

Autonomic smart manufacturing

Daniel A. Menascé, Mohan Krishnamoorthy & Alexander Brodsky

To cite this article: Daniel A. Menascé, Mohan Krishnamoorthy & Alexander Brodsky (2015) Autonomic smart manufacturing, Journal of Decision Systems, 24:2, 206-224, DOI: [10.1080/12460125.2015.1046714](https://doi.org/10.1080/12460125.2015.1046714)

To link to this article: <http://dx.doi.org/10.1080/12460125.2015.1046714>



Published online: 09 Jun 2015.



Submit your article to this journal [↗](#)



Article views: 36






View related articles [↗](#)



View Crossmark data [↗](#)

Autonomic smart manufacturing

Daniel A. Menascé* , Mohan Krishnamoorthy  and Alexander Brodsky 

Department of Computer Science, George Mason University, Fairfax, Virginia USA

(Received 30 October 2014; accepted 12 March 2015)

Smart manufacturing (SM) systems have to optimise manufacturing activities at the machine, unit or entire manufacturing facility level as well as adapting the manufacturing process on the fly as required by a variety of conditions (e.g. machine breakdowns and/or slowdowns) and unexpected variations in demands. This paper provides a framework for autonomic smart manufacturing (ASM) that relies on a variety of models for its support: (1) a process model to represent machines, part inventories and the flow of parts through machines in a discrete manufacturing floor; (2) a predictive queueing network model to support the analysis and planning phases of ASM; and (3) optimisation models to support the planning phase of ASM. In essence, ASM is an integrated decision support system for smart manufacturing that combines models of different nature in a seamless manner. As shown here, these models can be used to predict manufacturing time and the energy consumed by the manufacturing process, as well as for finding the machine settings that minimise the energy consumed or the manufacturing time subject to a variety of constraints using non-linear optimisation models. The diversity of models used affords different levels of integration and granularity in the decision-making process. An example of a car manufacturing process is used throughout the paper to explain the concepts and models introduced here.

Keywords: autonomic computing; smart manufacturing; queueing networks; optimization

1. Introduction

Smart manufacturing (SM) systems have to provide the capability for optimising manufacturing activities at the machine, unit or entire manufacturing facility level, as well as adapting the manufacturing process on the fly as required by a variety of conditions (e.g. machine breakdowns and/or slowdowns) and unexpected variations in demands. The process of making manufacturing more efficient has to be efficient itself.

In the past few years, there have been significant technological advancements in different areas of process interactions. Some of these areas include product manufacturing, supply chain and assembly lines. In some of these process interactions, there is a notion of a physical or virtual inventory that is used to store the intermediate products or materials, or the data associated with them, as the final product is being produced. Also, over time, these products change their state of production or completion. Hence, at certain time intervals, raw materials move from one machine to another to add or remove parts or modify their characteristics. We call such problems a buffered temporal flow process (BTFP; Krishnamoorthy, Brodsky, & Menascé, 2014). Such problems can

*Corresponding author. Email: menasce@gmu.edu

be found in many different areas of manufacturing and supply chain process interactions. One such area is that of discrete manufacturing, in which output items are produced from input items. The BTFP formalism presented in this paper can be used to model problems that deal with inventories and whose state varies at discrete time intervals.

Manufacturing companies are always looking for opportunities to reduce costs at the manufacturing floor. Limits on cost reduction on the material, labour and maintenance costs make manufacturing companies look toward other venues to reduce costs such as energy consumption, emissions, water consumption and waste disposal. This has resulted in a greater need for research into techniques to optimise the operations of the manufacturing floor while taking into account sustainability metrics. Government and environmental agencies are also enacting tougher laws and regulations to induce companies to take the sustainable manufacturing route. In addition, customers have become more aware of the looming dangers to the environment. This has resulted in increased customer demand for greener products.

At the same time, manufacturing floors are becoming more complex, more dynamic and more automated. There is then a need to turn manufacturing floors into self-managed operations so their performance can be continuously and automatically optimised as conditions change (e.g. machines break down, the demand for products changes). In this paper, we adapt the well-known autonomic computing paradigm (Kephart & Chess, 2003) developed for complex computer systems to smart manufacturing, resulting in autonomic smart manufacturing (ASM). In particular, we adopt the MAPE-K (monitor, analyse, plan, and execute based on knowledge) model (Kephart & Chess, 2003) of autonomic computing to ASM. We explain the approach with the help of examples that show how different kinds of models can support ASM.

This paper shows that the MAPE-K framework can be used to turn smart manufacturing into an integrated decision support system that enables decision makers, such as production engineers, to express queries and goals for a variety of metrics, while the ASM infrastructure constantly steers the manufacturing floor towards optimality by dynamically reacting to a variety of conditions. Examples of queries include ‘What is the throughput and processing time of the manufacturing floor for a given setting of the machines?’ or ‘What is the energy consumed by the manufacturing floor to produce a given number of parts/minute?’ A production engineer may, for example, set goals for energy consumption, processing time and processing throughput. An autonomic manufacturing floor will constantly optimise the settings of various parameters, such as machine speeds, in order to meet the goals.

Manufacturing execution systems (MES) provide control and management of, and information about, the manufacturing floor. An MES provides real-time reporting of actual manufacturing operations with performance results such as resource utilisation, resource availability and cycle time. If limits are exceeded, this is reported, and the person responsible for the process can view the event types in order to take corresponding measures (Dhandapani & Hu, 2006; McClellan, 1997; Meyer, 2009). Although MES provide good monitoring and reporting tools, our approach allows for not only setting control variables, but also adjusting these variables autonomically. Thus, the model knowledge is used to minimise human intervention in the analysis, planning and execution phases of ASM.

Modelling and analysis of BTFP-like systems can be performed using customised domain-specific solutions. These solutions are designed for the specific, limited setting of a manufacturing process, and would typically provide a graphical user interface that

is easy to use by end users. Examples include Melouk, Freeman, Miller, and Dunning (2013), Raska and Ulrych (2012). However, while these solutions may be efficient (in terms of both optimality of results and computational time), they are (1) typically not extensible to additional aspects of machines, processes and metrics; and (2) perform a 'silo' optimisation, which would not achieve the system-wide optimum if an extended underlying system needs to be optimised. Simulation-based systems have also been proposed that allow users to accurately model a system and its inner workings. These systems tend to be object-oriented, modular, extensible and reusable, and provide an easy-to-use graphical user interface. Tools like SIMULINK (Dabney & Harman, 2001) and Modelica-based ones (Fritzson, 2004) like JModelica (Åkesson, Gäfvert, & Tummescheit, 2009), Dymola (Brück, Elmqvist, Mattsson, & Olsson, 2002), and MapleSim (Hřebíček & Řezáč, 2008) allow users to model complex systems in mechanical, hydraulic, thermal, control and electrical power. However, simulation-based optimisation is significantly inferior to optimisation solutions based on mathematical programming (MP)/constraint programming (CP) in terms of optimality of results and computational complexity for problems that can be expressed using MP/CP models. This is because simulation-based optimisation amounts to a heuristically guided trial and error search, which does not utilise the mathematical structure of the underlying problem the way MP/CP methods do. Optimisation solvers and modelling languages based on MP and CP are often the technology of choice, when optimality and computational complexity are the priority. To use them, one would have to use an algebraic modelling language such as A Modelling Language for Mathematical Programming (AMPL; Fourer, Gay, & Kernighan, 2002), Optimisation Programming Language (OPL; Van Hentenryck, 1999), or General Algebraic Modelling System (GAMS; Boisvert, Howe, & Kahaner, 1985). However, MP and CP modelling present a significant challenge for engineers and business analysts to model. These techniques typically require an operations research (OR) expert to model a problem and express it in an algebraic modelling language like the ones mentioned. Additionally, these formal models are typically difficult to modify, extend or reuse. Finally Giaglis (2001) proposes an evaluation framework and a novel taxonomy of business process modelling (BPM) and information systems modelling (ISM) to assist decision makers in comparatively evaluating and selecting suitable modelling techniques. Although these models provide a generic framework for solving a large category of problems, they fail to provide (1) a uniform and reusable modelling framework, where the same model can be used for composition, computation, prediction and optimisation tasks; (2) a dynamic model that can be used in all phases of ASM; and (3) a simple and easy-to-use analytics that can respond to the changing environment of the manufacturing floor.

