Digital Twin for Energy Optimization in an SMT-PCB Assembly Line

Neha Karanjkar*, Ashish Joglekar, Sampad Mohanty, Venkatesh Prabhu, D. Raghunath and Rajesh Sundaresan
Robert Bosch Centre for Cyber-Physical Systems, Indian Institute of Science Bangalore, India
*Corresponding author (email: neha.karanjkar@rbccps.org)

***Abstract*—This paper presents a case study for the use of an IoT-driven digital twin for energy optimization in an automated Surface Mount Technology (SMT) PCB assembly line containing legacy machines. The line was instrumented with multiple sensors for measuring machine-wise activity and energy consumption. A software platform for data aggregation and a discrete-event digital twin of the line were built entirely using open-source tools. Based on the insights gained from data collected over several days, we propose a buffering-based solution for improving the energy efficiency of the line, and evaluate its impact using simulations of the digital twin. The results show that a 2.7x reduction in the energy consumption is possible via buffer insertion without significantly affecting line throughput.**

***Index Terms*—Internet-of-Things, Digital Twin, Manufacturing**

I. Introduction

Modern Industry 4.0 era machines will come equipped with inbuilt sensors and standardized interfaces to provide real-time information about the machine's operational state and performance metrics such as throughput, energy consumption, raw-material consumption patterns, quality of output and diagnostics information. This machine data will stream from every networked system on the factory floor. Real-time analytics will work on the data to help unlock performance and efficiency improvements as envisioned by the Industry 4.0 revolution [1]. The analytics engine can take the form of a *digital twin* of the real networked system. A digital twin refers to a virtual representation of the real physical system that mirrors its state and behavior. A digital twin of the networked factory floor can be used for prediction and simulation-based optimization studies [2].

In this case study, we consider an SMT PCB assembly line located at a manufacturing facility in Mysuru, India consisting primarily of legacy machines. The machines provide very limited real-time performance information over proprietary interfaces. The quality of output from these machines relies on the alertness and skill of human operators. The throughput is dependent on how quickly drifts in calibration, exhaustion of raw material and machine faults are identified and corrected by the operators. The power consumption and machine-activity patterns are not known as there is limited information about the machine's operational state.

With the goal of energy optimization in mind, we have equipped each of these legacy machines with external sensors that monitor the machine's real-time energy consumption and operational state. A networked IoT infrastructure consisting of edge gateway-computers streams data from these sensors to a middleware platform. The sensor deployment and the IoT infrastructure details are described in detail in Section III. Large amounts of data from the sensors were recorded for a period of 3 months and various statistical techniques were used for inferring the machine state from multiple sensing modalities. The insights gained from the data are summarized in Section IV. In Section V we describe a digital twin of the assembly line built using an open source discrete-event simulation library. The digital twin was used for performing "what if" analysis of various process and configuration options. Based on the insights gained about the throughput bottlenecks and the energy consumption profile in the line, we propose a buffering-based solution for reducing the energy consumption in the line and evaluate its impact using simulations of the digital twin. The simulation results are presented in Section VI. We observe that up-to 2.7x reduction in the average energy consumption is possible via buffer insertion. In summary, this case study demonstrates the application of an IoT framework and digital twin technology for process optimization in assembly lines containing legacy machines.

II. Related Work

The use of discrete-event simulation for process optimization in the SMT PCB assembly process has been reported in the past [3]–[6]. In the absence of real-time data streaming from the assembly line, the model needs to be quite detailed in order to get reasonably accurate estimates for the cycle time and other performance measures. For instance, the processing delays for each machine are highly dependent on the particular PCB design, including the exact number and types of components to be placed, the position and type of component feeders in the placement machine, the stock levels of consumables, etc. On the other hand, a digital twin driven by data measured from the real assembly line can be modelled at a coarser level. For instance, the processing delays can be specified simply as a distribution. Not only is such a model easier to specify, but it can accommodate factors that are outside the cognizance of the operator, such as delays due to manual inspection, breakdowns, etc. Further, the model parameters can be kept continuously in sync with the state of the real assembly line. Case studies for the use of IoT-based digital twins in manufacturing have been described in [7], [8] and [9]. In this paper we describe the insights gained from data and the utility of a digital twin in an SMT PCB assembly process.

