A Study on Performance Evaluation and Status-based Decision for Cyber-Physical Production Systems

Feifan Wang, Feng Ju, Yan Lu

***Abstract*— In concert with advances in information and communication technology and their application to manufacturing environments, physical entities in factories are acquiring more intelligence via integration with cyber systems. This integration brings about Cyber-Physical Production Systems and leads to smart manufacturing, the next generation manufacturing paradigm. In the new paradigm, high levels of agility, flexibility, and real-time control make it possible to keep the system running efficiently and self-organized. At the same time, however, it becomes difficult in a self-organized and decentralized system to capture the system's status, evaluate the system's performance, and predict the system's future events. In this article, we suggest improvements to smart manufacturing systems where the intelligence from smart entities could be fully utilized without losing system control. To achieve this goal, a solution for integrating schedule-driven production (push systems) and event-driven production (pull systems) is proposed to optimize both material flow and information flow for manufacturing operations. For each entity in a smart manufacturing system, details of decision making are encapsulated and its status is exposed. The status-based decisions filter out unimportant information and make smart manufacturing systems loosely-coupled and predictable. A simulation case study based on Devices Profile for Web Services [1] is used to illustrate the effectiveness of such an approach. The case study suggests that status-based decisions could be applied to smart manufacturing and that they can be part of an approach that balances the self-organized control with overall performance. Therefore, we can make full use of intelligent entities in lower levels of a factory while keeping the entire system under control.**

***Index Terms*— Smart Manufacturing Systems, Cyber-Physical Production Systems, Status-based Decision, Devices Profile of Web Services**

I. Introduction

The rapid development in information and communication technology (ICT) provides an opportunity to improve current manufacturing systems. Smart manufacturing systems use ICT and intelligent software applications to optimize the use of labor, material, and energy, producing customized, high-quality products for punctual delivery. Such systems can respond quickly to the changes in market demands and supply chains, relying on the advances in Cyber-Physical Systems (CPS) [2][3]. CPS are integrations of cyber components and physical components in a system to add new capabilities to the original physical systems [4][5]. CPS have broad applications [6][7]. Specifically, CPS designed to achieve automation in production are called Cyber-Physical Production Systems (CPPS) [7][8][9][10]. Many case studies have been conducted in recent years to explore how to make use of CPPS. For example, a service-oriented industrial automation system was introduced by Colombo [11]. It is a customized dynamic assembly system with workstations, conveyors, and lifters. Reboredo [12] constructed a Semantic Field Device Integration Platform in discrete manufacturing that considers the computational limitations of field devices. Some other case studies focus on Digital Factory [13] and Smart Factory [14][15][16], where smart products, smart machines, and augmented operators cooperate with each other to form a flexible system with decentralized control.

To be flexible and agile, smart manufacturing systems apply existing technologies such as radio frequency identification (RFID) and embedded computing to a new design for material flow and information flow. Improvement comes from research on ICT, system architecture, production scheduling, and related standards. Much research has been done to improve the intelligence of devices in manufacturing systems, but limited efforts have been made on the design of material and information flow to achieve performance prediction and decision making. From the perspective of production systems, the following issues need to be investigated: (1) How to leverage local optimal decisions made by self-organized entities to contribute to improving overall performance, and (2) How to predict system performance in real time by incorporating endogenous and exogenous decisions and uncertainties. This paper incorporates previous improvements to smart manufacturing systems and proposes a guideline to address these two issues. Specifically, a solution for integrating schedule-driven production (push systems) and event-driven production (pull systems) is provided to optimize both material flow and information flow for manufacturing operations. For each entity in the system, details needed for decision making are encapsulated and its status is exposed. The status-based decisions filter out unimportant information and make smart manufacturing systems loosely-coupled and predictable. A simulation case study based on Devices Profile for Web Services is used to illustrate the effectiveness of such an approach. The case study suggests that status-based decisions could be applied to smart manufacturing to balance self-organized control with overall performance. Therefore, we can make full use of intelligent entities in lower levels of a factory while keeping the entire system under control.

