Dynamic Rerouting of Cyber-Physical Production Systems in Response to Disruptions Based on SDC Framework

Yassine Qamsane, Efe C. Balta, James Moyne, Dawn Tilbury and Kira Barton

Abstract— **The world is in the midst of a new industrial revolution driven by Smart Manufacturing (SM). Though this new paradigm promises increased flexibility, product customization, improved quality, efficient energy consumption, and improved productivity, SM systems are more susceptible to small faults that could cascade into major failures or even cyber-attacks that enter the plant. Flexibility and reactivity/proactivity represent important means to enhance SM systems' reliability, efficiency, and robust response to faults. Within this context, this paper focuses on dynamic rerouting of parts in response to a fault or attack that can change the system's behavior. The method is based on the use of our recently proposed Software-Defined Control (SDC) framework [1], which consolidates data from the different levels of the automation pyramid to provide a global view of the entire SM system. To solve the rerouting problem, a rerouting application accesses the global view of the system through a set of digital twins hosted in the SDC central controller, and provides new route alternatives to a decision maker that prioritizes these routes based on an optimization function. The new route alternatives are then sent to the operator as reconfiguration recommendations to be deployed to the plant floor. The proposition is illustrated using a small manufacturing system example.**

I. INTRODUCTION

Today's marketplaces are characterized by short lead times, tight product tolerances, pressure on costs, frequent changes in demand and continuous evolution of the technological requirements of the products [2]. Smart Manufacturing (SM), also called "the fourth industrial revolution" (Industry 4.0) [3] is a recent concept that promises improved operation of manufacturing systems in such markets. SM systems are defined as systems that are able to respond in real time to meet changing demands and conditions in the factory, in the supply network, and in customer needs in a fully-integrated and collaborative fashion [4]. SM has emerged thanks to the use of transformative information technologies, e.g., Industrial Internet of Things (IIoT), Artificial Intelligence (AI), cloud and service-oriented computing, and data science in the production domain. These technologies allow managing the physical and cyber assets of interconnected manufacturing components through the generation and analysis of high volume data which is known as Big Data [5]. The integration of the physical components of manufacturing with the cyberspace to form cyber-physical systems has been embraced by individual companies, industrial consortia, regions, and countries [6]. SM aims at flattening the top and the bottom management layers of the business organizational pyramid [7] by connecting the physical production process to the business planning and logistics. This promises to speed up the information flow in the ISA-95 hierarchy, where process data move up and decisions come down slowly, by providing more access to business data at different levels to more users. With SM, it becomes possible for operators and business leaders to get the right data to the right place at the right time, to support making well-informed decisions. SM will assist people in doing their jobs better. For instance, it will allow operators to act in real time with machines by providing interfaces that deliver production and quality data in order to tackle slowdowns or quality discrepancies.

Though these are not new concepts, flexibility and reconfigurability are key elements of SM. Related works on Flexible Manufacturing Systems (FMS) [8], [9] and Reconfigurable Manufacturing Systems (RMS) [10], [11] address the use of modular resources to realize and improve the flexibility and reconfigurability of systems. These concepts give the advantage to improve efficiency and thus lower production costs. Generally, flexibility falls into two typical categories: machine flexibility and routing flexibility. The former refers to the ability of a machine to carry out different operations required by different part types. The latter refers to the ability of processing different part types using more than one route (alternative routing) [12]. Alternative routing allows efficient scheduling, handling unexpected incidents such as machine breakdowns and rush orders, improving productivity, and minimizing system utilization and work-in-process [13], [14]. Nevertheless, the implementation of flexibility and reconfigurability concepts is uneasy to achieve in practice [15], [16]. These concepts induce complicated design, planning, scheduling, and control problems, which yield large data at different levels of the manufacturing system. On the other hand, most companies do not know what to do with the data they have, let alone how to interpret them to improve their processes and products. They lack software and modeling systems to analyze data [17].

The use of flexibility and reconfigurability concepts with cyber technologies, which allow to store and process structured data and knowledge, can support real-time decision making in production management (e.g., planning, scheduling, and control). The analysis of real-time data allows intelligent reasoning and provides value-added services to the decision makers. Within this context, we recently developed a Software-Defined Control (SDC) framework that enables integrated and programmatic management of SM systems. SDC extracts and consolidates data from the production

This work was funded in part by NSF 1544678.

Yassine Qamsane, Efe C. Balta, James Moyne, Dawn Tilbury and Kira Barton are with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA {yqamsane,baltaefe,moyne,tilbury,bartonkl}@umich.edu

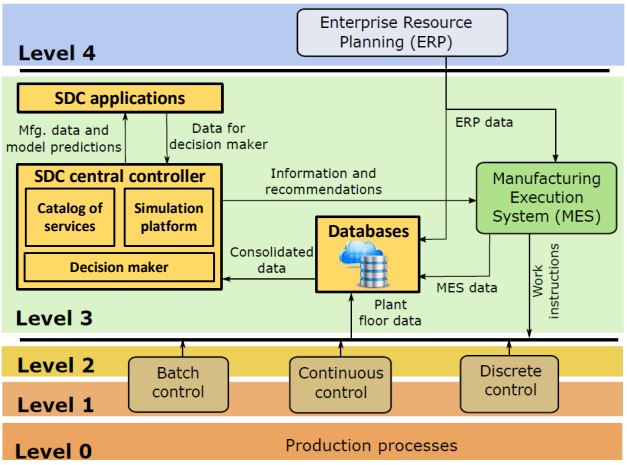

Fig. 1. Coexistence of SDC with ISA-95 in parallel with MES.

