A description and analysis method for reconfigurable production systems

Filippo Boschi, Giacomo Tavola, Marco Taisch

Politecnico di Milano, Milano, Italy, {filippo.boschi, giacomo.tavola, marco.taisch}@polimi.it

Abstract. New production systems are highly reconfigurable and interact with dynamic industrial environments. Their modelling, simulation and analysis of the operations and evaluation of performances are now much more complex than in the past when system had a static and predefined behavior. This paper proposes a formalism to describe complex production systems, based on utilization of FSA (Finite Status Automaton). This approach is enabling better understanding and sharing with stakeholders of how a system works, but it is also a good basis for computer based simulation and control. The interaction with external environments is structured in terms of External Events (inputs) and Trigger Outputs. The analysis of the system status evolution provides the possibility to calculate KPIs in specific conditions or their evolution along the time. In the paper it is proposed a simplified description language to describe the automaton including output generation and triggering of other functions of the production environment. The approach is implemented and demonstrated in a particular industrial domain: industrial machinery fabrication sector.

Keywords: Cyber Physical Systems, Finite Status Automaton, Reconfigurable Production Systems, Modelling

1 The evolution of production systems towards CPS controlled environments

An aggressive market competition on a global scale and the increasing frequency of new product introductions forcing companies to continuously upgrade their production capacities and the difficulty to estimate sales forecast, due to upcoming international competitors, led to a rapid changes of traditional manufacturing paradigm [1].

As a consequence, manufacturing companies, being challenged by volatile markets and uncertain sales forecasts, are challenged in aligning their production capacities and capabilities, hence reducing the ability to match high requirements in terms of short delivery times, low stock quantities and competitive costs [1]. For these reasons, the reconfigurability and the changeability (the ability to get early and foresighted adjustments of the factory's structures and processes on all levels to market change) are seen as key aspects that the current industrial production systems have to provide for a strong competitiveness [2].

In order to face these problems, Cyber Physical Systems (CPS) could be the key enabling technology. Thanks to their ability to connect the physical part of each component involved in the production system with its virtual concept, CPS are able to create a unique environment among the data coming from the shop floor and the information concerning the overall aspects of the value chain (i.e. dynamic market demand, products' and equipment life cycle data). In this way, CPS enable to take the right decision in real time and, therefore, they guarantee a rapid integration, a fast change-over, and a ubiquitous communication assuring an agile production environment [3]. The interaction between the environment and the reactive actions taken through CPS which translates physical input events into logical ones and logical output events into physical output events, can be mapped using a descriptive automaton-based method [4] Thus, this paper proposes a model based on Finite State Automaton (FSA) able to describe the overall status and evolution of production processes that allow also to manage in real time the right information, to take the correct decisions and, therefore, to guarantee the correct degree of flexibility and reconfigurability.

2 How a FSA (Finite State Automaton) can describe a reconfigurable production system

2.1 Automaton Types (Moore/Mealy machines)

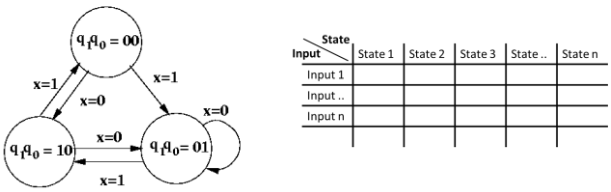
A generalized sequential logic system that can be described by a number of output (n,o) which depend on the present and the past values of the input (n,i) can be formalized as a finite state machine (FSMs) [5]. It is a mathematical abstraction where all states represent all possible situations in which the state machine may ever be. As the number of dinstinguishable situations for a given state machine is finite, the number of states is finite too . Hence, it is a behavior model composed of finite number of states, transitions between those states, and actions [6]. Such process that provides as a result the set of outputs of the machine starting from a sequence of values as input can be specified as a as a state machine (SM) by defining a 5-tuple $(\Sigma, Q, q0, F, \delta)$, as described in literature [7]:

- Σ is the set of symbols representing input to M,
- $Q = \{S1, S2, Sn\}$ is the set of states of M
- $q0 \in Q$ is the initial state which is the state at time 0 of M
- $F \subseteq Q$ is the set of final states of M
- $\delta: Q \times \Sigma \rightarrow Q$ is the transition function,

It may be the case that multiple inputs are received at various times, means the transition form the current state to another state cannot be known until the inputs are received (event driven). There are two types of finite state machines that generate output. They are called a Moore machine and a Mealy machine, named after their respective authors.. A Mealy Machine is an FSM whose output depends on the present state as well as the present input while Moore machine is an FSM whose outputs depend on only the present state.These behaviours can be described in a graphical and tabular form. The

first one, that is shown in Fig. 1, is a representation that uses, as symbols, circles and arrows that represent, respectively, the current state of the automaton and the transition from one state to another. Each transition is also described with the incoming input symbol that determines the passage of state.Within the tabular representation, the inputs are listed down on the left side, and the states are reported on the top. The table cell at the intersection of a particular row and column indicates the destination state of the FSM when the row's input is received when the machine is in the column's state.