The rest of the paper is organized as follows: Section II introduces the technical background in the architecture

Feifan Wang and Feng Ju are with School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281, USA.Feifan.Wang@asu.edu, Feng.Ju@asu.edu

Yan Lu is with National Institute of Standards and Technology, Gaithersburg, MD 20899, USA. yan.lu@nist.gov

Please address all correspondence to Professor Feng Ju.

of cyber-physical production systems and their operation strategies. In Section III, details are given to describe a solution for integrating push systems and pull systems, for optimizing both material flow and information flow, and for achieving a high level of manufacturing flexibility and agility. A simulation case study is conducted in Section IV to illustrate how smart manufacturing systems could be loosely coupled and predictable. Section V is devoted to conclusions and future work.

II. Background

A. Service-oriented Architecture

SoA is an architecture style that promotes integrations of heterogeneous applications and technologies. SoA has successful applications in software engineering, where software systems are loosely coupled and reusable components act as services to the whole system. Details of each component are encapsulated. Interfaces and descriptions are published through platform- and language-independent standards. Several reasons motivate applications of SoA in manufacturing systems. SoA-based manufacturing follows along the early ideas of constructing a more flexible and reusable physical system, where a production system can be decomposed into several modules, and each module acts as a service provider. Modules collaborate with each other in a manufacturing system to realize a variety of products and handle unexpected changes. The flexibility of these machines can be fully taken advantage of when they are viewed as service providers in flexible manufacturing systems. Another motivating factor for applying SoA in manufacturing systems is to extend the cost-effective software systems integration approach in the cyber world to a cyber-physical world.

There are some existing technologies that support SoA in manufacturing systems. Devices Profile for Web Services (DPWS) [1] is one web service technology that defines a minimal set of web service implementation for device integration. DPWS enables secure messaging, discovery, description, and eventing on resource-constrained endpoints. The basic components in DPWS are clients and devices. Devices encapsulate information processes and publish their services using Web Services Description Language (WSDL) [17]. Clients get access to devices in order to acquire data or call methods. OPC Unified Architecture (OPC UA) from the OPC Foundation is another technology implementing SoA. It provides solutions via OPC UA Client and OPC UA Server for data exchange and data modeling in manufacturing systems [18].

B. Push and Pull Systems

Material flow and information flow are two important aspects that should be considered in manufacturing system design. Generally, manufacturing systems can be classified into two types with regards to material and information flow [19]. The first type is called the push system. Production activities in push systems are triggered by schedules. Schedules make demands of parts in the workstation level consistent with demands of products at the system level. The second type is pull systems, where production activities are triggered by status-changing events. In pull systems, only the last workstation in a production line receives demands of production from the system level. Operations happened in the last workstation change the work-in-process (WIP) in buffer, which triggers operations of upstream workstations. Material flows from upstream workstations to downstream workstations, while demand information flows from downstream workstations to upstream workstations. Both push and pull systems have their strengths and drawbacks. Pull systems perform well in controlling WIP and handling changes to the systems. One strength of push systems is that every activity follows a plan. The traditional way to combine push systems and pull systems is to create a push-pull interface, and make the systems partly pull and partly push. In smart manufacturing systems, more tools are available than in traditional manufacturing systems to optimize both material flow and information flow. However, they call for a new system design approach to make use of those tools to make material and information flow smoothly and efficiently.

III. Operations in Smart Manufacturing Systems

A. A Flat Architecture

It has been a long time since people first made use of computers to assist production. In product development domains, Computer Aided Design (CAD) was initiated in the mid 1970s. Later, Product Data Management (PDM) software was designed to manage product data like CAD drawings. In production systems, Material Requirements Planning (MRP) acts both as a method and a software application, coming into factories during the 1970s. These technologies enabled Computer-Integrated Manufacturing (CIM). CIM technologies played an important role in manufacturing this past decade. These technologies resulted from a convergence of the virtual world and the physical world in manufacturing systems [7]. They do not, however, fundamentally change how production activities are organized. In the CIM paradigm, computers usually replace humans and take over trivial work that is time-consuming but well structured.

Manufacturing systems handle two types of orders. One type (customer requests, for example) comes from outside the system. The other type originates inside the system (such as instructions to a workstation). To fulfill orders effectively, smart manufacturing calls for convergence between the cyber world and the physical world in manufacturing systems, but it differs fundamentally from traditional CIM-based manufacturing systems. Advances in ICT enable machines in a factory to request and respond to messages. When machines on the shop floor obtain the ability to collaborate and take care of more complicated duties in the production activities, the traditional rigid automation pyramid structure can be partly broken, and a flat architecture can be created. The traditional automation pyramid works when there is a lack of intelligence in lower levels. In that case, entire production processes are centrally controlled. If the system works based on a central plan, it becomes vulnerable to unpredictable variances. In smart manufacturing systems,

physical equipment has corresponding cyber nodes. Sensors and RFID enable the status of equipment to be detected and the equipment to receive instructions. Enterprise functions make production orders upon customer's requests assuming a factory as a black box, while the intelligence in lower levels of a factory enables resources to be dynamically allocated to fulfill the orders.