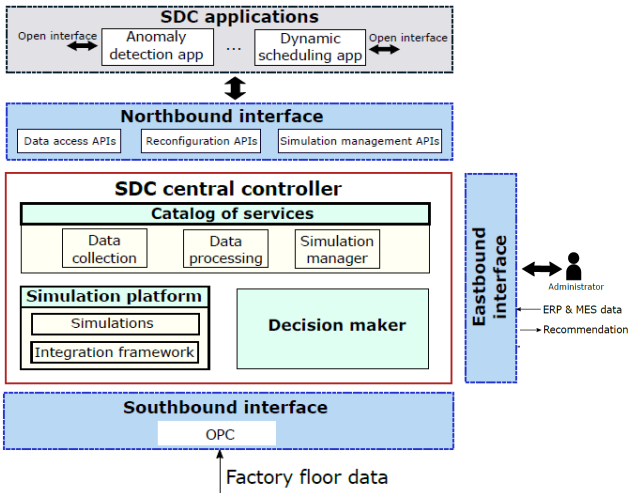

Fig. 2. SDC architecture [1].

and enterprise levels to a central controller that monitors performance and detects changing conditions [1], [18]. The integrated global view of the system provided by the central controller supports the development of applications, which in turn provide the central controller with new information to support reconfiguration.

This paper proposes a method for determining part rerouting in response to detection of a system anomaly, and applying the rerouting decision in reconfiguring an SM system based on the SDC framework. The method incorporates the following capabilities. (i) real-time modeling at multiple levels using the concept of Digital Twin (DT); (ii) the use of an SDC rerouting application that uses the DTs to explore all possible alternative routes; (iii) the use of SDC decision maker to prioritize the alternative routes based on optimization functions, e.g, cost, throughput, quality, etc.

The rest of this paper is structured as follows. Section II provides a background on the research related to this work. Section III presents a new method to reconfigure the system in case a disruption is detected for an optimal performance. Section IV illustrates the application of the proposed method via an example. Concluding remarks and future work are in Section V.

II. Background

A. Rerouting in manufacturing systems

The aim of production planning and control is to achieve the best utilization of available manufacturing resources to ensure high productivity. Effective production planning and control is affected by the stochastic and dynamic production environment, which is characterized by the occurrence of unexpected asynchronous events and interruptions. These may be caused by faults, machine degradation, or cyber-attacks [19]. Their occurrence may lead to unpredictable downtimes, which are considered as one of the major contributors to a low Overall Equipment Effectiveness (OEE). Because large-scale manufacturing systems comprise several resources that are able to perform the same task, a key answer to minimizing the effect of downtime on OEE is to reconfigure the plant to reroute parts to available resources.

A vast literature on routing flexibility and dynamic scheduling is available and some of the proposed works study the interaction with design, planning, scheduling and control problems in FMS. Four rerouting policies, namely no re-routing, queue re-routing, arrival re-routing, and all re-routing to deal with the problem of machine failure were proposed in [20]. In [21], different metaheuristics are adapted for solving the alternative routing selection problem in real time in order to reduce the congestion in the system by selecting a routing for each part among its alternative routings. Simulation results of the applied algorithms show that the real-time rescheduling outperforms the case without rescheduling, but it has a negative impact on the work in process. A multi-agent scheduling system based on bio-inspired scheduling approach for scheduling dynamic job shop problems is proposed in [22]. The method provides a promising way for the efficient scheduling of dynamic manufacturing systems based on agents but doesn't consider the effect of machine breakdown on the rerouting policies. A mathematical model to solve the dynamic rescheduling problem in an FMS, which considered the energy consumption and the schedule efficiency as multi-objective optimization functions is proposed in [23].

These methods propose policies that follow a selected option from a set of predefined paths of travel giving due consideration to the system's layout without considering the changing environment. The method proposed herein also considers predefined paths, but in addition, SDC can compute new non predefined routes considering the changing environment. This is gained thanks to the use of real-time data from the production floor to build a global view of the system, which supports decision making.

B. Software-Defined Control

We recently proposed Software-Defined Control (SDC), an approach to flexible control of manufacturing systems within SM paradigm [1]. SDC uses a global view of the entire manufacturing system, including physical components (e.g., machines, robots, and parts to be processed) together

with cyber components (e.g., logic controllers, RFID readers, and networks) to help improve manufacturing productivity, quality, and sustainability. To obtain this global view, SDC consolidates data at the control, operations, and business levels to support operations management solutions with global information and recommend reconfigurations to deploy ultimately to the plant floor. As shown in Fig. 1, SDC operates on level 3 of the ISA 95 hierarchy in parallel with the Manufacturing Execution System (MES). SDC is composed of three main components: a set of centralized databases, a central controller, and a set of applications (Fig. 2). Databases with big data capabilities are used to consolidate data from the plant floor, operations management, and business levels. The central controller, which is the main piece of SDC, makes use of the consolidated data to provide an up-to-date global view of the manufacturing system. Applications such as anomaly detection, scheduling/dispatching, rerouting, etc., use the global view to support the central controller in its decision making tasks. Depending on the storage capacity and speed requirements, any of these three components could be hosted in edge or cloud platforms. The separation of the applications from the central controller allows easier implementation of new algorithms, models, and optimization techniques, without refining the central controller.

