end user. Consequently, disadvantages of the current practice are:

- not reusable and maintainable unit operation models, control models, and recipes,
- poor facilities for control and logistical computer system integration.

By using an abstract framework of object-oriented models an important breakthrough can be expected. The data with the functions are ordered so that a high degree of reusability and maintainability can be provided. Also the framework acts as a checklist and speeds up the analysis phase.

In this paper a framework is presented for the application domain of the industrial production. It consists of seven structures. Thus far, this has been worked out roughly for logistics and control of continuous, batch processes, and pseudo continuous processes, such as milling. (Remark: information technologists use the word model in stead of structure. However, structure leads to less confusion, as model is already in use for a mathematical descriptions)

INTRODUCTION

The functional hierarchy, mostly reflected by a pyramid of functions (Fig. 1), is the basis of the current information systems. The implementation is the mapping of the required functions on the computer environment. DCS's are a typical example of this approach.

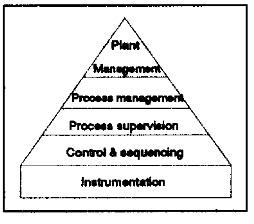

Figure 1 Functional approach of process automation

Most instrumentation and system firms aim for generally applicable concepts which are good maintainable and extendable. However, each firm introduces its own implementation-directed structure, which seriously restricts integration across the departments involved in production, such as: plant operations, planning, maintenance, quality control, and R&D. Systems intended for specific levels of operation bring about overlapping functionalities and conflicting forms of data management.

Dellisanti (1) describes in a strategic study the main characteristics of the process industries in the European Community. He concludes that:

- automation has to support the total production chain,
- discrete, continuous, batch etc. are conventional classifications that do not reflect the actual complexity of the computer integrated manufacturing and engineering (CIME) market,
- most plants are hybrid in nature, consisting of a mixture of process and discrete steps,
- fully computer-integrated production becomes more and more a factor of survival in the fierce international competition.

The above-mentioned considerations have lead to the following requirements for process automation:

- the main objective is to design a system that supports the business objectives and is capable of increasing profit;
- the system should be flexible: capable to support continuous, batch, and discrete production, and adaptable to product specification and equipment changes;
- the system should support cooperation between departments, and should be usable by the entire organisation; not only to accomplish the technical tasks but also to improve the job of the management;
- the system structure must fulfil the requirements with regard to maintainability and portability in line with good software design.

By using an object-oriented framework these objectives can be met. Data, models, and functions of the whole problem domain are arranged in inheritance or part-of structures, so that a maximum of reusability can be expected.

Until now, the object-oriented approach is a neglected area in production control. Only a few recent articles stress this subject. Menga, Elia and Mancin (2) have developed a framework for Computer-Integrated Manufacturing based on an object relationship diagram, but the essential hierarchical structures are not mentioned.

OBJECT-ORIENTED APPROACH

The object-oriented approach describes a system in terms of objects. Objects are entities in the problem domain that have a unique identity, a state and a behaviour. An example of an object is a distillation column. The distillation object has an identity e.g. C1001, a state, amongst others, the temperature profile, and a behaviour, represented by functions that can be performed by the distillation column, e.g. increase reflux flow, decrease boil-up flow, stop distillation, etc. As the state of the object is encapsulated, objects can only interact by means of messages. A message sent to an object is a request for executing one of the functions of the object.

Groups of similar objects are described by classes that contain the common properties of objects. A group of distillation column objects is described by a distillation column class that describes the state (temperature, etc) and the behaviour (increase temperature, etc) of all objects. Objects are instances of a class.

A problem description results in a collection of objects that are organized into two hierarchies: the part-of and the inheritance hierarchy. For instance, a distillation column object is part of a plant, that itself is part of a site. The distillation column is not only a part, it also contains several parts, like trays that can be decomposed into smaller parts. The inheritance hierarchy defines generalization and specialization relations between objects. For instance, the distillation column is a specialization of a physical component separator and is the

CONTROL '94, 21–24 March 1994. Conference Publication No. 389, © IEE 1994.

generalization of several types of distillation columns, like: sidestream column, steam distillation column, vacuum column.