The optimisation of composed BTFP processes over a time horizon was recently considered by Krishnamoorthy et al. (2014), who reduced the problem to a mixed integer linear programming (MILP) formulation. However, in many manufacturing cases, it is sufficient to consider a steady-state optimisation. In these cases, queueing theory-based approaches are considerably more efficient computationally, as compared to approaches based on mathematical programming. Manufacturing systems have been extensively modelled with queueing theory. S. Yaghoubi et al. propose a queueing network for a finite capacity multi-class multi-stage assembly system with only one server in each service station (Yaghoubi, Noori, & Azaron, 2013). Another approach is the optimisation of the supply chain by modelling it as an analytical queueing network (Kerbache & Smith, 2004). Wu (2014) explores single and batch job arrivals and models a manufacturing process with the capability for serial or parallel batch processing.

In addition, intelligent agents have also been used to do dynamic scheduling and load balancing in manufacturing systems. For instance, (Balic & Cus, 2007; Nadoli & Biegel, 1993) use the concept of intelligent agents to simulate the manufacturing process so that the information returned can be used to balance the processes on the floor. All these approaches either model the manufacturing process as a simple mechanism with inventory that helps produce multiple classes of products, or the mathematical formulation of the model is too complicated to be used in a dynamic environment. It is required that the process model can capture all the characteristics of any discrete manufacturing floor, and that the mathematical representation of the model can be solved quickly and with ease so that the different phases of ASM can respond to the changing environment of the manufacturing floor. This is exactly the focus of this paper.

The main contributions of this paper are: (1) A framework for ASM that relies on a variety of models for its support; (2) A process model to represent machines, part inventories and the flow of parts through machines in a discrete manufacturing floor; (3) A predictive queueing network model to support the analysis and planning phases of ASM; and (4) Optimisation models to support the plan phase of ASM. The diversity of models used affords different levels of integration and granularity in the decision-making process.

The rest of this paper is organised as follows. Section 2 introduces the framework of ASM. The subsequent section describes three models used to support ASM: process models, predictive queueing models and optimisation models. These models are illustrated by an example of a car manufacturing floor. The models presented here allow for the prediction and optimisation of manufacturing time and energy consumed, among other metrics. Section 4 discusses work related to this paper. Finally, Section 5 presents some concluding remarks.

2. Autonomic smart manufacturing

The term ‘autonomic computing’ was first coined by IBM when its Senior Research Vice-President Paul Horn stated that ‘the main obstacle to further progress in IT is a looming complexity crisis’, at a keynote to the National Academy of Engineers in 2001. Complex software systems have tens of millions of lines of code, which require skilled IT professionals to install, configure, tune and maintain. Because organisations run many different large complex software systems, there is a need for the integration of heterogeneous systems. Moreover, the workload submitted to these complex systems varies dynamically in nature (different types of requests) and intensity (low to high transaction arrival rates) in unpredictable ways. All of these considerations imply that it is no longer feasible for human beings to control the myriad of configurable parameters that would make a complex system operate at its best at any point in time. Therefore, computers are needed to make complex systems self-managing, which implies that complex computer systems should be self-configuring and self-optimising.

A well-known framework to guide the design of autonomic computer systems is IBM’s MAPE-K model. In this paper, we make the case that complex smart manufacturing systems could benefit from the MAPE-K framework. We use Figure 1 to explain the details of the MAPE-K model as it applies to smart manufacturing. We call it ASM. ASM as defined here is broader in scope than MES, which control and manage the manufacturing floor and interact with enterprise resource planning (ERP) systems at the top, and individual machine and automation control at the bottom.

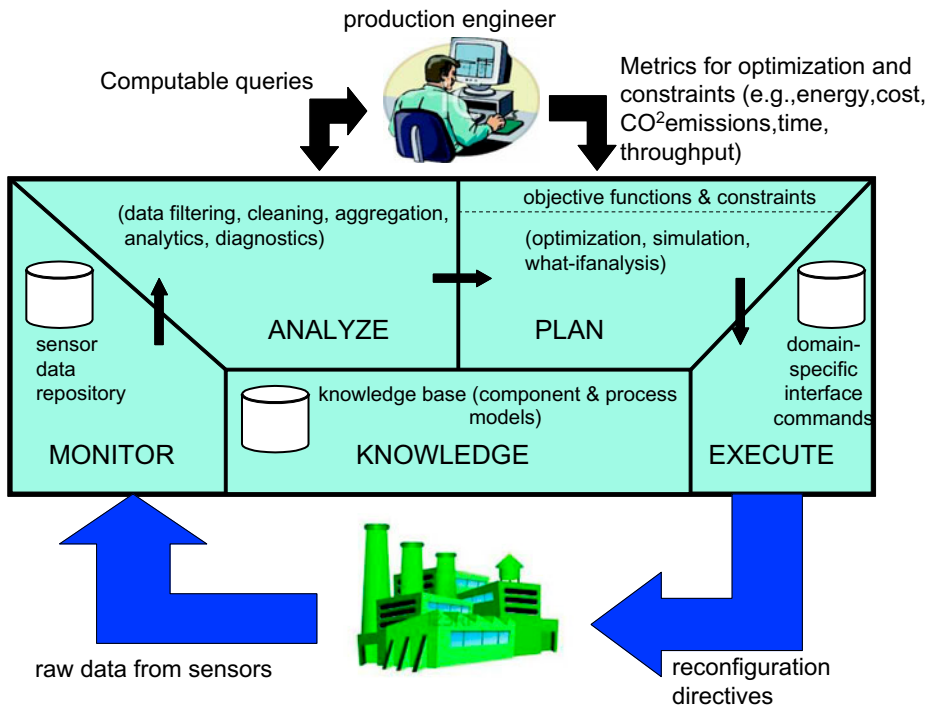


Figure 1. Autonomic computing paradigm.

The *monitor* phase of the MAPE-K loop is used to collect information through sensors, in an automated or semi-automated way, on individual machines as well as on the entire environment. Examples of data collected by sensors include speed and acceleration of specific machines, pressure applied to specific materials, density of input materials, CO_2 emissions (which can be measured or computed from other metrics), amount of water used, temperature on the manufacturing floor and energy consumed. Data collected by a variety of sensors is input into a sensor data repository. The monitoring phase is also used for preprocessing, cleaning, filtering and aggregating the raw data stored in the sensor data repository.

