DEVS FORMAL MODELING AND SIMULATION IN MANUFACTURING SYSTEMS

Cristina Ruiz Martin[a] and Gabriel Wainer[a]

[a] Department of Systems and Computer Engineering, Carleton University, Canada
{cristinaruizmartin, gwainer}@sce.carleton.ca

ABSTRACT

Bottlenecks are a major problem for manufacturing companies because they limit the throughput of the production line. Although analytical methods have been widely studied, these methods are impractical in many cases, and simulation-based approaches, where a model of the system is developed, are needed. In this work, we show how to use DEVS as a tool to model and simulate manufacturing systems. More specifically, we propose to use it to apply the Theory of Constraints, identifying bottlenecks in the manufacturing plants and large amounts of Work in Progress.

Keywords: DEVS, bottlenecks, manufacturing simulation.

1 INTRODUCTION

As highlighted in the book "The Goal" [1], bottlenecks are a major problem for manufacturing companies because they limit the throughput of the production line. Additionally, bottlenecks increase the amount of Work in Progress (WIP) in the factory, increasing inventory costs. Even before this problem was widely recognized, some companies started to develop methods and philosophies focused on how to reduce WIP, eliminate bottlenecks, and reduce inventory costs.

For example, in the 1960s and 1970s, Toyota introduced the Toyota Production System (TPS), also known as Just-In-Time (JIT) [2]. TPS or JIT is a methodology that aims to reduce the production times of goods, but also the response times to clients and from suppliers. In the 1990s, these two terms evolved into the concept of Lean Manufacturing. The final aim of Lean Manufacturing is to improve quality and reduce waste, production times, and costs. These goals are achieved by applying five principles [3]:

- **Specify Value**: clients should highlight which capabilities are valuable and at which price
- **Value Stream**: the company needs to define all the actions needed to deliver the product to the clients. They must also identify which of these activities add value to the product. The activities that do not add value should be classified as avoidable (and be eliminated) or unavoidable.
- **Create flow**: define the value chain in such a way that there is a smooth progression from the beginning to the end. This involves eliminating bottlenecks as they interrupt the flow. Everything that interrupts the flow contributes to Lean waste and reduces the value for the client.
- **Pull**: clients pull the product order instead of the traditional push philosophy. The production lines produce to fulfill the orders from customers, both internal (e.g. the next workstation in the production line) and externals (i.e. clients). This minimizes the WIP and the final product inventory.
- **Perfection**: the system is continuously monitored and measured so we can improve it over time.

The main benefits of applying these principles within a manufacturing system not only include a reduction of cost and production time, but also an increase of value for the customer and a reduction of waste.

To achieve the above-mentioned principles, several management disciplines have been proposed. For example, Supply Chain Management (SCM) focuses on the planning process, execution, and operation control of all the activities needed to satisfy the needs of the clients as efficiently as possible [4]. Enterprise

Proc. of the 2024 Annual Simulation Conference (ANNSIM'24), May 20-23, 2024, American University, DC, USA
C. Ruiz-Martin, B. Oakes and R. Cárdenas, eds.

Resource Planning (ERP) is a pioneering enabling and support system that helps with the effective implementation of business process management principles [5]. It often uses real-time data and the aid of software and technological tools. Manufacturing Execution Systems (MES) are software tools to help manufacturing management, from the top level to the operations on the plant [6].

Within those disciplines, different theories have appeared and applied. One of them is the Theory of Constraints (TOC) [7], which views manufacturing processes or organizations as "chains". In these chains, the strength of the system is determined by the weakest element (i.e., the bottleneck). The weakest element is the limiting factor for the organization, and it requires to be strengthened. No matter how strong a chain is, it always will have a weak element that can be improved, pushing forward the limits of the systems. Although it was introduced in 1986, TOC is still widely used [8].

Although this theory is useful, nowadays, bottlenecks are not identified manually. The detection methods can be classified into two categories: analytical and simulation-based. For analytical methods, the system performance is assumed to be described by a statistical distribution. However, this analytical approach cannot be applied to real production processes with complex dynamic structures. In those cases, the simulation-based approach, where a model of the system is developed, is needed [9]. As analytical methods are impractical in many cases, simulation-based methods are widely extended in the manufacturing field.

Two simulation-based methods are popular for studying manufacturing systems: System Dynamics (SD) and Discrete-Event Simulation (DES) [10]. Although Brailsford and Hilton [11] claim that DES is more used and suitable for modeling problems at an operational/tactical level, whereas SD is more suited to modeling problems at a strategic level, Tako and Robinson [10] did not find any evidence of this claim after reviewing more than 120 papers. In fact, they found that DES was used more frequently than SD.