The most important difference between the object-oriented approach and the conventional approaches is that the problem is not decomposed into data and functions, but modeled as objects, each incorporating its own data and functions. Additionally are these objects organized into hierarchies.

The advantage of the object-oriented approach is twofold:

- *uniformness*: the object-oriented model can be used in the analysis, design and implementation phase without conversion problems when going from one phase to the next. As software is generally developed iteratively, the absence of conversions will reduce the amount of errors and improve the clarity of the overall product.

- *extensibility & reusability*: the concept of inheritance allows designers to define new entities as specializations of existing entities, thereby reusing the state and behaviour specifications of the existing entities. The designer only needs to add the extension of the functionality, rather than redefining the complete entity. This concept facilitates both the extension of an existing system, when the functionality of the system needs to be extended and the reuse of parts of existing systems when building a new system.

FRAMEWORK ANALYSIS

Actually, the framework can be considered as a *global model*. It is a counterpart of the real world production organisation: the processes under study and the departments that control or support these processes. It can be applied to specific problem areas as an infrastructure, to define the data, functions and also their degree of automation. This gives a 'fill-in-the-blanks' solution which promises a maximum of extensibility and flexibility. The framework is the result of the problem domain analysis. The realisation can be included as a first phase in the automation lifecycle. Of course, this will require additional effort, but it has great profits. It is not recommendable nor necessary to implement the object-oriented global model in one go for every department. Rather a basis is created for integration of departments, changes in functionality, extensions, and a stepwise increase of the degree of automation.

The following analysis considerations have been made:

- *definition of the problem domain*
  What is the required scope for the problem under study ?
- *inventarisation objects concerned*
  What are the relevant object classes in the problem domain ?
- *transformation versus coordination*
  The cooperation, required between the different operational functions imply coordination functions. How should the relation between the operational and control functions be structured ?
- *reality versus abstraction*
  Operational functions are mutual related by the actual equipment structure as well as by an abstract equipment partitioning. The same applies to the control functions. In which way should the operational and the control functions be structured ?

Determination of problem domain

The framework should not only apply to the particular problem area, but cover the whole application domain. In this paper, this domain has been restricted to order handling for three production forms. The orders can be external ('make-to-order') as well as in-house ('make-to-stock'). Higher levels such as operational planning and adjacent areas such as machine maintenance and inventory-holding are not included yet. By using object-oriented independent structures, we expect that the framework can be extended easily.

An overview over all kind of functions related to production can be found in Rijnsdorp (3). They form the basis to the problem domain delimitation. In this book, also, the relations between the departments involved are described.

Inventarisation objects

The data of a production environment can roughly be grouped into three categories: temporal, static and dynamic data (Fig. 2). Orders represent the first category. They come and go, as they pass through the various departments in a relatively short time. When the order has been completed, only historical data will be left behind. The products associated with the orders do not change frequently. They have a more static character. The third category of data, associated with the manufacturing of the product orders, has a real-time character and changes continuously.

The conversion from raw material into various endproducts is performed via different steps. For batch and discrete production the equipment cycles through several different phases. So the status and the events that invoke state changes of a process can be described in terms of intermediate product movements as well as equipment phases. In practice, it is simpler to control production based on the status of the equipment than on the status of product progress. The measurements are equipment-related (not product-related) and not all equipment phases are covered by production steps (e.g. cleaning or start-up).

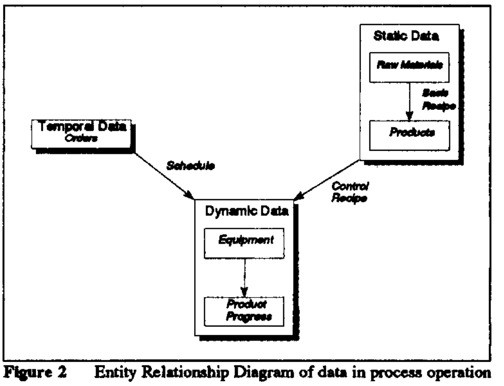

Figure 2 Entity Relationship Diagram of data in process operation

Three basic object classes are distinguished: order, product and equipment.