B. A Status-based Decision

A series of decisions is required in a manufacturing system to get an order fulfilled. Smart manufacturing systems distribute decisions to different entities in different levels of the systems. With each entity understanding its environment by communicating with other related entities, it makes a local optimal decision based on that understanding. Then, the statuses of the entities are updated to support future decisions. Entities from higher levels check the statuses of lower-level entities and make decisions on task assignments. Entities from lower levels receive the assignments, find a way to finish the tasks, and then expose the updated statuses. As such, each entity from lower levels to higher levels uses its own intelligence to optimally finish its assigned tasks, and the relationship between said entities is therefore loosely coupled. The structure of the system is shown in Fig.1.

Consider a scenario when a customer places an order for a customized product. Based on the status of the factory, the customer is promised to receive the product within a lead time. The demand for the product is decomposed into demands for different kinds of parts. Production planning is then based on the capabilities and capacities of production modules in the factory. Schedules from production planning enable each part to be finished on time. The production modules receive requests to make different parts. Each part gets admitted into the right module every time the module status is changed to be available. In this way, resources inside the module are assigned to parts dynamically.

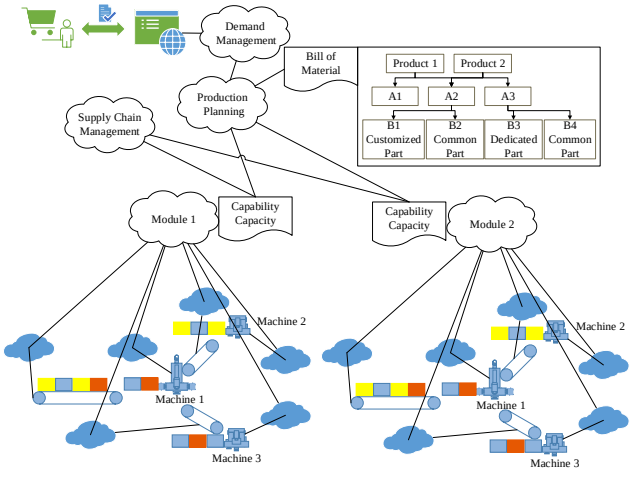

Fig. 1. A Status-based Decision

C. Integration of Push - Pull Systems

In order to achieve better performances, many efforts have been made integrate push and pull systems. Traditionally, this integration is achieved by dividing a production line into two segments, one push system and one pull system. A part in a smart manufacturing system could carry its own process information. In this situation, the material flow follows the information flow, as seen in Fig.2. From the perspective of an individual module, it is a push system. Modules receive orders from schedules, and the work for each module consists of producing the required parts in time. However, the implementation inside the module follows a pull system style. Parts come into a module based on the status of said module. The processing path for each part is carried on the part and it depends on the status of workstations in the module. The process work of workstations is triggered by different events.

The material and information flows can be achieved by using advanced ICT. For each part, an RFID chip can be installed on the carrier of the part. There is an RFID writer and a reader for each module and workstation. In order to accomplish a scheduled job in time, a module allows the right part to enter the module at the right time and writes an optimal path on the RFID for the part. When the part finds its path in the production module, all the process and path information is stored in the RFID chip. Each machine reads the customized process requirements from the RFID chip and provides a proper processing service. Similarly, a conveyor reads the path information and sends the part to the right workstation.