The *analyse* phase of the MAPE-K loop is used for analytics, what-if predictions, parameter calibration and diagnostics. Analytics is used to infer unknown relationships among data using data mining, statistical learning and machine learning techniques, for example. A production engineer may want to learn about a functional relationship between energy consumed and the throughput of the manufacturing process (in parts produced per hour). What-if predictions can be made based on relationships inferred through the analytics process. The production engineer may want to know what will be the increase in energy consumed if the manufacturing process throughput is increased by 15%. Diagnostics is the process of detecting faults (i.e. any deviation from what is considered to be normal behaviour) in the manufacturing process so that the best corrective actions can be determined (in the plan phase) and performed (in the execute phase). Examples of faults include breakdown of a machine, a machine operating at substandard performance in terms of speed or quality of parts produced, or excessive total CO_2 emissions. Production engineers may ask questions (i.e. computable queries)

whose answers can be obtained from the data made available in the analysis phase to determine how individual components are performing or how the manufacturing process as a whole is evolving.

The *plan* phase uses the results of the analysis phase to plan the optimal actions necessary for prognostics, i.e. to fix faults or to improve the efficiency of the manufacturing process as a whole. The plan phase requires the availability of models that represent, at the desired granularity level, the behaviour of individual components and of the manufacturing process as a whole. An example of a component-level model is a functional relationship between the energy consumed by a specific machine and the speed at which it produces parts. At the process level, a typical model would capture the flows of physical and information elements through the manufacturing process as well as the attributes of its components and flows. The models used by the plan phase need to support the descriptive, predictive and prescriptive analytical tasks, including optimisation, sensitivity analyses and answering what-if questions based on predictive models and not on data analysis. A variety of modelling approaches (e.g. deterministic and stochastic optimisation, simulation and queueing theory) may be used to determine action plans including optimal production schedules, optimal machine and process configurations and optimal process parameter settings. A plan may result, for example, in the following actions: stop machine A, and start machines B and C at 10,000 rpm. This action plan is passed on to the *execute* phase, which determines how to carry out the plan by looking up a database of domain-specific interface commands for the specific components involved (e.g. determining the command that should be sent to machine B to start it at 10,000 rpm).

Therefore, ASM is an integrated decision-support framework for smart manufacturing that is based on a variety of models including process models, performance models and optimisation models, as exemplified in the next section.

3. Models for ASM

Figure 1 also illustrates that a production engineer decides on the relevant metrics and their desired values so that the autonomic controller is capable of continuously going through the MAPE-K loop to maintain the industrial process operating at its best. A knowledge base (the K in MAPE-K) stores information about components, process models, optimisation and predictive models. This section describes these three types of models and provides an example used throughout the paper to illustrate the usefulness of these models.

3.1. Process Models for ASM

Figure 2 shows a simplified diagram for a process model for the first phase of a car manufacturing process. The diagram shows machines, represented by rectangles, and part buffers (aka inventories) connecting the machines. These buffers are used to store parts produced by one machine while waiting for the next machine in the sequence to be available to process that part. Aluminium coils are fed as input into the manufacturing floor and are sent to two uncoiling machines that work in parallel to uncoil the aluminium coils they receive. The flattened aluminium plates are sent to four different cutting machines: for the left side of the car, for the underbody, for the front and for the right side of the car. After being cut, the aluminium plates are sent to die press machines for each of the four sides (left side, right side, front and underbody). The front and underbody plates require additional processing by die press machines.

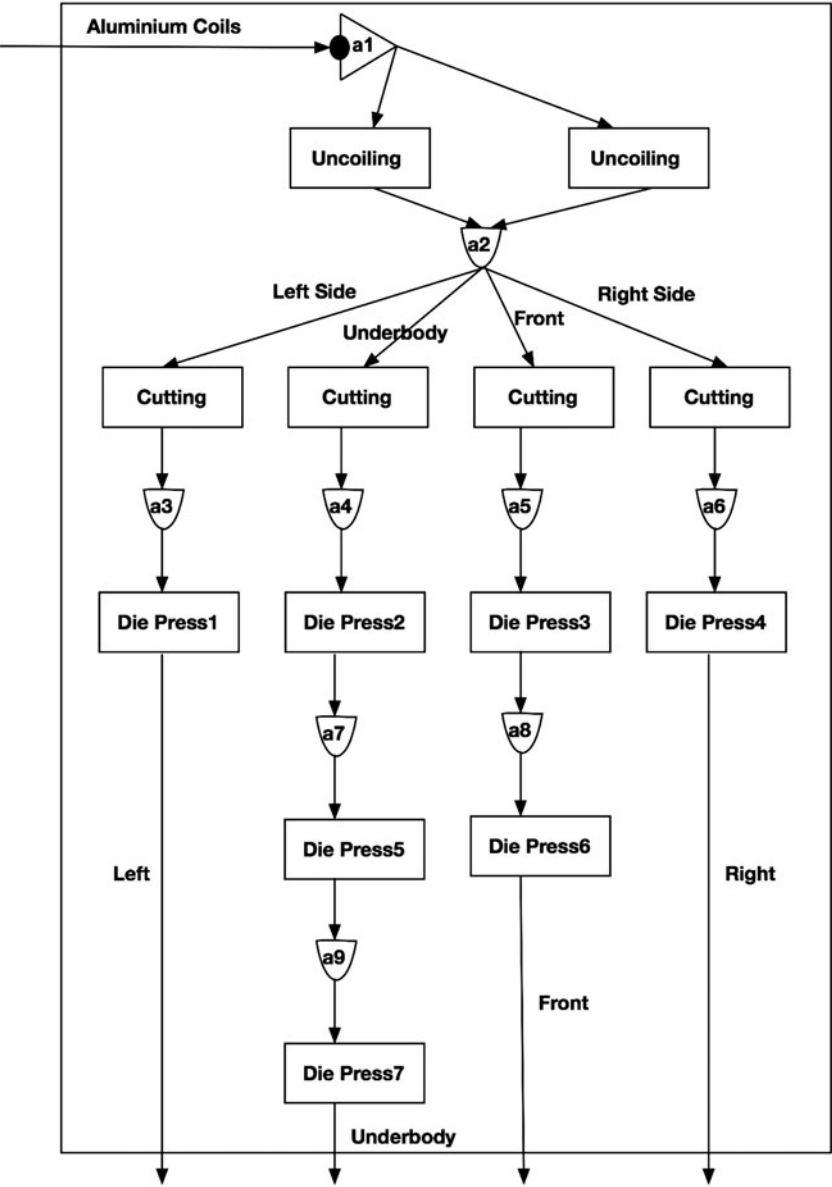


Figure 2. Process model for phase 1 of car manufacturing.

3.2. Predictive Queueing Models

This section presents models that can be used in the analysis phase of the MAPE-K loop as it applies to ASM. These models can be used to answer a variety of ‘what-if’ questions.

The process model of Figure 2 shows that each machine performs some processing on each part stored in the inventory of parts waiting for the machine. Each machine has a processing time, which is a function of its settings and of the physical

characteristics (e.g. density) of the part to be processed. These characteristics of the process model allow it to be modelled using a queueing network (QN) model (Menascé, Almeida, & Dowdy, 2004). A QN is a network of queues where the output of a queue feeds the input of another queue. Each queue consists of a machine, which has a finite speed for processing its input parts, and an input inventory of parts waiting to be processed. QN models allow one to predict the value of a series of performance metrics of the manufacturing floor as a function of a series of parameters. Examples of metrics of interest include: average manufacturing time (in hours), throughput (in cars/hr) and total energy consumed (in GJ/car).

