Modelling, simulation and evaluation of energy consumptions for a manufacturing production line

Andrea Cataldo
Machine and Manufacturing Control System
Institute of Industrial Technology and Automation - National Research Council
Milan, Italy
andrea.cataldo@itia.cnr.it

Marco Taisch, Bojan Stahl
Department of Management, Economics and Industrial Engineering
Politecnico di Milano
Milano, Italy
{marco.taisch, bojan.stahl}@polimi.it

***Abstract*—The use of discrete event simulation integrated with energy consumption computation is nowadays very useful for the efficient design of an industrial production plant.**

In this paper a serial manufacturing line made by four operating machines has been modelled by means of SIMIO discrete event simulator, focusing on the production line energy consumption computation. Each operating machine has been described according to three different aspects. Firstly the mechanical behaviour, that is guaranteed from the simulator modelling functionalities itself. Secondly the control functionalities that have been designed by means of Finite State Machines (FSMs), then translated into the C# programming language and finally integrated into the SIMIO platform. Finally the energy consumptions, that have been computed by modelling the energy states associated to the different operating machine working functions.

In the end a simulation experiment has been run and the regarding results have been presented. The power absorbed and the energy consumed from the whole plant and from each single machine have been compared in order to allow the discussion about possible actions to be taken in order to respect the maximum amount of energy supplied to the plant, value that often represents a primary economic constraints for the industrial plant management.

Main limitations coming from the proposed work regard the missing of plant power and energy measurement and the level of approximation used to model the different machines energy consumption.

Keywords—Discrete event simulation; Plant energy consumption; Manufacturing plant simulation; Energy machine modelling; Energy efficiency; Finite State Machines control

I. INTRODUCTION

Considerations towards sustainability have gained importance in the public over the last decades, lastly being lighted by debates on global warming and resource depletion [1]. The manufacturing industry being responsible for ca. 35% of world-wide CO2 emissions [2] makes strong demands on energy and resource efficiency. Eco-efficiency has become a major driver for optimization and improvement programs due to increasing commodity prices, customer-awareness and legislative regulations.

Although long-term improvements should be initiated [12], coordinated and made along the supply chain [16], Small and Medium Enterprises (SMEs) typically lack the power to have impact on their supply chain partners, so that the area of improvement is identified as their own operations [13]. The major concern is the translation of the strategic goal on increasing environmental sustainability of a production system into concrete actions and steps at shop floor level [10], [11], [15]. For this reason manufacturing companies require pragmatic tools which support them in the analysis of the production system [14] and in particular in the evaluation of energy items associated to the production system management [17], aiming towards a higher environmental sustainability and improved economic perspectives at the same time.

According to the scientific efforts carried out to cope with the production system energy evaluation, and that are discussed in the proposed literature above mentioned, this paper proposes a simulation-based approach to be used for the evaluation of the energy consumption of an industrial automotive engine assembly line, which is based on SIMIO discrete event simulator [3].

We have modelled the global behaviour of each operating machine of the production line considering three different, even if complementary, aspects. At first we have described the macro-operations or the mechanical functional behaviour like milling, welding, screwing and drilling, by splitting them in micro-operations characterized by specific parameters, in particular the processing times. Then we have implemented the simplified control functionalities that must be run by the machine control system. We have used FSMs because it allows keeping simple the control code implementation by smoothly translating the FSMs algorithm into C# software code that can be easily integrated into the SIMIO simulator. In the end we have described the energy consumption of each machine by considering their energetic states and by implementing them again by means of FSMs. The so modelled energy behaviour of the machine has been translated into C#

978-1-4799-0223-1/13/$31.00 ©2013 IEEE

code to be run into the simulator, together with the mechanical behaviour and the control functionalities code.

The so built simulation model has allowed running an experiment focusing on the different machine energy consumption and consequently the whole production line one. In particular both the absorbed power and the energy consumption of the whole production line have been plotted versus time, by allowing a discussion about the maximum peaks of power required from the plant.

The paper is organized as follows. Firstly a short excursus about the energy items related to the Discrete-Event Simulation (DES) is presented. Secondly the modelling methodology adopted to build the assembly line simulation model is explained, by focusing on the different aspects that characterize the mechanical, control and energetic behaviour of each operating machine. Then a simulation experiment is discussed in order to evaluate the usefulness of such a modelling and simulation methodology approach. Finally, the simulation based approach is summarized in the conclusions and the results of the simulation experiment are evaluated together with the limitations coming from the adopted modelling method. Future resulting developments close the present work.