Fig. 1. Automaton behaviour representation [7]



2.2 Status of a production systems

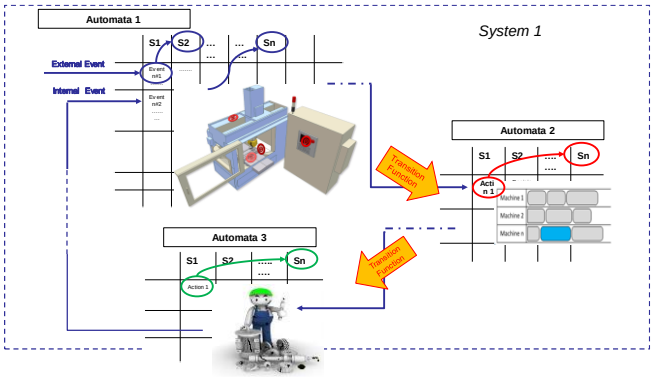
Replicating finite state-machine approach, it is possible to model and design a production system by describing each component (machine, line, shop floor or application) as an automaton. Therefore, it is needed to identify what states each system can be in, what inputs or events can trigger state transitions, and how the production system will behave in each state. In this model, the system behaviour is as a sequence of transitions that move the system through its various states [8]. From this, it is needed to identify several key characteristics of the system that can be modeled with a finite state machine:

- The system must be describable by a finite set of states and it must have a finite set of inputs and/or events that can trigger transitions between states.
- The behavior of the system at a given point in time, considered as discrete, depends upon the current state and the input or event that occur at that time.
- The system has a particular initial state
- A system is triggered by external inputs or event, or it can be triggered or it can trigger another system

A description of simple behaviour of such responsive system is described in in **Fig. 2** where 3 automata describing three subsystems interact each other. The system 1 has one initial state q0, one input set Σ, and the output set O. Both parts are built upon a set E of interactions with the environment, called events. The input is a conjunction of events and it describes a condition generated by the environment to which the system reacts. The events can be external or internal (to the global production system) depending on where they come from. During the initial phase, Automaton 1 is in S1 state. Once an external event occurs, based on instructions associated to such event, it transitions from S1 to a specific status S2. The Automaton 1 status change is the output of the transition function δ, which represents the Actions that have to be executed by the Automaton 1 and eventually cause its status change. One of the actions of Automaton 1 is

the generation of an event that Automaton 2 receives as an internal input, determining its status change. For this reason, Automaton 2 transitions from status S1 to a specific Sn status, depending on the specific event that occurs. Same reasoning can be done for Automaton 3., which will react to event coming from Automaton 2 by changing its current status.

Fig. 2. Multiple Automaton system example



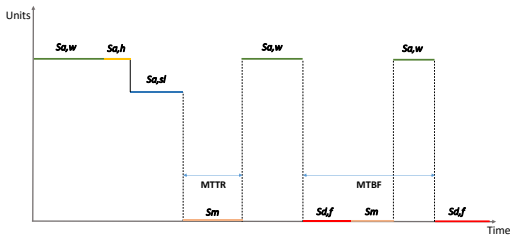
It is important to note that each Automaton can represent one or multiple similar physical components of the production system or a group of them, but it can also represent specific functional components as planning, operational or human intervention and for this reason, a production system can be represented as a combination of Automaton able to interact with each other. As a consequence, knowing the status space, the event space and the transition functions of each Automaton, it is possible to be aware a priori of the reactive behaviour of the system modelling and simulating them.

3 KPI measurements

The proposed model describes both the whole set of status in which an automaton can be and tracks the time spent by the automaton in each status. This ability is guaranteed by the fact that the automaton changes its status every time an event occurs and by the fact that the model is able to list the temporal sequence of the occurred events. In this way, the model can provide information about the status history of the automaton and about their duration. Hence, it is possible to estimate time-based performance indicators associated to each status. In fact, knowing that the time is considered as discrete, it is possible to describe some machine parameters as an event the machine is subject to, constant over time intervals and memoryless. The figure below depicts that considering, for example, the overall time spent in failure and maintenance status, an estimation of MTTR indicator for that machine can be provided. Following the same reasoning, the MTBF estimation can be obtained. Finally, knowing that machine availability depends on these values, also its estimation can be evaluated. Calculating the time spent

in not-productive status, it is possible to describe the real utilization rate of that machine. Therefore, it is shown how it is possible to estimate different kind of KPIs (i.e. OEE), starting from the analysis of an automaton status evolution in the time domain.