Figure 3 shows the diagram for the QN that corresponds to the process model of Figure 2. Following the diagrammatic convention used in queueing theory, the processing component of a queue (the machine, in our case) is represented by a circle, and the waiting line (the inventory of parts waiting to be processed, in our example) is represented by a rectangle in front of the circle. The QNs considered here represent systems in steady state. We also assume that operational assumptions such as operational equilibrium, homogeneous arrivals and homogeneous service times (Buzen & Denning, 1980) are met.

The QN of Figure 3 has branches in which the flow of parts branches and then joins. This is called a fork-and-join (FJ) construct. While there are well-known solutions for QNs without FJ constructs (see e.g. Menascé et al., 2004), there are no exact solutions for QNs with FJ constructs. However, Firas Alomari and D. Menascé have presented a very accurate approximation for the time spent in a FJ construct (Alomari & Menascé, 2014). That approximation was extensively validated through simulation and covers the general cases of multiclass open and closed queueing networks with fork branches with non-homogeneous passage times as well as probabilistic forks. Consider that an FJ construct has M branches ($m = 1, \dots, M$). Let the time spent in branch m be T_m . Then, we renumber the branches as $v(1), v(2), \dots, v(M)$ such that $T_{v(1)} \geq T_{v(2)} \geq \dots \geq T_{v(M)}$. Then, according to the approximation in Alomari and Menascé (2014), the time T_{FJ} spent in an FJ construct is given by:

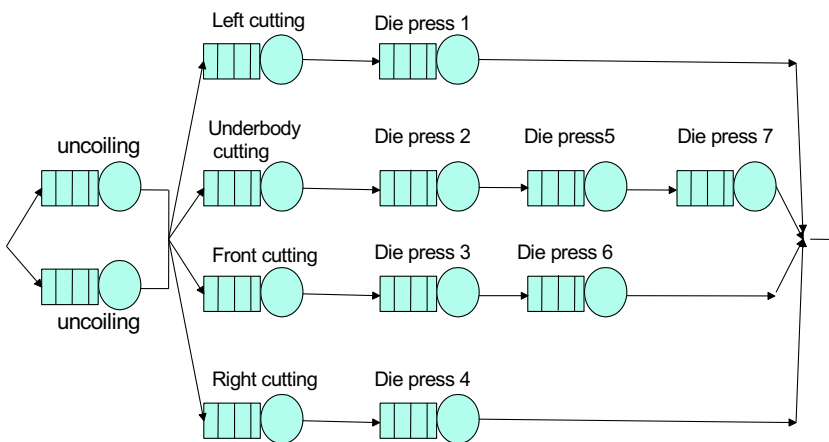


Figure 3. QN for process model for phase 1 of car manufacturing.

$$T_{FJ} = \sum_{m=1}^M \frac{1}{m} \times T_{v(m)} \quad (1)$$

For example, assume that an FJ construct has three branches ($M = 3$) and that the average execution times of each of the branches are $T_1 = 2$, $T_2 = 4$, and $T_3 = 3$. Then, $T_{FJ} = \left(\frac{1}{1}\right)T_2 + \left(\frac{1}{2}\right)T_3 + \left(\frac{1}{3}\right)T_1 = 4 + 1.5 + 2/3 = 6.17$.

According to well-known results of queueing theory (e.g. Menascé et al., 2004), the average time T_i spent at queue i with average service time S_i is given by:

$$T_i = \frac{S_i}{1 - \lambda S_i} \quad (2)$$

where λ is the average arrival rate of requests (e.g. car manufacturing requests in our example) to the manufacturing floor. Note that λS_i is the utilisation of machine i . Therefore, combining these results with the FJ approximation of Equation (1), we can derive the average time T_{man} to execute phase 1 of manufacturing the car as

$$T_{man} = T_{uncoil} + T_{cut-press} \quad (3)$$

where

$$T_{uncoil} = \frac{S_{uncoil}}{1 - \lambda \cdot S_{uncoil}} + \left(\frac{1}{2}\right) \frac{S_{uncoil}}{1 - \lambda \cdot S_{uncoil}}. \quad (4)$$

Equation (4) uses Equations (1)-(2) and assumes that both uncoiling machines have the same speed S_{uncoil} . In order to compute $T_{cut-press}$ we need to use the FJ approximation of Equation (1) again. But first, we need to compute the time spent in each branch (left, under, front, and right) as follows:

$$T_{left} = \frac{S_{left-cut}}{1 - \lambda \cdot S_{left-cut}} + \frac{S_{die-press1}}{1 - \lambda \cdot S_{die-press1}}$$

$$T_{under} = \frac{S_{under-cut}}{1 - \lambda \cdot S_{under-cut}} + \frac{S_{die-press2}}{1 - \lambda \cdot S_{die-press2}} +$$

$$\frac{S_{die-press5}}{1 - \lambda \cdot S_{die-press5}} + \frac{S_{die-press7}}{1 - \lambda \cdot S_{die-press7}}$$

$$T_{front} = \frac{S_{front-cut}}{1 - \lambda \cdot S_{front-cut}} + \frac{S_{die-press3}}{1 - \lambda \cdot S_{die-press3}} +$$

$$\frac{S_{die-press6}}{1 - \lambda \cdot S_{die-press6}}$$

$$T_{right} = \frac{S_{right-cut}}{1 - \lambda \cdot S_{right-cut}} + \frac{S_{die-press4}}{1 - \lambda \cdot S_{die-press4}} \quad (5)$$

We can now write down $T_{cut-press}$ as:

$$T_{cut-press} = \sum_{k=1}^4 \frac{1}{k} \times \frac{T_{v(k)}}{1 - \lambda \cdot T_{v(k)}} \quad (6)$$

where $T_{v(k)} \in \{T_{left}, T_{under}, T_{front}, T_{right}\}$ and $v(1), v(2), \dots, v(4)$ are numbered such that $T_{v(1)} \geq T_{v(2)} \geq \dots \geq T_{v(4)}$.

The QN formulation shown above considers a single product being processed at the manufacturing floor. However, in many real cases, the same manufacturing floor may be shared by several products (e.g. different models of the same car). This can be easily modelled by using multi-class queueing network models (Menascé et al., 2004). The FJ approximation used above (Alomari & Menascé, 2014) is also valid for multiclass QN models.

Table 1 shows an example of the parameters for phase 1 of the process model of Figure 2. The first column indicates all machines involved. The second column lists the service times, S_i , for each machine. The next column lists the utilisation $\rho_i = \lambda S_i$ of each machine assuming a value of λ equal to 3 cars/hr. Column 4 shows the average processing time T_i of a part at machine i . This time is the sum of the service time S_i plus the average time spent by a part in the inventory before it is acted upon by the machine (i.e. the average waiting time). As indicated in Equation (2), this is computed as $T_i = S_i / (1 - \rho_i)$.