SDC is designed to coexist with current automation technologies comprising the Manufacturing Execution system (MES) and Enterprise Resource & Planning (ERP) software in a value-add and optional manner (Fig. 1). Based on the global view of the entire manufacturing system, and with the help of the SDC applications, the central controller suggests reconfiguration recommendations to the MES. Once approved by the operator, the new configurations could be swiftly deployed to the plant floor. The information flow within the central controller is supported by the interfaces shown in Fig. 2. A southbound interface is used for the collection and transformation of plant floor data by means of unified and standardized protocols (e.g., MTConnect [24] and OPC UA [25]) prior to its use for analytics; A northbound interface enables communication between SDC applications and the central controller; An eastbound interface allows the back and forth communications with the MES and the ERP, and allows system administrators to configure and manage the central controller. The central controller consists of: a catalog of services that manage the information flow to and within the central controller, a decision maker that determines the recommendations to transmit to the MES, and a simulation platform that hosts digital twins of the system used to identify the changing conditions, and to predict the effect of suggested configurations before their deployment.

III. Dynamic rerouting based on SDC framework

In this section, we present how SDC can solve the problem of rerouting parts in case a disruption occurs in the system. Real-time data is used to build models of the system at different levels, which allows the SDC central controller to keep an up-to-date global view of the system. The real-time global view will support a developed rerouting application to define the possible alternative routes, then return these to the decision maker, which in turn prioritizes the alternatives based on an optimization function. In the remainder of this section we first revisit the concept of Digital Twin and we propose two types of DTs used within the SDC simulation platform to build the real-time global view of the manufacturing system. Then, we present the rerouting app, which uses the DTs to define new alternative routes and send them back to the decision maker. Finally, we show how the decision maker sorts the new alternatives based on an optimization function.

A. Digital Twin

Digital Twins (DT) are virtual representations of physical assets, processes, and systems which can be used for different goals, e.g., to better understand and predict problems, prevent downtimes, provide early warnings, develop new products, etc., by using simulations [26], [27], [28]. DT is one of the key concepts of SM, which allows to exploit real-time data coming from the plant floor to monitor and analyze performance of machines and equipment (e.g., evaluate machine health, detect faults, and control production), develop predictive maintenance strategies (e.g., eliminate unplanned downtimes), and manage the plant floor expectations to support reconfiguration. Considering this need, the SDC framework uses a set of DTs at different levels of the manufacturing system. These help to better understand the physical system digitally in order to improve its global performance. In the following, we present two types of DTs likely to support rerouting.

1) Topology Digital Twin: we view a manufacturing system as a material flow network that consists of a number of production units. The topology DT provides a real-time representation of the system's layout including all active units and their interconnections. To form an up-to-date global view of the manufacturing system, the topology DT stores the initial layout which consists of all the manufacturing units constituting the overall system. Then, the layout is updated based on the real-time data, e.g., if a machine is down it will be omitted from the updated topology.

In [29], manufacturing systems are also viewed as networks that consist of a set of nodes (vertices) and a set of links (edges) that connect some of the nodes. Similarly, to build the topology DT, we divide the manufacturing equipment into nodes and links. Each work station (milling/turning machine, assembly/welding robot, quality inspection, buffer, etc.) in the manufacturing system is a single node, whereas a possible material flow between two work stations (conveyor, AGV, gantry, etc.) is represented by a directed link. Links could be unidirectional or bidirectional.

Formally, a topology DT is a tuple $T = (N, L, In, Out, \Delta)$ where:

- N is a finite set of nodes;
- L is a set of links that connect some of these nodes;
- In is the set of material flow inputs;
- Out is the set of material flow outputs;

The system may consist of unique or multiple material

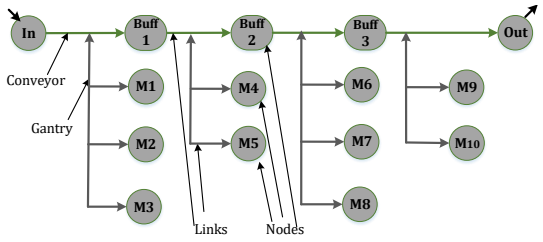

Fig. 3. Example of a topology digital twin.

flow inputs/outputs. In and Out are defined as nodes, i.e., $\{In, Out\} \in N$.

- $\Delta : N \times L^* \rightarrow N$ is a function that defines the flow transitions, where L^* denotes the set of all finite links concatenations in L. An element $l \in L^*$ is a sequence of links. The length of a sequence l is given by the number of its involved links. A sequence $l \in L^*$ consists of at least 1 link. For instance, consider $\{l_1, l_2, l_3\} \in L$ three links to move material to the four nodes $\{n_1, n_2, n_3, n_4\} \in N$. The flow transition functions between n_1 and n_2 and between n_1 and n_4 are respectively defined by: $\Delta(n_1, l_1 \cdot l_2) = n_2$, and $\Delta(\Delta(n_1, l_1), l_3) = n_4$.

Fig. 3 represents an example of a topology DT of a manufacturing system that consists of 4 cells where parallel CNC machines perform a sequence of machining tasks. A conveyor moves parts between the input of the flow and its output, and at each cell, a gantry performs the loading and unloading of parts to and from the machines. Three buffers exist between the 4 cells. Parts which are processed in one cell are transferred to any of the machines of the next cell.