Fig. 3. State transition on time



4 Implementation of FSA for industrial environment on computer

Implementation of FSA is a well defined topics, for example in string parsing and regular expression matching. Adoption of FSA is also well known in monitoring and control of real-time system, where the computer operating system is able to generate (asynchronous) inputs as reaction to external (asynchronous) event collected by I/O devices (e.g. sensors).

4.1 Architecture and coding of a FSA

In order to implement a FSA for industrial purposes on a computer, the only requirement is that the operating system is able to manage asynchronous events; most of the programming languages can be adopted to code such application. The Core automaton is implemented by a process in Hibernate state that after a system and environment setup is on hold waiting to manage external inputs. A specific automaton can be instantiated many times, that is can manage multiple similar physical components (e.g. multiple machines of the same department) working in the same way, utilizing the same automaton, but evolving autonomously. It is required that each instance (each machine) is properly described by a dedicated set of data (the Context). The Context contains all the specific information describing the history and the characteristic of the specific instance; it is usually implemented utilizing a static memory area. When an event is generated, the operating system is able to associate the event to the specific indicator of the instance (the physical component) it refers to (this is called Context Pointer). The description of the automaton is usually carried out in a matrix (as described in **Fig. 1**) where for each status the automaton can have, are described the actions to take for each possible event. The actions to take are described by ***ACTIONS,*** they are portion of code implementing the operation to execute. The ACTIONS are described by a syntax composed by the following keywords:

- ACTION (ID, Action_Name): they keyword launches the execution of the procedure Action_Name passing the parameter ID as argument to identify the instance of the automaton and pointing to the associate Context. The ACTION is a portion of code with the activities required, including the physical output as printing, displaying or driving an actuator;
- NEXTSTATUS(ID): this keyword is the last of the instruction to be executed to manage the event and describe the next status the automaton is transitioning to;
- OUTPUT (AUTOMATON, ID, event): this keyword generates an event on the instance ID of the automaton.

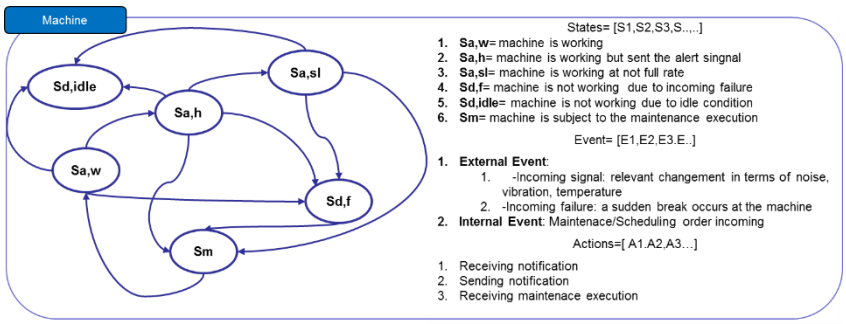
4.2 Simulation and Monitoring

The utilization of the FSA is a powerful for simulating the behavior of the physical system, just assigning an Arbitrary sets of Initial Status and generating the desired sequence of events E(t). Moreover, the memorization of the sequence of the status each instance of the automaton along the time, allows to monitor the evolution and behavior of the system, assuring the capability to asses and quantify KPIs (see chapter 3)

5 Industrial Machinery: case study

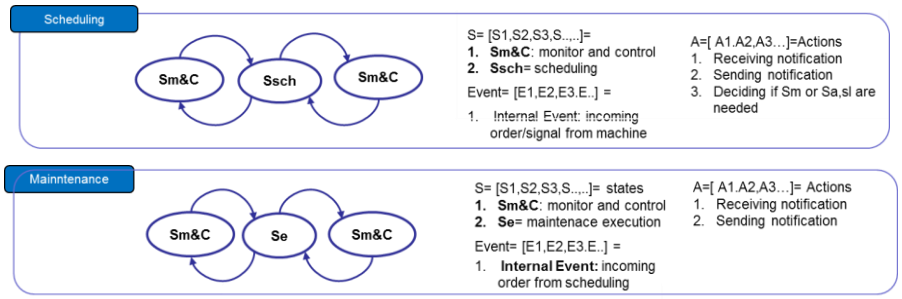
The industrial real case has been implemented in EU-project PERFoRM ([3]) and it analyzed belongs to the machinery sector. The real production environment is composed of a production line made of different machines, a common scheduling and a maintenance system. In order to model the production system, the real production environment has been simplified considering one block made of only three resources: one machine, the scheduling and the maintenance system. In this way, it is possible to model the overall production environment by replying this block every time is needed. These three resources are considered as three different Automaton, each of them having a finite number of status in which it can be, a finite number of possible events which could have an impact on those status, some transition functions and actions it could take. In the figure below, the automata describing the machine (lathe) is depicted.