This work is supported in part by the UAY scheme (project no. 13) by MHRD, Govt. of India, in collaboration with Tata Consultancy Services (TCS), and in part by the Robert Bosch Centre for Cyber-Physical Systems. We deeply thank Vinyas Innovative Technologies for permitting us to use their facility for instrumentation and analysis.

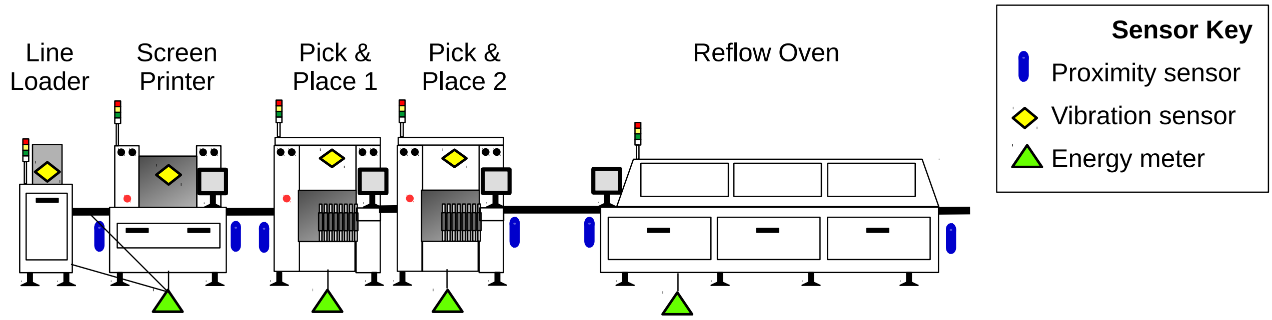

Fig. 1. Schematic of the SMT PCB assembly line and the location of sensors deployed.

III. The IoT Framework and Deployment

A typical configuration for an SMT assembly process consists of the following automated machines connected in sequence: A *Line Loader* loads PCBs one-by-one from a stack onto the conveyor belt. A *Screen Printer* performs application of the solder paste onto the board. *Pick and Place* machines select individual components from reels and place them onto the board. The boards are then forwarded to the *Reflow Oven* which consists of a conveyor belt moving inside a heated chamber. Here, the solder paste melts to form contacts.

A schematic of the assembly line chosen for this case study is shown in Fig. 1. An IoT framework consisting of sensors, edge gateways and a middleware platform was developed and deployed on the factory floor to enable real-time data streaming and offline analytics on a remote cloud. The positions of the sensors deployed onto the line are also indicated in Fig. 1. Energy meters (comprising 3-phase VAF meters and current transformers) were deployed for measuring the instantaneous power drawn by each machine. A three axis vibration sensor (developed in-house) was attached to the body of each machine. The vibration signatures were used along with the energy meters for identifying the operational states and the corresponding power drawn in each machine. IR-based Proximity sensors were placed at the entry and exit of every machine to detect the flow of PCBs. The vibration data was sampled at 75 Hz whereas a sampling rate of 1 Hz was used for the energy meter. The non-intrusive nature of these sensors allowed for their seamless deployment on a fully operational line.

The sensors communicate with edge gateways which are single board computers (Raspberry Pi Zero W+). The gateways support multiple interfaces such as MODBUS, I^2C and BLE for communication with the sensors. Our edge gateway architecture has provisions for time synchronization and data stream buffering. An NTP service helps synchronise the clocks of all the IoT gateways to the accuracy of a few milliseconds. Each gateway runs an MQTT broker service using the open source Eclipse Mosquitto broker. The sensor data samples arriving at the gateway are timestamped and buffered, and then forwarded to a middleware layer over a dedicated WiFi network using REST APIs. The middleware layer consists of an ElasticSearch database hosted on a server-grade system located inside the factory. The middleware acts like an immediate data sink for the gateways and regularly synchronises the local copy of the sensor data with the data stored on the cloud for analysis.

