Published in Proc. 2012 Intern. Conf. on Mechatronics and Automation, Chengdu, China, August, 5-8, 2012. pp. 1536-1543

Manufacturing algebra. Part I: modeling principles and case study

Enrico Canuto[1], Manuela De Maddis[2], Suela Ruffa[2]
[1]Dipartimento di Automatica e Informatica,
[2]Dipartimento di Ingegneria Gestionale e della Produzione
Politecnico di Torino
Torino, 1 1029 Italy
{enrico.canuto, manuela.demaddis, suela.ruffa}@polito.it

Abstract - **Manufacturing Algebra provides a set of mathematical entities together with composition rules, that are conceived for modeling and controlling a manufacturing system. In this first paper, only the modeling capabilities are outlined together with a simple case study. Though the algebra is formally introduced, the scope of the paper is to familiarize the reader with the proposed methodology, and to highlight some peculiarities. To this end formulation is reduced to a minimum and no theorem is included. Among the algebra peculiarities, both manufacturing process and the factory layout are neatly defined in their basic elements, and the link between them is given. A manufacturing model (parts, operations) need to include time and space coordinates in order it could be employed by factory elements like Production Units and Control Units. This asks for the definition of event and event sequence and of the relevant discrete-event elements and operators. A further peculiarity to be clarified in the second part, is the capability of aggregating algebra elements into higher level components, thus favoring hierarchical description and control of manufacturing systems.**

Index Terms -Manufacturing Algebra, discrete-event, state equations, manufacturing systems, modelling.

I. I. INTRODUCTION

The paper deals with the problem of modelling and controlling manufacturing systems. The proposed method is based on the mathematical framework offered by the Manufacturing Algebra (MA) and on discrete-event state equations as outlined in [1] and [2] . Manufacturing Algebra has been developed at the end of nineties, and since then for different reasons, .developments and applications have slowed down. This conference may be an occasion to review the algebra fundamentals, to test them through a simple case study in view of applications e.g. to the automotive industry. The algebra has been developed to model dynamics of production processes taking place in factories using mathematical objects and operations, at different degrees of accuracy and detail. It is conceived around a few concepts:

1) Direct relation with manufacturing. Algebra elements and their semantics are related with concepts and objects found in industrial manufacturing systems (Section III).
2) Separation principle. While manufacturing model describes the materials flows and the operations that occur during the manufacturing process of each product, factory model describes the physical layout, accounting for machines, transport means and storage places. Both models are kept as separate entities, but their link is defined and formulated (Section V.A)
3) Aggregation principle. The principle states that lower-level entities can be combined to provide a reduced number of higher-level entities having the same properties of the lower-level.
4) Multilayer modeling. The manufacturing process is described at different levels of detail in an incremental way, based on the aggregation principle.
5) Hierarchical control. Control units are part of the model of the manufacturing system and are compatible with the multilayer architecture of the model. The result is a hierarchical control scheme.

Previous work on manufacturing algebra has been already published, see [3] , [4] and [5] . Here, the main concepts of the algebra are recalled and referred to a simple case study, by reducing the formal burden and avoiding formal proofs. The following problems are treated: how to describe a manufacturing process (Section IV) and how to link this description with the factory layout (Section V). How to simplify models using the aggregation principle, how to obtain a multilayer model, how to design control strategies are treated in a companion paper [19] .

Several methods for modelling and controlling manufacturing systems have been proposed. A comprehensive discussion is in [6] . Several textbooks are available: relations to and discrepancies from [7] will be outlined. The need for a flexible, self-programmable, closed-loop and distributed technique to design and implement complex manufacturing operations seems still an open problem [8] . Proposed solutions are usually in the form of software architectures and packages (for simulation, monitoring and control) [9] , empirical approaches [10] or network structures [11] , and usually do not stem from a formal and comprehensive mathematical background, unless they are based on well known formal methods such as Petri Nets [12] , or approximate decomposition techniques and Markov models. Problems of distributed scheduling [13] , parts routing [14] , service rate selection have been extensively studied, though explicit solutions can be hardly achieved [15] [16] [17] . Since efficient analytical models are hardly available, discrete-event simulation models are extensively used for analysis and design issues [18] .

Modelling capabilities are introduced through a case study (Section II) similar to those presented in [16] , [15] and [17] . The following characteristics are outlined.

1) A single workshop can perform different operations. It means that more than one finished product can be manufactured.
2) A single machine can work more than one part at a time.
3) The capacity and utilization of storage places is modeled and monitored.
4) The process is described at two levels of detail.

II. THE CASE STUDY

As a case study we consider a simple assembly plant often studied in the literature ([16] , [15] [17]). To make clear how MA tackle modelling and control problems, the plant is first described using queuing network elements and concepts. It consists of two shaping machines M_0 and M_1 , and a final assembly machine M_2 separated by buffers (Figure 1).

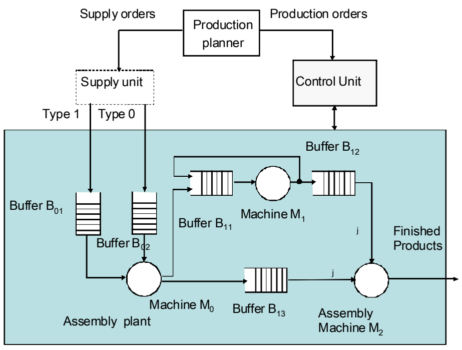

Figure 1 Block-diagram of the assembly plant.