Fig. 4. Lather machine status, events and actions



The lathe machine can have 6 different status and can change from one to another if some external or internal event occurs. It is defined that only 2 external events and 1 internal event can let the lathe machine status change. Please consider that the same automaton can describe (utilize multiple Contexts) multiple physical machines. The work processed by this machine is defined by the scheduling system which has 2 possible status as shown in the fig5. It can monitor and control how the lathe machine is performing (S1) or it can schedule the production for the lathe machine (S2). Only 1 event can let its status change: the arrival of a signal from the lathe machine. When this event occurs, it reacts by communicating with the maintenance system, sending it a notification and connecting it to the lathe machine which sent the signal. Also the maintenance system can find itself in 2 different states (**Fig. 5**): monitor and control and execution. It changes from the monitoring status to the execution status when scheduling systems requires its intervention on the lathe machine. Once the maintenance execution has been finished, it turns back to the scheduling system sending it a notification. In this way, thanks to this communication, scheduling system can provide a new schedule for the lathe machine, which will start working again.

Fig. 5. Scheduling & Maintenance system status, events and actions



In order to facilitate the understanding of the method, an example is provided here follow. Let's assume that at t=0 the lathe machine is working and so it is in Sa,w status. At a certain time, smart sensors embedded on it send an alert to the machine, which leaves the Sa,w status and turns to the Sa,h one. When the lathe machine is in Sa,h status, it sends a notification to the scheduling system in order to let it aware of what is happening on the production flow. The scheduling system, which goes to Sm&c status to Ssch one, can now decide what kind of actions are needed to let the lathe machine start working again. Scheduling can decide to let the machine continue working but at a low rate, sending it this order and moving it to the Sa,sl status, or can decide to stop the production and send an alert to the maintenance system. In the first case, if machine has already sent an alert signal to the scheduling system but this system decided to let machine continue working, at a certain time, it can happen that the lathe machine can fail, moving itself from Sa,sl status to Sf status. Both in the second case mentioned above and in this last case, scheduling system send an alert to the maintenance system, which turns its status from Sm&c to Se. At the same time, the lathe machine changes

its status in Sm. Once the maintenance finishes its work, it goes back to the scheduling system, which can now generate a new schedule for the repaired lathe machine.
This narrative description of the automaton can be also described utilizing matrixes as described in **Fig. 1** b.

6 Conclusions

The proposed approach allows description of the production system, regardless its complexity and the number of each components and their actual configuration; the utilization of multiple automaton synchronized each other allows to expand the domain we are describing both in terms of additional function and multiple instances of the same components. The flexibility of the approach associated with embedded ability to manage asynchronous events make very easy to integrate interaction of external systems (e.g. planning, maintenance, logistics, etc.) and the interaction with human operators. The definition of the admissible status and input event defines the boundary of the systems and its representation on the status/input matrix allows an easy and understandable analysis and communication of the way the system behaves. This can be the input for implementation of simulation systems on other platforms. Last the tracking of the transition of the status in the time domain allows to calculate and consolidate KPIs.

7 Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 680435.

8 References

[1] D. Spath, S. Gerlach, S. Schlund, "Cyber-physical system for self-organised and flexible labour utilisation," in *22nd International Conference on Production Research*, 2013.

[2] P. Nyhuis, N. Duffie, M. Brieke, "Changeable Manufacturing - Classification , Design and Operation," *CIRP Ann. - Manuf. Technol. Ann. Technol.*, vol. 56, no. 2, pp. 783–809, 2007.

[3] PERFoRM – Production harmonizEd Reconfiguration of Flexible Robots and Machinery, "http://www.horizon2020-perform.eu," 2016. .

[4] F. Maraninchi, "The Argos Language: Graphical Representation of Automata and Description of Reactive Systems," in *IEEE Workshop on Visual Languages*, 1991, p. Vol.3.

[5] J. F. Wakerly, "Sequential logic design principles," in *Digital design: principles and practices*, Prentice Hall, Ed. 2005, pp. 431–558.

[6] M. Thirumaran, "Evaluating Service Business Logic using Finite State Machine for Dynamic Service Integration," *Int. J. Comput. Appl. (0975 8887)*, vol. 22, pp. 33–39, 2011.

[7] J.Hennessy and D. Patterson, "Large and Fast: Exploiting Memory Hierarchy," in *Computer Organization and Design The hardware/software interface*, Ed. MORGAN KAUFMANN, 2013, pp. 450–548.

[8] Wright, D. R. (2012). Finite state machines. *CSC215 Class Notes. Prof. David R. Wright website. N. Carolina State Univ. Retrieved July, 14, 1-28.*.