Coordination versus transformation

The equipment-related objects represent the dynamic operational side. In all functional real-time analysis methods a distinction is made between transformation and control functions. Although they act on the same objects, the mutual control functionalities have to be separated to obtain independent transformations.

Control functions coordinate objects such as: instructions from a higher hierarchical control level, interventions to a lower level, control models, proper goals, as well as status and measurement information from the plant. The common communication mechanism fail in describing coordination amongst various objects. Coordinated behaviour modelled by class inheritance is not possible, as the objects do not inherit from the coordinator. If the rules for coordination are implemented by the different separated objects involved in the coordination, then the system becomes poorly maintainable. Then, when the coordination changes, all models concerned should be adapted. As proposed by Aksit and Bergmans (4), a better maintainable solution is to abstract the control rules and to modulate them in a dedicated class. The operation will be described by separated equipment and equipment control classes.

Reality versus abstraction

The actual equipment has a partitioning according to a hierarchical organisation. For instance: a distillation column is a part of a plant, that itself is a part of a site. However, the function "distillation" can also be described by a more abstract hierarchy: distillation is a kind of liquid separation, that itself is a kind of physical component separation. Both structures are essential. The first reflects the reality, but the second gives the possibilities to define inheritance relations between equipment functions that lead to a structure of maintainable operation models.

Nearly the same applies for equipment control. For a plant two alternative control structures can be defined: the partitioning of the process into operational levels based on actual equipment units (eg plant scheduling, unit optimization, instrument control and switching), and the layering of abstract control functions. Although the control functions operate in conjunction with the equipment, it is necessary to build two structures: a structure for the operational units and a separate structure for the control functions. In this way it is possible to define abstractions of control functions.

The equipment structure and the equipment control structure are strongly related as both are reflections of reality. To avoid contradictions the desired control hierarchy can be constructed by mapping the abstract control structure on the equipment structure. In the applications this will be illustrated.

Process models at different levels of abstraction (e.g. dynamic black-box model versus a static physical model) offer examples of a multiple view obstacle: different descriptions of the same object. This object-oriented obstacles (and others) have been discussed by Aksit and Bergmans (4). The multiple view problem can be solved by using composition filters in the objects. These filters are a modular extension of the conventional object model. Messages need to pass the filter before being passed to the right function. Herewith the equipment functionality model can contain all kinds of views.

Practically, it is difficult to define inheritance relations for the mutual process and control models. Inheritance can only be implemented based on the mathematical equations used. But this problem domain is not yet elaborated for the object-oriented approach.

OBJECT-ORIENTED FRAMEWORK

After these preliminary considerations, we come to the framework promised in the introduction. The framework contains the following seven structures:

- order structure:
  historical object of orders.
- product structure:
  inheritance hierarchy of products classified on application,
- equipment structure:
  part-of hierarchy of area equipment,
- operation step structure:
  inheritance hierarchy of models for process functions,
- instrumentation structure:
  inheritance hierarchy of models for instrumentation functions,
- control structure:
  hierarchical structure of control functions,
- equipment control structure:
  hierarchical structure of coordinator objects.

The framework is object-oriented and it covers the functions and data mentioned under conventional systems. A stringent separation has been made between the data concerning general process functions, which are stored in the operation step structure, and the equipment-related data, which are put into the equipment structure. The object *operation step* is the link by which the product, the equipment structure, and the operation step structure are connected to find a relation between products and equipment. The operation step is defined as an in time indivisible process operation. Intentionally, the notion of unit operation is not used because it causes too much misunderstandings. In the equipment structure and the equipment control structure, the term *phase* is used to indicate the various operation stages (statuses) of the equipment unit. The equipment control structure contains only the coordinators of the various control levels. The control models, the equipment-related procedures, and the operation step models are grouped in their proper structures.

