Integrated Manufacturing Modeling System

by H. Engelke
J. Grotrian
C. Scheuing
A. Schmackpfeffer
W. Schwarz
B. Solf
J. Tomann

The objective of the Integrated Manufacturing Modeling System (IMMS) project was to build a generic software package for manufacturing modeling. The package allows interactive model building, testing, and experimentation. IMMS integrates the fundamental components of modeling, namely system description, data acquisition and management, graphic input and output, and performance simulation. The fundamental idea is to describe the flow of information, materials, and resources by using parametric standard model building blocks. Inputting is done by placing function symbols on a graphic screen and by filling in blanks for parameter data. The output includes all common performance measures, such as utilization and work in process (WIP), in graphic form. The simulation language used to implement the model building blocks is RESQ2. The Integrated Manufacturing Modeling System combines the advantage of simulation modeling with the advantages of end user programming capability. Simulation results allow intelligent decision making based on analysis (instead of intuitive experience), and an end user package reduces the turnaround time for modeling and eliminates the need for specialized programming skill. All this translates into faster and better-quality manufacturing system design and operation.

Introduction

The complexity of the structure and dynamics of manufacturing systems tends to exceed human comprehension.

- The functions of production equipment, data processing equipment, and human workers are diverse and interact in a complicated way which often is not readily apparent.
- The division of labor promotes one-sided system views. Also, system views may be distorted by the fact that not all functions are equally obvious. For example, the activities of a machine on the floor can be watched, whereas a control decision may be noticed only because of its consequences.
- Knowledge about a given manufacturing system is rendered obsolete more rapidly by fast-changing product and production technologies. This makes thorough understanding of a system difficult simply because of lack of time.
- Increasing automation makes workers less familiar with production. Passed-on experience may be lost. Also, the creativity stimulated earlier by direct contact with a system is now suspended.

The transparency of manufacturing systems can be improved and decision support can be obtained by advanced modeling techniques, which should be applied during the different planning phases from first design to the operation of a system. In the past and generally today, modeling activities have been focused only on performance simulation. The simulation of manufacturing systems is done by simulation "artists," highly skilled programmers who use simulation languages such as GPSS, RESQ, SIMPLIX, GASP, SLAM, SIMSCRIPT, or SIMULA.

In order to cut down the time-consuming model building and validation processes, some parametric general simulation models were developed, such as the Cycle Time Simulator (E. Fishkill) [1], Manufacturing Model (J. Schneider, Endicott), Multiple Application Simulation System (E. Nelson, Poughkeepsie), Mercury/Zephyr Models (D. Burns, Lexington), and PCMODEL (D. White). These models are valuable within their preplanned range of applications. However, the end user cannot change the model characteristics; for instance, he cannot model a special information flow if the model architect did not allow it.

A breakthrough in modeling efficiency and user acceptance will be achieved when the decision maker (i.e., the technical specialist or the manager) who has to solve a logistical task obtains a tool allowing him to model his task himself. Steps in this direction are represented by modeling tools like SIMULAP (Fraunhofer Institute, Stuttgart) [2] and MAP/1 (Pritsker & Associates, Inc.) [3].

This paper describes a truly generic modeling package based on a structured approach which describes manufacturing systems in general terms and allows graphic editing of models.

IMMS goals

The Integrated Manufacturing Modeling System (IMMS) is designed to be a generic modeling tool for the manufacturing planner or system designer. Typically, the life cycle of a manufacturing system is composed of the following phases: design, building implementation and debugging, operation, modification, and end of life. During each of the subsequent phases, appropriate methods to model the manufacturing system are required.

The Integrated Manufacturing Modeling System can be applied particularly during design, implementation and debugging, operation, and modification.

During system design or modification, modeling with IMMS results in faster and better planning of the flow of products, information, and resources; better resource allocation; better specification of the production system; and better understanding of the system behavior regarding throughput, turnaround time, work in process, flexibility, and reliability. During the operation phase, modeling supports product scheduling and line control, resulting in an optimized utilization of resources.

IMMS concepts

Design of the Integrated Manufacturing Modeling System began with an analysis of the user requirements for such a system.

- The modeling functions should allow a structured manufacturing description, hierarchical top-down modeling, graphic input/output, and a broad application scope.
- The user interface should provide an interactive dialogue, a fast system response, user-oriented input and output, a high-level modeling language (building blocks), modeling control from the workstation, help functions, and ergonomic hardware and software design [4].
- With regard to system integration, the modeling tool should 1) have an intelligent workstation providing color, interactive devices (light pen or mouse), graphic processor and picture storage; 2) use central computing capacity for running simulation software with high requirements of virtual storage and computer power to run a data base system; and 3) integrate an advanced data base system to store, retrieve, and combine manufacturing data, modeling data, and graphic data.

To meet these requirements, the IMMS development effort was subdivided into five major tasks:

- Manufacturing system description.
- Graphic editing and experimentation.
- Data acquisition.
- Data management.
- Performance simulation.

These components of IMMS are being integrated into one software package.

- *Manufacturing system description*

The key to improved modeling techniques is a concept for describing manufacturing processes in generalized terms. The method should permit modeling of all production system activities (material flow, information flow, resource flow), structuring of the system in a top-down approach [5], and use of standard modeling functions; it should be easy to understand and use.

Available description methods [6–8] were analyzed and found to be unsatisfactory for IMMS purposes. A Manufacturing Description Method (MDM) was developed in order to achieve a clear and accurate description of manufacturing systems. The Manufacturing Description Method describes the execution of manufacturing activities in terms of *manufacturing entities* and *manufacturing modeling functions*. It permits specification of the structure as well as the behavior of a manufacturing system [9]. The structure specifies what manufacturing modeling functions a system includes and how these are related to one another. Since detailed system descriptions should be derivable from less detailed ones and vice versa, the method allows hierarchical structuring.

In this way one can cope with complexity and minimize description expenditures by balancing the level of accuracy according to the needs of the user. While the system structure explains what functions a manufacturing system includes, the behavior of a system is described by functions performing activities which cause the flow of objects, so-called manufacturing entities, through the system. **Figure 1** shows the IMMS view of a production system.

Manufacturing entities
Three classes of entities are used to describe the execution operations of a manufacturing system: *information, material,* and *resources.*

Information exists in human- or machine-executable form. In machines information occurs as all kinds of digital and analog data which may represent speech, pictures, and text. It is stored on a variety of media including semiconductor memories, magnetic and optic materials, and paper. A manufacturing system receives information from its environment. It gets orders from the market and technical information on products and manufacturing processes from development engineering and vendors. Production planning generates production orders and provides this information to the production execution functions. In addition to the information coming from outside, information is generated by manufacturing functions. This information is exchanged and used by other functions of the manufacturing system or may be communicated to the outside world.