The assembly plant can manufacture different finished products, obtained by varying the mix of β_j. items of the part type 0 and γ_j items of the part type 1, $j = 0,\ldots,3$. Leaving the mix undefined, but to be designed for optimizing production capacity, may be interpreted in different ways: (i) plant capacity has been designed for a specific component mix; (ii) finished products are actually a mix of elementary finished products which can be produced together. In summary, lot sizing is underlying this formulation. The input parts are assumed being provided by a supplier, being queued in the input buffers B_{01} and B_{02} , and then transformed by two different processes at machine M_0 . At the exit of machine M_0 , parts of type 1 are ready be assembled, while parts of type 0 need to be further processed by machine M_1 . Such a process may end into defective parts with probability α , but faulty parts are reworked on the same machine M_1 to provide the second component of the final assembly at machine M_1 . All buffers are assumed with finite capacity.

The arrival process of the input parts 0 and 1 is assumed to be synchronized with the sequence of production orders made by a production planner (automatic or manual). Each order defines the type of finished product to be manufactured and its quantity. The sequence of production orders can be completely arbitrary in types and quantities: no stationarity is assumed, also in the average.

Production orders are sent to the control unit of the assembly plant, which takes care of scheduling in real time the start time of the manufacturing operations on each machine, taking into account the buffer state. The real-time control strategy of the control unit is one of the objectives of the pair of papers [19] . It is designed to meet the following requirements.

1) The output sequence of finished products must respect the sequence of the production orders supplied by the planner.
2) Production orders must be processed as soon as possible.
3) Buffers must contain the least possible part items, in other words just the right quantity that makes feasible the next manufacturing operation.

The plant is subject to production irregularities (variability in [7]) of two kinds.

1) Micro-irregularities, described as small random variations of machine cycle times and part supply and delivery times with respect to their averages (say <±20%).
2) Macro-irregularities, i.e. rare events (preemptive or not) deeply modifying the manufacturing process, such as long machine breakdowns or long reworking of faulty parts (variability from recycle in [7]). Here for simplicity only part reworking will be accounted for, by assuming a not negligible reworking time (>100% of the typical cycle times). In this case, the control unit modifies its usual control strategy, capable of accommodating micro-irregularities, in order to manage the macro ones.

III. THE BASIC ELEMENTS

The algebra uses six elements, divided in three groups. The elements of the first group are used to build the manufacturing model, which is related to the logical sequence of the manufacturing process and describes the operations that progressively transform raw materials into finished products:

1) Objects (also articles, parts). They describe all the distinct though possible equal and interchangeable material parts that can be consumed, employed and produced during a manufacturing process. The term ‘part’ is defined in [7] meaning an object to be worked on in some workstation. Here ‘object’ is preferred, to include also parts that are not just worked on, like containers, fixtures, tools.
2) Manufacturing operations (MO for short). They describe the manufacturing stages required to transform and transport material parts. A similar concept is not defined in [7] , though the word is employed, probably because the emphasis is given to the concept of ‘job’, defined as a set of parts that traverse a factory plant together with the associated logical information, like drawings, Bill-of-Materials (BOM). MO is indeed the ‘logical information’ to be distinguished from parts (‘objects’).

The elements of the second group can be used to build the factory model, related with the physical layout and facilities of the factory. The concepts of space and of time must become explicit which leads to factory dynamics.

1) Storage units (SU). They describe factory locations where objects wait between consecutive operations. They are not defined in [7] , but are referred to as 'inventory locations'.
2) Production units (PU). They describe factory facilities capable of transforming parts. Production units are dynamic elements capable of performing only one operation at a time; they employ a finite time to complete the current operation. A similar concept in [7] is that of 'workstation ', but here 'production unit' is meant to be more generic as it may encompass transport units.
3) Resource units (RU): Describe significant and reusable resources (workers, fixtures, dies, …) which are employed by different production units in order to complete a manufacturing operation. It is not defined in [7] .

The separation between MO and PU arises from the difference between what the factory must do and where and when it must do it. It is possible that a single operation may be performed in more than one PU or that a single PU can perform several operations. A factory is defined as a network of SU, PU and RU and a factory together with its real-time control system defines a manufacturing system.

The third group is related with the real-time control and is composed of a single element, the Control Unit (CU), which is described in the second part [19] , and is not defined in [7] , though a chapter is dedicated to shop-floor control.. It describes factory facilities in charge of scheduling and dispatching operations to production units. It is assumed that all the factory and control elements modify their state only at countable time instants and, therefore, their evolution in time is described by a set of discrete-event state equations, whose fundamentals are presented in the Appendix.

IV. Manufacturing model

A. *Objects, types, quantities and events*

Let $\mathscr{A}$ be a countable set representing the universe of the objects $a \in \mathscr{A}$. A relation, the type-equivalence, makes a partition of the universe into a finite set of disjoint object classes $\mathscr{A}_n,\ n = 0,...,N-1$, called object types (this concept seems absent in [7]). Let $A \subset \mathscr{A}$ be a set of objects. A type-equivalence partitions the set A into N disjoint sub-sets $A_n \subset A$ whose cardinality is denoted by $q(n)$ and called the quantity of the (part) type n in the set A . The integer n is referred to as the part (type) number. The vector $\mathbf{q} \in \mathscr{Q} \subset \mathscr{I}^N$ collects the quantities of all types and is called the quantity vector of the set A . Very often integer quantities $q(n)$ may be expressed as a ratio $q(n)/\underline{q}$ with respect to a nominal or maximum quantity $q_{\max}$, as for instance $q(n)$ objects out of a container of capacity $q_{\max}$. Thus it seems natural extending the quantity space to be the rational space. Two types of random quantities can be defined.

1) An independent random quantity $\mathbf{q}(S)$ is a (right) stochastic matrix $S,\ N \times \Pi$ defined by

$$\sum_{\pi=0}^{\Pi-1} S(n,\pi) = 1 , \qquad (1)$$

2) The quantity $q(n)$ is random extracted independently of the other types. A joint random quantity $\mathbf{q}(P)$ is the probability matrix P satisfying

$$\sum_{n=0}^{N-1}\sum_{\pi=0}^{\Pi-1} P(n,\pi) = 1 . \qquad (2)$$

The row sum gives the occurrence probability $P(n,\cdot)$ of the type n , whereas the column sum gives the occurrence probability $P(\cdot,\pi)$ of π parts in total.