Order structure

The order structure takes care of all order-related data which are only temporarily present in production. The structure is a historical object that can be arranged to batches, or customer or in-house orders. Historical data, such as measurements taken during the production of an batchrun, are stored here.

Product structure

The product structure (Fig. 3) is an inheritance hierarchy to describe the relations between products. It contains the recipes based on operation steps (Fig. 4). The structure should be equipment independent. The products are ordered to application area, to simplify searching.

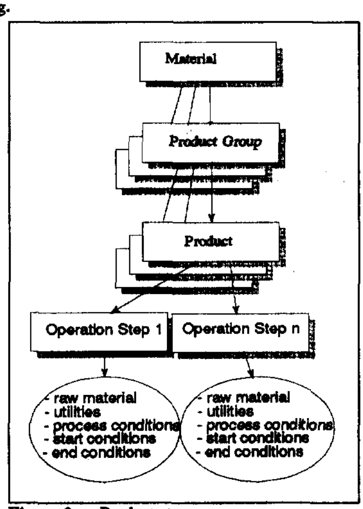

Figure 3 Product structure

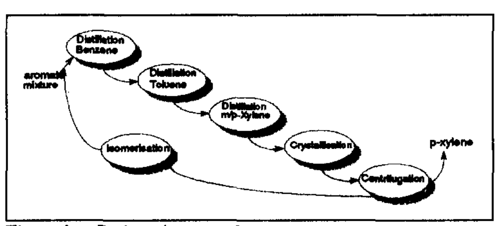

Figure 4 Basic recipe example

Equipment structure

The equipment structure is a part-of hierarchy and describes the structure of the actual production environment (Fig. 5). It reflects the process flow sheet. Every equipment unit or set of units is included and provided with its actual name. This structure contains equipment-related data:

- references to the equipment-independent operation step structure containing the process models, which describe the general processing functionality of a phase,
- equipment-related parameters of process models,
- the typical equipment-related procedures (models and sequences) for all phases which the equipment can perform (e.g. filling, emptying, warming-up, cleaning and regeneration),
- operation goal functions,
- (possible) connections to other equipment,
- maintenance data.

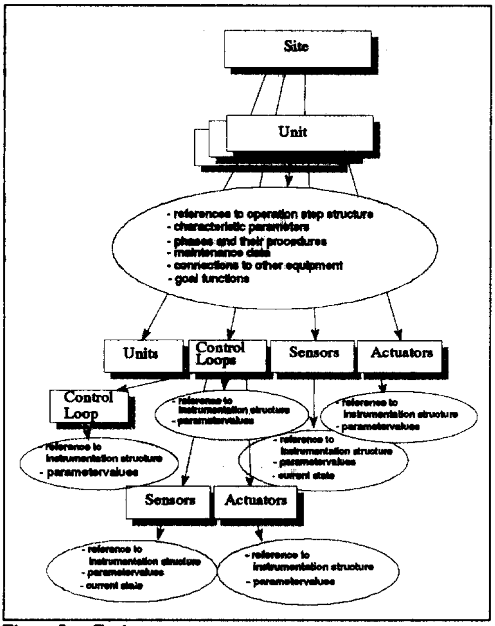

Figure 5 Equipment structure

Operation step structure

This structure (Fig. 6) is an inheritance hierarchy of process models. The indivisible operation steps are at the bottom of the structure. This structure provides all required static and dynamic models to describe an operation step. Typical operation steps are: reaction, evaporation, heating, distillation.