Manufacturing functions use or generate the following types of information: orders, order status reports, technical control information, and technical status reports.

Orders invoke the execution of manufacturing functions. Orders may be characterized by a list of attributes specifying order identity, order type, identity of related technical information, order priority, order release date, or order completion date.

Order status reports refer to the status of orders. Orders may be active, pending, or finished.

Technical control information contains the data necessary for specification of a manufacturing function. There is a close relationship between orders and technical control information. Most of the technical information is transformed into order information at some lower level of subfunctions. A main purpose of technical control information is to serve as a basis for generating lower-level orders. Since technical control information is necessary for manufacturing execution, it can be considered a manufacturing resource. In fact, this corresponds to a present trend in industrial management science, in which information is considered an independent production factor beyond the classical production factors of equipment and human work. The Manufacturing Description Method emphasizes the independent role of information by treating it separately from resources of a physical nature.

Technical status reports contain information on the present availability of entities, previous activities of a function, and reasons for the unavailability of entities, e.g., entity is missing, entity is being used, entity has wrong quantity, entity has wrong attributes.

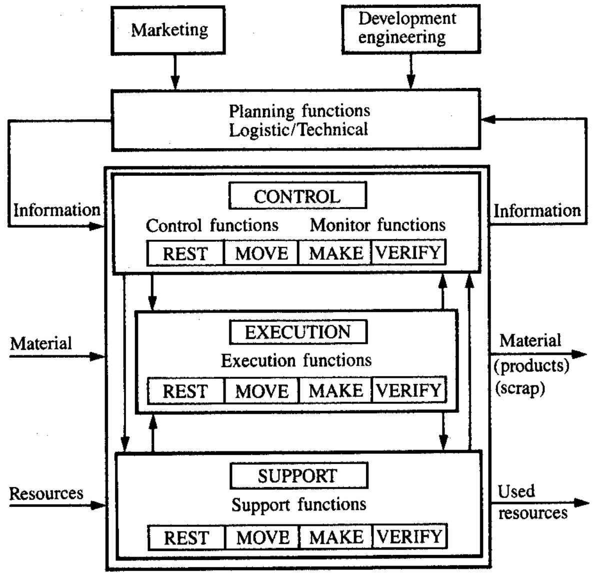

Figure 1

IMMS view of a production system.

Material is the production object of the manufacturing execution function. Material includes all the physical matter that enters the product during the production process: raw materials, parts/assemblies, auxiliary materials, products, and scrap material.

Resources are all the physical means that carry out the manufacturing process. Resources may be defined as all physical means used in the manufacturing process that are not material. Resources include process, test, transport, and storage equipment; tools and fixtures; data processing equipment; basic resources: supply material, energy, space, and time; personnel.

Manufacturing modeling functions
IMMS uses three classes of functions: *CONTROL, EXECUTE,* and *SUPPORT.*

- CONTROL functions are used to model the information flow. Examples of CONTROL functions are modification, change, generation, storage, transmission, or checking of information.
- EXECUTE functions are used to describe the materials flow. Examples of EXECUTE functions are storage, transportation, processing, assembly, or testing of materials.
- SUPPORT functions model the state of resources and describe all activities that support the manufacturing process. Examples of SUPPORT functions are set-up of equipment and provision or feeding of tools and fixtures.

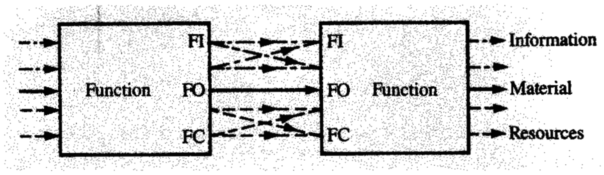

Figure 2

Horizontal connection of manufacturing functions.

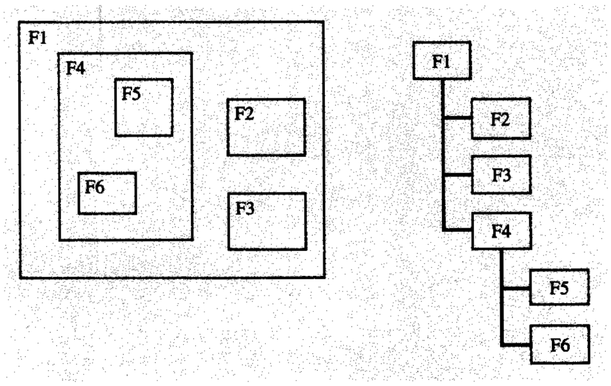

Figure 3

Vertical structure of manufacturing functions: Nesting structure and corresponding function tree.

Each class of the manufacturing modeling functions is related to a prime manufacturing object (function object) which is processed by the function. This is information for CONTROL, materials for EXECUTE, and resources for SUPPORT. However, during the manufacturing operation the manufacturing functions of different classes are interlinked by the exchange of entities. CONTROL, EXECUTE, and SUPPORT functions require information input to start their activity and they require resources to perform their activity.

Each of the manufacturing function classes CONTROL, EXECUTE, and SUPPORT is divided into four function types according to the function purpose. The purpose may be that of the original system designer or that seen by a system user or modeler. The four function types are REST, MOVE, MAKE, and VERIFY. These functions form a complete set, since all activities which cannot be identified by a MOVE, MAKE, or VERIFY function are described as a REST function.

REST means a planned storing or an unplanned waiting of function objects for the beginning of a MOVE, MAKE, or VERIFY activity. REST differs from the other three functions by the fact that function objects are not necessarily changed. In this respect REST can be viewed as a nonactivity, even though the resource space is used.

MOVE is a change of the location or spatial arrangement of function objects. Manufacturing activities in this sense are transport or handling of parts, including positioning and orientation, insertion or removal of tools into or from a tool magazine, or sorting or transfer of data.

MAKE is a change of all attributes of the function objects except those describing position, orientation, or information content. Manufacturing activities in this sense are manufacturing of parts, e.g., drill, punch, and expose, set-up of equipment, or generation of data.

The function type MAKE also includes activities that consist in assembling or disassembling different entities of the same class. Examples are assembly of parts for a product, wafer-cutting into chips, assembly of manufacturing equipment, linkage of data, assembly of wafer and substrate, or disassembly for repair.

VERIFY is any kind of information gaining about function objects and includes a comparison with reference information. Manufacturing functions in this sense are inspection, measurement, and testing.

Inspection is the examination of parts (material or resources) with respect to their physical appearance, e.g., identity, shape, or surface. Inspection may also be applied to information entities and refers here to identity, type, etc.

Measurement compares attributes of function objects, e.g., physical parameters, quantity, weight, etc., to given nominal values. Data files may be measured, e.g. for their date of origin or some other parameter.

Testing investigates how, for example, a product meets its functional specifications. Examples are the functional test of a circuit card, a drill machine, or a computer program.