The concept of event and event sequence is fundamental in making a object set $A(t)$ capable of evolving in time. Time evolution is driven by the following mechanisms:

1) change of the object population occurs at a countable set of time instants $\mathscr{T} = \{t(i), i \in \mathscr{I}\}$, called the time set, that can be bounded or unbounded;
2) objects can be added (produced, delivered) or dropped (consumed, drawn) from an object set A as arbitrary quantities.

The time sequence of their variations is described by a quantity event sequence $\sigma_q = \{e_q(i) = [t_q(i), \mathbf{q}(i)]\}$, where $e_q(i)$ is a quantity event (see the Appendix), and i the counter. Positive components $q(n,i) > 0$ describe production (or delivery) of the type n , while negative components describe drawing. The events for which no type is drawn, $\mathbf{q}(i) \geq 0$, are called delivery events; the events for which no type is produced $\mathbf{q}(i) \leq 0$ are called drawing events.

Object types can be further partitioned into a pair of super-types called consumable and reusable. Consumable refers to objects that exist for short times - between their production and consumption - but their type can be produced in very large quantities. Their production and consumption are described by delivery and drawing events. Reusable refers to factory facilities or expensive equipments, whose life time can be considered unbounded with respect to manufacturing times. Reusable types may exist in very small quantities - usually a single item - and their time evolution is a sequence σ_w of quantity events called start and end events, $e_w(i) = [t_w(i), \mathbf{w}(i) = \pm 1]$ marking employment $w(n,i) = -1$ or release $w(n,i) = 1$ of the type n . Unfortunately the term 'consumable' has in [7] a rather different meaning, as those materials like chemicals, that are employed and consumed at workstations, but do not become part of the product to be sold. As such they are not listed in the BOM. Materials of this kind are modelled by MA as 'reusable objects' that have a finite life time, like tools that wear out.

B. *Manufacturing operations*

A way of describing a manufacturing process is to provide the list of the operations taking part to the process, as input/output representations; that is by listing the types and their quantities needed to perform each operation and the types and their quantities produced by each operation. In other terms, the Bill-of-Materials of the operation defines the operation itself. Although no explicit description of time is given, precedence relations are made explicit since input objects must be available before output objects are produced.

A manufacturing operations m is therefore described by a pair of quantity vectors $(\mathbf{u}(m), \mathbf{y}(m))$, where $\mathbf{u}(m)$ is the vector of the consumed objects, and $\mathbf{y}(m)$ enumerates the produced quantities. A set of operations is denoted by $\mathcal{M} = \{0,...,m,...,M-1\}$. A generic set of operations can also be denoted with m_j, $j = 0,1,...,J-1$. A stochastic operation is defined to have either random input or output quantities or both quantity vectors as random.

Input/output representations can be given graphical symbols: an operation is a box, object types are circles. For each operation, arrows are drawn from input types to the left side of the box, and from the right side to output types. If the consumed/produced quantity of an object type is zero, no arrow is drawn; in the unitary case a plain arrow is drawn, whereas the quantity is marked over the arrow if greater than one. A simple example is shown in Figure 2, which provides the following information: four units of the type 0 are used by the operation $m = 0$ to produce one unit of the type 2.

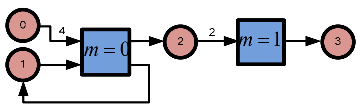

Figure 2 Graphical input/output representation of operations.

A more compact form of the input/output representation can be obtained through the balance vector. A balance vector $\mathbf{b}(m) = \mathbf{y}(m) - \mathbf{u}(m)$ specifies the order between input and output types through the sign of the quantities, which is negative for input and positive for output ones. Balance vectors allow linear algebra application, but they miss the precedence property of the input/output representation, since any object (referred to as reusable) having the same quantity in $\mathbf{u}(m)$ and $\mathbf{y}(m)$, will have zero quantity in $\mathbf{b}(m)$. This is the case of the type 1 and the operation $m = 0$ in Figure 2.

A balance matrix B is the ordered collection of the balanced vectors of the operations set $\mathcal{M}$

$$B = \left[\mathbf{b}(0) \quad \dots \quad \mathbf{b}(m) \quad \dots \quad \mathbf{b}(M-1)\right], \qquad (3)$$

and if $\text{rank}B = M$, the columns of B become a basis of the quantity space $\mathcal{Q}$. Moreover, the operations in $\mathcal{M}$ can be referred to as linearly independent, and called elementary operations. The basis B is such to partition the object types into four classes: (i) raw materials such that $B(n,m) < 0$, (ii) finished products (end items in [7]) such that $B(n,m) > 0$, (iii) semifinished products (the term subassembly in [7] seems rather restrictive) such that $B(n,m)$ may be positive and negative , (iv) reusable objects (of a single operation) such that $B(n,m) = 0$. By reordering the rows of $\mathbf{b}(m)$, the matrix B can be partitioned into four sub-matrices, B_r of the raw materials, B_s of the semifinished products, B_f of the finished products, $B_u = 0$ of the reusable types:

$$B^T = \left[B_r^T \quad B_s^T \quad B_f^T \quad 0\right]. \qquad (4)$$

C. *Manufacturing model of the case study*

The model of the manufacturing process of the assembly plant in Figure 1 consists of $m = 8$ elementary operations listed in Table I. Four assembly operations, $m = 4 + j$, $j = 0,\dots,3$, have been defined, each one yielding a different finished product. Connections between operations and object types are shown in the block-diagram of Figure 3. Each operation has been already linked to the Production Unit (to be defined in Section V) where it can be performed. Production Units correspond one-to-one to machines in Figure 1. Arrows entering or leaving a circle carry the quantity that is either delivered or requested by the linked operation. Some quantities are integer numbers; those denoted with letters are rational numbers less than 1.