In the topology DT of Fig. 3, the set of nodes is identified $N = \{In, \{M_1, ..., M_{10}\}, \{B_1, ..., B_3\}, Out\}$, with $\{M_1, ..., M_{10}\}$ is a set of 10 machines, $\{B_1, ..., B_3\}$ is a set of 3 buffers, and $\{In, Out\} \in N$, the input and output of the material flow. The set of links is given by $L = \{conv, g_1, g_2, g_3, g_4\}$ where $conv$ refers to the conveyor that routes the parts between the input and output of the system going through the 4 cells and 3 buffers; and $\{g_1, g_2, g_3, g_4\}$ refer to the 4 gantries loading and unloading the machines. At the first cell of the manufacturing system, the flow transition function is identified as:

- $\Delta(In, conv \cdot g_1) = M_1 \vee M_2 \vee M_3$, which indicate that a part can be moved from the input of the system to any of the machines M_1, M_2, or M_3 for further processing through the conveyor ($conv$) and the gantry g_1 consecutively.
- $\Delta(M_1 \vee M_2 \vee M_3, g_1 \cdot conv) = B_1$, which indicate that a processed part can be moved from the corresponding machine (M_1, M_2, or M_3) to the downstream buffer through the gantry (g_1) and the conveyor ($conv$) consecutively.

The flow transitions at the other cells of the system are identified in a similar way.

2) Machine Asset Digital Twin: manufacturing units often operate on a set of discrete states where some continuous dynamics take place. Transitions among these discrete states are event-driven or time-driven, which allows to model their asynchronous behavior as Discrete Event Dynamic Systems (DEDS). The continuous dynamics within certain discrete states could be studied by using differential equations of some continuous variables. Discrete and continuous models are treated separately within the SDC digital twin environment. As the work in this paper is interested in rerouting in manufacturing systems, we limit the study to a DEDS modeling abstraction.

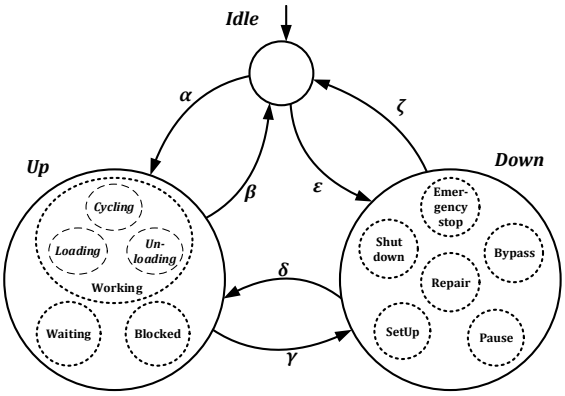

Fig. 4. Machine asset digital twin.

This section introduces the machine asset DT as a generic discrete model that provides access to the structure, the status, and the behavior of an individual manufacturing unit. We define the machine asset DT as a Finite State Machine (FSM) with 3 global states: *Idle*, *Up*, and *Down*. A machine could have one *Idle* state and multiple *Up* and *Down* states as shown in Fig. 4. Transitions between these states are whether event-driven or time-driven, i.e., annotating edges $\{\alpha, \beta, \gamma, \delta, \varepsilon, \zeta\}$ refer to the occurrence of an event or to the elapse of some time. It is also possible to have transitions between states inside the *Up* and *Down* macro-states.

The following describes the different states of a machine within the machine asset DT:

- A machine is described as *idle* when it is not being used. The machine goes to the idle state if it has completed all tasks.
- *Up* states are those wherein the machine can run automatically to move or produce parts when the corresponding conditions are fulfilled. These are divided into:
  - *Working* states: the machine is executing a task on a part. Loading (material is entering work position) and unloading (material is exiting work position) are parts of working.
  - *Blocked* state: the machine has finished its working task but the part cannot be cleared.
  - *Waiting* state: the machine is capable and ready to start executing a working task, and a part is available but it cannot start because additional conditions are not fulfilled.
- *Down* states are those wherein the machine is not capable to move or produce parts. These are divided into:

- *Setup* state: the setup state is initiated when a set up procedure (e.g., warm up, tool change, mastering, clean machine table and other surfaces, etc.) is launched. The machine stays in this state until it transitions to another state.
- *Bypass* state: the bypass state is used when the machine is not functioning correctly and it is simply passing entering parts without working on them. The machine stays in the bypass state until this latter is locally or remotely deactivated.
- *Pause* state: the pause state is triggered manually by an operator to go for a break. The machine stops in a "controlled" manner in this state. The resumption of operation after pressing pause/resume button allows the machine to go back to the up state it was in and complete its normal task.
- *Emergency stop* state: a safety state which is triggered by an emergency stop button or by another safety system. It is configured to abort the operation of the machine and place it in a safe condition as quickly as possible. Once initiated, the machine stays in the emergency stop state until it is forced to idle state.
- *Repair* state: the state wherein the machine has a physical intervention such as opening a gate for maintenance. The machine stays in this state until it is forced to idle state.
- *Shutdown* state: the shutdown state is triggered manually by an operator usually using a button on the machine control panel. Once initiated, the machine stays in the shutdown state until it switches to another state.