Connection of functions

Entities and functions of a manufacturing system may be connected with each other in three different ways:

- Entities engage in an activity. This leads to the basic entity relation *function input/function output.*
- Functions exchange entities. They are thus connected to a precedence network which we call a *horizontal function structure.*
- A function is substituted for an entire horizontal function structure. This relation is called a *vertical function structure.*

Horizontal function structure Two functions are horizontally connected if an output entity of the first function is used by the second function as an input entity. The first function is called the predecessor, the second the successor (see **Figure 2**). Not all horizontal connections are meaningful. This is a consequence of the definition of entity classes. Material is always the function object of EXECUTE functions. It is clearly distinguished from resources and information. Resources may not be used as information, and vice versa [10].

Vertical (hierarchical) function structure A vertical relationship exists between a function and a horizontal function structure. One may be substituted for the other: i.e., the function is either replaced by the structure or the structure is replaced by the function.

A vertical structure is defined in two steps:

- First, a function is repeatedly replaced by subfunctions, giving rise to a nesting structure of all functions involved or a tree structure. A series of unique subordination (hierarchic) levels is thereby produced (see **Figure 3**).
- In a second step the substitution of entities in functions and subfunctions is defined. This leads to vertical connections between functions and their subfunctions. By this step the subfunctions of a given function or subfunction are decomposed into *internal* and *boundary* functions. Boundary subfunctions have direct connections to their superposed function, whereas internal functions are connected to functions only via horizontal connections between subfunctions (see **Figure 4**).

A generalized network of functions is shown in **Figure 5**, which contains all horizontal and vertical function connections between the prime function and first-order subfunctions. Second-order subfunctions are also indicated.

The function of the highest level, called the prime function, determines which entities are the production objects. For manufacturing systems the prime function is an EXECUTE function. The prime function normally represents a whole system. Its resources may be very complex, and the information required to operate the system is generally of considerable volume.

Consider the lowest level of subfunctions. There is certainly a level of subfunctions that cannot be replaced by lower-level subfunctions. However, such indivisible functions are too simple to be of practical use. Therefore, those functions are defined as lowest-level or *basic* functions that do not need any technical control information. In some cases these functions are very simple and require only basic resources.

The vertical structure describes the organization of functions in a system. The number of hierarchic levels is determined by the modeler and depends on the level of detail required to describe the logistic task. **Figure 6** is an example of a material flow hierarchy (EXECUTE hierarchy) from the plant level down to the basic function level with intermediate levels for manufacturing sectors and manufacturing equipment. Which function on a given level is called REST, MOVE, MAKE, or VERIFY depends on the prime task of the function. At plant level there is production

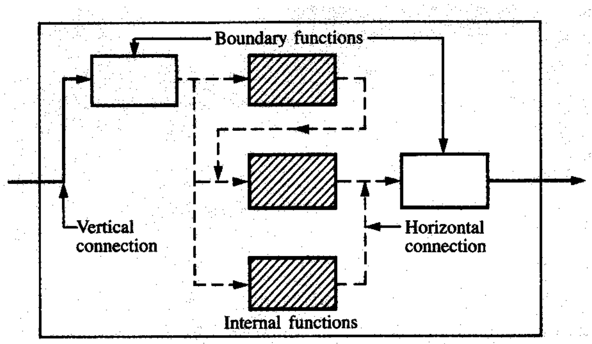

Figure 4

Function/subfunction relation.

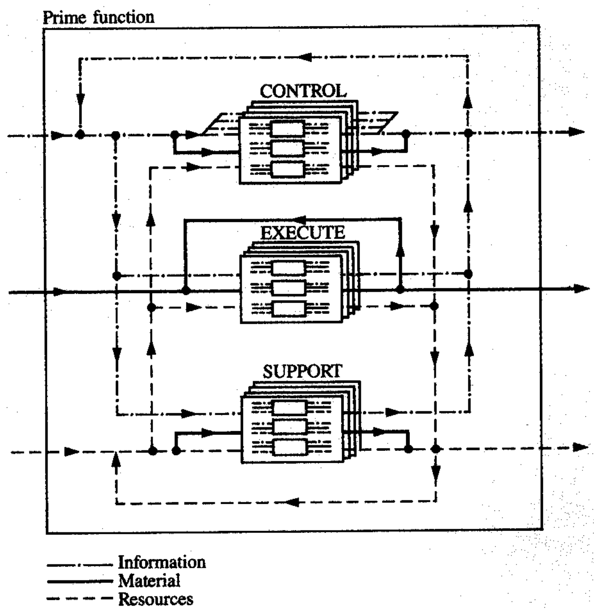

Figure 5

Horizontal and vertical function connections.

(MAKE), warehousing (REST), interplant transportation (MOVE), and quality (VERIFY). Within production there are REST, MOVE, MAKE, and VERIFY manufacturing sectors, each of which can include equipment for REST, MOVE, MAKE, and VERIFY. In almost all cases IMMS simulations will use this level as the lowest. However, the methodology permits replacement of equipment functions by their basic functions. Three stages of a material flow hierarchy are given in Figs. 14–16, shown later. They are taken from a model of a multilayer-ceramic production line.

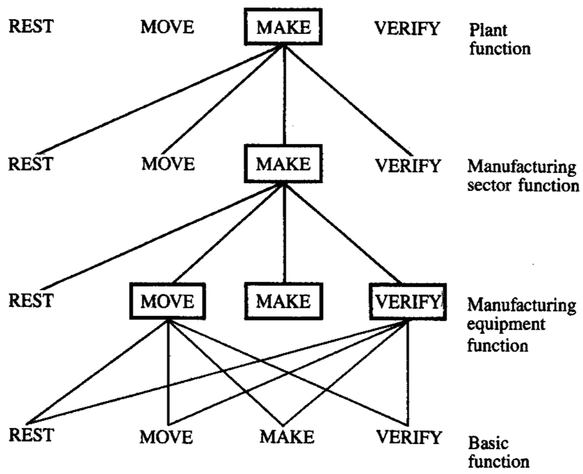

Figure 6

Material flow hierarchy. Any of the REST, MOVE, MAKE, VERIFY functions may be replaced by a set of subfunctions REST, MOVE, MAKE, VERIFY on a lower level. Here only the replacement of MAKE functions is shown.

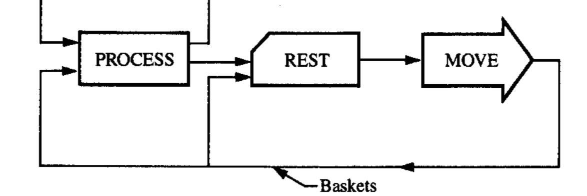

Figure 7

IMMS-1 execution functions (example of wet process line).