Operations $m = 2$ and $m = 3$ are defined as stochastic operations since their output type can be different according to some probability distribution. They can produce type 5 with probabilities $1-\alpha$ and $1-\varepsilon$, respectively, and defective parts 4 or 6 with the complementary probabilities α and ε. The output vectors are probability vectors as in (2), and are reported in the form of balance vectors in (5). The assembly operations $m = 4 + j$ are parameterized by the fractional quantities β_j, γ_j and δ_j to be designed in the companion paper [19] . Each type which is input/output of an operation must be unambiguously defined; in other terms, different parts should belong to the same type n if they are interchangeable during the production process. The elementary operations in Figure 3 are listed in Table I, together with data (cycle time, manufacturing time, number of events) to be defined in Section V. Times are given in arbitrary units. Their standard deviation (micro irregularities) is reported within brackets. A 10% standard deviation has been assigned to cycle times, whereas a constant deviation has been assigned to manufacturing times being defined by the last delivery event.

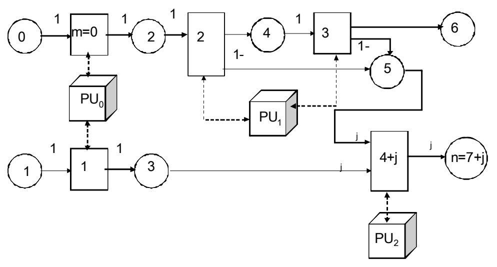

Figure 3 Block-diagram of the manufacturing process.

TABLE I ELEMENTARY MANUFACTURING OPERATIONS

MO m	Cycle time	Operation type	Manufacturing time	Number of events	PU p
0	10 (1)	Shaping	20 (1.5)	4	0
1	20 (2)	Shaping	31 (1.5)	4	0
2	36 (3.3)	Shaping, stochastic	52 (1.5)	4	1
3	80 (8)	Reworking, stochastic	110 (1.5)	4	1
4	75 (7.3)	Assembly	102 (1.5)	5	2
5	60 (6)	Assembly	85 (1.5)	5	2
6	90 (9)	Assembly	125 (1.5)	5	2
7	90 (9)	Assembly	125 (1.5)	5	2

Operations in Figure 3 and Table I employ and produce 11 types that are listed in Table II. Types are classified into raw materials, semifinished and finished products as it follows from the balance matrix in (5).

TABLE II OBJECT TYPES

Object type n	Object class
0	raw material
1	raw material,
2	semi-finished product,
3	semi-finished product,
4	semi-finished product, defective
5	semi-finished product,
6	finished product, discarded
7 to 10	finished product

The balance matrix B is reported in (5) leaving zero entries in blank. The matrix rows are partitioned into the first three sub-matrices of (4), corresponding from top to bottom to raw materials, semifinished and finished product. The rank is full, marking the columns of B as basis vectors and the relevant operations as elementary in $\mathcal{M}$.

$$B=\begin{array}{c|cccccccc|c}
\text{MO} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & \text{Type} \\
 & -1 & & & & & & & & 0 \\
 & & -1 & & & & & & & 1 \\
\hline
 & 1 & & -1 & & & & & & 2 \\
 & & 1 & & & -\gamma_0 & -\gamma_1 & -\gamma_2 & -\gamma_3 & 3 \\
 & & & \alpha & -1 & & & & & 4 \\
 & & & 1-\alpha & 1-\varepsilon & -\beta_0 & -\beta_1 & -\beta_2 & -\beta_3 & 5 \\
\hline
 & & & & \varepsilon & & & & & 6 \\
 & & & & & \delta_0 & & & & 7 \\
 & & & & & & \delta_1 & & & 8 \\
 & & & & & & & \delta_2 & & 9 \\
 & & & & & & & & \delta_3 & 10
\end{array} . \quad (5)$$

V. THE FACTORY MODEL

Manufacturing and factory models are linked together by providing manufacturing operations with space and time. To this end input/output representation must be replaced by event representation.

A. *Operations represented as an event sequence*

Event representations explicitly include the occurrence times of the actions that complete an operation, and from such a standpoint they should be defined together with the factory model. In fact the same input/output representation may give raise to different delivery/drawing times if performed by different Production Units. Input/output and balance representations are therefore simplifications.

The event representation of an operation m consists of a pair of sequences: the quantity sequence $\sigma_q(m)$ and the start/end (or activation) sequence $\sigma_w(m)$. They are typical of the pair (MO,PU), after a MO has been assigned to a PU. The quantity sequence $\sigma_q(m)$ is in turn the addition of two quantity sequences, the input sequence $\sigma_u(m)$ and the output sequence $\sigma_y(m)$:

$$\sigma_q(m)=\sigma_u(m)+\sigma_y(m). \quad (6)$$

Sequences are re-locatable, as their occurrence times are relative to a start time $t_w(m)$ that must be scheduled by a real-time control. The scheduled sequence is denoted with $\sigma_q(m,t_w(m))$. Input and output sequences are defined as

$$\begin{aligned}
&\sigma_u(m)=\left\{e_u(m,i)=\left[\tau_u(i),-\mathbf{u}(m,i)\right]\right\} \\
&\mathbf{u}(m,i)\ge 0,\ 0\le\tau_u(i)\le T_u(m) \\
&\sigma_y(y)=\left\{e_y(y,i)=\left[\tau_y(i),\mathbf{y}(m,i)\right]\right\} \\
&\mathbf{y}(m,i)\ge 0,\ 0\le\tau_y(i)\le T_y(m)
\end{aligned} . \quad (7)$$

Events e_u are drawing events from input storage units. The output events e_y are delivery events to output storage units.