The DEVS formalism (*Discrete EVent System Specification*) is a formal Modeling and Simulation methodology for DES [12]. It is derived from Systems Theory, and it provides several advantages in the field of modeling and simulation. It is a methodology to develop hierarchical models in a modular fashion. This modularity allows model reuse and, thus, reduces the development time and testing. The model definition, implementation, and simulation are separated. Hence, the same model can be implemented on different platforms, facilitating the reliability of models and results. Moreover, the simulation algorithm for DEVS models has already been verified and validated. In fact, it has been shown that SD models can be defined using DEVS [13].

Based on these facts, we here show how to use DEVS as a tool to model and simulate manufacturing systems. More specifically, we propose to use it to apply the TOC to find the weakest in the manufacturing plant. The objective of this work is to explain how to use the DEVS formalism to help managers make sound decisions in the design and control of manufacturing systems. We will focus the explanation on the identification of bottlenecks and the reduction of WIP as they are the main factors that affect the performance of the factories in terms of inventory cost and throughput.

The structure of the paper is as follows. In Section 2, we explain related works that apply DEVS formalism in the manufacturing and production field and summarize the DEVS methodology and tools used in this research. In Section 3, we focus on a first case study to show how to apply DEVS to study the workflow in a Pharmaceutical Plan. In Section 4, we present a second case study where we focus on a production line with the aim of identifying bottlenecks. Finally, in Section 5, we present the conclusions and the future research lines.

2 RELATED WORK

DEVS has been used to study and solve problems in different fields, such as biology, defense, environmental science, construction, architecture, or networking among others [14]. The application fields also include Manufacturing Engineering, as we detail in the rest of this section.

Business and manufacturing managers need to model the processes that happen within the organization in order to share them with other stakeholders and optimize them. These processes can easily be modeled as standard workflows using the Business Process Model and Notation (BPMN). The advantage of BPMN is that it is easy for stakeholders and managers to understand. However, although there have been advances in the field [15], there is no widely accepted simulation algorithm behind the notation to study the dynamics of these workflows. To fill this gap, [16] introduced a transformation of BPMN models into DEVS models, which can be simulated. In this context, they present an extension to the SLMToolBox tool to support this transformation and the simulation of the workflows [17]. The research presented in [18] is also focused on the transformation of engineering workflows into DEVS but they do not require to use the BPMN approach for their definition.

Not only the definition and simulation of workflows within a company are important. The product design and services offered by the manufacturers are key in providing value to the clients. DEVS has been applied in this context to create Product-Service Systems models to be simulated in different service scenarios based on G-DEVS/HLA. The simulation results support decision-makers in choosing the right design scenario to be manufactured [19] and the design phase, including the behavior of the clients in the models before the product is launched to the market [20].

When studying a company and its manufacturing processes, it is also important to consider that there may be differences between the designed processes and the processes actually implemented in the company. To be optimum, it is important to identify these discrepancies. Viale et al. [21,22] focus on this problem, combining experts' knowledge and activity information in the design of DEVS models.

Optimization is also a key problem within companies. In this area, DEVS has been combined with different optimization techniques to study the supply chains and develop planning and control algorithms. For example, Godding et al. [23] developed a methodology for integrating different types of models using a Knowledge Interchange Broker (KIB). Gholami et al. [24] also use KIB to integrate Linear Programming models with DEVS models and generate them automatically from industry relational databases. Model Predictive Control paradigms have also been combined with DEVS using KIB [25]. All this research resulted in the development of a simulation platform called Optimization, Simulation, and Forecasting (OSF) for the domain of manufacturing and logistics supply-chain systems. The platform supports the composition of DEVS, Linear Program (LP), and forecast models through a KIB [26].

Other research has used DEVS for validation purposes. For example, Pujo et al. [27] present a method to reallocate a Flow-Shop without stopping the production and use a DEVS model of the flow-shop to validate their method. Rajaoarisoa and Sayed-Mouchaweh [28] presented an adaptive fault diagnosis approach for manufacturing systems also using DEVS formalism.

Our work differs from the previous ones in that we focus on how to apply the DEVS formalism as a tool to improve the manufacturing system using the Theory of Constraints. We identify the weakest element of the manufacturing system (i.e., the bottleneck) to reduce the amount of WIP and, therefore, reduce the inventory cost and production time. We also focus on showing how to apply the methodology with a tool (CD++) that is suitable for DEVS model development.