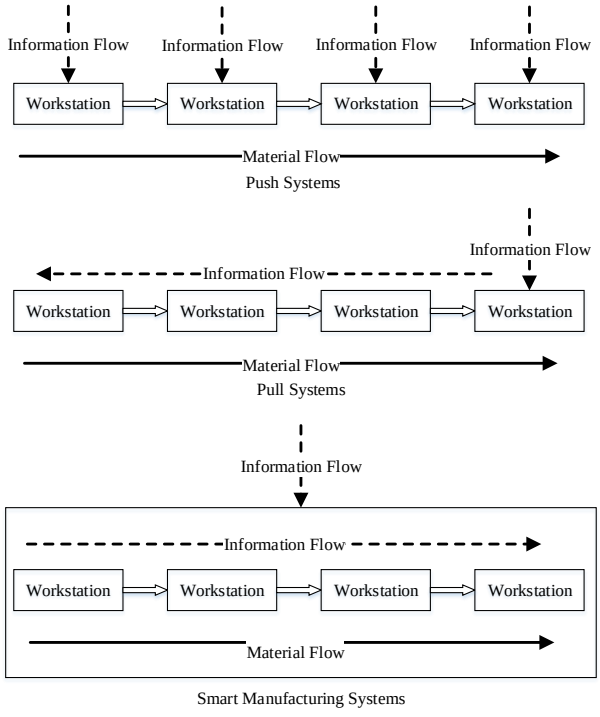

Fig. 2. Integration of Push - Pull Systems

D. Plug-N-Play

With the support of Plug-N-Play [14], a workstation can be removed during run time from the system without reconfig-

uration. Similarly, a workstation can be added to the system without reconfiguration. The property of Plug-N-Play in smart manufacturing systems keeps manufacturing systems reliable. In a traditional factory, failures and maintenance schedules of machines might have a big impact on the entire manufacturing system. Unplanned failures of machines may cause impossible schedules and create a large number of WIPs. Any maintenance requires a configuration suitable with the production plan. In smart manufacturing systems, the function of Plug-N-Play can help solve these problems.

Plug-N-Play for smart manufacturing brings a lot of advantages. Via Plug-N-Play, smart manufacturing systems are able to handle unplanned variances. Since details of a module are hidden from other modules, all changes that happen to any workstation are represented by a status change. Status-based decisions from different entities keep the entire production process going normally.

E. Capability and Capacity

Higher levels decision making keeps work going to the right modules that are capable of the processes, while lower levels decision making keeps the workload in balance. Compared with traditional manufacturing systems, smart manufacturing systems enable distributed equipment in factories to share the responsibility for decision making and avoid contradicting decisions. Real-time control in discrete manufacturing becomes possible in smart manufacturing systems. Smart manufacturing systems automatically adjust their behaviors to keep load balance and optimize their performance in real time. However, the performance of a system needs to be predictable in order to make these real-time controls effective. This is motivated by the customer requirements for reliable delivery times. When a customer places an order, the smart manufacturing system should be able to give a lead time with high confidence. The more self-organized the manufacturing systems are, the more difficult it is to predict the performance of the systems. The performance can be cycle time, throughput, or WIP. If the capabilities and capacities of modules are defined as statuses, the module can work just like a single machine. Even though a module is a combination of several machines (and/or operators), it can be viewed as a single machine during the planning. If the capability and capacity of a module can be obtained based on the capabilities and capacities of the machines within the module, the scale of a smart manufacturing system can be reduced.

For each module in a smart manufacturing system, only the status of the module is exposed to higher level functions. The status of a module is calculated from its components. Fig.3 is an example of a module, which is a distributed system. The availability of different processes is based on the availability of machines in the module. The cycle time and throughput of different processes can be calculated based on the statuses of all the machines. Production planning is made based on the calculated capability and capacity of the module, and production control is realized by the self-organized module.

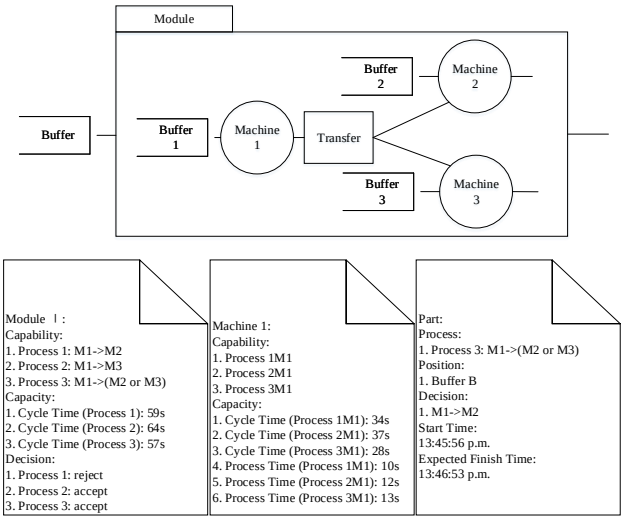

Fig. 3. Capability and Capacity of A Module

Plug-N-Play can be conducted for this case. Consider a situation where Machine 2 is down during the run time. First, Machine 2 is removed from the module. The module can detect the event and change its own status. Then the capability of the module is changed – Process 1 is removed, and Process 3 is changed. The system as a whole is still running normally without any modification and reconfiguration. After Machine 2 gets repaired and connected back to the module, the capacity and capability recover.