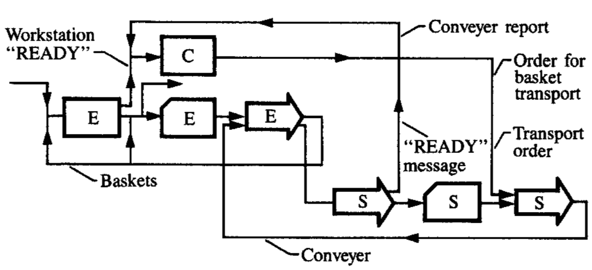

Figure 8

IMMS-2 control, execution, and support functions (example of wet process line).

Modeling features

In the Integrated Manufacturing Modeling System project two modeling packages are being developed. The first package, IMMS-1, covers the major portion of the application field, and the second package, IMMS-2, deals with more complex manufacturing models.

IMMS-1 will provide explicit modeling of the material flow while assigning data about information flow and resource flow via tables to the material EXECUTION functions (**Figure 7**). In particular, IMMS-1 will allow one to model limited space and processing capacity, resource availability, process yield, rework, product mix, job priorities, and lots and batches.

IMMS-2 will, in addition to the material flow, consider the information and resource flows (**Figure 8**). This will add to the IMMS-1 features

- Assembly of parts,
- Explicit resource behavior (e.g., set-up cycle),
- Feedback loops for manufacturing control, and
- Cascades of SUPPORT and CONTROL functions.

- *Graphic editing and experimentation*

The IMMS user will perform all modeling activities on a graphic workstation [11, 12]. The user will utilize the concepts of the Manufacturing Description Method to build or change models of the manufacturing system and to use the model during system analysis.

To ease the modeling activity, a variety of standards have been developed [13] to graphically represent the flows of information, material, and resources, to provide a coherent vertical and horizontal picture organization, and to allow fast and ergonomic user/system interaction:

- Graphic representation standard.
- Picture organization standard.
- Model building standard.
- Model usage standard.

Graphic representation standard

Symbols As shown in **Figure 9**, symbols are used to graphically represent the manufacturing modeling functions in IMMS models. They have been created in accordance with flowchart symbols used in data processing [14]. The function REST, for example, is represented by the punch card symbol. The symbols were designed to meet the following requirements [15, 16]:

- The representation of the models should be limited to their essential functions. This is the only way to keep the model in a clear and consistent form.
- The symbols used should describe the manufacturing functions clearly and completely.

- The symbols used should be obvious and easily distinguishable.

Specific information is assigned to these symbols in the form of text, numbers, bars, or curves.

Screen partitioning Since overcrowded displays make locating and identification difficult, an appropriate partitioning (segmentation) of the screen is necessary. A display image therefore is composed of at most one fourth of the available character fields on the screen, and contains up to three quarters of the dynamic information [17]. System messages are placed in the upper right corner. The communication lines as well as the virtual keyboard are in the lower part of the screen (see **Figure 10**).

Clearly arranged images are obtained by grouping. For this reason the symbols are represented in three rows and five columns. Above, below, left, and right of the symbols there are fields for inserting the data relevant to the symbol (e.g., bars, simulation values, etc.). To prevent mistakes, the relative arrangement of the insertion with reference to the symbol should invariably apply to all images of the system, e.g., scale to the left, bar to the right, digits above the symbol. Graphs and bar diagrams are always shown in the same location and in addition are marked with colors. Identifiers and representations are chosen so as to minimize subjective interpretation in taking a reading. Moreover, only unambiguous terms having the same meaning for all users are selected [18].

Analog displays (bars) allow a rapid overview. The scale has the graduation 0 5 10 or 0 2 4 6 8 10 with the decadic multiples.

Digital displays are used only for items where a precise reading is necessary, but they do not employ more than four digits. They are arranged one above the other, with the appropriate positions in alignment. The display of curves is organized in the image annotation field, curve field, curve annotation field, and information field.

Colors These are used consistently; allocation of a specific meaning to a color applies to all images in the system. However, colors are used sparingly (a maximum of seven). Any increase in this number creates a too-brightly-colored, confusing presentation ("fairground effect"). Colors are preferably used as a redundant coding aid. Whole colored areas are avoided, since they make it difficult to look for and identify certain objects. Moreover, large bright-colored areas tend to flicker [19, 20].

The following contrasting colors for the foreground/background or vice versa make symbols easier to recognize:

BLUE/WHITE	YELLOW/BLACK
RED/WHITE	ORANGE/BLACK
GREEN/BLACK	WHITE/BLACK

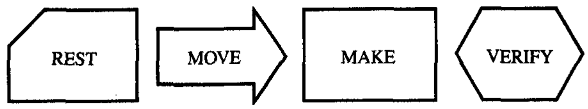

Figure 9

Symbols representing IMMS manufacturing modeling functions.

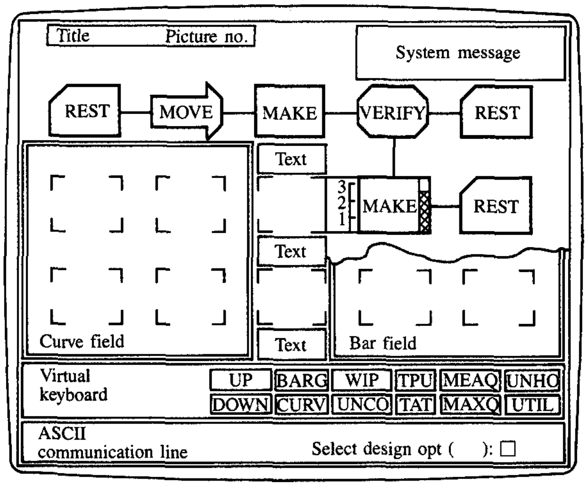

Figure 10

Screen partitioning.

Colors proposed by standards [21, 22], e.g., red for danger, green for safety, must be adopted for displaying the conditions of a system.

Annotations Horizontal annotation provides clearly legible designations. Names, terms, and information are objectively better to read in capitals than in lowercase or mixed letters. However, this does not apply in continuous text. Longer denominations are broken up into groups with spacings, e.g., AAAA 1234 BBBB.