II. Energy in Discrete-Event Simulation

The assessment of a production system in solely economic terms is nowadays a straightforward and mature process. Key performance indicators are commonly known and accepted, and required data for analysis can be quickly gathered through factory Information Technology (IT) systems. Conducting environmental assessments is more challenging, since the impact of the production system on different dimensions and remarkable less data availability makes high demands on the assessor. An integrated assessment of environmental and economic performances requires a system-oriented approach. Here, DES is a widely accepted tool for studying and improving the behaviour of systems [4]. Commercially available discrete event simulators do have an extensive selection on analysis metrics and tools from the economic perspective, but do not consider environmental issues at the moment. In the past years simulation towards energy efficiency has gained strong interest in the research community and remarkable progress has been done until now. Several different simulators, programmed or based on commercial engines, have been developed in order to integrate environmental performances in the DES engine. A comprehensive overview can be found in [5]. Diverse studies have shown that the states of production assets have a strong impact on the total energy consumption and are an appropriate subject for studies [6]. In this paper, we will use SIMIO as a discrete event simulator to study energy consumption under certain control policies.

III. Modelling of the Assembly Line

A simple serial engine assembly line has been taken into account as case study in order to apply and verify the modelling and simulation approach aimed to the manufacturing line energy consumption evaluation. It is based on four different operating machines (M1, M2, M3 and M4), placed in sequence one to each other, see layout showed in Fig. 1.

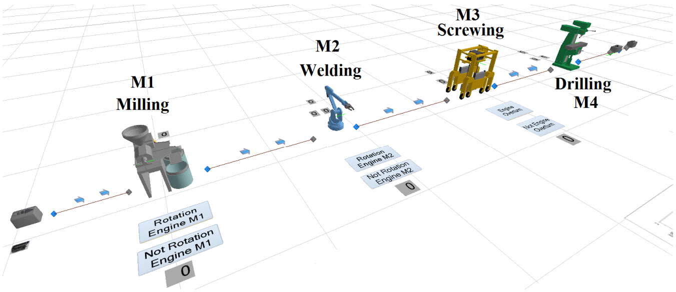

Fig. 1. Engine assembly line layout

The machines M1, M2 and M3 are characterized by two kind of mechanical operations over the engines while the machine M4 performs just only one machining operation. M1, M2 and M3 are respectively a milling, a welding and a screwing machine. The main operations, milling, welding, and screwing respectively, are executed over every kind of engine, while the second one, that is carried out to perform a spatial rotation of the machined engine, is only performed over specific kind of engines. Each machining operation of every operating machine is characterized by specific electric power requirements. M4 is a drilling machine and is characterized by only one machining operation, associated to specific electric power requirements too.

In order to implement a simulation model in SIMIO, besides the machine objects, it is necessary to add some standard library elements: a *Source*, needed to inject a sequence of *entities* into the simulated line (an *entity* corresponds to an engine in the real production line), a *Server*, able to emulate the line saturation, and a *Sink*, which destroys the *entities* exited from the modelled line.

Each machine is represented by means of the *Workstation* SIMIO standard library object, suitable to process an entry entity according to the parameters specified in the apposite user-interface mask. The most frequent parameters that have to be set are the Transfer-In time (represents the time that an *entity* takes to enter the *Workstation*), the Processing time (represents the time used from the *Workstation* to process the *entity*) and the Teardown time (represents the time that the *entity* takes to get out of the *Workstation*). In order to replace the real operating machines working function, two photo-sensors must be modelled so to simulate the detection of the presence of an engine at the input and output of the machine itself.

Since the *Workstation* object is characterized by a standard simulated behaviour that do not completely match with the real operating machine ones, it has been necessary to customize the *Workstation* module functionalities by implementing a specific software function that, in the real industrial cases, constitutes the control code running in the operating machine automation system.

Also the operating machine energy consumption feature is not available in the SIMIO standard library objects. Hence, as like as for the control software function, specific software has to be implemented.

These three behaviour aspects of every operating machine are deeply described here after.