The next three columns deal with power consumption. Each machine consumes some static power P_{stat}^i (column 5) even when it is idle (powered up but waiting for a part to become available in its input inventory), and some dynamic power P_{dyn}^i (column 6) when it is in operation (i.e. executing an operation). The dynamic power increases as the average service time decreases, because a smaller service time may be due to a machine rotating at a higher speed or employing higher temperature and/or force to perform its operation. In general, we can write P_{dyn}^i as a function of S_i . An example of such function is a power law function, such as:

$$P_{dyn}^i(S_i) = \alpha_i \cdot S_i^{-\beta_i} \quad \alpha_i > 0, \beta_i > 1. \quad (7)$$

Table 1. Parameters and Metrics for Phase 1 of Car Manufacturing for Figure 2. $\lambda = 3$ cars/hour.

| Machine | S_i min | ρ_i | T_i min | P_{stat} kJ/s | P_{dyn} kJ/s | P_{avg} kJ/s | E_i GJ |
|-------------------|-----------|----------|-----------|-----------------|----------------|----------------|----------|
| Uncoiling 1 & 2 | 4 | 0.20 | 5 | 40 | 30,000 | 114 | 0.03 |
| Left cutting | 10 | 0.50 | 20 | 60 | 7200 | 72 | 0.09 |
| Underbody cutting | 14 | 0.70 | 47 | 80 | 4898 | 56.2 | 0.16 |
| Front cutting | 12 | 0.60 | 30 | 80 | 6667 | 75.1 | 0.14 |
| Right cutting | 10 | 0.50 | 20 | 60 | 7200 | 72 | 0.09 |
| Die Press 1 | 10 | 0.50 | 20 | 18 | 2160 | 21.6 | 0.03 |
| Die Press 2 | 8 | 0.40 | 13.3 | 20 | 3750 | 30.3 | 0.02 |
| Die Press 3 | 9 | 0.45 | 16.4 | 26 | 5926 | 38.5 | 0.04 |
| Die Press 4 | 10 | 0.50 | 20 | 18 | 3118 | 23.2 | 0.03 |
| Die Press 5 | 8 | 0.40 | 13.3 | 32 | 6125 | 48.6 | 0.04 |
| Die Press 6 | 7 | 0.35 | 10.8 | 16 | 9796 | 37.1 | 0.02 |
| Die Press 7 | 8 | 0.40 | 13.3 | 18 | 3375 | 27.2 | 0.02 |

The average power consumption P_i^{avg} (column 7) can be computed as

$$P_i^{avg}(S_i) = (1 - \rho_i) \cdot P_i^{stat} + \rho_i \cdot P_i^{dyn}(S_i) \quad (8)$$

because when the machine is idle (this happens $(1 - \rho_i)$ of the time) the power consumption is P_i^{stat} and when the machine is operational (this happens ρ_i of the time) the power consumption is P_i^{dyn} . Finally, column 8 shows the the total energy E_i consumed per machine per car. This number is obtained as

$$E_i(S_i) = P_i^{avg}(S_i) \times T_i(S_i) \times 60/1000,000 \quad (9)$$

because the time it takes each part to flow through machine i is T_i minutes. During that time, the average power consumption of the machine is P_i^{avg} kJ/s. The factor 60 is used to convert minutes into seconds and the whole expression is divided by 1000,000 so that the resulting value of E is given in GJ. Note that the energy consumed by a machine, its average power and its processing time are all a function of its service time S_i . If we add all the values in the last column we obtain the total energy $E = \sum_{i=1}^K E_i(S_i)$ consumed per car for phase 1 of a car manufacturing. This value is 0.7 GJ.

We can now use the formulas in Equations (4–5) along with the values in Table 1 to compute the average time spent performing uncoiling and cutting and die pressing in each of the four branches. The results are $T_{uncoil} = 7.5$ min, $T_{left} = 40$ min, $T_{under} = 86.7$ min, $T_{front} = 57.1$ min, and $T_{right} = 40$ min.

In order to compute $T_{cut-press}$ according to Equation (6), we need to sort the processing times of the four branches of the cut and die press phase in decreasing order. According to the values above, we get $T_{v(1)} = T_{under}$, $T_{v(2)} = T_{front}$, $T_{v(3)} = T_{le}$, $T_{v(4)} = T_{right}$. Thus,

$$T_{cut-press} = T_{under} + \frac{1}{2}T_{front} + \frac{1}{3}T_{left} + \frac{1}{4}T_{right} = 138.6 \text{ min.} \quad (10)$$

If we add T_{uncoil} to $T_{cut-press}$ we get 146.1 min, which is the total time to complete phase 1 of the car manufacturing process.

Figure 4 has two y-axes. The left one represents the completion time in minutes for phase 1 of the car manufacturing, and the right axis represents the energy consumed in GJ per car for phase 1. These values are a function of the car throughput λ measured in cars/hr and represented on the x-axis. For all numerical examples used in this paper, we considered $\beta_i = 2$ for all machines. That graph shows, for example, that if we wanted to limit energy consumption per car to 0.78 GJ, the maximum throughput of the manufacturing floor should not exceed 3.6 cars/hr.

The model discussed above allows one to answer several interesting what-if questions in the analysis phase of the MAPE-K loop. Some examples include:

- Your production engineer is proposing to increase the throughput of the manufacturing floor by 40%. That means that λ would increase from 3 cars/hr to 4.2 cars/hr. You then want to answer the question: What is the total time to complete phase 1 of the manufacturing process and what is the total energy consumed per car under the new throughput? Using the model described above, we get that the total time for phase 1 is 859 min, a 5.9-fold increase in the total time, and the total energy

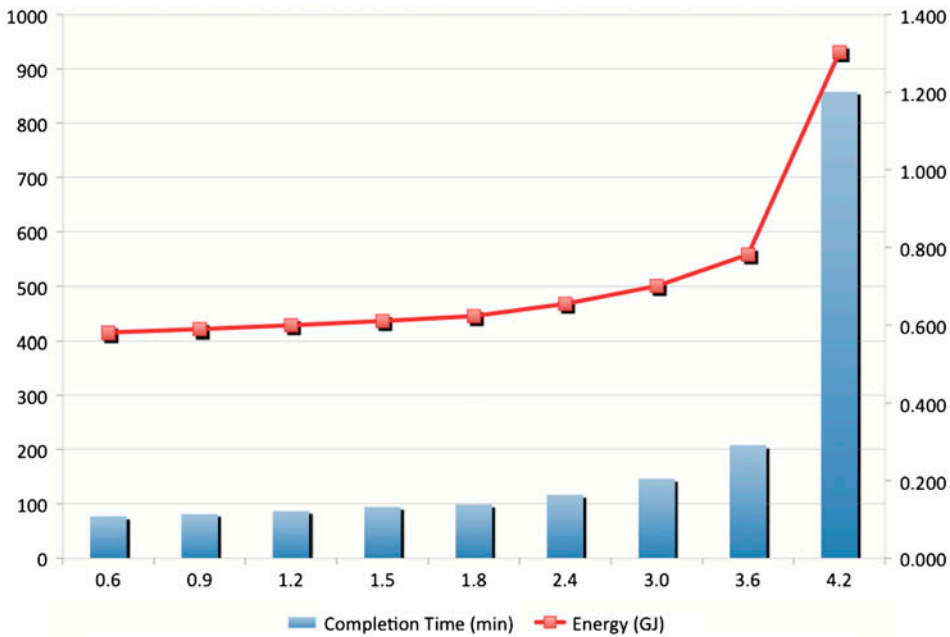


Figure 4. Left y-axis: completion time (in minutes) and right y-axis: energy consumed per car (in gjoules) vs. throughput (in cars/hr).

consumed per car is 1.3 GJ, a 1.9-fold increase in energy consumption. The reason for the significant increase in the total time is that the underbody cutting machine is the bottleneck of the manufacturing process. When λ is equal to 3 cars/hr, the utilisation of this machine is 70%, the highest among all machines (see Table 1). But, when λ goes up to 4.2 cars/s, the utilisation of this machine goes to 98%.