Picture organization standard

A modeler should never get lost in a hierarchy. Therefore IMMS uses a dual-screen concept to display detail pictures on an interactive working screen and an overview picture on a separate screen (**Figure 11**). The detail that is displayed from the overview picture is marked "blue" in the overview picture. A light-pen operation allows the modeler to switch between different vertical levels and between pictures on a horizontal level. The modeling function symbols and

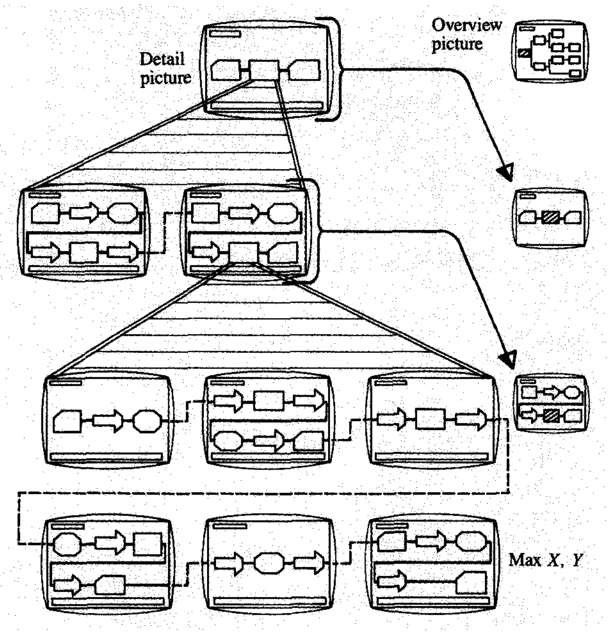

Figure 11

IMMS picture organization standard.

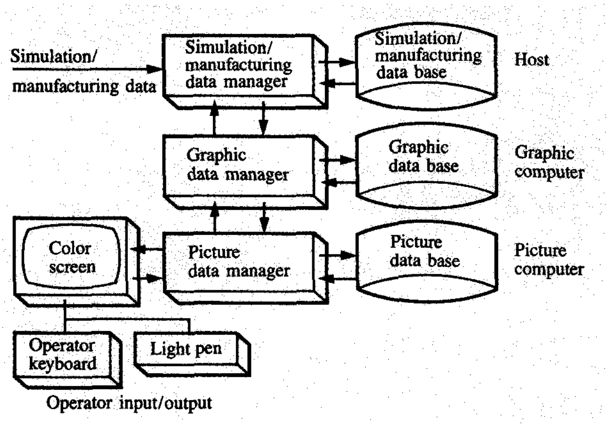

Figure 12

IMMS software configuration.

additional entry and exit symbols are used for this operation. This methodology is superior to scrolling and zooming.

It first of all takes into account the limited resolution of screens and of the human perception. By working with only one level of resolution, pictures are simply ordered and picture layouts can be standardized.

Graphic details are not hidden due to resolution as in unzoomed pictures. By organizing pictures hierarchically, searching for details is formalized. There is less danger of "blind search," as in scrolling at high magnification levels. As a major advantage, a graphic hierarchy promotes hierarchical system views.

Model building standard

The user/system interaction to build and use models is done with conventional means such as fill-in blanks, menus, system messages, and help functions. Input devices can be light pen, mouse, or keyboard.

The IMMS graphics package supports the individual operations necessary to build a hierarchical model of a manufacturing system:

- Place modeling functions from a symbol set at the wanted screen positions (light pen).
- Describe the picture: picture number, title, function names.
- Describe the functions by filling blanks in function tables (product-independent data).
- Define routing for each product, including loops.
- Define process parameters for each product.
- Link pictures horizontally and vertically.
- View the model.
- Define flow of repair, rework, and scrap parts.
- Define process yield.

Model usage standard

As in model generation, experimentation with the models is supported with standard graphic functions and tables:

- Define experimental frame parameters: sources, sinks, initial population, job priority, lots/batches.
- Define external control of function availability.
- Define run control parameters.
- Run simulation.
- View experimental frame and run control parameters.
- View simulation results.
- Document simulation results.

- *Graphic workstation*

The interactive color graphic workstation is the interface to the user, giving full access to all system functions. The user may create models and drawings, manage data input and output, and execute simulation runs.

The current workstation comprises six major components:

- Two or more monitors—video display units for viewing graphics, text, and dynamic representations, e.g., overview and detail pictures.
- A light pen for menu and virtual switch selection.
- A keyboard for alternative command entry.
- A color hard copy to provide hard copies of displayed graphics.

- A terminal for the nongraphic dialogue with the picture computer.
- The picture computer, consisting of a graphic processor for displaying pictures and a disk for picture storage.

The system's command structure uses conversational syntax, with verbs identifying requested actions.

The graphic system provides features such as overlay techniques, horizontal and vertical curve and bar representations, base and partial image techniques, and macro symbols. Image fields may be updated and can be made to blink or to change color. Complex graphic symbols including lines, text, color, and virtual switches may be stored in a library.

In addition to graphics, the system gives the user excellent programming aids and features for easy operation.

- *Data management*

The IMMS software configuration is shown in **Figure 12**. Images are designed on the graphic workstation, handled by the picture data manager, and stored in a picture data base. An intermediate computer with the graphic data manager is located between the host computer and the graphic computer. Any instructions to alter images or update them, including commands from the host computer and reactions from the virtual keyboard (change image, insert data into predetermined fields, etc.), are effected by this system. These image manipulation data are stored in image description tables in the graphic data base. In addition, the graphic data manager controls the data exchange between the host computer and picture data manager. Image description tables are created and changed by means of editor modules. Data in these tables are converted to representation instructions. This graphic data manager is written in a structured manner.

The simulation and manufacturing data are managed on the host computer and stored in a relational data base. The model generator, simulation, and data collecting programs execute on the host system.

The entire hardware and software configuration has a modular design. For this reason hardware improvements do not induce the exchange of the complete system, but only cause the replacement of single components. An essential advantage of the modular design is the inclusion of new or additional components without impairing the overall conception. Changes, improvements, or new software components can easily be integrated as modules into the overall conception.

Figure 13 shows the hardware configuration, realized as a development system to show the feasibility of the IMMS concepts.

The simulation programs, written in the simulation language RESQ2 (Research Queueing Package), run on an IBM 4341 under VM/SP. SQL/DS (Structured Query Language/Data System) has been installed as a relational data base [23, 24]. An IBM Series/1 is connected to the host computer through a synchronous, serial interface (on-line). The Graphic Data Manager, a program written in EDL (Event-Driven Language), manages the data requested by the simulation or process and stores them in the image description tables.

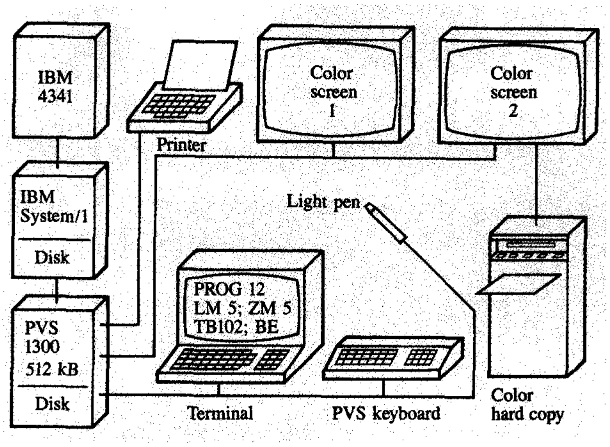

Figure 13

IMMS hardware configuration (development system).