IV. Insights from Data

Sensor data was used for identifying operational states in each of the machines. The data revealed causal relationships among machines on the serial assembly line, with each individual machine cycling through distinct states. As an example, Fig. 2 presents a snapshot of the time-series data recorded from the Line Loader's vibration sensor and the Screen Printer's RMS current sensor. The operational states were identified using window-based variance and thresholding of the streaming data. The Screen Printer is seen to cycle through three distinct states, namely *printing*, *cleaning* and *idling*. The cleaning operation is seen to occur periodically after printing two boards. Using the automated state identification, average values for the duration and the power drawn in each state were measured for each of the machines. These values were then plugged in as parameters in the discrete-event digital twin for simulations. The data also revealed the production shifts and the usage patterns for the machines. For example, the Reflow oven was found to be turned off only on non-working days. Analysis of data collected over several days showed that the Reflow oven clearly accounts for the largest fraction of energy consumed among all the machines in the line. On an average, the Reflow oven was seen to account for more than 85% of the total energy consumption in each production period. A comparison of the average processing latencies for each of the machines revealed that the Pick and Place operation is typically the throughput bottleneck in the line.

V. A Discrete-Event Digital Twin of the SMT-PCB Assembly Line

We have built a detailed, parameterized model of the assembly line using SimPy [10], a discrete-event simulation library in Python. The model is open source and freely available at [11]. Individual machines, conveyor belts and human operators in the line are modelled as SimPy processes and PCBs are modelled as passive objects flowing between the machines over the conveyor belts. The behavior of each machine is defined by the set of states that it can be in (for example, *idle*, *waiting to output*, *processing/printing*) and the transitions between them. Some of the machines also have consumables (for example, solder paste or component reels). A human operator is interrupted whenever the level of a

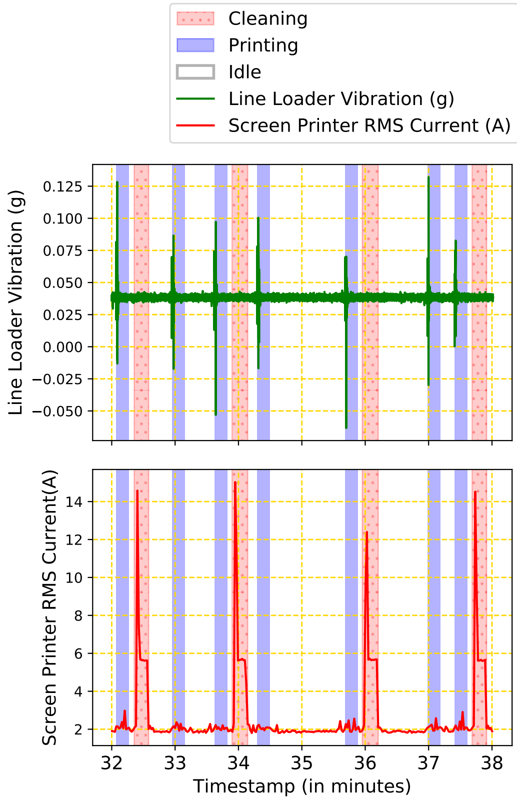

Fig. 2. Identification of the operational states in the Screen Printer using data from the vibration and current sensors. The detected states are indicated using red and blue regions in the plot. The printer is seen to cycle through *cleaning*, *printing* and *idle* states.