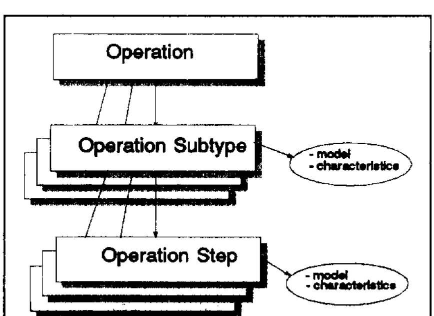

Figure 6 Operation step structure

Instrumentation structure

The instrument functions differ essentially from process equipment functions, so they are placed in a separated class. The instrumentation structure is an inheritance model for sensors, actuators and control loop models (Fig. 7). In this structure, for instance, the description of a PID-algorithm can be found. This structure has the same purpose and structure as the operation step structure

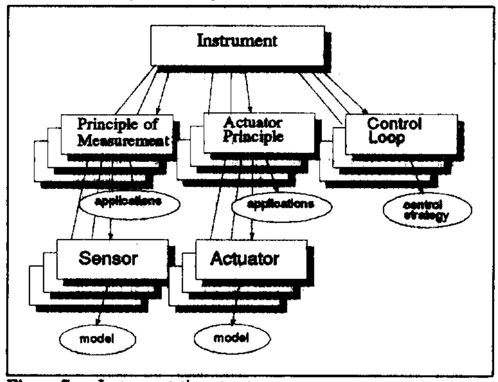

Figure 7 Instrumentation structure

Control structure

The control structure is a three level structure of the control hierarchy. It contains the models for scheduling, conditioning and (setpoint) control.

Scheduling determines place and time for an (internal) order. The schedule contains the timings for the different process stages, the product quality, and product quantity. The conditioner determines the required process conditions (setpoints). The controller maintains the conditions by directing the manipulated variables. Only models for model-based controllers are stored. PID-algorithm is placed under the instrumentation model.

This hierarchy is for all functions that are control-related and execute the functions of the different four decision steps as mentioned in the organisation hierarchy.This also applies to decision support functions like: adaptation, optimization, state estimation, and reporting. These subfunctions can be applicable at all three control levels.

Equipment control structure

This structure is the result of a mapping of the control structure on the equipment structure. The mapping is performed during the (re)implementation of the system, so this structure is directly available during production. The equipment control structure contains only the coordinators of the various control levels. The advantage of this approach is a minimum of redundancy. No fixed control hierarchy is imposed, but a proper hierarchy can be designed corresponding to the actual equipment arrangement and fulfilling the specific control needs. The structure contains the parameters of the control models. Three different examples are described below.

CONTINUOUS OPERATION

During implementation of the framework for a certain company or company location the equipment control structure is created by combining the control structure with the equipment structure. In this case the following mapping is made (Fig. 8):

- the plants on a site are scheduled by the *site scheduler*,
- the site is optimized by the *site conditioner*,
- the conditioning of the plant is established by the *plant*

conditioner,
the control of the process is carried out by the *process controller.*

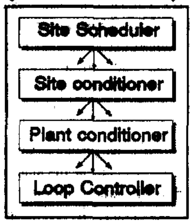

Figure 8 Equipment control model for a continuous operation

Site scheduling

The goal of site scheduling is the assignment of orders to plants, and indicating the period of production. For this purpose the scheduler needs information about the product and the available equipment.

The product structure contains the recipe consisting of operation steps together with conditions, utilities and raw materials per operation step. As continuous plants are dedicated to a certain product, a direct link between product and equipment is required. This link can either be present in the product structure, or can be introduced by the operating personnel.

The scheduler must check if the required raw materials and utilities are available. If all the selection criteria are fulfilled the plant can actually be scheduled. Here a rough estimate has to be made of the duration of production.

Site conditioning

Site conditioning is important for integration of processing and utility plants to allocate scarce raw materials and energy. The conditioner can take care of the following points (Rijnsdorp (3)):

- finding optimal loads for production and utility plants,
- bilateral coordination between plants for material and energy exchange,
- reckoning with bottlenecks.

To calculate setpoints for a production site, the site conditioner needs information about the process conditions to be adjusted at this level and a model describing the behaviour of the site as a whole (Fig. 9). This model is simpler than the combination of the plant models used for plant conditioning. The process conditions can be found in the product structure, while the model describing the equipment functionality can be found in the operation step structure. The equipment parameters are stored in the equipment structure.