The images are designed and represented on a Krupp-Atlas-Electronic Process Video System PVS 1300 connected in real time to the IBM Series/1 via an asynchronous interface. Peripheral devices [25] directly connected to the PVS 1300 include a display station, a typewriter, a PVS keyboard, a light pen, and two 66-cm color screens (one monitor shows the overall image, the other a detailed image). A color hard copy system which provides color drawings or overhead masters by means of RGB signals is used as an image output system. It is also possible to record images on video tape through an RGB/FBAS encoder.

- *Performance simulation*

Simulation language

IMMS uses the Research Queueing Package RESQ2 as the basic simulation language. However, the IMMS user need not have a knowledge of RESQ because he is modeling the manufacturing system with prefabricated building blocks. RESQ is hidden in the building blocks and after graphic editing of the model the IMMS user starts a model generator which automatically produces a running simulation model [10]. RESQ has a number of properties which make it very suitable for implementing the Manufacturing Modeling concept. RESQ is a high-level language which enables the construction and solution of queueing network performance models. Solutions can be obtained through discrete simulation, through numerical methods, or through a combination of both, enabling the RESQ user to trade off model detail against the computer time required to obtain performance measures from the model. Special characteristics are the following:

- RESQ uses a process-oriented world view for describing the flow of parts (jobs) within a set of queueing networks (chains).
- RESQ provides a convenient representation of system characteristics such as simultaneous resource possession, dynamic routing mechanism, and synchronization of parallel activities.
- Models may be described in a multilevel hierarchical fashion with the definition of parametric submodels to be called by a main model routine.
- RESQ facilitates graphic network representations by an explicit routing definition scheme.
- RESQ provides dynamic simulation data as different kinds of traces.

RESQ has been developed by the IBM Research Division in Yorktown Heights for performance analysis of computer systems and communication networks [26, 27]. It is available as an IBM Research Distributed Program [28].

Model building blocks

In IMMS, prefabricated building blocks are connected to a running model in much the same way as electronic modules are connected to a circuit card. If the building blocks are properly designed, the flexibility of the models exceeds that of parametric models. Building blocks are the basic Manufacturing Modeling (MM) functions, i.e., those functions that are not further resolved by subfunctions. For the implementation of the building blocks, the idea of different function layers, each representing independent or orthogonal tasks of the basic MM function, was introduced. This concept was adopted from the design of communication networks [29].

Four different layers were defined: the WORK layer, the FLOW layer, the PATH layer, and the LINK layer.

The WORK layer of a function has to do with the main purpose of an MM function. It creates/destroys entities or changes entity attributes.

The FLOW layer controls how entities flow in relation to one another. Entities may be collected into batches, i.e., groups of entities that are processed in parallel. Entities may represent lots, organizational groups of parts that are split and merged at various locations of the flow network.

The PATH layer controls the paths that entities take through a network.

The LINK layer synchronizes the activities of the MM functions of the network, which may all work in parallel.

Model generation

The RESQ source files for MM functions, sources and sinks, IMMS submodel bodies, and a main model body are kept in a library. They contain that part of the simulation programming work which can be done before an IMMS model exists. The second part of the work has to be performed by a model generator.

The model generator uses the tables created during graphic model input and the prefabricated RESQ files for MM functions and assembles them with (sub)model *body* files to a RESQ source file.

The model generator modifies the body files record by record. In particular, it

- Inserts the code for submodel invocations.
- Inserts the code for routing.
- Inserts model parameter names.
- Copies or modifies records.

It actually does the work an electrical engineer would do in order to assemble a circuit board: places modules on the card, wires the card, attaches a connector, and gives it a serial number.

The generator is implemented as a PL/I program. It is able to read model descriptions directly from a relational database.

For hierarchical models, the generator builds up an invocation tree structure and uses it for assembly of the submodels.

Measurement facilities

Performance values are provided for all MM functions. In IMMS-1, performance values are measured for each product type separately. Primary performance measures are as follows:

TAT	Mean turnaround time	(in RESQ: QT).
TP	Mean throughput	(in RESQ: TP).
WIP	Mean work in process	(in RESQ: QL).

The corresponding normalized performance values are the following:

SV	Mean serviceability (defined as the mean value of the ratio between current service time and current turnaround time).
UT	Mean utilization (defined as the mean value of the ratio between the current throughput and the current processing capacity of the MM function).
FF	Fill factor (defined as the ratio between the current work in process and the number of entity spaces or number of servers).

As in RESQ, standard deviations, maximum values, and number of departures are available, together with final queue lengths.

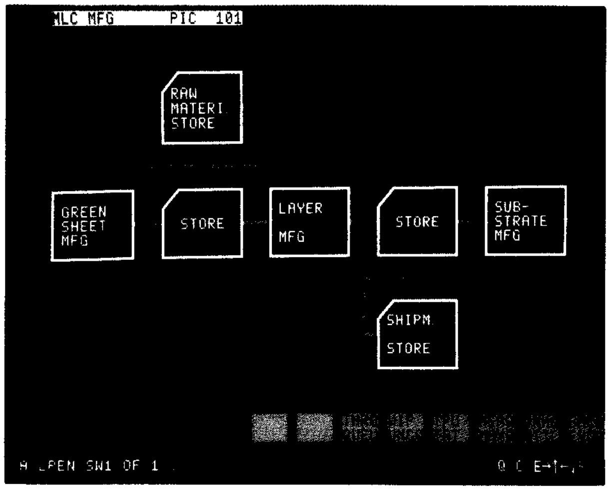

Figure 14

General image of a production line.

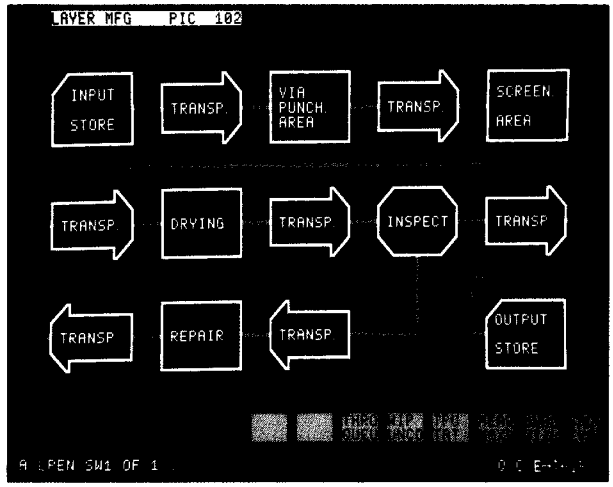

Figure 15

Detailed image of a production section.

RESQ USER-traces are provided for tracing the activities of any MM functions and to derive entity histories.