Since state transitions of a machine asset DT are event- or time-driven, it is appropriate to represent a machine asset DT as a timed automaton $M = (S, s_0, \Sigma, C, E, I)$ where:

- S is a finite set of states.
- $s_0 \in S$ is the idle state.
- Σ is a finite alphabet of events.
- C is a finite set of clocks.
- $E \subseteq S \times \Sigma \times B(C) \times 2^C \times S$ is the set of edges that describe the states transitions, where $B(C)$ is the set of guards expressing clock constraints and/or variables to be fulfilled in order for the transitions to be taken, and 2^C is the set of clocks to be reset and/or variables to be assigned when the transitions are taken.
- $I : S \longrightarrow B(C)$ assigns invariants to states. Invariants are local clock constraint values expressing how long M may remain in a particular state.

An example of a machine that has one idle state, two up states, and one down state is shown in Fig. 5. The model depicts that the machine has the capability of performing two different jobs (2 different up states), which could be detected for example by two different sensors that enable these jobs to start. The times to realize these jobs (*cycle time*) and time to recover from a failure (*time to repair*), are specified by the variables t_1, t_2, and t_3.

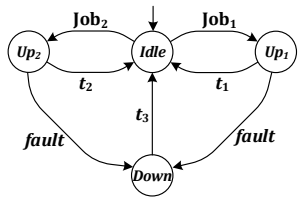

Fig. 5. Machine DES model example.

Within machine asset DTs, some machine level performance metrics described by cycle time, downtime, equipment failure rate, setup and adjustments, etc., are tracked by analyzing real-time and historical data. This allows the central controller to keep an up-to-date integrated global view of the system. This global view supports the development of applications which in turn support the central controller in its analysis tasks. For instance, the DTs may be used by an anomaly detection app to detect faults and evaluate machine health; a rerouting app to decide the path (route) of work and the sequence of operations; a scheduling/dispatching app to allocate resources over time, and to assign the next job to be processed from a set of waiting jobs.

This paper focuses on performing rerouting by using the proposed DTs. We assume that rerouting is solicited after an anomaly detection app detects a disruption (e.g., a machine is down) in the system and send feedback to the central controller. The latter requests the rerouting app to provide new route alternatives. The following shows how the rerouting app makes use of the DTs to compute new routes for the parts.

B. Rerouting App

The task of the rerouting app is to provide all possible part routes in the entire or in a specific portion of the system when it is requested. Rerouting is needed to avoid states that may slowdown production, e.g., in the instance of anomalies, machine health degradation, changing conditions, etc. Once solicited by the central controller, the rerouting app accesses the global view of the system provided by the DTs to compute a list of alternative routes. It first accesses the topology DT to gain insight into available machines and links. Then, it reaches the machine asset DTs of the available machines and links in order to check their capabilities.

The concept of the rerouting app consists of first building a capability model for each available machine. The capability model is a standard FSM $A = (Q, \Sigma, \delta, q_0, Q_m)$, where Q is a set of states with $q_0 \in Q$ the initial state, and $Q_m \in Q$ the set of marked states which refers to the number of capabilities of a machine; Σ is the event alphabet where each event represents a capability, e.g., the event $M_1\_P_1$ refers to that machine M_1 is capable of realizing process P_1; $\delta : Q \times \Sigma \rightarrow Q$ is a transition function that connects the initial state q_0 to the marked states Q_m with transitions labeled with machines' capabilities. Fig. 6 shows an example of a machine M_1 with 2 capabilities. There might be some particular constraints to specify for the rerouting app such as avoiding a machine or avoiding routing a specific part

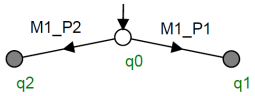

Fig. 6. Example of machine capability model.

to a specific machine. These specifications are also defined as FSM. The composition of all machine capability models and the specifications (if any) returns a model of all possible route configurations. The latter are provided to the decision maker within the central controller, which sorts these based on an optimization function.

C. Decision Maker

After the possible new routes are evaluated by the rerouting app, the decision maker utilizes the DTs to evaluate the optimality of these routes. Here we propose a Mixed Integer Linear Program (MILP) to evaluate the route with minimum time. Assume that the production plan is known, and the task of the decision maker is to make sure that each of the required parts will be produced with minimum possible time. For each process, we will have the transportation time t^{tr}, setup time if the consecutive processes are of different kind t^{se}, and the cycle time t^{cy}. Let C_i denote the possible configurations for the machines, that are known a priori through the machine asset DT. A configuration is a possible set of part assignments to available machines. For example, $C_1 = [P_1, P_2]$ denotes a configuration where part 1 (P_1) and part 2 (P_2) are respectively assigned to potentially different machines. We define $\psi = (\mathbf{M}_i, C_j)$ as a decision path pair where, $\mathbf{M}_i$ denotes the machines assigned to the configuration C_j. For example, $\psi_0 = ([M_1, M_2], C_1)$ means that P_1 is assigned to M_1 and P_2 is assigned to M_2. Each route in the FSM tree provided by the rerouting app will correspond to a decision where machines M_i are utilized with a certain configuration C_i. We denote $b_k \in \{0, 1\}$ as the decision variable to show if a certain path in the FSM tree is chosen for the optimal solution or not, therefore b_k implies certain machines and configuration combination as $b_k \implies \psi_k$. The required use of each combination is denoted by c_i. Then, we write the MILP for a system with n_c configurations and n_b decision variables as follows.

$$\begin{aligned} \min \quad & \boldsymbol{\tau}^T \mathbf{b} && \text{(1a)} \\ \text{subject to:} \quad & \boldsymbol{\Phi}\mathbf{b} = \mathbf{c} && \text{(1b)} \\ & \boldsymbol{\tau}_i = t_i^{tr} + t_i^{se} + t_i^{cy} && \text{(1c)} \\ & b_i \in \{0, 1\} && \text{(1d)} \end{aligned}$$

where, $\mathbf{b} = [b_1, b_2, \ldots, b_{n_b}]^T$ is the vector of decision variables, $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_{n_b}]^T$ are the total time for each decision b_i and consequently the machine and configuration combination, $\mathbf{c} = [c_1, c_2, \ldots, c_{n_c}]$, and $\boldsymbol{\Phi} \in \{0, 1\}^{n_c \times n_b}$ is a matrix with each row denoting the decision variables related to a single configuration such that $c_i = \sum_{k=1}^{n} b_k$. Times $\boldsymbol{\tau}_i$ are updated from the machine asset DT at the time of decision for making decisions with real-time data.