A. *Mechanical behaviour*

The mechanical behaviour of each machine is obtained starting from the *Workstation* object, see Fig. 2.

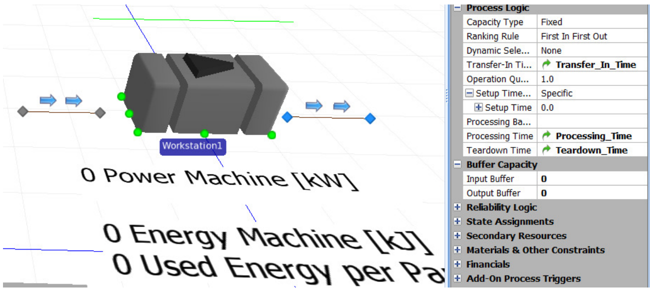

Fig. 2. Workstation SIMIO standard library module

By setting the *Transfer-In time*, *Processing time* and *Teardown time* parameters it is possible to characterize the working function behaviour of this standard library module. Instead of setting specific numeric values associated to the three parameters, it becomes more powerful to link them to specific variables in order to obtain a model as much flexible as possible. In fact, thanks to the object oriented programming principles (heredity in this case), it is possible to use the *Workstation* module as a "father or parent" class and reuse it as many times as necessary in order to get the specific operating machines modules by simply customizing their parameters.

In this way each operating machine module is able to strictly simulate the processing operation of an entered *entity*, but, for example, it is not suitable to describe the coordination among the previous and the following operating machine. In order to do that it is necessary to define and implement control functionalities.

B. *Control function behaviour*

The control functionalities of each operating machine are based on two main items. The previous one regards the presence of the photo-sensors used to synchronize each machine working function with the previous and following ones. The second item is concerning the machining functions performed by the machine.

1) *Photo-sensors*

Each machine is characterized by two photo-sensors placed both at the input (FTC_In) and at the output (FTC_Out) of the machine itself, see Fig. 3. Such sensors are used for the identification of the position of the engine that flows through the machine. In particular, as an engine crosses the input stage of a machine, then the input photo-sensor rises its associated Boolean variable at higher logical level (=1), which falls down again to the lower logical level (=0) as soon as the engine has completed the machine input crossing. From the control functionalities point of view this means that the machining operation can start.

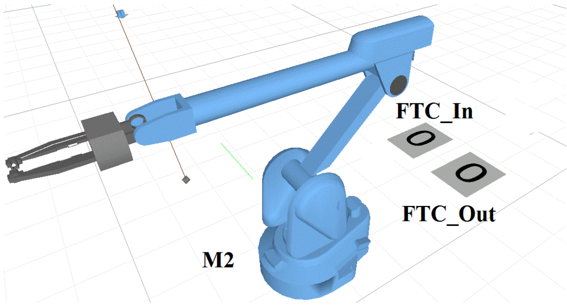

Fig. 3. Operating machine M2 photo-sensors

A similar behaviour is associated to the output photo-sensor. As soon as the engine is released from the processing section of the machine, then the output photo-sensor switches-on its associated Boolean variable at higher level (=1), which becomes again at low level (=0) as the engine has completely left the machine.

2) *Finite State Machine control algorithm*

In order to implement the machining functionalities of each operating machine FSMs have been adopted; since there are few functionalities to be implemented in the machine software module the FSM diagrams do not become too much complex to be managed. In order to support the detailed description of such methodology the machine M2 FSM diagram has been taken into account as an example, see Fig. 4.

The state S0 corresponds to the idle state of the machine in which it waits for a new engine to be machined. As soon as a new engine crosses the input of the machine, then the signal FTC_IN changes to the high logic level (=1) and the FSM switches from the state S0 to the state S1 through the regarding arc associated to which the control action Start_welding is executed, that means that the engine starts to be machined. As soon as the processing time is elapsed, such parameter is defined in the machine module user-window, then the transition between the states S1 and S2 is active and then the FSM switches from the state S1 to the state S2.