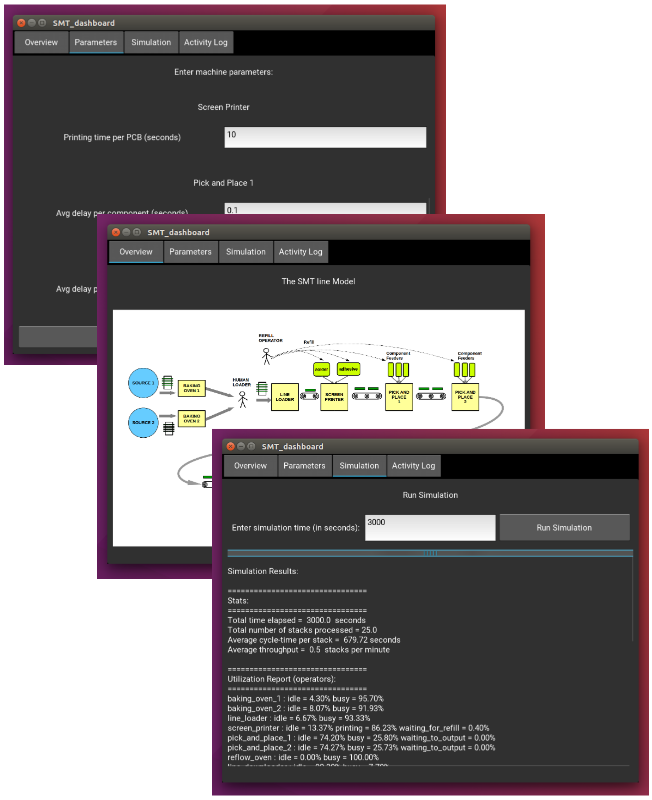

Fig. 3. Screenshots of the simulator's GUI. The simulator is available for download at [11].

consumable falls below a set threshold and the machine stalls until the consumable is refilled. The frequency and duration of the refill operations, the state-wise processing delays and power ratings for each machine, and the conveyor belt speeds are some of the model parameters. The parameter values are set to correspond to the sensor data from the real line over a specified time interval. For the simulation-based optimization results presented in this paper, the parameter values were set to match the line data over a period of three days.

Each simulation run generates a detailed activity log along with aggregate performance measures such as the average cycle-time and energy consumption per-PCB, average system throughput, and machine-wise utilization and energy profiles. The model has been validated by comparing the aggregate performance measures predicted by the model to those inferred from the real line data. A cross-platform graphical user interface (GUI) for the model has been implemented using Kivy, an open-source Python framework for interactive application development. Fig. 3 presents some screenshots of the simulator. The GUI is targeted for use by the factory floor-manager for prediction and what-if analyses, whereas a text-based parameter setup can be used for the automated optimization runs.

VI. Energy Optimization via Buffering

From the data collected over several days we observe that the Reflow oven alone accounts for more than 85% of the total energy consumption. However the throughput bottleneck in the line is often the Pick and Place operation which has the highest latency. Each Pick and Place machine can process only a single PCB at a time. As a result, the arrival rate of PCBs at the entry of the Reflow oven is much lower than the maximum rate that it can support. Thus the Reflow oven remains powered-on, yet unutilized for a large fraction of the time. This is, in-fact a common concern in high-mix low volume production lines. To address this, some Reflow oven manufacturers provide additional energy-saving (sleep/hibernate) modes in recent designs.

However, for assembly lines containing legacy machines, the energy efficiency of the Reflow oven remains a concern. For such lines, buffering may help improve the utilization of the Reflow oven thereby improving the energy efficiency. PCBs can be buffered after the Pick and Place operation and sent through the Reflow oven in a burst. While the buffer fills up, the Reflow oven can remain powered off, thus improving the energy efficiency of the line. Fig. 4 shows the configuration of the assembly line with buffering introduced between the Pick and Place and Reflow operations. The buffering module consists of vertical stacks for holding the PCBs, and capability for automatic loading and unloading of the PCBs one at-a-time. Vertical PCB buffering modules suited to such a purpose are available from major SMT equipment manufacturers.

We perform simulations of the digital twin to evaluate the impact of such a buffering-based solution on the energy consumption and the throughput of the line. For this evaluation we consider two types of buffering schemes: a *single buffering* scheme and a *double buffering* scheme. For each scheme, the buffer-size (that is, the maximum number of PCBs that can be stored in the buffer) is an optimization parameter.