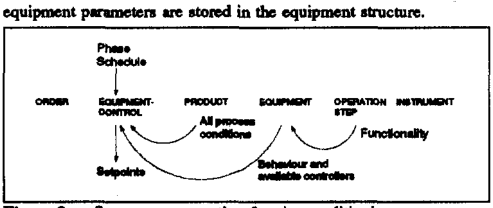

Figure 9 *Structure cooperation for site conditioning*

The conditioner must now obtain the controllers assigned to the site level from the equipment structure. With the gathered data, the setpoints can be determined and passed on to the pertaining controllers. Plant conditioning is done in the same way.

BATCH OPERATION

For batch operation the following mapping is made (Fig. 10):

- the plant is scheduled by the *site scheduler,*
- the phases of production are scheduled by the *plant scheduler,*
- the conditioning of the phases is done by the *phase conditioner,*
- the control of the process is carried out by the *process controller.*

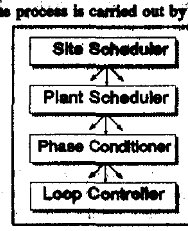

Figure 10 Equipment control structure for a batch operation

Differences between continuous and batch production especially concern site scheduling (the way to find the most suitable production location) and handling of the various production unit phases which are a consequence of the dynamic behaviour of batch processes.

Site scheduling

Unlike continuous site scheduling, for batch site scheduling a direct link between product and equipment is not necessary. If this link is not present, a search for a plant capable of producing the ordered product has to be carried out (Fig. 11).

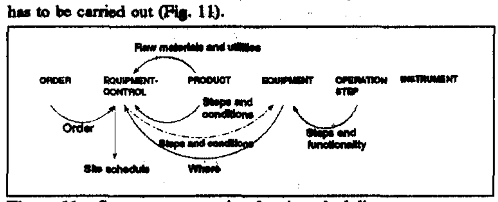

Figure 11 Structure cooperation for site scheduling

The scheduler inspects the basic recipe in the product structure to find the necessary operation steps. Then, the scheduler uses the equipment structure to find a plant to carry out these steps. For this task the *equipment structure needs the operation step structure to determine* which operation step (or steps) a piece of equipment can perform. This option can result in one or more plants which can carry out the requested operation steps.

A check must be made if the necessary process conditions can be established in a plant. This check can be done simultaneously with the equipment selection. The conditions to be realized in the various production steps are stored in the product structure while the equipment structure can determine which conditions can be realized in a certain plant.

Plant scheduling

In case of batch processing, the overall site schedule is not sufficient. Each production run consists of phases, which have to be scheduled as well. For this purpose one requires the model from the operation step structure and the parameters from the equipment structure describing the behaviour of the piece of equipment that carries out the step. Also the production phases have to be obtained from the equipment structure (Fig. 12).

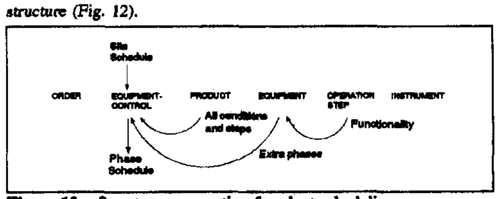

Figure 12 Structure cooperation for plant scheduling

The operation steps in the product structure must be mapped on the production phases in the equipment structure. Comparison of these two lists of production phases can furnish extra (equipment-related) phases to the list obtained from the product structure. The duration of each phase is calculated with the production demands obtained from the product structure, and from the model and parameters describing the equipments behaviour obtained from the operation step structure and equipment structure.

Phase conditioning

This conditioner differs from continuous production conditioners for his scope: a phase instead of a site or plant. Further, it checks for end conditions of the current phase and the start conditions for the coming phase (some phases can operate concurrently e.g. fill and heat). When the phase conditioner determines a start or stop condition this is passed on to the plant scheduler.

Comparison with ISA-SP88

In many respects the object-oriented approach differs from the structures defined by the ISA-SP88 group (5). An important one is found in the recipe handling procedure, which, of course, is characteristic for batch processes. In this paper the recipe as well as the recipe control function are a result of mapping of two structures. By using inheritance hierarchies, redundancy is avoided and reuse of recipes and structures is strongly promoted. In contrast, ISA-SP88 chooses for separated recipe models at four levels. At the top level one finds the basic recipe and every lower level adds specific information coming from different parts of the organization. In our opinion this structure is less maintainable.