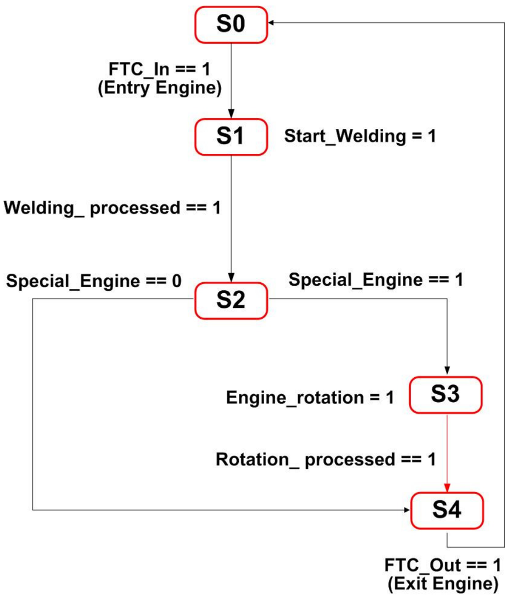

Fig. 4. Operating machine M2 FSM control function algorithm

According to the state S2 no control actions have been foreseen so the two conditions, associated to the regarding transitions from the state S2 to the states S3 and S4, are checked. In the same way, all the other states become reached so that the control actions can be performed in the right way and consequently the functionalities of the operating machine can be executed.

The control function algorithm formalized by means of the FSM is translated into C# software code in order to be executed inside the SIMIO simulator. To do that it has been used a *switch-case* instruction, as showed below.

```
//Arc Transitions
switch (M2_FSM_state_variable) {
        case 0 : {//From State S0 to State S1
                if (FTC_In == true) {
                        M2_FSM_state_variable = 1;}
        } break;
        case 1 : {//From State S1 to State S2
                if (Welding_processed == true) {
                        M2_FSM_state_variable = 2;}
        } break;
        case 2 : {//From State S2 to State S3 or State S4
                if (Special_Engine == true) {
                        M2_FSM_state_variable = 3;
                }else {
                        M2_FSM_state_variable = 4;}
        } break;
        case 3 : {//From State S3 to State S4
                if (Rotation_processed == true) {
                        M2_FSM_state_variable = 4;}
        } break;
        case 4 : {//From State S4 to State S0
                if (FTC_Out == true) {
                        M2_FSM_state_variable = 0;}
        } break;
}//switch

//Output
switch (M2_FSM_state_variable) {
        case 0 : {//Output State S0
        } break;
        case 1 : {//Output State S1
                Start_welding = true;
        } break;
        case 2 : {//Output State S2
        } break;
        case 3 : {//Output State S3
                Engine_rotation = true;
        } break;
        case 4 : {//Output State S4
        } break;
}//switch
```

An integer variable *M2_FSM_state_variable* is used to jump into the right *case* to be managed. The first part of the code has been designed to model the transitions among the states of the FSM associated to the regarding conditions. For example as the FSM has the state S0 active then the variable *M2_FSM_state_variable* is equal to 0 and the *case* 0 is selected. As soon as the condition FTC_In is true, that is the engine crosses the input of the operating machine, then the *M2_FSM_state_variable* is set to 1 that means the change of the state in the FSM. The same *M2_FSM_state_variable* value is just used in the second part of the code in which the control actions associated to each state are implemented. In fact if the FSM active state is S1 and then the *M2_FSM_state_variable* value is equal to 1, then the *case* 1 of the *switch-case* instruction is selected by executing the *Start_welding* control action. And so on for what concern all the other FSM states and transitions.

In order to integrate the C# code into SIMIO simulator the formalism called *process* must be used, see Fig. 5.

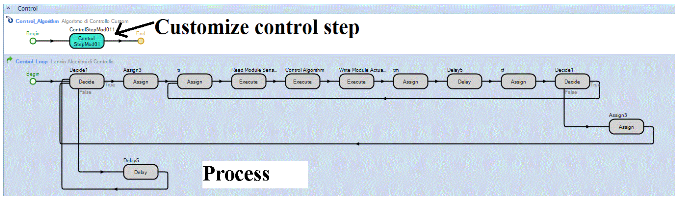

Fig. 5. SIMIO process formalism for the definition of the machine behaviour

A generic SIMIO *process* is structured by means of single *steps* in which it is possible to see a sequence of basic *steps* connected one to each other; each *step* implements a specific software function (for example variable assignment, executing parallel *processes*, managing events, writing data on files, etc.). By means of this formalism it is possible to build any logic functionality and then characterize any kind of behaviour for a SIMIO library module and then for the different operating machines. Moreover there is the possibility to design and implement customized software functionalities to be integrated into custom *steps*, provided that the customized software code is written in C# language. That explains why we have adopted the above described methodology in order to translate the FSMs scheme into C# code *switch-case* instructions.