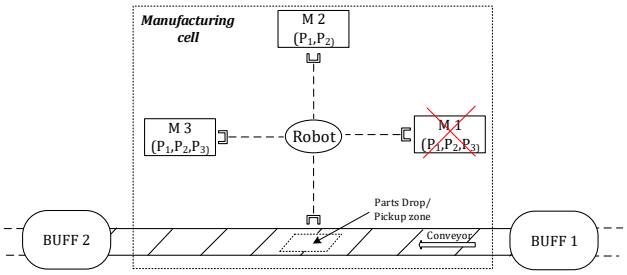

Fig. 7. Illustrative manufacturing system example.

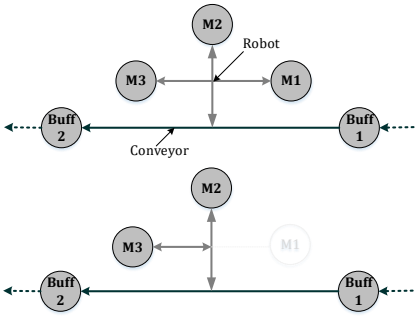

Fig. 8. Topology DT of the manufacturing system example. (a) initial topology (b) updated topology after loss of M_1's connection.

IV. Case Study Example

A small manufacturing system example is used to illustrate our proposed rerouting solution. The manufacturing cell shown in Fig. 7 processes 3 different types of parts (P_1, P_2, and P_3), which are assembled together in a downstream station to form a final product. The manufacturing cell has 3 milling machines (M_1, M_2, and M_3), each can mill different parts, and a robot (R) that transports these. A conveyor ($Conv$) moves the parts through the system. When a part arrives from the entrance buffer ($Buff_1$) to the cell, the robot will pick it up and drop it into the corresponding milling machine for further processing. After processing, the robot will pick it up from the milling machine and drop it into the conveyor which transfers it to the exit buffer ($Buff_2$). We assume that both buffers have the capacity of 3 slots, and that a downstream assembly station needs the 3 different parts at buffer 2 to put together a final product, i.e., buffer 2 should not contain 2 parts of the same type. For the sake of brevity, the assembly station is not treated in this example.

Initially, the finished-part quantity per time unit (throughput) is maximized when parts P_1, P_2, and P_3 are routed to machines M_1, M_2, and M_3 respectively. After being processed by the corresponding machines, the parts are picked from buffer 2 and assembled together. We consider the instance where an anomaly detection app detects that M_1 has an unexpected failure and must be shot down for maintenance. SDC is required to compute new route configurations for the physical parts through the manufacturing cell, thus avoiding the defective machine. The objective is to keep production operating (albeit at a reduced level). The following steps illustrate how the problem is solved.

(i) The topology DT is automatically updated based on

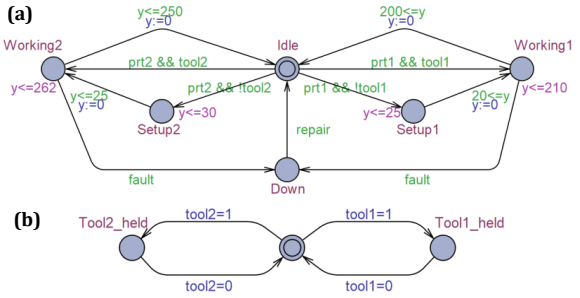

Fig. 9. Machine asset DT for machine M_2. (a) Machine M_2 model, (b) Tools setup model.

real-time data, e.g., the loss of the heartbeat of M_1. The set of nodes in the topology DT is updated from the initial set $N = \{M_1, M_2, M_3, Buff_1, Buff_2\}$ to the up-to-date set $N' = \{M_2, M_3, Buff_1, Buff_2\}$ where M_1 is omitted. The set $L = \{Conv, R\}$ remains unchanged. The flow transitions become: $\{\Delta(Buff_1, Conv \cdot R) = M_2 \vee M_3,\ \Delta(M_2 \vee M_3, R \cdot Conv) = Buff_2\}$. Figure. 8 illustrates the update of the topology of the system.

(ii) A request is sent to the rerouting app to provide route alternatives for the parts.