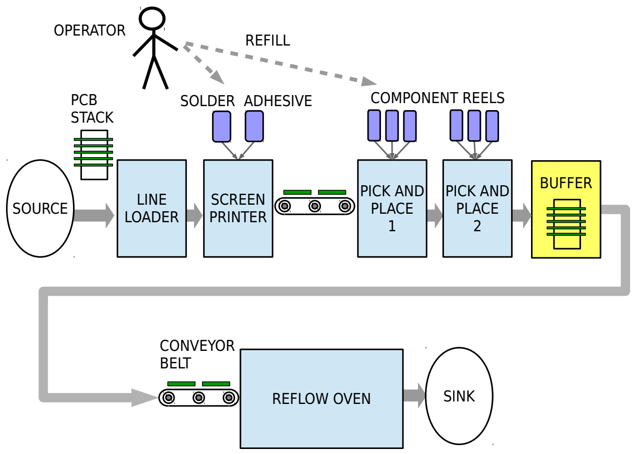

Fig. 4. SMT-PCB assembly line configuration with buffering.

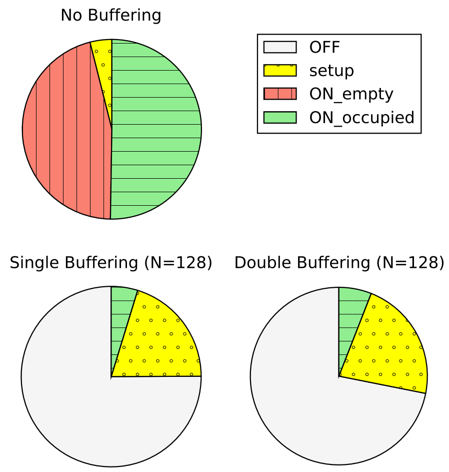

Fig. 5. Fraction of time spent by the Reflow Oven in each of its states.

In the **single buffering** scheme, the buffering module consists of a single stack that can hold at-most N PCBs. The module can be in one of two modes: *filling* and *emptying*. In the *filling* mode the buffer accepts PCBs leaving the Pick and Place machines and stores them onto the stack. When the buffer is full, it enters the *emptying* mode and entry of PCBs from upstream is blocked. The Reflow oven is turned ON and PCBs already stored onto the stack are then unloaded one-by-one and sent to the Reflow oven. When the stack is empty, the oven is turned OFF and the cycle repeats. Thus, in the single-stage buffering, the throughput is limited as the entry of items into the buffer is blocked while the buffer is emptying.

In the **double buffering** scheme, the buffering module consists of two stacks, each of capacity N. PCBs leaving the Pick and Place machine fill up into the input-side stack while PCBs already stored on the output-side stack can be *simultaneously* unloaded towards the Reflow oven, thus allowing for a higher throughput. Whenever the input-side stack is full and the output-side stack is empty, the positions of the two stacks are interchanged. Thus the entry and exit of PCBs from the buffer can occur simultaneously, leading to a higher throughput. In fact, the use of double-buffering for decoupling the input and output has been widely used in domains such as computer graphics.

For SMT PCB assembly, there are a few factors which limit the extent of buffering possible in a line. The *tack time* for a PCB refers to the maximum time that can elapse between the placement and the Reflow operations for reliable contact formation. The tack time depends largely on the composition of the solder paste and typically varies between 8 to 24 hours. Introduction of a buffering module can increase the cycle time but must not violate the limit on the tack time. As a conservative limit, we assume that the total cycle time for a PCB must not exceed six hours. Further, the buffer has a limited capacity (typically varying between tens to hundreds of items). In addition, each time the Reflow oven is turned off and then turned on again, it incurs a setup delay. Whenever the oven is turned off, its temperature decays with time (in a manner governed by Newton's law of cooling). When it is turned on again, the setup time depends on the difference between the current and the preset oven temperatures. The total amount of time spent in the setup state must not be very large as the setup state draws a large current and this may offset the energy savings. We consider all of the above constraints and limitations during simulations for evaluating the impact of buffering.