MILLING OPERATION

The hot-strip mill is an example of a pseudo continuous process. After heating in a furnace, slabs of different thickness and material composition are reduced in thickness by several roller-stands with a factor forty till eighty. The resulting band is cooled by water and winded up. The mill works based on client orders. For a milling process the following mapping of control functions is made (Fig. 13):

- the mill orders are scheduled at two different levels by the functions *day and hour scheduler*,
- the *pacing scheduler* tracks and optimizes the production progress and signals the furnace operator when the next slab can enter the milling section,
- the conditioning is performed by the function *unit conditioner* for every slab,
- the control of the process is carried out by the *process sequencing and control* functions.

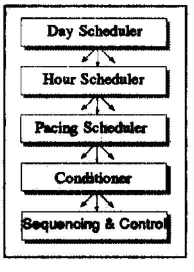

Figure 13 Equipment control structure for a milling operation

The most interesting control functions are the pacing scheduler and the slab conditioner, as they are unique for this process type.

Pacing Scheduler

Every time a slab enters the mill, the initialisation function of the pacing scheduler estimates for the next slab the required processing time of each unit with simplified grey operation step models. With these data, a schedule is composed that ensures a minimal slab interval time at the bottle-neck position. Most parameters are equipment related and fetched from the equipment structure. Some of the data go with the order, such as: actual and required length and thickness. The initialisation function is stored in the control structure, while the schedule itself is stored in the equipment control structure.

When a slab enters the mill, its progress control is attended by the scheduler. After the head or the tail of the slab passed a tracking point, the schedule is adapted to the actual situation. No new calculations of processing times are necessary. If the bottle-neck interval time becomes too short, then the slab (and all the following slabs) are delayed. The update functions are available in the control structure.

When a slab exits the mill, the slab is removed from the schedule. The remove functions is also located in the control structure.

Conditioner

For every slab entering a certain area, new unit setpoints as well as unit controls have to be calculated. Before a slab enters, the roller-stand is directed to a starting-position by the conditioner. When the slab enters, the setpoint controllers take over. The setpoints and controls depend on the product (composition, hardness and thickness) and on the equipment status (e.g. roller wear). The models for the setpoint calculations are stored in the operation step model, the product data are stored in the product structure, and the equipment parameters are available in the equipment structure. Many parameters in the product and the equipment structure are adapted after rolling. The adaptation function is stored in the control structure. The control, setpoints and measurements which are of interest, are stored in the order object.

CONCLUSIONS

A framework consisting of seven object-oriented structures have been designed to cover the domain of order production. On one hand, some structures are reflections of the real world: order, product, equipment, and equipment control structure. On the other hand, various structures are developed to assure a maximum of reusable functions and models: operation step, instrument, and control structure.

The framework is well suited to incorporate functions supporting other adjacent departments. Examples are: order handling for sales (order structure), maintenance support (equipment and instrumentation structure), and flow sheeting for engineering purposes (equipment and operation step structure).

Still, the state of the art of object-oriented design has some limitations. These can be solved by the introduction of composition filters and coordinator objects.

REFERENCES

1. Dellisanti G., 1992. Trends in Process Industry; Report of the CIM-Europe Interest Group, Brussels october.
2. Menga G., Elia G., Mancin M., 1992, "An Environment for Object Oriented Development of CIM Distributed Automation"; Interkama Congress 1992, pp.613-621; R.Oldenbourg Verlag, München.
3. Rijnsdorp J.E., 1991, "Integrated Process Control and Automation", Elsevier, Amsterdam.
4. Aksit M., Bergmans L., 1992, "Obstacles in Object-oriented Software Development"; OOPSLA'92, ACM SIGPLAN-notes Vol.27-10.
5. ISA-dS88.01, 1992, "Batch Control Systems (models and terminology)"; Draft 5.