The advanced methods of RESQ for analysis of simulation output—the method of independent replications [30], the regenerative method [31], and the spectral method [32]—will also be available in IMMS.

IMMS status

• *IMMS for process monitoring and control*

A subset of the IMMS package is used for process monitoring and control applications. The hardware consists of a PVS 1300 graphics system and a Series/1 (no host computer). The IMMS software allows fast generation and manipulation of graphic models of manufacturing processes, the display of real-time process data (MONITORING), and direct interaction with the process from the graphic screen (CONTROL) [33–35]. Two pilot applications, "Compressed Air Supply" and "Heating/Ventilation/Air Conditioning," are currently running successfully.

• *IMMS for performance modeling*

Work on the development of IMMS-1 has been completed. The graphic data manager and the simulation data manager are operational, and all software links are established. A user can edit models graphically by picking building blocks from a symbol set and connecting them to a network. This information is used to set up a simulation model automatically. After parameters have been specified for the manufacturing functions and the experimental frame, simulation runs can be started from the graphic workstation. During the experimental phase the performance parameters are displayed upon request.

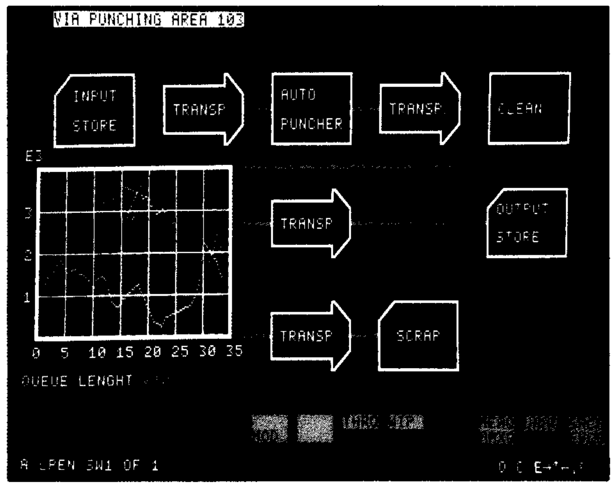

Figure 16

Detailed image of a production subset with curve set in.

At present, different alpha (first test) users are testing the system by simulating the production of computer components such as semiconductor chips, multilayer-ceramic packages, and multilayer printed circuit boards.

Example

Figures 14, 15, and **16** provide an overview of the possibilities offered by the system.

Acknowledgments

We wish to express our sincere thanks to Burghilde Wieneke of the Fraunhofer-Institut fuer Produktionsanlagen und

Konstruktionstechnik, Berlin, for her research contributions in the development of IMMS; to Dave Withers and Ed McNair of the IBM Yorktown Research Center and Wayne Oates of IBM Lexington for their valuable assistance at various stages of this project; and to Wolfgang Gruber and Wolfgang Hufnagel of Hartmann Software Inc., Stuttgart, for their contribution in implementing the software packages.

References and note

1. *Cycle Time Simulator for a Multiproduct Manufacturing Facility,* IBM Installed User Program No. 5796-PBG, 1973.
2. *SIMULAP, ein Anwendungsorientiertes System fuer die Planung von Foerder- u. Lagersystemen,* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation, Stuttgart, 1982.
3. *MAP/1, Modeling and Analysis Program,* Pritsker & Associates, Inc., P.O. Box 2412, West Lafayette, IN 47906, 1983.
4. R. W. Bailey, *Human Performance Engineering: A Guide for System Designers,* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982.
5. W. E. Beregi, "Architecture Prototyping in the Software Engineering Environment," *IBM Syst. J.* **23**, 4–18 (1984).
6. J. L. Peterson, "Petri Nets," *Comput. Surv.* **9**, 223–252 (1977).
7. D. T. Ross, "Structured Analysis," *Proceedings, International CAM Congress,* CAM-I Inc., Arlington, TX, 1974, pp. 388–421.
8. *Introduction to IDEF*, SofTech, Inc., Waltham, MA, 1979.
9. G. A. Blaauw, "Computer Architecture," *Elektronische Rechenanlagen* **14**, 154–159 (1972).
10. H. Engelke, J. Grotrian, C. Scheuing, A. Schmackpfeffer, and B. Solf, "Structured Modeling of Manufacturing Processes," *Proceedings of the 16th Annual Simulation Symposium*, Tampa, FL, IEEE Computer Society Press, Silver Spring, MD, 1983, pp. 55–68.
11. J. D. Foley and A. van Dam, *Fundamentals of Interactive Computer Graphics,* Addison-Wesley Publishing Co., Reading, MA, 1982.
12. H. Freeman, *Interactive Computer Graphics*, IEEE Computer Society, Long Beach, CA, 1980.
13. C. Scheuing and B. Solf, "Graphics for Manufacturing Planning, Control and Monitoring," *Proceedings of the CAMP '83, Computer Graphics Application for Management and Productivity,* VDI (Verein Deutscher Ingenieure) Verlag, Berlin, 1983, pp. 49–66.
14. W. R. Mullee and D. B. Porter, "Process-Chart Procedures," *Industrial Engineering Handbook*, McGraw-Hill Book Co., Inc., New York, 1963, pp. 2.20–2.41.
15. "Sinnbilder fuer Zubringefunktionen," *VDI 3239,* VDI Verlag, Duesseldorf, 1966 (in German).
16. "Materialfluss-Untersuchungen," *VDI 3300,* VDI Verlag, Duesseldorf, 1973 (in German).
17. M. M. Danchak, "CRT Displays for Power Plants," *Instrum. Technol.* **10**, 29–36 (1976).
18. E. R. Tufte, *The Visual Display of Quantitative Information,* Graphics Press, Cheshire, CT, 1983.
19. R. E. Christ, "Review and Analysis of Color Coding Research for Visual Displays," *Human Factors* **17**, 542–570 (1975).
20. P. J. Robertson, "A Guide to Using Color on Alphanumeric Displays," *Technical Report TR. 12.183,* IBM UK Laboratories, Hursley, 1979.
21. *Human Engineering Design Criteria for Military Systems and Facilities,* Publication No. MIL-STD-1472 A, U.S. Department of Defense, Washington, DC, 1970.
22. "Kennfarben fuer Leuchtmelder und Druckknoepfe," *DIN (Deutsche Industrie Norm) IEC 73*, Beuth Vertrieb, Berlin, 1978 (in German).
23. *IBM Program Product: SQL/Data System: General Information,* Order No. GH24-5012-3, available through IBM branch offices.
24. *IBM Program Product: SQL/Data System: Concepts and Facilities,* Order No. GH24-5013-2, available through IBM branch offices.
25. J. Encarnaçao and E. G. Schlechtendahl, *Computer Aided Design,* Springer-Verlag New York, Inc., New York, 1983.
26. Charles H. Sauer, Edward A. MacNair, and Silvio Salza, "A Language for Extended Queueing Network Models," *IBM J. Res. Develop.* **24**, 747–755 (1980).
27. We-Min Chow, Edward A. MacNair, and Charles H. Sauer, "Analysis of Manufacturing Systems by the Research Queueing Package," *IBM J. Res. Develop.* **29**, 330–342 (1985, this issue).
28. Inquiries concerning the program should be addressed to E. A. MacNair, IBM Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598.
29. *IBM System Network Architecture, Concepts and Products,* Order No. GC30-3072-0, available through IBM branch offices.
30. W. D. Kelton and A. M. Law, "Analytic Evaluation of Alternative Strategies in Steady-State Simulation," *Oper. Res.* **32,** 169–184 (1984).
31. M. A. Crane and A. J. Lemoine, *An Introduction to the Generative Method for Simulation Analysis*, Springer-Verlag New York, Inc., New York, 1977.
32. P. Heidelberger and P. D. Welch, "A Spectral Method for Confidence Interval Generation and Run Length Control in Simulation," *Commun. ACM* **24**, 233–245 (1981).
33. P. Dreher, R. Grimm, and I. Hertlin, "Flexible, Hierarchical Process Monitoring and Operation of Decentralized Automation Systems," *Proceedings of the INTERKAMA Congress,* Springer-Verlag New York, Inc., New York, 1983, pp. 726–735.
34. W. A. Schumacher, "Integrated Computer Control in a Fertilizer Plant," *Chemical Engineering Progress (CEP)*, American Institute of Chemical Engineers, New York, November 1981, pp. 54–60.
35. B. Shneiderman, "The Future of Interactive Systems and the Emergence of Direct Manipulation," *Behav. & Info. Technol.* **1**, 237–256 (1982).