We perform simulations with varying buffer sizes (N) and compare the performance measures to those obtained for the original (unbuffered) line. Each simulation is run until a single batch (of 1024 PCBs) is processed completely. For each simulation, we assume that the buffer size N is fixed and is a power of two. The simulation results are listed in Table I. We observe that the energy consumption drops and the cycle time increases with increasing buffer capacity (N). The constraint on cycle time (maximum cycle time ≤ 6 hours) is satisfied for $N \leq 128$. Thus at $N = 128$ indicated by the shaded rows, a reduction in the energy consumption of about 2.3x is seen for the single buffering scheme and 2.7x for the double buffering scheme. In the single buffering case, the reduction in energy is achieved at the cost of a 20% drop in the system throughput, whereas in the double buffering case the throughput remains largely unaffected. In Fig. 5 we plot the fraction of the total production time spent by the Reflow oven in each of its states (OFF, SETUP and ON) in the original as well as the buffered line. The buffering eliminates the time spent in the ON and unutilized state, but also increases the fraction of the time spent in the setup state. The net effect however is a significant reduction in the total energy consumption.

A further optimization possible, is to overlap in time the setup of the Reflow oven with the filling up of the buffer. In this scheme, the buffer is used as before, but the Reflow oven is turned on as soon as the buffer accumulates $N - k$ items (where $k \in \{0, 1, 2, \ldots, N-1\}$ is some fixed constant). Thus the time taken for the buffer to accumulate k more items to become completely full overlaps with the setup time for the Reflow oven. For the double buffering scheme and a fixed buffer size ($N = 128$) we plot the average energy and throughput as a function of k in Fig. 6. The optimum value for k is found to be 24 in this case, indicating that for the current line configuration the Reflow oven should be turned on as soon as the input-side buffer accumulates 104 items out of its capacity of 128.

TABLE I
SIMULATION RESULTS SHOWING THE EFFECT OF BUFFERING ON THE AVERAGE THROUGHPUT AND ENERGY CONSUMPTION IN THE LINE.

Buffer capacity per-stage (N)	Avg Energy per-PCB (kJ)	Avg throughput (PCBs per-hour)	Avg cycle-time per-PCB (hours)	Max cycle-time per-PCB (hours)
No Buffering				
	2,427.29	40.35	0.74	1.62
Single Buffering				
4	1,411.65	40.17	0.81	1.74
8	1,328.58	40.10	0.90	1.88
16	1,438.52	34.10	1.22	2.05
32	1,463.27	30.88	1.67	2.55
64	1,316.22	30.68	2.31	3.59
128	1,042.13	32.21	3.32	5.47
256	736.76	34.64	5.03	8.90
512	516.00	36.73	8.24	15.43
1024	396.82	37.98	14.67	25.59
Double Buffering				
4	1,384.90	40.31	0.82	1.75
8	1,298.07	40.40	0.89	1.70
16	1,232.62	40.58	1.06	1.60
32	1,157.41	40.96	1.36	1.83
64	1,060.68	40.91	1.94	2.60
128	910.05	40.33	3.03	4.28
256	688.54	39.68	4.88	7.59
512	487.61	39.01	8.19	13.80
1024	373.14	37.98	14.67	25.59

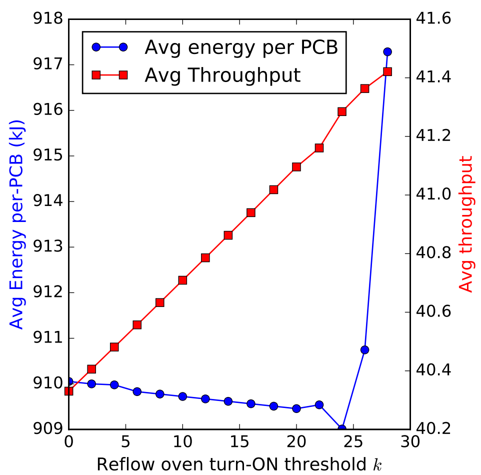

Fig. 6. A plot of the average energy consumed per-PCB and the system throughput as a function of the Reflow-oven turn-on threshold K for the double buffering scheme with $N = 128$.