Since it is possible to characterize whatever operating machine simulated behaviour this approach has bee exploited for the implementation of the energy behaviour of the different operating machines too.

C. Energy behaviour

The concept that has been adopted for the energy behaviour modelling and simulation of each operating machine is based on the fact that machines pass through different states in their production activity [9]. Each energetic state is characterized by different power requirements for the mix of energy carriers and could be constant or variable over time, according to the type of process and the product to be worked.

The behaviour of the operating machines follows different productive states: *off*, *starvation*, *set-up*, *failure*, *blocking* and *working*. The productive states change continuously according to discrete events which affect the machines themselves.

The productive states do not correspond one-to-one to the energetic states of the machines [7]. The *blocking* production state can correspond to an *off*, *stand-by* or *idle* state from the energy point of view. Analogue results can be deduced for the *failure* state. The same considerations can be drawn for the *set-up* state which can happen with the equipment in the *off*, *stand-by*, *idle* or *set-up* energy state.

The specific energetic states of a machine have been formalized as like as for the control functionalities: firstly by means of FSMs, then translated into C# code and finally implemented into SIMIO custom *steps*. The mathematical computation used to calculate the energetic state energy consumption values are according to [8].

In particular for the engine assembly line here described, an approximated constant power profile is associated to each of the operating machines. Starting from this power profile the computation of the energy consumption is carried out as follows: for each machine, the simulator calculates the time interval related to the specific active productive state that is used to integrate the approximated constant power load profile by allowing the calculation of the energy consumption. By associating the machine productive state and the regarding power and energy computation to a FSMs which transitions are synchronized with the FSMs control behaviour, by translating it into C#code and by integrating it into the SIMIO *processes*, the whole assembly line simulation model is then able to perform the computation of the consumed energy.

IV. SIMULATION EXPERIMENT

In order to validate the production line energy consumption modelling approach presented in this paper, a simulation experiment has been carried out. As said in the pervious paragraphs, the machines M1, M2 and M3 are characterized by two kind of machining operations while M4 by only one. Moreover, since each machine has electronic control devices which require a certain amount of electric power in all different productive states (apart the of state in which the machine is off), such consumed energy contribution has been taken into account even if it is lower compared to the machining power requirements. Such data are listed in Table I.

TABLE I. MACHINES REQUIRED POWER

Operations	[a]M1	[a]M2	[a]M3	[a]M4
Milling	4,45			
Welding		6,84		
Screwing			1,25	
Drilling				2,33
Rotation	0,75	0,75	0,75	
Minimum power	0,50	0,50	0,75	0,65

a. Power expressed in [kW]

According to the mechanical behaviour the process parameters needed for the *workstation* modules configuration are listed in the Table II.

TABLE II. MACHINES OPERATING TIME

Time	[b]M1	[b]M2	[b]M3	[b]M4
Transfer_In	1	1	1	1
Processing	10	5	2	5
Teardown	1	1	1	1

b. Time expressed in [s]

The interval time used for the mathematical integral operation has been set to 0,1 [s]. Moreover in this simulation experiment some boundary conditions have been introduced in order to highlight the machines energy consumption aspects. As already said in the previous paragraphs another SIMIO standard library object, called *server*, has been placed at the end of the production line simulation model; it is similar to the *workstation* module even if it is simpler. Its scope consists of creating a block for the entities flow. In this way the machines are forced to switch to the block productive state one after the other because they cannot discharge the machined engine (entity), so that their energy consumption falls down to the minimum rate corresponding to the minimum absorbed power value. In Fig. 6 the simulation results are plotted.