2.1 The DEVS Formalism

The DEVS formalism is a formal DES methodology [12]. It is derived from Systems Theory, and it allows one to define hierarchical modular models that can be easily reused. In DEVS, an atomic model defines the behavior of a component. It is specified as a black box with a state and a duration for that state. When state duration time elapses, an output event is sent, and an internal transition takes place to change the model state. A state can also change when an external event is received. Then, a DEVS model is defined by describing the set of states the model goes through, the internal and external transition functions, the output function, and the state duration function. DEVS models can be put together by linking the outputs of a model to inputs of other models to form coupled models. Models made from more than one component are called coupled models. We can also link coupled models.

The use of DEVS offers the following advantages:

- **Reliability**: DEVS is based on system theoretical roots and sound mathematical theory. This formal base provides logical and timing correctness to the models.
- **Model reusability**: It has been proven that DEVS is closed under coupling. It has well-defined concepts that allow coupling components in a hierarchical and modular way.
- **Hybrid modeling**: DEVS is the most general discrete event formalism and many techniques to model both continuous and discrete systems have been mapped into DEVS. Therefore, we can use different modeling techniques for different parts of complex systems. Then we translate them into DEVS for integration.
- **Process flexibility**: Hybrid-modeling capabilities are transparent for the simulator, which is defined by an abstract mechanism that is independent of the model itself.
- **Testing**: Because the model definition and the simulation mechanisms are independent, we can focus on the model variation and validation being sure that the simulation mechanism is correct if we use a verified and validated DEVS simulator.

DEVS atomic models can also be defined using DEVS-graph notation as explained in [14,29].

The model's graphical description is composed of bubbles, arcs, and labels, as shown in Figure 1. Bubbles represent model phases/states. Each bubble includes an identifier (ID) and a state lifetime (LT). The transitions between states are represented by arcs. Internal transitions are presented by dotted lines, and external transitions by full lines. The output that happens before the internal transition is represented with a label close to the dotted line. The notation is "*port!value"* where port represents the output port of the model and value the content of the output. For inputs that trigger the external transitions, the notation is similar. The difference is that *port* stands for the input port of the model and value for the content of the input message.

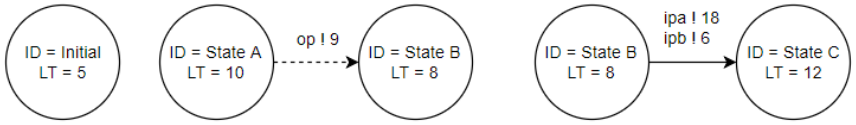

Figure 1: DEVS-Graphs notation.

Several simulators implement the DEVS simulation algorithm using different approaches. We used the CD++ tool combined with CD++Builder [30], which allows the definition of DEVS models using the graphical notation explained above. Expert programmers can implement more complex models and use the advanced features of the simulator. Additionally, it is well documented and has large libraries of models. The tool can be downloaded from cell-devs.sce.carleton.ca/index.php/installation/.

3 CASE STUDY 1: A PHARMACEUTICAL FACTORY

We present a case study focusing on a piece of the whole supply chain of a pharmaceutical product. The supply chain covers the system from the provision of raw materials, production of the tablets, distribution of the final products to retailers, and delivering them to the hands of customers. For the case study, we selected the mother factory, which has the main role in this process. This factory includes an administration that controls all the transactions inside the factory and acts as a communicator between different sections. It also has a warehouse that holds the inventory of raw materials and final products in a way that whenever new batches of raw materials are delivered, or a batch of the final product is produced, they are sent by the administrator to the warehouse. The batches of final products are produced at the Pharmaceutical Manufacturing Plant (PMP) and with the order that is placed by the administrator to restock the capacity of the warehouse. Another role of the administrator is to send the requested final product to retailers as soon as enough stock is available in the warehouse. The PMP itself has some workstations to transform the raw material into the final packed product.

Once the factory is modeled and tested, it can be integrated with the other elements of the supply chain, which will be modeled in a similar way, using DEVS hierarchical and modular model interfaces.

3.1 DEVS Model Definition

The DEVS coupled model of the factory is presented in Figure 2.