IV. Case Study of Smart Manufacturing Systmes Based on DPWS

In order to illustrate the task completion of smart manufacturing systems, we build a model of a distributed system to simulate a typical module in smart manufacturing systems, as shown in Fig.3. The model contains three layers: a physical system, a cyber system, and a user interface. The physical system is represented by a simulation model with three machines and three kinds of parts. The cyber system is constructed by DPWS through Java Multi Edition DPWS Stack (JMEDS) [20], which supports web service messaging, description, discovery, and evening on resource constrained devices. Several parts wait in a buffer before entering the module. The cooperation of the three machines enables the module to finish several different process tasks. Part A calls for Process 1 and requires a path from Buffer to Machine 1 and then to Machine 2. Part B calls for Process 2 and requires a path from Buffer to Machine 1 and to Machine 3. To finish Process 3, Part C goes from buffer to Machine 1 and then to Machine 2 or Machine 3. This model follows the specifications of DPWS and is programmed using Java. JMEDS is used to support DPWS in the Java environment. JMEDS provides Java classes, like DefaultDevice, DefaultClient, DefaultService and Operation, that contain basic functionalities and can be inherited to create new classes for an application.

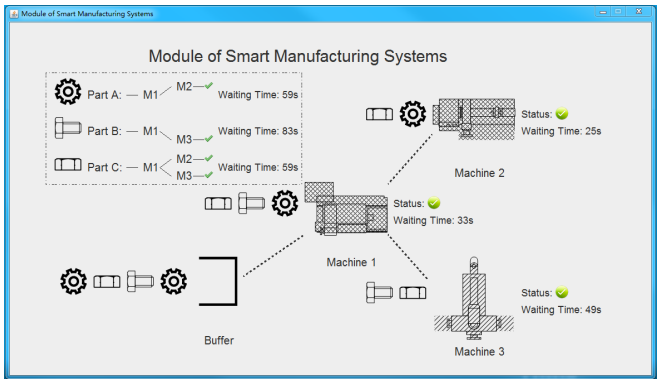

Fig. 4. Simulation of a Module Based on DPWS

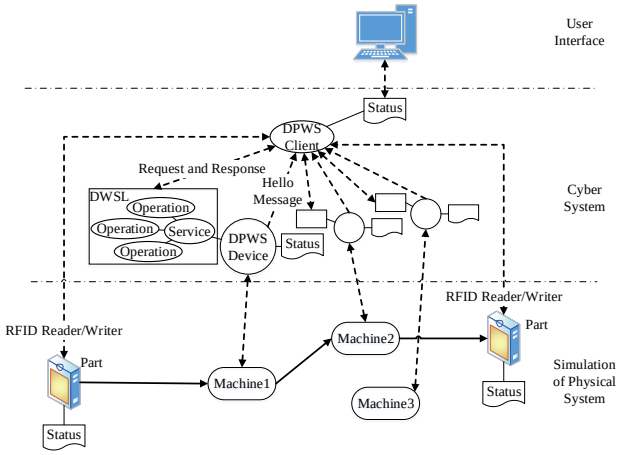

Fig. 5. Structure of Simulation Model

A. Simulation Model Structure

This model can be analyzed in different layers, as seen in Fig.5. The lowest layer is composed of machines and parts manufacturable by the machines. There is no endpoint in the network for parts, but parts can carry machine-readable information about their required processes. When a part enters a machine, the part triggers operations to be performed by that machine automatically. These events are recorded at the cyber system layer, and the system can monitor the status. The Java class is created to represent parts, and each part in the simulation is an instance of this class. In the lowest layer, machines read the information carried by each coming part and process the part. Additionally, machines record their status in their local databases, publish services through Web Service Description Language (WSDL), and respond to requests from clients. The status of machines includes their capacities and capabilities. The machine class in the simulation extends the DefaultDevice class in JMEDS. When a machine object in the simulation starts, a new thread is created for the object and an IP address is automatically assigned to the machine. The layer higher than the machine layer contains clients as modules. The client class extends DefaultClient class in JMEDS and represents modules. It collects data from machines by sending HTTP requests to services published by machines. A client determines which process is available based on the status of all machines in a module. It also gives the estimation of cycle time for each process. The client makes decisions for the module and gives instructions to machines. The module client encapsulates details in the machine layer and enables the module to act as a single machine. The client also exposes the status of the module to an even higher layer through an interface which constructs data readable by human users.