- What is the maximum throughput you can obtain from your manufacturing floor? Because the utilisation ρ_i of each machine cannot exceed 100%, the maximum throughput cannot exceed $1/S_i$ for all machines. So, the machine with largest service time value will constraint the maximum throughput. Thus, $\lambda < 1/(\max_{i=1}^K S_i)$. For the example of Table 1, the maximum throughput is $1/14 \times 60 = 4.29$ cars/hr.
- You decide that you have enough demand to increase your throughput to 4.2 cars/hr. You are considering investing in a more modern and faster underbody cutting machine. The service time for the new machine is 10 min. With this machine, the total time to complete phase 1 of the manufacturing process is 204 min (24% of the time spent with the original machine) and the total energy consumed is 0.66 GJ (50% of the energy consumed with the original machine). The reason for the decrease in energy consumption is that even though the faster underbody cutting machine consumes more dynamic power, its utilisation goes down from 98 to 70%, and its average processing time goes down from 700 to 33 min. This significantly reduces the energy consumed by this machine. Consider, on the other hand, that we decide to replace the uncoiling machines by machines 4 times faster, while still keeping the original underbody cutting machine. The total energy now goes up to 1.34 GJ from 1.3 GJ. In the previous case, the energy consumption decreased

because we made a high-utilisation machine faster. In the case of the uncoiling machines, we are making a low-utilisation machine faster, so the energy consumed increases. The reason is that, on the one hand, a decrease in the service time of a machine increases its dynamic power (see Equation 7), which increases its energy consumption. But, at the same time, a decrease its processing time reduces the energy consumption. So, depending on which particular effect dominates, it is more or less advantageous to replace a machine by a faster one.

3.3. Optimization Models

The plan phase of the MAPE-K loop for ASM requires models that can change the configuration of the manufacturing floor to make it operate in an optimal or near-optimal way as conditions change. This requires the use of optimisation models such as the ones discussed in this section. These models are based on the predictive analysis models discussed in the previous section.

The queueing and energy model discussed above can be used as the basis for solving a range of optimisation problems of the form described below. The decision variables are service times S_1, \dots, S_K of the machines on the manufacturing floor. The key performance indicators are the completion time $T = T(S_1, \dots, S_K)$ and the total energy consumption $E = E(S_1, \dots, S_K)$, which are both functions of S_1, \dots, S_K .

It is important to note that the service time decision variables S_1, \dots, S_K can serve as a proxy for machine control settings. For example, for a turning machine, it is easy to preprocess a function that would associate the service time to process a part with the optimal control setting of (1) depth of cut, (2) feed rate and (3) spindle speed.

A decision-maker, typically a manufacturing process engineer, would be able to formulate general optimisation problems of the form:

Minimise Objective (T, E)

s.t. Constraints (T, E)

$$U_i(S_i) < 1 \forall i$$

$$S_i \geq S_{\min} \forall i$$

where Objective (T, E) is an expression that may involve the completion time T and the total energy consumption E , and so, in turn, is a function of S_1, \dots, S_K , and Constraints (T, E) is a set of inequalities involving T and E . Below are a couple of examples of problems of this form:

- Objective $(T, E) = T$ and Constraints $(T, E) = \{E \leq E_{\max}\}$. This problem minimises the completion time within the total bound of energy consumption.
- Objective $(T, E) = E$ and Constraints $(T, E) = \{T \leq T_{\max}\}$. The goal here is to minimise the energy consumption within the total bound on completion time.
- Objective $(T, E) = 0.1 \times E + \max(0, 10.0 \times (T - T_{\max}))$ and Constraints $(T, E) = \{\}$. This problem minimises the total cost that consists of the energy cost, $0.1 \times E$, plus a penalty on the excess over the maximum time delay T_{\max} .

Furthermore, the optimisation framework can be used to construct a Pareto-optimal frontier in the (T, E) space. To do that, we can discretise the possible completion times,

say T_1, \dots, T_n , and for each such completion time T_i solve the optimisation problem where Objective $(T, E) = E$ and Constraints $(T, E) = (T \leq T_i)$. A graph with Pareto-optimal frontier can be useful to the decision-maker to decide on the tradeoff between energy consumption and the completion time.

Due to the nature of the objective functions and constraints involved in the problems described above, we need to solve continuous non-linear optimisation problems. Some examples of techniques for dealing with these problems include Sparse Nonlinear OPTimizer (SNOPT; Gill, Murray, & Saunders, 2002), Interior Point OPTimizer (IPOPT; Biegler & Zavala, 2009) and WORHP (Christof Büskens, 2013).

We now discuss a numerical example of a problem of the first type described above, i.e. minimise the total energy subject to a maximum completion time T_{max} . We used the following values: $\lambda = 3$ cars/hour, $T_{max} = 120$ minutes, and $S_{min} = 1$ minute and used the Generalised Reduced Gradient solver included with Microsoft Excel. In a typical ASM environment, optimisation solvers that can be invoked programmatically should be used. The results are given in Table 2. The minimum energy consumed per car is 0.407 GJ and the total car completion time in phase 1 is 32.2 min.

The second problem referred above is an inverse problem to the one discussed above, in that the goal is to minimise the total car manufacturing time for phase 1 while keeping the total energy consumed below a maximum value of E_{max} .

As an example, we solved this problem using $E_{max} = 0.5$ GJ. The minimum value of $T(S_1, \dots, S_K)$ is 13.1 min for the following values of service times: $S_{uncoil} = 1.16$ min, $S_{left-cut} = 2.31$ min, $S_{under-cut} = 1.85$ min, $S_{front-cut} = 1.95$ min, $S_{right-cut} = 2.21$ min, $S_{die-press1} = 1.66$ min, $S_{die-press2} = 1.11$ min, $S_{die-press3} = 1.63$ min, $S_{die-press4} = 1.81$ min, $S_{die-press5} = 1.35$ min, $S_{die-press6} = 1.69$ min, and $S_{die-press7} = 1.06$ min.

If we compare the solutions of the two problems, we see that by increasing the energy consumed by 23% (from 0.407 to 0.5 GJ), we are able to reduce the car manufacturing time to 41% (from 32.2 to 13.1 min) of its time under the previous setting of the machines.

The above optimisation problems are just two examples of what can be done to keep the manufacturing floor operating at an optimal level, where the optimality criterion is defined by the production engineer.