The activation sequence is a start/end sequence indicating the least time interval $\tau(m)$ which is necessary to perform the operation m on a specific Production Unit

$$\sigma_w(m)=\left\{e_w(m,0)=[0,-1],e_w(m,1)=\left[\tau(m),1\right]\right\}. \quad (8)$$

Re-locatable times $\tau_u(i)$ and $\tau_y(i)$ in (7) and $\tau(m)$ in (8) are used by production control to synchronize the different operations of a manufacturing process. Two of them deserve attention.

1) The manufacturing time indicates the time duration of an operation and is defined by

$$T(m)=\max\left\{T_u(m),T_y(m)\right\}. \quad (9)$$

In [7] a similar time is referred to as 'cycle' time: 'the average time from release of a job at the beginning of a routing until it reaches an inventory point'. Here we prefer the alternative meaning (cited in a note of [7]) of the 'time allotted for each station to complete its task' (MO here).

2) The working time $\tau(m)$ which is defined in (8), may be also referred to as the cycle time of m when it remains constant and the same operation m is repeated (in series) on the same PU. In this case, the inverse $\pi(m)=\tau^{-1}(m)$ defines the production rate (or throughput in [7]) under steady conditions. In general working times are much shorter than manufacturing times especially for complex operations. Think of a car factory that assembles 1000 vehicles each 8-hour shift, corresponding to a cycle of about 30 s. Were the factory employing four 8-hour shifts to complete each vehicle from first operation to final delivery, the manufacturing time would amount to 32 hours. Capacity in [7] is the upper limit of the throughput.

The input/output quantity sequences of m can be shrunk to input/output vectors $(\mathbf{u}(m),\mathbf{y}(m))$ by summing their facts as

$$\begin{aligned}
\mathbf{u}(m)&=\textstyle\sum_i\mathbf{u}(m,i) \\
\mathbf{y}(m)&=\textstyle\sum_i\mathbf{y}(m,i)
\end{aligned} . \quad (10)$$

B. *Factory elements*

1) *Storage and resource units*

The elements of the factory models are dynamic elements, since they evolve in time according to some state equation driven by input signals.

A storage unit $s = 0,...,S-1$ is a dynamic operator used to represent factory places where objects can be stored. The state of the storage units s is a quantity vector $\mathbf{x}_s(s,t)$. Given a finite set of quantity sequences

$$\sigma_q(\pi) = \{e_q(\pi,i) = [t_q(\pi,i), \mathbf{q}(\pi,i)]\}, \tag{11}$$

indexed by $\pi = 0,1,...,\Pi(s)-1$, a storage unit can be modelled as an event adder (see the Appendix) which is forced by the addition of the sequences (input sequences) and provides as output the staircase time function $\mathbf{x}_s(s,t)$

$$\mathbf{x}_s(s,t) = \Sigma\left(\sum_{\pi=0}^{\Pi(s)-1} \sigma_q(\pi)\right). \tag{12}$$

A similar model applies to resource units $r = 0,...,R-1$, as they can be kept as the location of reusable types. In this case input sequences are start and end sequences $\sigma_w(\pi) = \{e_w(\pi,i)\}$, and the state $\mathbf{x}_r(r,t)$ is the output of an event adder as follows

$$\begin{aligned} &\mathbf{x}_r(r,t) = \Sigma\left(\sum_{\pi=0}^{\Pi(r)-1} \sigma_w(\pi)\right) \\ &\sigma_w(\pi) = \{e_w(\pi,i) = [t_w(\pi,i), \mathbf{w}(\pi,i) = \{-1,1\}]\} \end{aligned}. \tag{13}$$

The capacity of each storage unit is usually bounded by some linear inequality.

2) Production unit

A production unit $p = 0,...,P-1$, is a dynamic operator used to represent places or machines of the factory where manufacturing operations are carried out. A set of admissible operations $\mathcal{M}(p) \subset \mathcal{M}$ is associated to each PU p. A PU has an input-output dynamic model, which can be represented by a state equation. Start from the input-output sequences.

1) The input sequence is the sequence $\sigma_c(p)$ of the command events

$$\begin{aligned} &\sigma_c(p) = \{e_c(p,j) = [t_c(p,j) = t_w(m,j), m(p,j)]\} \\ &m(p,j) \in \mathcal{M}(p) \end{aligned}. \tag{14}$$

scheduling the start time $t_w(m,j)$ of the operation m. The counter j keeps record of the event sequence.

2) The output sequences include the sequence $\sigma_w(p)$ of the start/end events

$$\sigma_w(p) = \{e_w(p,i) = [t_w(p,i), \{-1,1\}]\}, \tag{15}$$

and the quantity sequence $\sigma_q(p)$ of the drawing and delivery events imposed by the commanded operations.

The forced response $\sigma_x(p, e_c(j))$, $x = q, w$, to an input event $e_c(p,j)$ in (14) that has been scheduled the MO m, is the time shift (see the Appendix) of the relocatable sequences of m in (7) and (8), i.e.

$$\begin{aligned} &\sigma_w(p, e_c(p,j)) = \sigma_w(m(p,j), t_w(m,j)) \\ &\sigma_q(p, e_c(p,j)) = \sigma_q(m(p,j), t_w(m,j)) \end{aligned}. \tag{16}$$

The forced response to the whole command sequence $\sigma_c(p)$ follows by adding the sequences in (16) over the counter j. Figure 4 shows the block-diagram of the input-output dynamics, including an output (static) function distributing the quantity sequence σ_q to different (input/output) storage units. By passing the activation sequence σ_w through an adder, provides the activation state $x_w(p,t)$ marking whether the PU is busy, $x_w = 0$, or not, $x_w = 1$.