Received November 1, 1984; revised February 7, 1985

Helmut Engelke *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Dr. Engelke is responsible for the development of simulation methods in the Manufacturing Line Architecture Department at the IBM German Manufacturing Technology Center. Since joining IBM Germany in 1970, he has developed electron beam technology techniques such as contactless test methods and statistical methods for advanced lithography systems. For the latter project he was on U.S. assignment in East Fishkill, New York, during 1974 and 1975. He has been active in the field of finite element analysis for several years, with applications including stress analysis and thermal problems. He has also worked on geometrical problems in semiconductor lithography, developing and implementing a method for solving the open mask stencil problem. Dr. Engelke received an M.S. in physics in 1965 and a Ph.D. degree in solid state physics in 1969 from the University of Stuttgart.

Jens Grotrian *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Mr. Grotrian was responsible for the development of the Manufacturing Description Method (MDM) in the Manufacturing Line Architecture Department at the IBM German Manufacturing Technology Center. He is currently working with Bernhard Solf to develop a graphical data manager system for the IBM System/1. He joined IBM in 1978 at the Information Systems Department in Sindelfingen, where he developed software for IBM System/1 tool control applications. In 1980 he joined the Manufacturing Line Architecture Department. Mr. Grotrian received an M.S. in communication technology in 1973 and another M.S. in economic science in 1975 from the Technical University of Aachen. He studied operations research methods and gained experience in the simulation and functional description of manufacturing systems while he was at the Laboratory for Werkzeugmaschinen und Betriebslehre of the Technical University of Aachen from 1975 to 1978.

Claus Scheuing *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Mr. Scheuing is responsible for the development of graphic representation tools in the Manufacturing Line Architecture Department at the IBM German Manufacturing Technology Center. Before joining IBM Germany in 1974, he worked in a development department of Bosch in Stuttgart. Since joining IBM he has developed, designed, and built various optic and laser measurement tools. He also has been active in the field of robotics and participated in the first robot installation at IBM Germany. Mr. Scheuing is a graduate of the Fachhochschule Esslingen.

Arno Schmackpfeffer *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Since joining IBM in 1967, Dr. Schmackpfeffer has worked at the German Manufacturing Technology Center. Since 1979 he has been manager of the Manufacturing Line Architecture Department, where he is developing methods and tools to improve the productivity of manufacturing system planning and operation. In 1980/81 he chaired the Corporate Automation Subcommittee on Automation Strategy. From 1969 to 1979 he was manager of the Optics Technology Department, where he applied laser technology in high-precision micromachining, geometric measurement, and photolithography. Dr. Schmackpfeffer received a Ph.D. in physics from the Technical University of Berlin in 1967.

Walter Schwarz *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* On the IMMS project Mr. Schwarz is working to develop building blocks using RESQ2 as a simulation language. After joining IBM Germany in 1970, he worked for ten years in equipment engineering for semiconductor process and test equipment. Later he participated in product transfer from IBM Sindelfingen to Montpellier/France, the installation of the first IBM-Robot 7535, and the introduction of Value Analysis as a method for cost improvement. He joined the Manufacturing Line Architecture Department in 1983.

Bernhard Solf *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Mr. Solf is responsible for integrated data management in the Manufacturing Line Architecture Department at the IBM German Manufacturing Technology Center in Sindelfingen. Since joining IBM Germany in 1967, he has worked on control of electronic applications in the measurement area: for example, a bus interface between an IBM 1130 and an optical measurement system, development of communication interfaces between microprocessors, and optical measurement and inspection systems, as well as microprocessor hardware and software for instrument control applications. Mr. Solf developed an IBM System/370 channel attachment feature for a microprocessor software development system and worked on plant process control and monitoring applications using an IBM System/7 distributed bus system. He has been active in developing communication software interfacing IBM 4341, IBM System/1, IBM 5100, and vendor systems. At present he is working on a graphical data manager system for the IBM System/1 which controls a vendor graphic system for monitoring, control, and modeling. This work includes integration of a relational data base management system for an integrated manufacturing modeling system. Mr. Solf received an M.S. in electronics in 1967 from the Technical University of Karlsruhe.

Josef Tomann *IBM Germany, German Manufacturing Technology Center, Max-Eyth-Strasse 6, D-7032 Sindelfingen, Germany.* Mr. Tomann is responsible for the development of data acquisition methods in the Manufacturing Line Architecture Department at the IBM German Technology Center. Before joining IBM Germany in 1974, he worked for Continental Gummi in Hannover, at the U.S. Army Maintenance Plant in Boeblingen, and for Brueninghaus Hydraulics in Horb/Neckar. Since 1974 he has planned material flows for the Sindelfingen plant and has participated in planning the new Hulb/Boeblingen plant. Responsible for the technical planning of disk manufacturing in Sindelfingen, he built up the first eight-inch disk manufacturing line in IBM Germany and the Encore disk preproduction line, and also was active in the Sindelfingen-Mainz disk transfer plan. Mr. Tomann is a graduate of the Fachhochschule Ulm/Donau.