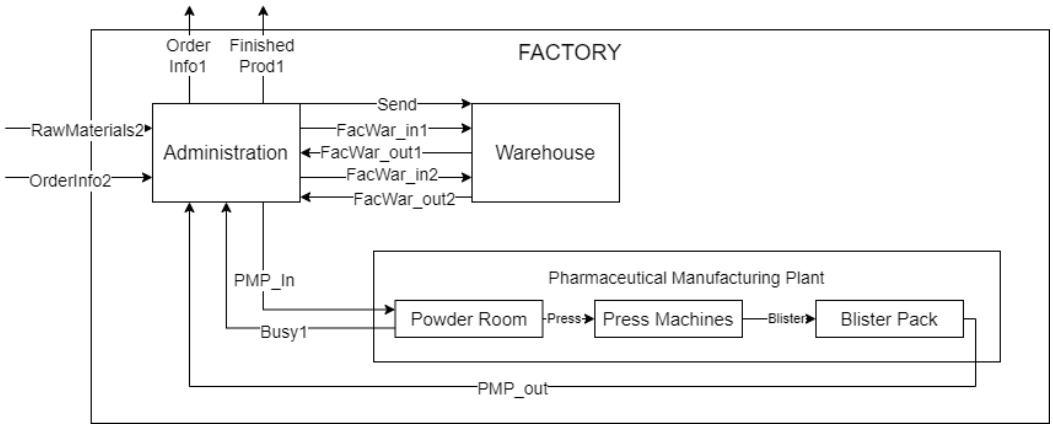

Figure 2: Factory DEVS coupled model.

As we can see, the Factory has an administration, a warehouse, and the PMP, described following.

Administration: The administrative duties include receiving good from the supplier, attaining the orders from the distributor, placing orders with the supplier and shipping finished products to the distributor. All these duties are modeled as an atomic model (Administration in Figure 2) that always tries to maintain the full capacity of finished products.

It is comprised of seven input ports:

- *RawMaterials2* is the port through which the supplier transports raw materials to the factory. For the supplies to be transferred from the supplier to the factory there exists variable lead times (due to factors such as transportation delay). However, for the purpose of simplicity, we will model this as a constant time delay of 1 day.
- *OrderInfo2* is used to place orders to replenish the distributor's inventory. Realistically the delay of information flow can vary from case to case. To avoid complexity, we set this delay to a constant value of 12 hours.
- *FacWar_out1* is the port through which the warehouse sends its raw materials to the administration.
- *FacWar_out2* is the port through which the warehouse sends its finished products to the administration
- *Busy1* is the port first workstation of the manufacturing plan is busy or not.
- *PMP_out* is the port through which the factory's pharmaceutical manufacturing plant communicates with the administration.
- *Send* is the port through which the factory's administrator communicates to the warehouse how many raw materials or finished products it needs.

In addition to input ports, the Administrator also has four output ports:

- *FinishedPro1* port is used by the factory to ship its finished products to the distributor. The shipment delay is modeled as a constant value of 1 day.
- *FacWar_in1* is the port through which the administration transports raw materials to the factory's warehouse.
- *FacWar_in2* is the port through which the administration transports finished products to the factory's warehouse.

- *PMP_in* is the port through which the administration communicates with the factory's pharmaceutical manufacturing plant.

Factory Warehouse: It is a storage facility for the factory's raw materials and finished products. It has a maximum carrying capacity of 80 batches for raw materials (represented as positive integers ranging from 111 to 180) and 20 batches for finished products (represented as positive integers ranging from 181 to 200). In the event of an overflow of either raw materials and/or finished products, the redundant batches will just be discarded without notification.

The warehouse receives raw materials and finished products from the administrator through the ports FacWar_in1 and FacWar_in2, respectively. In addition, it sends raw materials and finished products based on the administrator requests through the ports FacWar_out1 and FacWar_out2, respectively. The number of products (both raw materials and finished products) that need to be sent to the administrator is communicated through the port Send.

Pharmaceutical Manufacturing Plant (PMP): The PMP manufactures pills. I.e., it converts raw materials into a pill that will be stored in the warehouse until it is sent to the distributor. The factory is modeled as a coupled DEVS model with three atomic components, each of them representing a workstation. Each workstation is modeled as an atomic model that receives "raw material" as input, and after the processing time, it outputs the processed item and requests new raw material.

3.2 Formal Definition and Implementation

Because all the atomic models are defined similarly, we just present the formal definition and implementation of one of them, the Powder Room atomic model, which is formally defined as follows:

$$\text{PowderRoom} = \langle X, Y, S, \delta ext, \delta int, \lambda, ta \rangle,$$

where:

X = {("PMP_in", XPMP_in = {1,2,3,4})},

Y = {("Busy1", YBusy1 = {true,false}),("Press", YPress = {1})}

S = B x P(V) states of the model, where B = {wait, ready, Powdering, InvalidInput, Keep}, V = {Powder, PowderStat}

The internal, external, time advance and output functions are represented in Figure 3 using DEVS-Graphical notation.