Table 2. Optimal Service Time Values for Phase 1 of Car Manufacturing for Figure 2. $\lambda = 3$ cars/hour.

| Machine | S_i min | ρ_i | T_i min | P_{stat} kJ/s | P_{dyn} kJ/s | P_{avg} kJ/s | E_i GJ |
|-------------------|-----------|----------|-----------|-----------------|--------------------|----------------|----------|
| Uncoiling 1 & 2 | 3.11 | 0.16 | 3.7 | 40 | 4.97×10^4 | 150.5 | 0.033 |
| Left cutting | 3.11 | 0.16 | 3.7 | 60 | 7.45×10^4 | 225.5 | 0.050 |
| Underbody cutting | 3.11 | 0.16 | 3.7 | 80 | 9.92×10^4 | 300.5 | 0.066 |
| Front cutting | 3.11 | 0.16 | 3.7 | 80 | 9.95×10^4 | 300.9 | 0.066 |
| Right cutting | 3.11 | 0.16 | 3.7 | 60 | 7.44×10^4 | 225.3 | 0.050 |
| Die Press 1 | 3.11 | 0.16 | 3.7 | 18 | 2.24×10^4 | 67.7 | 0.015 |
| Die Press 2 | 3.12 | 0.16 | 3.7 | 20 | 2.47×10^4 | 75.0 | 0.017 |
| Die Press 3 | 3.80 | 0.19 | 4.7 | 26 | 3.32×10^4 | 97.4 | 0.027 |
| Die Press 4 | 3.70 | 0.19 | 4.5 | 18 | 2.28×10^4 | 67.3 | 0.018 |
| Die Press 5 | 3.14 | 0.16 | 3.7 | 32 | 3.98×10^4 | 120.3 | 0.027 |
| Die Press 6 | 4.76 | 0.24 | 6.2 | 16 | 2.12×10^4 | 59.7 | 0.022 |
| Die Press 7 | 3.10 | 0.16 | 3.7 | 18 | 2.25×10^4 | 67.9 | 0.015 |

3.4. Model integration for decision support

The three models described in the previous subsections are examples of the type of models that would be part of the knowledge base of the MAPE-K framework for smart manufacturing. Other models could include statistical learning models and simulations models. These models provide integrated and seamless support for decision-makers such as production engineers.

For example, a decision-maker may pose the question ‘What is the maximum throughput (in parts/s) of the manufacturing floor for the given setting of the machines and for a different setting that speeds up some of the machines?’ This query would be answered through predictive performance models such as the one described in Section 3.2. The decision-maker could then ask the question ‘What is the new processing time of a part if the manufacturing process is changed so that new machines are added in order to increase the level of parallelism in one of the manufacturing stages?’ Answering this query would entail modifying the process model of Section 3.1 and using a predictive performance model such as the one in Section 3.2.

In other circumstances, the production engineer may specify that the energy consumed be minimised while maintaining the manufacturing time of a part below a certain threshold even when some machines experience degraded performance. Achieving this goal requires a continuous use of the optimisation models discussed in Section 3.3, which require the predictive performance models of Section 3.2.

Therefore, the control loop provided by the MAPE-K framework provides for dynamic and integrated model-based decision-making.

4. Related work

Most previous research efforts tried to develop autonomic models for scheduling, resource allocation and task allocation for manufacturing managers using the intelligent agent approach. For instance, the dynamic scheduling problem in manufacturing floors is addressed in Pereira & Madureira (2010) by combining the methods of multi-agent systems, autonomic computing and nature-inspired techniques. Madureira, Santos, & Pereira (2008) propose a multi-agent autonomic and bio-inspired framework with self-managing capabilities to solve complex scheduling problems using cooperative negotiation. Madureira & Pereira (2010) proposed the use of multi-agent systems to support dynamic and distributed scheduling in manufacturing systems in order to reduce the complexity of managing manufacturing systems and human interference. In Kota, Gibbins, & Jennings (2012), autonomous agents enable the modification of structural relations to achieve a better allocation of tasks in a simulated task-solving environment. BastosBastos, de Oliveira, & de Oliveira (2005) present an autonomic solution based in a multi-agent model for resource allocation in a manufacturing environment. Lee, Choy, Law, & Ho (2014) aim to provide effective and timely decision-making for resource allocation by using a database management system (DBMS) and fuzzy logic to analyse data for intelligent decision-making, and radio frequency identification (RFID) for result verification. Delen & Pratt (2006) developed an intelligent decision support system for manufacturing systems (IDSS-MS), capable of providing structuring, analysis-tool selection, autonomic model generation and execution for a set of problem-symptoms. Finally, Park & Tran (2012) propose an autonomous manufacturing system based on a swarm of cognitive agents (AMS-SCA) in order to adapt to the disturbances autonomously based on the reaction of each agent or the

cooperation among them. While these approaches successfully produce an autonomic model for a complex manufacturing floor, they are limited by the fact that queries can only be posed against the entire model. This limits the algorithms from breaking the complex model into small sub-models and posing queries to these sub-models to assess the problem at varying levels of granularity. Also, there is a lack of visualisation support for these models that restricts managers from monitoring the autonomic actions from time to time. Our approach provides a simple additive mathematical formulation, based on queueing networks, for a complex manufacturing model that can easily be supported by a visualisation software.

Some research efforts also focus on the use of approximate queueing models to compute capacity requirements or the control variables of the model, or for performing optimisation on these models. For instance, Avsar & Zijm (2014) present an approximated queueing model for base-stock controlled multi-stage production-inventory systems with capacity constraints by replacing some state-dependent conditional probabilities using recursive algorithms. Schönlein, Makuschewitz, Wirth, & Scholz-Reiter (2013) suggest the use of fluid network analysis to quantify robustness using a single number called the stability radius, which represents the worst-case measure of the magnitude of the smallest shift of the expected value of the inter-arrival and/or service times distributions. Tüysüz & Kahraman (2010) devise an approach for the modelling and analysis of time-critical, dynamic and complex systems using stochastic Petri-nets together with fuzzy sets. Finally, Xia & Cao (2012) use perturbation analysis and sensitivity-based optimisation to enable performance optimisation of queueing systems. These QN models can be used to solve complex problems quickly to provide approximate results. But there is a lack of a general approach that works for a domain of problems in manufacturing or supply chain. Our approach overcomes this limitation by providing a generic autonomic framework for any discrete manufacturing domain problem that can be used to make queries in the monitoring, analysis, planning and execution phases. Another limitation of the papers discussed here is that it is not possible to reuse the components of the QN without paying the penalty of having to remodel the underlying mathematical formulations. In our approach, it is possible to compose QN for different process models without this penalty. This is especially useful in the manufacturing domain where explanations and changes to the floor happen frequently.

5. Concluding remarks

This paper presents an autonomic framework for smart manufacturing using the MAPE-K model used in autonomic computing. The K in MAPE-K stands for knowledge and includes a variety of models such as process models, predictive QN models, energy consumption models and optimisation models. An example shows how to map a process model to a QN model that can be used to predict overall manufacturing time, machine utilisation and energy consumption, and overall energy consumed in the manufacturing process. The QN and energy models can then be used to solve optimisation problems of interest as shown in the examples discussed in the paper. The MAPE-K framework provides for dynamic and integrated model-based decision-making.

We are currently working on several extensions to the current work. For example, we plan to investigate algorithms to learn, from collected data, the relationship between power and machine service time. We are also in the process of designing a graphical user interface (GUI)-based tool that will hide the complexity of building QN models

from the production engineer. Such models will be built and solved automatically from a visual description of the process model. We are also investigating how to extend our approach to deal with composability, i.e. a manufacturing process is seen as part of a larger manufacturing process. For that purpose, we will examine the use of the theory of decomposability (Courtois, 1985).

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was partially supported by National Institute of Standard and Technology [grant number 70NANB12H277].