The core of the PU model is a discrete-event dynamics converting a command sequence into output sequences. The core consists of two event delays (see the Appendix) where the future (potential, pending) events of the scheduled MO are registered in a queue and then made to occur (if not changed or cancelled).

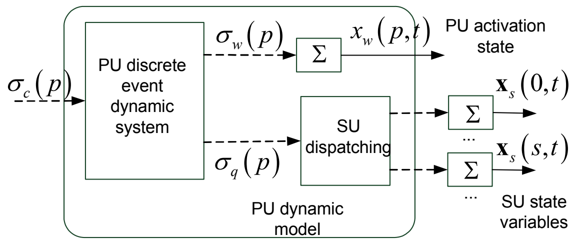

Figure 4 Block-diagram of the PU input-output dynamics.

A pair of state variables are defined.

1) The first state variable, denoted by $z_p(p,t)$ and called capacity state (in agreement with 'capacity' as defined in [7]) , is the queue of the future start-end events. It must be distinguished from the activation state $x_w(p,t)$ in Figure 4.
2) The second state, denoted by $x_p(p,t)$, is the program state, i.e. the queue of the quantity events generated by the commanded operations. Each command event $e_c(j)$ in (14) updates, at time $t = t_c(j)$, the event queue $x_p(p,t)$ into $x_{p+}(p, t_x(k) = t_c(j))$ through a static function G converting $e_c(j)$ into $[t_x(k), x_{p+}(p, t_c(j))]$, where $t_x(k)$ is the sequence of times when x updates. The counters k differs from j, so as to account for events of the queue that occur and drop from the queue itself (free response). The fact $x_{p+}(\cdot)$ of the latter event is registered by the delay, implying $x_p(t)$ jumps to the new value x_{p+}. Since x_{p+} is a queue of future events, the first occurrence time of the queue defines $t_x(k+1)$, unless a new command event has occurred meanwhile. State equations are omitted, but Figure 5 shows the block-diagram of the dynamic system in Figure 4. The concept that future events are the state variable of discrete-event state equations is the key asset of the formulation and the basis for design, simulation and control [2] .

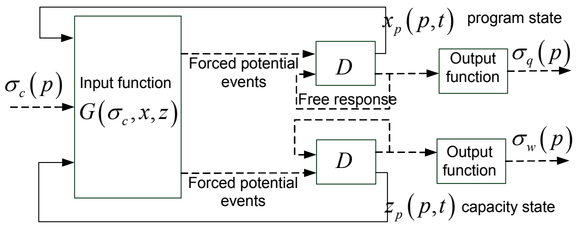

Figure 5 Block-digram of the PU dynamics.

C. *Factory model of the case study*

The event sequences of the different MOs in Table I are reported in Table III. The triple (relocatable time, quantity, type) is indicated in the case of quantity events. The undetermined quantities β_j, γ_j and δ_j in Figure 2 have been given numerical values descending from the design in the second part [19] . To separate between drawing and delivery events, the cells of the drawing events are grey coloured.

TABLE IIIEVENT SEQUENCES OF MOS

MO	Start	End	Drawing	Drawing/ Delivery	Delivery
0	(0,-1)	(10, 1)	(1, -1, 0)	None	(20, 1, 2)
1	(0, -1)	(20,1)	(0.5, -1, 1)	None	(31, 1, 3)
2	(0, -1)	(36, 1)	(0.5, -1,2)	(52, α , 4)	(52, $1-\alpha$, 5)
3	(0, -1)	(80, 1)	(1, -1, 4)	(110, ε , 6)	(110, $1-\varepsilon$, 5)
4	(0,-1)	(75,1)	(0.7, -2, 5)	(0.8, -3,3)	(102, 1, 7)
5	(0, -1)	(60,1)	(0.6, -2, 5)	(0.9, -2,3)	(85, 1, 8)
6	(0, -1)	(90, 1)	(0.9, -2, 5)	(1, -4,3)	(125, 1, 9)
7	(0, -1)	(90, 1)	(0.9, -1, 5)	(1, -3,3)	(125, 1, 10)

The factory plant includes three production units: PU$_0$, PU$_1$ and PU$_2$ corresponding to the machines M$_0$, M$_1$ and M$_2$ in Figure 1. Buffers in Figure 1 are described by a pair of storage units, SU$_0$ is the input unit encompassing B$_{01}$ and B$_{02}$. The three intermediate buffers B$_{11}$, B$_{12}$ and B$_{13}$ are collected into SU$_1$. A third storage, SU$_2$, accounts for the output buffer where finished products and discarded types are stored. It is possible to aggregate more buffers into a single storage unit as far as no constraint is imposed to delivery and drawing policies, and each original buffer stores a single type. The input storage is assumed to contain all the raw materials required to process the planned finished products..

Figure 6 shows the block-diagram of the factory model. control units (CU) are also indicated [19] . The whole factory can be aggregated into a single aggregated PU, denoted with $\underline{\text{PU}}_0$ and explained in the second part [19] .

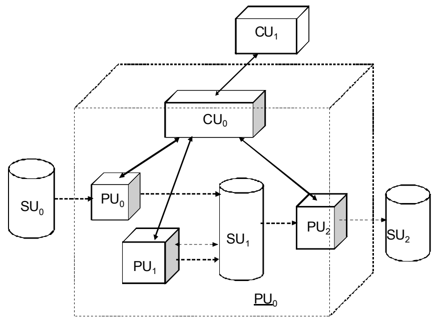

Figure 6 Block-diagram of the factory model.