(iii) The rerouting app accesses the global view of the system through the central controller's DT platform. First, the available machines within the system are verified through the topology DT. The available machines for the studied example are M_1 and M_2. Second, the machine asset DTs allow to know which machines are capable of taking over M_1 in processing parts P_1. Figure. 9 shows the machine asset DT of machine M_2. The latter has the capability of working on parts P_1 (state "$Working1$") and P_2 (state "$Working2$"). If a "$fault$" is detected, the machine goes to "$Down$" state. A tool change is necessary to switch from processing P_1 to processing P_2 and vice versa, which requires passing by the setup states "$Setup1$" and "$Setup2$". The model of Fig. 9(b) allows to know which tool is used at anytime. For instance, if "$tool1$" is used by the machine, the model of Fig. 9(b) switches from the initial state to the state "$tool1\_held$". Then if a part P_1 is available ("$prt1$"), the condition "$prt1 \wedge tool1$" becomes true, which allows the transition from the state "$Idle$" to state "$Working1$". If tool2 is used and part1 is delivered, the condition "$prt1 \wedge \neg tool1$" forces the machine to the state "$Setup1$" to outfit the appropriate tool. This operation takes between $20s$ to $25s$, the automaton may leave "$Setup1$" at any time point when the clock y is in the interval between 20 (associated to the edge) and 25 (associated to the state invariant). The machine processes a part P_1 within $200s$ to $210s$. The automaton may leave "$Working1$" at any time point when the clock y is in the interval between the value 200 associated to the edge, and 210 associated to the state invariant. The model behaves in a similar way if a part P_2 is delivered.

On the other hand, M_3 can process parts P_1, P_2, and P_3 and it is modeled in a similar way.

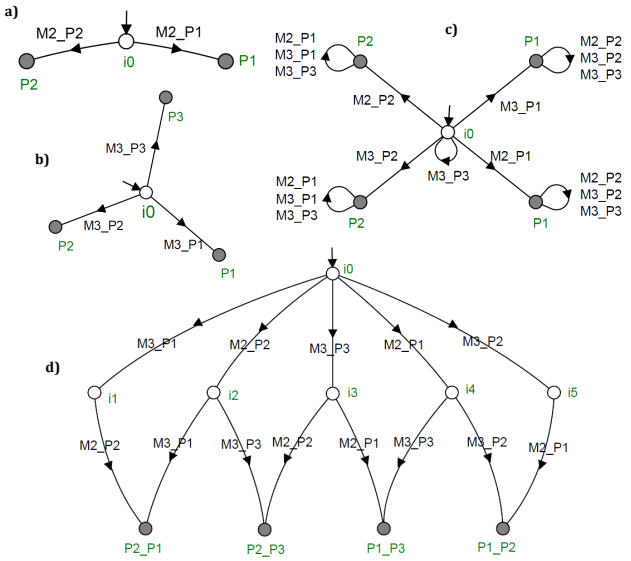

Fig. 10. Steps of computing possible routes. (a) Capability of M_2 (b) Capability of M_3 (c) Constraint on $Buff_2$ (d) Synthesis of possible existing routes.

(iv) The rerouting app builds a capability model for each available machine (M_2 and M_3) and a specification model on the overflow of buffer 2. M_2 and M_3 capability models are shown in Fig. 10(a) and (b), respectively. Fig. 10(c) represents a specification that restricts the system from processing and delivering 2 parts of the same kind to the exit buffer ($Buff_2$) as required by the downstream assembly station. The composition of these 3 models results in the model of Fig. 10(d), which provides all possible and admissible combinations of the active machines to keep producing the parts with respect to the specification. Once the alternative route configurations are identified, the rerouting app provides these to the decision maker which prioritize them based on an optimization function.

(v) To maximize the number of assembled products at the downstream station, parts should be delivered to buffer 2 in a sequence of the pairs (P_1,P_2), (P_1,P_3), and (P_2,P_3). The decision maker identifies the possible configurations for the available machines M_2, M_3 as $C_1 = \{P_1, P_2\}, C_2 = \{P_1, P_3\}, C_3 = \{P_2, P_3\}$, thus we have $n_c = 3$. Additionally, using the machine asset DTs, the vector $\boldsymbol{\tau} = [481, 575, 547, 603]$ is formed for the optimization. Based on possible paths, there are 4 candidates ($n_b = 4$) as shown in Fig. 10(d). Here, we assume that the required manufacturing steps to apply in the reconfiguration and the requirement of each configuration is known by the decision maker. The requirement is to have one of each configuration on the decision, thus we have $\boldsymbol{c} = [1, 1, 1]^T$ and

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Therefore we can easily see that the optimal decision includes $b_2 \implies ([M_2, M_3], C_2)$ and $b_3 \implies ([M_2, M_3], C_3)$. Also note that in the given example, the total transportation time is identical for all possible paths since the robot will transfer parts to both CNCs in each path. In a complex environment with additional paths, the transportation time would affect the optimal path. Optimization in Eq. 1 evaluates that based on the cycle, setup, and transportation times the configuration with the assignments $M_3 \leftarrow P_1$ and $M_2 \leftarrow P_2$ is the optimal configuration for C_1.

V. Conclusion

The work presented in this paper focuses on a method to solve the problem of rerouting parts in a manufacturing system by using the SDC framework. Presented work leverages digital twins to utilize real time-data from the production floor in making decisions. A topology digital twin is presented to capture the real-time topology formed by the manufacturing units in the plant floor and a machine asset digital twin is introduced to provide insight into machine status, capabilities and process times. An important assumption in this work is that the required production steps for individual parts in the system are known by the central controller and the requirements for the reconfiguration are predetermined. Compared to conventional methods, the proposed SDC method offers more flexibility and agility to help improve quality and throughput of the production while reducing waste in time and resource. This is gained thanks to the use of real-time data from the production floor to build a global view of the system, which supports monitoring the changing environment and optimize performance of SM systems.