B. Simulation Run

- Step 1: A client is created to represent a module. The module is set to be a constant-work-in-process (CONWIP) system. The numbers of CONWIP and WIP are initialized.
- Step 2: Machine 1 is created and an IP is automatically assigned to Machine 1. A proper service is added to Machine 1, and proper operations are added to the service of Machine 1. Machine 1 publishes its service to the network through WSDL.
- Step 3: Machine 1 broadcasts a hello message with meta-data of Machine 1. The client receives hello message from Machine 1 and records the meta-data. The client sets the capability of Machine 1 and indicate that it's available.
- Step 4: Similarly, Machine 2 and Machine 3 are created. The capability of Machine 2 and that of Machine 3 are set. Since both Machine 1 and Machine 2 become available, Process 1 and Process 3 become available. Since both Machine 1 and Machine 3 become available, Process 2 becomes available.
- Step 5: When the processes become available, the first part goes into the module from the buffer. The number of WIP is increased by 1. The client requests status from all the machines and finds the best path for the part. The path information is recorded on the part.
- Step 6: When the part finishes its process on Machine 1, it goes to Machine 2 or Machine 3 based on its path.
- Step 7: When the part finishes all processes and leaves the module, the number of WIP is updated. A finished part triggers a decision to take a new part into the module.

Fig.6 shows a sequence diagram describing the simulation, where only Machine 1 is included and the transfer for parts from one machine to another is omitted. The sequence diagram gives a rough illustration of several Java objects communicating with each other to execute the simulation.

C. Discussion

The case study in this section mimics the behavior of a smart manufacturing system. Issues in communication technologies and production scheduling can be explored further in the future research. For communication technologies, the reliability of existing ICT needs to be analyzed to build a robust manufacturing system, because accidental failures or delays in communication channels might happen with multiple concurrent client requests. The potential consequences

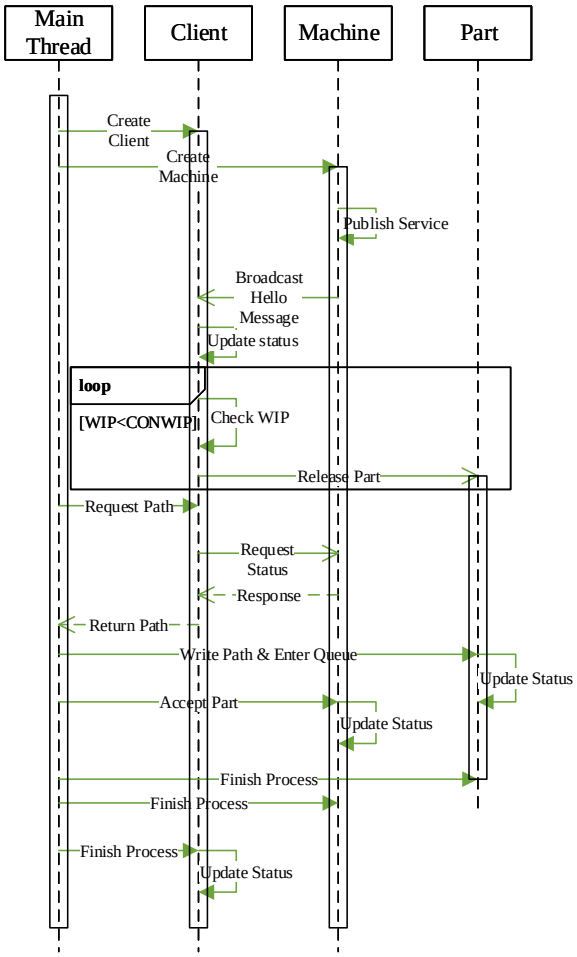

Fig. 6. Sequence Diagram

of failures and delays should be evaluated. For production scheduling, we call for better measurements and control decisions. More precise methods are needed to calculate the status of a module. It is also valuable to determine the best processing sequences for parts in buffers to better balance the production.