VI. CONCLUSIONS

The method for modelling manufacturing systems based on the Manufacturing Algebra has been outlined together with a simple case study in which the basic concepts of the algebra have been put in evidence. Special attention has been given to a description of the elements, starting from the model of a manufacturing process (types and MO) and then proceeding to the factory model, as the latter requires that the MO are modelled through event sequences. To this end the fundamentals of event sequences have been summarized in the Appendix. Last but not least, such models can be easily converted to a simulation code.

The factory real-time control must solve the inverse problem of converting production orders expressed in terms of finished products, into a sequence of operations capable of transforming raw materials into the requested quantity of finished products at the right time. Manufacturing algebra suggests to solve the problems though aggregation and disaggregation operations to be outlined in the second part.

VII. APPENDIX

A. *Events and their sequences*

Time is discrete and is defined by a numerable set of time instants $t_i = t(i) \in \mathcal{R}$, called occurrence times, i being their counter. An event $e_i = e(i)$ is defined as an ordered pair $[\text{time,fact}] = [t(i), \xi]$ belonging to an event set $\mathcal{E} = \mathcal{T} \times \Xi$. Ξ is the fact set possessing the null element 0 and $\mathcal{T} \subset \mathcal{R}$ is the time set. An event $e(i) = [t(i), \xi]$ is said to occur when the output $c(t)$ of a clock equals $t(i)$. The fact set can be either a set without algebraic structure or a set endowed with algebraic operations like addition. In the latter case, two or more events $e_1(i) = [t(i), \xi_1]$ and $e_2(i) = [(i), \xi_2]$ having the same occurrence time $t(i)$ can be added into a single event $e(i) = [t(i), \xi_1 + \xi_2]$ by adding their facts: one may say that they coalesce.

An event sequence σ is a strictly ordered set of events $\sigma = \{e(i) = [t(i), \xi(i)], i \in \mathcal{I}\}$ defined over the time set $\mathcal{T}$ and the counter set $\mathcal{I} \subset \mathcal{Z}$ of i . It means that no simultaneous events exist and all sequence events are strictly ordered by their occurrence times. The set of all possible sequences defined over $\mathcal{E} = \mathcal{T} \times \Xi$ is denoted by $\mathcal{S}(\mathcal{E})$. A sequence may be indicated by a generic element $e(i)$. A finite sequence (also queue) is defined by a finite counter set $\mathcal{I} = \{i = 0, ..., n-1\}$ and a bounded time set $\mathcal{T}$. The difference $\tau = t(n-1) - t(0)$ is called the sequence length.

B. *Sequence operations*

Algebraic operations can be defined over a sequence set $\mathcal{S}(\mathcal{E})$. The most important are:

1) Addition. Given two sequences $\sigma_1 = \{e_1(i)\}$ and $\sigma_2 = \{e_2(j)\}$ with time sets $\mathcal{T}_1$ and $\mathcal{T}_2$, their sum $\sigma = \sigma_1 + \sigma_2$ is the union $\sigma = \{e_1(i)\} \cup \{e_2(j)\}$ of the two sequences. Possible simultaneous events coalesce into a single event only when facts can be added.
2) Time-shift. A sequence may be shifted backward and forward in time. Given a sequence $\sigma = \{e(i)\}$ and a time shift T (it may positive or negative), the time-shifted sequence $\sigma' = S(\sigma, T)$ is defined by the new occurrence times $t'(i) = t(i) + T$. For instance, when the first occurrence time $t(0)$ of a sequence σ is finite, all times may be shifted to $\tau(i) = t(i) - t(0)$, in which case occurrence times $\tau(i)$ are referred to as relative times, and the sequence as re-locatable or potential, to mean that starting time has not yet been assigned an absolute instant t . In fact, it is a task of the real-time control to schedule the absolute starting time $t(m) > 0$ of each operation

$m \in \mathcal{M}$. The resulting event sequence is denoted by $\sigma(t(m))$, which can be read as "the event sequence σ starts execution at the absolute time $t(m)$".

3) (Future) restriction. Given an event sequence $\sigma = \{e(i) = [t(i), \xi(i)]\}$ and a time t, the restriction $\sigma' = F(\sigma, t)$, is the subsequence such that $t(i) \geq t$. If $c(t) = t$, the restriction $x = R(\sigma, t)$ is the subsequence of the future (potential) events at time t. The state variables of discrete-event state equations at time t are sequences (queues) of potential events.

Events and sequences may be collected into appropriate classes depending on the fact set Ξ. A class may be denoted with a letter x to be appended as a subscript to the event and sequence symbol like e_x and σ_x.

An event sequence can be converted into a staircase function and vice versa through a pair of operators: the event register $z(t) = R(\sigma)$ and the event actuator $e(i) = A(z(t))$, whose symbols are in Figure 7. Event sequences are denoted by dashed lines, piecewise functions by continuous lines. They are defined by

$$\begin{aligned} & z(t) = R\big(e(i) = [t(i), \xi(i)]\big) = \xi(i),\ t(i) \leq t < t(i+1) \\ & e(i) = A(z(t)) = z(t) = [\tau(t), \xi(t)],\ \text{if } t = \tau(t) \end{aligned} \tag{17}$$

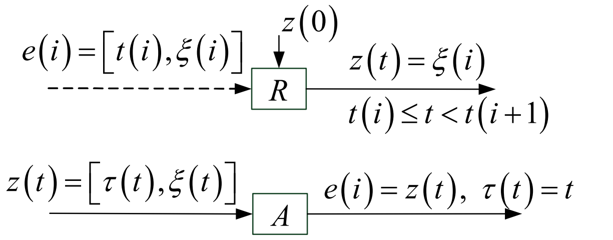

Figure 7 Symbols of register and actuator.