VII. CONCLUSIONS

This paper presented a case study for the deployment of an IoT framework in an SMT PCB assembly line containing legacy machines. Multiple sensors were installed on the line for measuring machine-wise activity and energy consumption profiles. A data aggregation platform and a discrete-event digital twin of the line were built entirely using open source tools. Data collected from the line over several days provided insights into the performance bottlenecks and the energy consumption patterns in the line. Based on these insights, we proposed a buffering-based solution for improving the energy efficiency and evaluated its impact using simulations of the digital twin. The simulations predicted a 2.7x reduction in the average energy consumption achievable via buffer insertion without significantly affecting the line throughput. The current limitations of this framework are that the sensor data is collected at a high sampling rate, but the machine state estimation is performed remotely on the raw data sent over a network. In future, the machine state inference could be implemented at the edge node itself so that only the processed information is sent to the cloud. Further, a variable sampling rate for the sensor data needs to be supported, so that the sampling rate could be adjusted as per the requirements dictated by the application.

ACKNOWLEDGEMENT

The authors wish to thank Dr. Devadutta Kulkarni and Dr. Rajeev Shorey (TCS Innovations Lab) and Prof. Bharadwaj Amrutur, Dr. S.Sridhar, Dr. Alexandre Reiffers-Masson (Robert Bosch Centre for Cyber-Physical Systems) for their valuable suggestions and fruitful discussions.

REFERENCES

[1] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, “Industry 4.0,” *Business & Information Systems Engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[2] E. Negri, L. Fumagalli, and M. Macchi, “A Review of the Roles of Digital Twin in CPS-based Production Systems,” *Procedia Manufacturing*, vol. 11, pp. 939 – 948, 2017, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy.

[3] J. Li, N. Nagarur, and K. Srihari, “Modeling PCB assembly lines in EMS provider’s environment: integrating product design into simulation models,” in *Proceedings of the Winter Simulation Conference*. Winter Simulation Conference, 2011, pp. 2341–2350.

[4] S. Gebus, O. Martin, A. Soulas, and E. Juuso, “Production Optimization on PCB assembly lines using discrete-event simulation,” *University of Oulu, Department of Process and Environmental Engineering, Control Engineering Laboratory, Report A*, no. 24, 2004.

[5] T. M. Tirpak, P. K. Mohapatra, P. C. Nelson, and R. R. Rajbhandari, “A generic classification and object-oriented simulation toolkit for SMT assembly equipment,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 32, no. 1, pp. 104–122, 2002.

[6] M. Jackson and C. Johansson, “Real time discrete event simulation of a PCB production system for operational support,” in *Proceedings of the 29th conference on Winter simulation*. IEEE Computer Society, 1997, pp. 832–837.

[7] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui, “Digital twin-driven product design, manufacturing and service with big data,” *The International Journal of Advanced Manufacturing Technology*, vol. 94, no. 9-12, pp. 3563–3576, 2018.

[8] F. Tao and M. Zhang, “Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing,” *IEEE Access*, vol. 5, pp. 20 418–20 427, 2017.

[9] T. H.-J. Uhlemann, C. Schock, C. Lehmann, S. Freiberger, and R. Steinhilper, “The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems,” *Procedia Manufacturing*, vol. 9, pp. 113 – 120, 2017, 7th Conference on Learning Factories, CLF 2017.

[10] “SimPy: A discrete-event simulation library for Python,” https://simpy.readthedocs.io/en/latest/.

[11] “Online repository for the SMT-PCB assembly line simulation model,” https://github.com/NehaKaranjkar/IIoT_RBCCPS/.