Future work will look into considering a multi-objective optimization and integrating the product requirements into the decision making to optimize the reconfigurations based on the required production steps. The generalization of the research results in this paper will be conducted for more complex systems. Future work will implement the concepts of digital twin and decision maker in a manufacturing testbed to evaluate their effectiveness.

References

[1] F. Lopez, Y. Shao, Z. M. Mao, J. Moyne, K. Barton, and D. Tilbury, "A software-defined framework for the integrated management of smart manufacturing systems," *Manufacturing Letters*, vol. 15, pp. 18–21, 2018.

[2] T. Tolio, *Design of flexible production systems*. Springer, 2008.

[3] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–4.

[4] N. I. of Standards and Technology, "Smart manufacturing operations planning and control program," 2014. [Online]. Available: https://www.nist.gov/programs-projects/smart-manufacturing-operations-planning-and-control-program

[5] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015.

[6] A. Kusiak, "Smart manufacturing," *International Journal of Production Research*, vol. 56, no. 1-2, pp. 508–517, 2018.

[7] B. Scholten, *The road to integration: A guide to applying the ISA-95 standard in manufacturing*. Isa, 2007.

[8] U. A. Tetzlaff, "Flexible manufacturing systems," in *Optimal Design of Flexible Manufacturing Systems*. Springer, 1990, pp. 5–11.

[9] H. Tempelmeier and H. Kuhn, *Flexible manufacturing systems: decision support for design and operation*. John Wiley & Sons, 1993, vol. 12.

[10] M. G. Mehrabi, A. G. Ulsoy, and Y. Koren, "Reconfigurable manufacturing systems: Key to future manufacturing," *Journal of Intelligent manufacturing*, vol. 11, no. 4, pp. 403–419, 2000.

[11] Y. Koren, W. Wang, and X. Gu, "Value creation through design for scalability of reconfigurable manufacturing systems," *International Journal of Production Research*, vol. 55, no. 5, pp. 1227–1242, 2017.

[12] H. Tsubone and M. Horikawa, "A comparison between machine flexibility and routing flexibility," *International Journal of Flexible Manufacturing Systems*, vol. 11, no. 1, pp. 83–101, 1999.

[13] N. Nasr and E. Elsayed, "Job shop scheduling with alternative machines," *The international journal of production research*, vol. 28, no. 9, pp. 1595–1609, 1990.

[14] M.-C. Yu and T. J. Greene, "An operational measure of routing flexibility in a multi-stage multi-product production system," *The International Journal of Advanced Manufacturing Technology*, vol. 43, no. 3-4, pp. 357–364, 2009.

[15] Y. Qamsane, A. Tajer, and A. Philippot, "Towards an approach of synthesis, validation and implementation of distributed control for ams by using events ordering relations," *International Journal of Production Research*, vol. 55, no. 21, pp. 6235–6253, 2017.

[16] ——, "A synthesis approach to distributed supervisory control design for manufacturing systems with grafcet implementation," *International Journal of Production Research*, vol. 55, no. 15, pp. 4283–4303, 2017.

[17] A. Kusiak, "Smart manufacturing must embrace big data." *Nature*, vol. 544, no. 7648, pp. 23–25, 2017.

[18] E. C. Balta, D. M. Tilbury, and K. Barton, "A centralized framework for system-level control and management of additive manufacturing fleets," in *Automation Science and Engineering (CASE), 2018 14th IEEE Conference on*. IEEE, 2018.

[19] F. Lopez, M. Saez, Y. Shao, E. C. Balta, J. Moyne, Z. M. Mao, K. Barton, and D. Tilbury, "Categorization of anomalies in smart manufacturing systems to support the selection of detection mechanisms," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1885–1892, 2017.

[20] E. Kutanoglu and I. Sabuncuoglu, "Routing-based reactive scheduling policies for machine failures in dynamic job shops," *International journal of production research*, vol. 39, no. 14, pp. 3141–3158, 2001.

[21] M. Souier, Z. Sari, and A. Hassam, "Real-time rescheduling meta-heuristic algorithms applied to fms with routing flexibility," *The International Journal of Advanced Manufacturing Technology*, vol. 64, no. 1-4, pp. 145–164, 2013.

[22] X. Yu and B. Ram, "Bio-inspired scheduling for dynamic job shops with flexible routing and sequence-dependent setups," *International Journal of Production Research*, vol. 44, no. 22, pp. 4793–4813, 2006.

[23] L. Zhang, X. Li, L. Gao, and G. Zhang, "Dynamic rescheduling in fms that is simultaneously considering energy consumption and schedule efficiency," *The International Journal of Advanced Manufacturing Technology*, vol. 87, no. 5-8, pp. 1387–1399, 2016.

[24] M. Institute, "Mtconnect standard version 1.2.0. www.mtconnect.org/gettingstarted/developers/standards.aspx."

[25] M. Schleipen, "Opc ua supporting the automated engineering of production monitoring and control systems," in *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*. IEEE, 2008, pp. 640–647.

[26] A. Canedo, "Industrial iot lifecycle via digital twins," in *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. ACM, 2016, p. 29.

[27] 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.

[28] M. Grieves and J. Vickers, "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," in *Transdisciplinary perspectives on complex systems*. Springer, 2017, pp. 85–113.

[29] T. Becker, M. Meyer, and K. Windt, "A manufacturing systems network model for the evaluation of complex manufacturing systems," *International Journal of Productivity and Performance Management*, vol. 63, no. 3, pp. 324–340, 2014.