In practice the register generates a staircase function whose values equal the fact sequence (a fact may be an event). The event actuator takes a future event $z(t)$ and makes it to occur. The sequence of register and actuator becomes an event delay operator (see Figure 8), as it registers in a staircase state $z(t)$ an input future event $e(i+1)$ entering at time $t(i)$ until it occurs at time $t(i+1)$. The simplest event delay is an alarm clock: the input event at $t(i)$ is the alarm clock that sets the potential event $e(i+1) = [\tau(i+1), \xi(i+1)]$ to occur at time $\tau(i+1) > t(i)$.

When a fact set is endowed with addition, a sequence $\sigma = \{e(i) = [t(i), \xi(i)]\}$ can generate a constant piecewise time function $x(t)$ by progressively adding the event facts as they occur. This can be formally obtained by associating σ with an ideal pulse train

$$\sigma(t) = \sum_i \xi(i)\delta(t - t(i)), \tag{18}$$

and by integrating the train. The corresponding dynamic operator is called event adder, denoted with Σ and graphically represented in Figure 8. The event adder $x(t) = \Sigma(\sigma)$ can be represented by the following state equation

$$x(t) = x(t(i)) + \xi(i),\ x(0) = 0,\ t(i) \leq t < t(i+1). \tag{19}$$

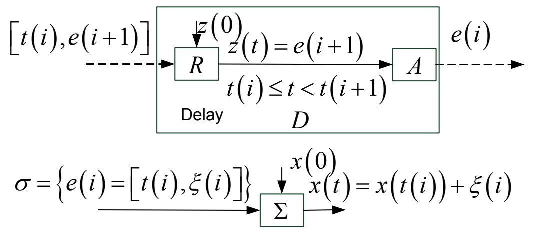

Figure 8 Symbols of event delay and adder.

References

[1] E. Canuto., F. Donati and M. Vallauri "Manufacturing Algebra: a new mathematical tool for discrete-event modelling of manufacturing systems", in *Systems Theory and Practice, Adv. Comp. Sciences*, R. F. Albrecht ed., (Springer, Wien, 1998), pp. 269-312.

[2] F. Donati, E. Canuto and M. Vallauri, "A new Approach to Discrete-Event Dynamic System Theory," *Periodica Polytechnica. Ser. Electr. Eng.*, Vol. 42, No. 1, p. 79-89, 1998.

[3] E. Canuto "Manufacturing Algebra", *Intelligent Automation and Soft Computing*, Vol. 2, No. 3, 1997, pp. 389-406.

[4] E. Canuto and F. Balduzzi "Manufacturing Algebra a formal method for modelling manufacturing systems", in *Formal Methods and Manufacturing,* J. C. Gentina, A. Giua and M. Silva eds. (Prensas Universitarias de Zaragoza, Zaragoza, 1999), pp. 109-124.

[5] F. Donati, E. Canuto and M Vallauri "Manufacturing Algebra an overview", in A new mathematical approach to manufacturing eng., *Proc. 2nd HIMAC Work.,* Vallauri M. ed. (CELID, Torino, 1998), p. 23-41.

[6] S. Stidham Jr. and R. Weber, "A Survey of Markov Decision Models for Control of Networks of Queues," *Queuing Systems*, Vol. 13, pp. 291-314, 1993.

[7] W.J. Hopp and M.L. Spearman, *Factory Physics. Foundations of manufacturing management,*2nd ed., Irwin McGraw-Hill, Boston, 2000.

[8] A. Luder, A. Klostermeyer, J. Peschke, A. Bratoukhine, T. Sauter. "Distributed automation: PADABIS versus HMS". *IEEE Trans. on Ind. Informatics.* Vol 1, Feb. 2005, pp 31-38.

[9] A. Ferrolho and M. Crisostomo "Intelligent control and integration software for flexible manufacturing cells", *IEEE Trans. on Ind. Informatics*, Vol. 3, No. 1, Feb. 2007, pp. 3-11.

[10] K. Thramboulidis. "Model-Integrated mechatronics – Toward a new paradigm in the development of manufacturing system", *IEEE Trans. on Ind. Informatics*. Vol. 1, Feb. 2005, pp 54-61.

[11] V. Vyatkin, J. Christensen, J. Martinez. "OOONEIDA: An open, object-oriented knowledge economy for intelligent industrial automation". *IEEE Trans. on Ind. Informatics*. Vol. 1, Feb 2005, pp 4-17.

[12] J.-M. Proth and X. Xie. *Petri Nets: a tool for design and management of manufacturing systems*, John Wiley and Son, 1996.

[13] J.R. Perkins, C. Humes Jr. and P.R. Kumar, "Distributed Scheduling of Flexible Manufacturing Systems: Stability and Performance," *IEEE Trans. Rob. Autom.*, Vol. 10, No. 2, pp. 133-141, 1994.

[14] R. Zhang and Y.A. Phillis "Multiple Control Policies of Two-Station Production Network with Two Types of Parts Using Fuzzy Logic" *Proc. 1998 IEEE Conf. Rob. Autom.* (Leuven, Belgium), pp. 2759-2764.

[15] H. Chen, P. Yang and D.D. Yao, "Control and Scheduling in a Two-Station Queuing Network: Optimal Policies and Heuristics," *Queuing Systems*, Vol. 18, pp. 301-332, 1994.

[16] J.M. Harrison and L.M. Wein, "Scheduling Networks of Queues: Heavy Traffic Analysis of a Two-Station Closed Network," *Operations Research*, Vol. 38, No. 6, pp. 1052-1064, 1990.

[17] L.M. Wein, "Optimal control of a two-station Brownian network," *Math. Opns. Res.*, Vol. 15, pp. 215-242, 1990.

[18] C. G. Cassandras and S. Lafortune *Introduction to discrete-event systems,* Kluwer Academic Pu., 1999.

[19] E. Canuto, M. De Maddis and S. Ruffa "Manufacturing Algebra. Part II: aggregation, control and simulation", submitted to *ICMA 2012*.