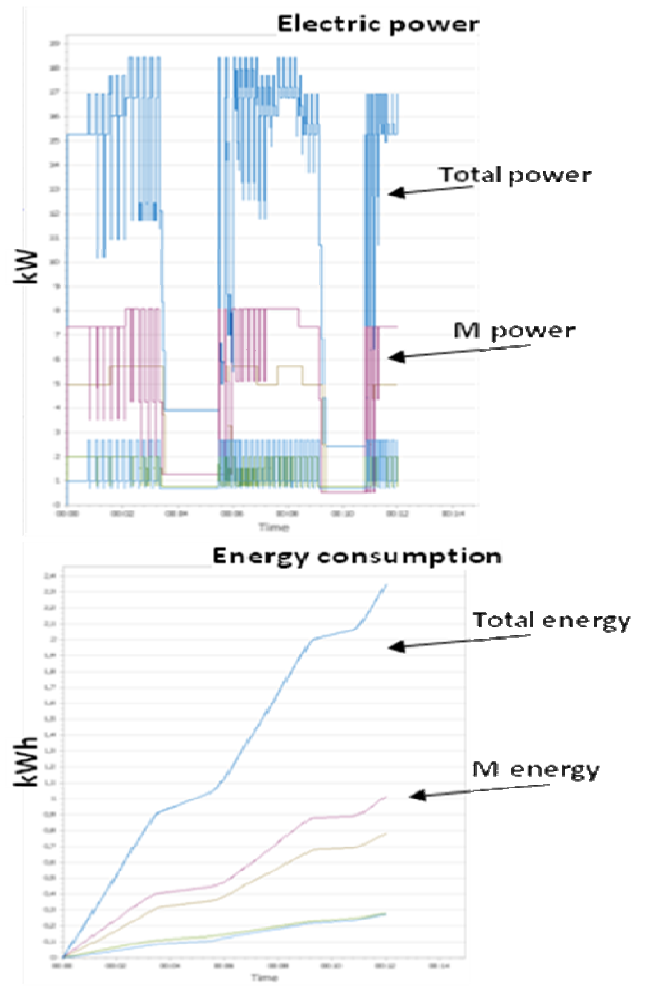

Fig. 6. Simulation experiment results with power and energy plots

The upper graphic is related to the power absorbed from each machine (M_i power) and the total production line one. Each time the machine is in blocking state then the required power falls down to the minimum value. This aspect is highlighted in two interval time in which the *server* has blocked the production line and so has caused its saturation. This plot is very useful for the identification of the maximum power peak; this kind of data is a very important economic parameter often used for a productive plant design or / and management. In particular for the plant design phase, the absorbed power peaks that are related to the operating machines production states allow the engineers to analyze the simulated line and then to evaluate the efficiency of the sketched layout, by searching for an optimal plant solution. For what concern the energy consumption plot, corresponding to the line saturation interval time, it is very clear the rate changing both for the machines and for the production line trend; in fact, since the energy is the mathematical integral of the power, then the absorbed power reduction causes the consumed energy rate decreasing. It must be pointed out that, even if it is not possible to see it from the energy diagram because of the scale, the energy rate of a machine (M_i energy) changes each time an engine enter and exit the machine itself; in the first case the machine switches into the working productive state while in the second one it changes into the waiting state. The economic evaluation that can be argued from this information regards both the total cost associated to a machine working function and the productive line management over a certain period of time, by multiplying the consumed energy with the energy unit cost.

V. CONCLUDING REMARKS

In this paper an industrial engine assembly line has been modelled and simulated in SIMIO discrete event simulator platform, focusing on the energy consumption aspects. The simulation model of each operating machines has been obtained by considering three different and complementary aspects: the mechanical behaviour, the control functionalities and the energy consumption. While the first aspect has been modelled only with SIMIO features, the other two ones have been implemented by means of the FSMs that have been translated into C# code and finally integrated in SIMIO custom *steps*.

A simulation experiment has been run in order to test the whole modelling methodology. The different process times used to characterize the mechanical behaviour of the operating machines have been obtained from the process measurements while the absorbed power values associated to each machine productive state have been deduced from the machine datasheets.

Some limitations characterize the results of this work. The first one regards the missing of process energy measurement so we were not able to evaluate the accuracy of the simulation model energy consumption results. The second limitation comes from the specific machines productive state modelling method and the regarding algorithm used to calculate the energy consumption: in fact since the energy consumption value is obtained after the machine productive state has changed, the user cannot get the refreshed energy values at each simulation time step.

Nevertheless this simulation approach represents a very important support for the industrial engineers especially during the plant design phase in which the plant designers are able to evaluate the energy consumption of the conceived plant solution before the real plant implementation.