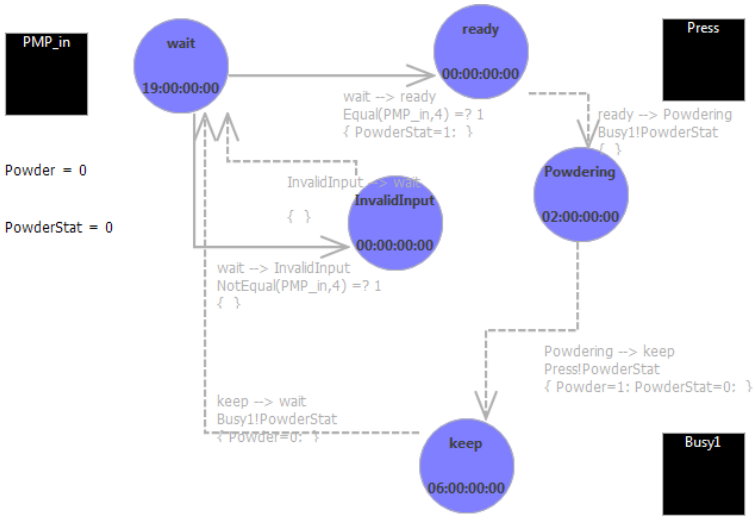

Figure 3: PowderRoom implementation in CD++ using CD++Building and DEVS Graphs notation.

The models can be implemented using CD++ metalanguage or the graphical interface provided by CD++ Builder. In Figure 3, we show the CD++ implementation of the Powder Room atomic model using CD++ Building and DEVS Graph notation. The squares represent the input and output ports and contain the port names. The circles or bubbles represent the model state and contain the identifier and the elapsed time for the state. Finally, the internal transitions with the output message and the external transitions with the input messages that trigger them are represented with labeled dotted and solid lines, respectively.

3.3 Simulation Results and Analysis

Once the model is implemented, we can run simulations for different studies. For example, we can study how the factory performs in satisfying different demand distributions based on the availability of raw materials from the suppliers. In Table 1, we show a simulation log generated when we study what happens when the distributor places an order, and there is not enough finished product in the factory's warehouse. These simulation results show that the factory is trying to replenish its finished product stock even before the order has been placed since the number of finished products stored in its warehouse was not at full capacity.

Table 1: Sketch of the simulation log file.

Inputs	Outputs
	00:00:00:000 send 104 00:00:00:000 facwar_out1 4
00:00:01:01 OrderInfo2 10 00:00:02:01 RawMaterials2 20	
	01:00:00:050 pmp_in 4 00:00:01:090 send 190 00:00:01:090 facwar_out2 10 00:00:01:137 out 10 ….. 90:00:00:810 send 104 90:00:00:810 facwar_out1 4 91:00:00:860 pmp_in 4 96:00:00:818 facwar_in2 181

If we compare the time the order was placed and the time the final product was sent to the distributor, we could identify if the factory has an acceptable response time or not. If the time to accomplish the order is not acceptable, improvements in the PMP are needed. In the next case study, we explain how possible improvements in the manufacturing plant can be identified, and how the impact of their implementation in the plan performance is analyzed.

4 CASE STUDY 2: A MANUFACTURING PLANT

In this section, we present a case study to show how to apply DEVS to study the bottlenecks in a Manufacturing Plant and being able to improve throughout and production time. We choose a simple case study to be able to explain how to apply the methodology and its potential use to study large and complex production systems. Our case study is based on the one presented in [31], where the authors present a manufacturing system of a company in Shanghai (China) that produces booster cables; it includes several productions lines with a similar structure. Each production line has different workstations connected by a conveyor belt. Each workstation is equipped with a buffer for the raw material needed in the process, and the specific workers and tools they need to release the WIP products. As all the production lines are similar, in this case study, we focus on the analysis of one of them. The other production lines would be modeled and analyzed in a similar way.

Our production line consists of the sequence of processes illustrated in Figure 4. For each work station, we show the average processing time in seconds. For example, for the Peel Cable station, the average processing time is 8 seconds. We have two inventories, one for raw material and another one for the finished product. We also have five areas in the production line: (1) cable zone, (2) clip zone, (3) assemble clip-cable zone and (4) quality zone and (5) packaging zone.