V. Conclusions

In order to make full use of intelligent entities in lower levels of a factory and at the same time keep the entire system under control, status-based decisions are proposed to balance self-organized controls and overall system performance. Specifically, capabilities and capacities are used as statuses to describe states of machines and modules. Future work will extend CPPS to cyber-manufacturing service models. Requirements on service models will be derived from the use cases. Existing manufacturing information modeling standards and ontologies will be used as bases to model CPPS capabilities, performance, states, and interfaces.

References

[1] F. Jammes, A. Mensch, and H. Smit, "Service-oriented device communications using the devices profile for web services," in *Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing*, pp. 1–8, ACM, 2005.

[2] Y. Lu, K. Morris, and S. Frechette, "Current standards landscape for smart manufacturing systems," *National Institute of Standards and Technology, NISTIR*, vol. 8107, 2016.

[3] Y. Lu, K. Morris, and S. Frechette, "Standards landscape and directions for smart manufacturing systems," in *Automation Science and Engineering (CASE), 2015 IEEE International Conference on*, pp. 998–1005, IEEE, 2015.

[4] H. Song, D. B. Rawat, S. Jeschke, and C. Brecher, *Cyber-physical systems: foundations, principles and applications*. Morgan Kaufmann, 2016.

[5] E. A. Lee, "Cyber-physical systems-are computing foundations adequate," in *Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, vol. 2, Citeseer, 2006.

[6] L. Wang, M. Törngren, and M. Onori, "Current status and advancement of cyber-physical systems in manufacturing," *Journal of Manufacturing Systems*, vol. 37, no. Part 2, pp. 517–527, 2015.

[7] L. Monostori, "Cyber-physical production systems: Roots, expectations and r&d challenges," *Procedia CIRP*, vol. 17, pp. 9–13, 2014.

[8] K.-F. Seitz and P. Nyhuis, "Cyber-physical production systems combined with logistic models–a learning factory concept for an improved production planning and control," *Procedia CIRP*, vol. 32, pp. 92–97, 2015.

[9] H. Kühnle and G. Bitsch, *Foundations & Principles of Distributed Manufacturing*. Springer, 2015.

[10] Y. Lu and F. Ju, "Smart manufacturing systems based on cyber-physical manufacturing services (cpms)," in *IFAC world congress*, 2017.

[11] A. W. Colombo, S. Karnouskos, J. M. Mendes, and P. Leitão, "Industrial agents in the era of service oriented architectures and cloud based industrial infrastructures," *Industrial Agents: Emerging Applications of Software Agents in Industry*, pp. 67–87, 2015.

[12] P. Reboredo and M. Keinert, "Integration of discrete manufacturing field devices data and services based on opc ua," in *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*, pp. 4476–4481, IEEE, 2013.

[13] J. Wermann, E. C. Moraes, and A. W. Colombo, "A service-oriented architecture implementation in the digital factory of the university," in *International Conference on Industrial Applications of Holonic and Multi-Agent Systems*, pp. 73–83, Springer, 2015.

[14] S. Weyer, M. Schmitt, M. Ohmer, and D. Gorecky, "Towards industry 4.0-standardization as the crucial challenge for highly modular, multi-vendor production systems," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 579–584, 2015.

[15] M. Loskyll, I. Heck, J. Schlick, and M. Schwarz, "Context-based orchestration for control of resource-efficient manufacturing processes," *Future Internet*, vol. 4, no. 3, pp. 737–761, 2012.

[16] P. Stephan, G. Meixner, H. Koessling, F. Floerchinger, and L. Ollinger, "Product-mediated communication through digital object memories in heterogeneous value chains," in *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pp. 199–207, IEEE, 2010.

[17] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, *et al.*, "Web services description language (wsdl) 1.1," 2001.

[18] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC unified architecture*. Springer Science & Business Media, 2009.

[19] W. J. Hopp and M. L. Spearman, *Factory physics*. Waveland Press, 2011.

[20] W. S. for Devices, "Stack: Ws4d-jmeds (java). http://ws4d.org/jmeds.."