Future works coming from this work regard the use of different and more accurate operating machine energetic models instead of using the productive state model. Moreover a different mathematical algorithm for the energy computation has to be defined in order to allow the calculation at each simulation time step. Last but not least it would be useful to remove from the simulator the control functional behaviour of the operating machines in order to have two different software platforms interfaced one to each other: the plant simulator and the control system emulator; in this way it would be possible to use the simulation platform as a control system test bed besides for plant energy efficiency evaluations.

REFERENCES

[1] M. Garetti and M. Taisch, “Sustainable manufacturing trends and research challenges”, in Production Planning & Control, Vol. 23, No. 2-3, pp. 83-104, 2012.

[2] IEA, “Tracking Industrial Energy Efficiency and CO 2 Emissions,” Paris, France, 2007.

[3] www.simio.com

[4] M. Jahangirian, T. Eldabi, A. Naseer, L.K. Stergioulas, and Young, T., “Simulation in manufacturing and business: a review”, in European Journal of Operational Research, Vol. 203, pp. 1-13, 2010.

[5] Thiede, S., "Energy Efficiency in Manufacturing", Dissertation, TU Braunschweig, 2012.

[6] T. Gutowski, J. Dahmus and A. Thiriez, "Electrical energy requirements for manufacturing processes", 13th CIRP International Conference on Life Cycle Engineering, 2006.

[7] A. Cannata, "A methodology to enhance energy efficiency at factory level: improvements for sustainable manufacturing", Dissertation, Politecnico di Milano, 2011.

[8] B. Stahl, M. Taisch, A. Cannata, F. Mueller, S. Thiede, C. Herrmann, A. Cataldo, F. Cavadini, "Combined Energy, Material and Building Simulation for Green Factory Planning", 20th CIRP International Conference on Life Cycle Engineering LCE 2013, 17th - 19th April 2013, Singapore.

[9] C. Khiang Pang, C. Vinh Le, O. Peen Gan, X. Min Chee, D. Hong Zhang, M. Luo, H. Leng Chan, F. Lewis, "Intelligent energy audit and machine management for energy-efficient manufacturing", 2011 IEEE 5th International Conference on Cybernetics and Intelligent Systems (CIS)

[10] Z. Sun, L. Li, "Opportunity estimation for real-time energy control of sustainable manufacturing systems", IEEE transactions on automation science and engineering, Vol. 10, No 1, January 2013

[11] M. Mashaei, B. Lennartson, "Energy reduction in a pallet-constrained flow shop through On-Off control of idle machines", IEEE Transactions on Automation Science and Engineering, Vol. 10, No 1, January 2013

[12] S.S. Krishnan, N. Balasubramanian, E. Subrahmanian, V. Arun Kumar, G. Ramakrishna, A. Murali Ramakrishna, A. Krishnamurthy, "Machine level energy efficiency analysis in discrete manufacturing for a sustainable energy infrastructure", Developing 21st Century Infrastructure Networks, (INFRA), Second International Conference on Infrastructure Systems and Services, Chennai, 9-11 December 2009

[13] A. Cannata, S. Karnouskos, M- Taisch, "Energy efficiency driven process analysis and optimization in discrete manufacturing", 35th Annual Conference of IEEE on Industrial Electronics IECON '09, 3-5 November 2009

[14] B. Heinzl, M. Robler, N. Popper, W. Kastner, I. Leobner, K. Ponweiser, F. Dur, F. Bleicher, "Interdisciplinary strategies for simulation-based optimization of energy efficiency in production facilities", 15th International Conference on Computer Modelling and Simulation, UKSim 2013

[15] C. Vinh Le, C. Khiang Pang, O. Peen Gan, "Energy saving and monitoring technologies in manufacturing systems with industrial case study", 7th Conference on Industrial Electronics and Applications (ICIEA), 18-20 July 2012

[16] B. Krellner, R. Kunis, G. Runger, "Modeling of energy-sensitive manufacturing processes", 9th IEEE International Conference on Industrial Informatics (INDIN), 26-29 July 2011

[17] S. Chiotellis, N. Weinert, G. Seliger, "Simulation-based, energy aware production planning", 43rd CIRP International Conference on Manufacturing Systems, 26-28 May 2010, Vienna