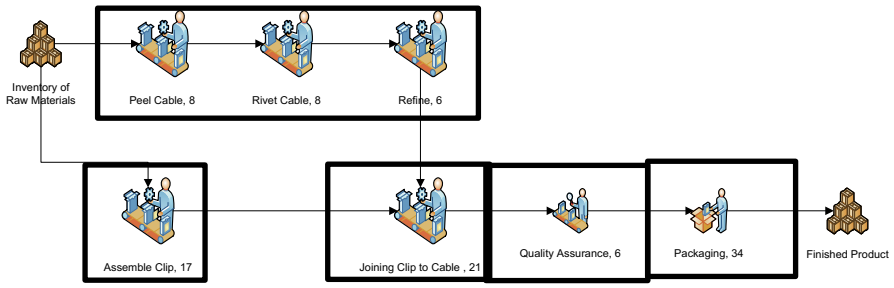

Figure 4: Work stations for the booster cable production line.

Cable Zone: The cable is prepared before the assembly process. The zone contains three processes: peel cable, rivet cable, and refine. The average processing time for these processes is 8 seconds, 8 second, and 6 seconds respectively.

Clip Zone: It is similar to the cable one except that it contains just one operation to prepare clips before the assembly process. This process takes 17 seconds on average.

Clip-Cable Assemble Zone: The clips and cables are assembled. This is critical because we need both a cable and a clip to perform the operation. On average, this operation takes 21 seconds to process.

Quality and Packaging Zones: Quality assurance and packaging processes take about 6 and 34 seconds respectively. At the check work station, they automatically verify that a booster cable is connected properly to a clip and that it satisfies their standard. Packaging takes a longer time because it is accomplished manually.

As we already mentioned, each workstation is equipped with a buffer for each raw material needed in the process. In our model, we assume that the buffers have an unlimited capacity to easily identify the bottlenecks. Considering this assumption, bottlenecks will be in the workstations where the buffer has the larger WIP inventory. Because our objective in this study is to identify bottlenecks in the manufacturing process, we can also make the following assumptions:

- There is unlimited raw material.
- Because the buffers for the peel cable and clip assembly can be considered as storages, we do not take them into account to identify the bottlenecks. Therefore, we set the capacity to 5 elements.
- Interruptions are not allowed in the work stations. Once a process starts in a work station, it cannot be interrupted until it finishes and asks for the next element.
- The time to move the raw material from the buffer to the process is constant through the whole manufacturing system. As we do not have an estimation for this value, we fix it in one second.
- Raw materials are sent from the buffer to the process without any delay.

The two last assumptions could be relieved, including these delays as parameters of the model.

4.1 DEVS Model Definition

In Figure 5, we show the DEVS top model for the production line of booster cable presented in Figure 4. We have 5 couple models representing each one of the zones in the production line as detailed below. Because we have two independent sources of raw material (i.e. cable and clip), we model them separately as two independent atomic models.

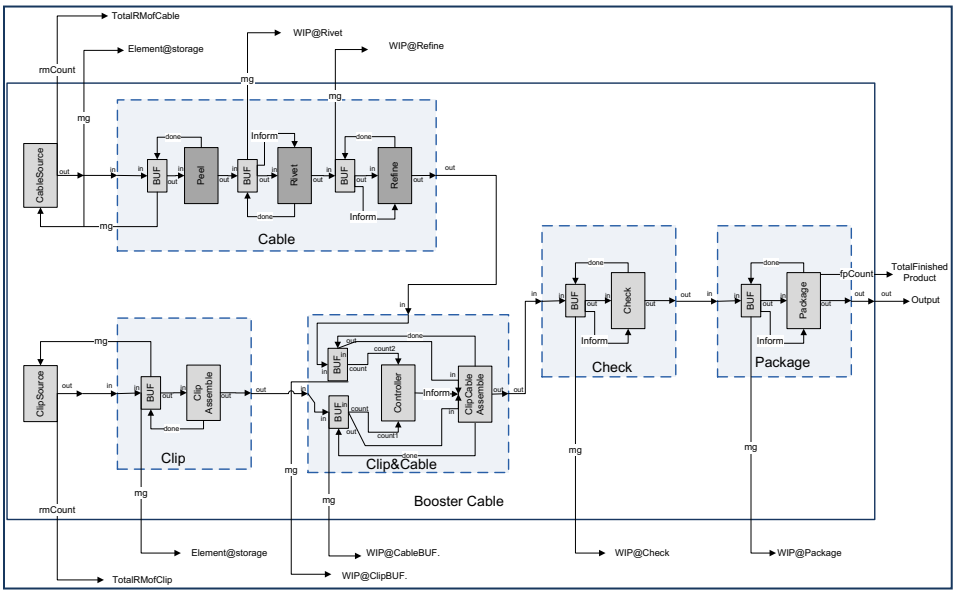

Figure 5: DEVS model of the production line of booster cable.