ORCID

Daniel A. Menascé  <http://orcid.org/0000-0002-4085-6212>

Mohan Krishnamoorthy  <http://orcid.org/0000-0002-0828-4066>

Alexander Brodsky  <http://orcid.org/0000-0002-0312-2105>

References

- Åkesson, J., Gäfvert, M., & Tummescheit, T. (2009). Jmodelica-an open source platform for optimization of modelica models. In *Proceedings of MATHMOD 2009–6th Vienna international conference mathematical modelling*. TU Wien, Vienna, Austria.
- Alomari, F., & Menascé, D. (2014). Efficient response time approximations for multiclass fork and join queues in open and closed queueing networks. *Parallel and Distributed Systems, IEEE Transactions on*, 25, 1437–1446.
- Avsar, Z. M., & Zijm, W. H. (2014). Approximate queueing models for capacitated multi-stage inventory systems under base-stock control. *European Journal of Operational Research*, 236, 135–146.
- Balic, J., & Cus, F. (2007, September). Intelligent modelling in manufacturing. *Achievements in Materials and Manufacturing Engineering*, 24, 340–349.
- Bastos, R. M., de Oliveira, F. M., & de Oliveira, J. P. M. (2005). Autonomic computing approach for resource allocation. *Expert Systems with Applications*, 28, 9–19.
- Biegler, L., & Zavala, V. (2009). Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, 33, 575–582.
- Boisvert, R. F., Howe, S. E., & Kahaner, D. K. (1985). Gams: A framework for the management of scientific software. *ACM Transactions on Mathematical Software*, 11, 313–355.
- Brück, D., Elmqvist, H., Mattsson, S. E., & Olsson, H. (2002, March 18–19). Dymola for multi-engineering modeling and simulation. In *Proceedings of modelica* (pp. 55-1–55-8). Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), Oberpfaffenhofen.
- Buzen, J., & Denning, P. (1980). Operational treatment of queue distributions and mean-value analysis. *Computer Performance*, 1, 6–15.
- Christof Büskens, D. W. (2013). *The ESA NLP solver WORHP*. New York, NY: Springer.
- Courtois, P. J. (1985, June). On time and space decomposition of complex structures. *Communications of the ACM*, 28, 590–603.
- Dabney, J. B., & Harman, T. L. (2001). *Mastering simulink 4* (1st ed.). Upper Saddle River, NJ: Prentice Hall PTR.
- Delen, D., & Pratt, D. B. (2006). An integrated and intelligent DSS for manufacturing systems. *Expert Systems with Applications*, 30, 325–336.
- Dhandapani, S., & Hu, W. (2006). *Performance analysis and improvement for manufacturing execution systems*. University of North Dakota, Grand Forks, North Dakota.

- Fourer, R., Gay, D., & Kernighan, B. (2002). *AMPL: A modeling language for mathematical programming*. Cengage Learning. 517 pages, ISBN 0-534-38809-4.
- Fritzson, P. (2004). *Principles of object-oriented modeling and simulation with modelica 2.1*. Piscataway, NJ: Wiley-IEEE Press.
- Giaglis, G. M. (2001). A taxonomy of business process modeling and information systems modeling techniques. *International Journal of Flexible Manufacturing Systems*, 13, 209–228. doi:10.1023/A:1011139719773
- Gill, P., Murray, W., & Saunders, M. (2002). SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12, 979–1006.
- Hřebíček, J., & Řezáč, M. (2008, June). Modelling with maple and maplesim. In *Proceedings 22nd European conference modelling and simulation ECMS* (pp. 60–66). Nicosia, Cyprus.
- Kephart, J., & Chess, D. (2003). The vision of autonomic computing. *IEEE Internet Computing*, 36, 41–50.
- Kerbache, L., & Smith, J. M. (2004). Queueing networks and the topological design of supply chain systems. *International Journal of Production Economics*, 91, 251–272.
- Kota, R., Gibbins, N., & Jennings, N. R. (2012). Decentralized approaches for self-adaptation in agent organizations. *ACM Transactions on Autonomous and Adaptive Systems*, 7(1), 1–28.
- Krishnamoorthy, M., Brodsky, A., & Menascé, D. A. (2014). *Temporal manufacturing query language (TMQL) for domain specific composition, what-if analysis, and optimization of manufacturing processes with inventories* (Tech. Rep. No. GMU-CS-TR-2014-3). Fairfax, VA: Department of Computer Science, George Mason University. Retrieved from <http://cs.gmu.edu/tr-admin/papers/GMU-CS-TR-2014-3.pdf>
- Lee, C., Choy, K., Law, K., & Ho, G. (2014). Application of intelligent data management in resource allocation for effective operation of manufacturing systems. *Journal of Manufacturing Systems*, 33, 412–422.
- Madureira, A., & Pereira, I. (2010). Intelligent bio-inspired system for manufacturing scheduling under uncertainties. In *Hybrid intelligent systems (HIS), 2010 10th international conference on* (pp. 109–112), August 2010, Atlanta, GA.
- Madureira, A., Santos, F., & Pereira, I. (2008, July). Self-managing agents for dynamic scheduling in manufacturing. In *Proceedings of the 10th annual conference companion on genetic and evolutionary computation* (pp. 2187–2192). Atlanta, GA.
- McClellan, M. (1997, August). Applying Manufacturing Execution Systems. Library of Congress, CRC Press.
- Melouk, S., Freeman, N., Miller, M., & Dunning, M. (2013). Simulation optimization-based decision support tool for steel manufacturing. *International Journal of Production Economics*, 141, 269–276.
- Menascé, D. A., Almeida, V. A., & Dowdy, L. W. (2004). *Performance by design: Computer capacity planning by example*. Upper Saddle River, NJ: Prentice Hall.
- Meyer, H. (2009). *Manufacturing execution systems: Optimal design, planning, and deployment*. The McGraw-Hill Companies, Inc. New York, NY.
- Nadoli, G., & Biegel, J. E. (1993). Intelligent manufacturing-simulation agents tool (IMSAT). *ACM Transactions on Modeling Computer Simulation*, 3, 42–65.
- Park, H.-S., & Tran, N.-H. (2012). An autonomous manufacturing system based on swarm of cognitive agents. *Journal of Manufacturing Systems*, 31, 337–348.
- Pereira, I., & Madureira, A. (2010, July). Self-optimizing through CBR learning. In *Evolutionary computation (CEC), 2010 IEEE congress on* (pp. 1–8). Barcelona, Spain.
- Raska, P., & Ulrych, Z. (2012). Simulation optimization in manufacturing systems. *23rd International DAAAM Symposium*, 23, 221–224.
- Schönlein, M., Makuschewitz, T., Wirth, F., & Scholz-Reiter, B. (2013). Measurement and optimization of robust stability of multiclass queueing networks: Applications in dynamic supply chains. *European Journal of Operational Research*, 229, 179–189.
- Tüysüz, F., & Kahraman, C. (2010). Modeling a flexible manufacturing cell using stochastic Petri nets with fuzzy parameters. *Expert Systems with Applications*, 37, 3910–3920.
- Van Hentenryck, P. (1999). *The OPL optimization programming language*. Cambridge, MA: MIT Press.
- Wu, K. (2014). Taxonomy of batch queueing models in manufacturing systems. *European Journal of Operational Research*, 237, 129–135.

- Xia, L., & Cao, X.-R. (2012). Performance optimization of queueing systems with perturbation realization. *European Journal of Operational Research*, 218, 293–304.
- Yaghoubi, S., Noori, S., & Azaron, A. (2013). Lead time control in multi-class multi-stage assembly systems with finite capacity. *Computers & Industrial Engineering*, 66, 808–817.