Source atomic models: At the initialization stage, the source will send a complete packet where its size is equal to the buffer capacity. After that, a raw material element is sent every time a request is received.

Cable Zone: It is modeled as a coupled model that contains six atomic models. Three of them represent the processes at this zone, and the other three are buffers for each process. This coupled model works as follows. It takes raw material as input for the peel buffer. The first element is sent directly to the operation, peel. After that, the buffer will not send any other element until the operation finishes and asks for another element. As soon as the element is sent to the operation from the buffer, the buffer sends a request to replace that element. Note that this buffer was considered just storage and will never have more than 5 elements and less than 4 since it asks for a new one after every element is sent to the peel operation, and the sources have unlimited capacity. At the next operations, rivet and refine, the buffers accumulate the elements received and do not send any element until getting a request from the operation. In that way, we ensure that the elements are processed one at a time. After all these three processes are accomplished, an element is sent through the coupled model "out" port.

Clip Zone: The coupled model is similar to the cable one, except it contains just one operation and, therefore, just two atomic modes: a buffer and an operation. As in the previous case, the first element is sent directly to the operation (i.e., clip assemble). After that, the buffer will not send any other element until it receives a raw material request. As soon as the element is sent to the operation from the buffer, the buffer sends a request to replace that element so that the buffer will not be empty.

Clip-Cable Assemble Zone: As we already mentioned, this zone is critical because there are two buffers connected to one operation. Therefore, it necessary to be sure that the operation will not start until it gets the two elements to assemble. To solve and control this issue, an atomic model called "Controller" is added. Every time the buffers receive an element, they are added to the queues and inform about their length (i.e. how many elements does the buffer have at a specific moment). The Controller takes as input the length of both buffers and compares them. When both lengths are greater than one, the controller informs the operation (i.e. Clip&CableAssemble) about the availability of elements in both buffers. If there are no elements under operation at that moment, the clip-cable assemble model asks immediately for an element from each buffer. The first element in each buffer will be sent after one second to the operation. Once the operation is finished, the assembled product is sent through the coupled model "out" output port. If there are any product being processed at "Clip&CableAssemble", the message received from the controller is ignored until the work is completed.

Quality and Packaging Zones: The behavior of these two coupled models (i.e. the buffer and the operation) is the same as the behavior of rivet or refine processes with their buffers in the Cable Zone.

4.2 Formal Definition and Implementation

Although there are 18 atomic models within the Manufacturing System Coupled model, only 4 of them are different: the source atomic model for the inventory, the buffer, the operation and the controller

With these 4 atomic models, we can define the manufacturing system. We just need to define the processing time for the operation model and the delay of the buffer as a parameter of the atomic model that can change every time it is instantiated.

Because all the atomic models are defined similarly, we present the formal definition of the Operation atomic model because it is one of the most used in the top coupled model.

Operation

The formal definition of the Operation is as follows:

$$Operation(ProcessingTime) = < X \ \ , Y \ \ , S, ta, \delta_{ext}, \delta_{int}, \lambda >,$$

where:

$$X \ = \{(\text{"in"}, d \in \mathbb{N}), (\text{"inform"}, d \in \mathbb{N})\};$$

$$Y \ = \{(\text{"out"}, d \in \mathbb{N}), (\text{"done"}, d \in \mathbb{N})\};$$

$$S = \{state \in (waiting, processing, requesting)\};$$

$$ta(S) = \{state = waiting \rightarrow \infty;\ state = requesting \rightarrow 0;\ state = processing \rightarrow ProcessingTime\};$$

$$\delta_{ext}(S, e, X) = \begin{cases} if(port = \text{"in" \& state !=processing}) \rightarrow \text{state = processing} \\ if(port = \text{"inform" \& state = waiting }) \rightarrow state = requesting \end{cases};$$

$$\delta_{int}(S) = \{state = waiting\};$$

$$\lambda(S) = \begin{cases} if\ (state = procesing) \rightarrow (\text{out}, 1)(done, 1) \\ if(state = requesting) \rightarrow (done, 1) \end{cases},$$

where “in”, “inform”, “out” and “done” are the names of the input and output ports respectively; d is the message received through the port.

The above formal definition can also be expressed using the DEVS-Graph notation as in the previous case study, which is more readable for the general public. The coupled models are also formally defined using DEVS notation. However, it is not necessary to write the formal specification as Figure 5 (i.e. the graphical notation for coupled models) contains all the information needed for implementation.

We implement both the atomic and couple models using CD++ DEVS simulator.

4.3 Simulation Results and Analysis

After running the simulation for the Cable Booster manufacturing system for a complete shift (i.e., 8 hours), the results presented in Table 2 were obtained after processing the log file. The simulation results are the amount of WIP at each buffer and the total amount of Cable Boosters produced during the shift.

Buffers at first operations of cable and clip (i.e. “Peel” and “ClipAssemble” respectively) will not be more than 5 elements because they have a limited size equal to 5 elements. As we have already explained, we assumed these two buffers work as storages of raw materials, so that they will not affect the study of WIP or bottleneck of the system. However, the number of WIP elements in the other buffers will vary from zero to infinity (as we assumed the buffers have unlimited capacity). The number of elements in those buffers is key to determine the bottlenecks. The bottlenecks are located on those stations with a larger amount of WIP.

Table 2. Simulation results for the Cable Booster manufacturing system for a complete shift.

Port name	Description	Value
wipPeel	Number of WIP elements at Peel	4
wipRivet	Number of WIP elements at Rivet	0
wipRefine	Number of WIP elements at Refine	1
wipClpA	Number of WIP elements at Clip Assemble	4
wipCblCCA	Number of WIP elements in Cable at Clip-Cable Assemble	2132
wipClpCCA	Number of WIP elements in Clip at Clip-Cable Assemble	534
wipCheck	Number of WIP elements at Check	0
wipPackage	Number of WIP elements at Package	245
totalFinishedProduct	Number of Cable Boosters produced	819

Based on the results in Table 2, the processes of clip-cable assemble and package are the bottlenecks of the production line. The number of WIP elements in cable buffer at clip-cable assemble is very high, 2132 elements, while the clip buffer at the same stage contains 534 elements. This difference between the two buffers reveals that the cable process (i.e. peel, rivet, and refine) is faster than the clip assembly. The other bottleneck is at packaging, where the amount of WIP elements is 245. If we tackle this bottlenecks, the number of Cable Boosters produced by shift should increase considerably

In order to improve the system, we need to minimize the WIP with the objective of increasing throughput. One possible solution is to replace the "Package" zone by three packaging work stations that work in parallel. We choose three considering the processing time at the Check zone. In more complex systems, we may need to test different solutions through simulation. Similarly, "Clip&CableAssemble" zone can be doubled (i.e. we can have two work stations working in parallel).

After implementing this enhancement in the model, we obtain the results show in Table 3.

Table 3: Simulation results for the Cable Booster manufacturing system for a complete shift after the enhancement.

Port name	Description	Value
wipPeel	Number of WIP elements at Peel	4
wipRivet	Number of WIP elements at Rivet	0
wipRefine	Number of WIP elements at Refine	1
wipClpA	Number of WIP elements at Clip Assemble	4
wipCblCCA	Number of WIP elements in Cable at Clip-Cable Assemble	1599
wipClpCCA	Number of WIP elements in Clip at Clip-Cable Assemble	1
wipCheck	Number of WIP elements at Check	0
wipPackage	Number of WIP elements at Package	0
totalFinishedProduct	Number of Cable Boosters produced so far	1597

If we compare both results, we notice that the throughput has almost doubled, and WIP has disappeared in the Package area. However, the number of elements in the Cable buffer at Clip-Cable Assemble is high (i.e., 1599 elements). This result indicates that a new bottleneck has appeared. In this case, we need to consider that the Clip-Cable Assemble has two buffers, the Cable buffer with 1599 elements and the Clip buffer with just one element. This means that the Clip zone is the new bottleneck.

We can address this issue in the same way, we can simulate a new configuration where we have two workstations working in parallel in the Clip zone. The analysis of the simulation results may reveal that the production line has been optimized for the current processing times or may indicate the need for taking further actions. With our model, we can also test the effect of adding more resources to the actual configuration to reduce the processing time at each workstation.

5 CONCLUSIONS

In this paper, we have presented DEVS as a tool to improve a manufacturing system using the Theory of Constraints. We developed a DEVS model of the manufacturing system in order to simulate it and identify the weakest element (i.e., the system bottleneck). In order to identify the weakest element, we need to find the workstation that accumulates the larger amount of WIP. The identification of the bottleneck allows us to think of strategies to remove it and improve the system capacity and production time.
The main advantage of using DEVS with CD++ simulator is that we have a formal methodology and simulation tool that allows us to define complex manufacturing systems easily. Additionally, DEVS is flexible and agile: we can develop models for the different parts of the system using a different level of detail. Because it is modular, if we need to include models for other components of the system, we can easily add them to the actual model without starting the whole modeling process again.
The flexibility to easily modify and include more components in the model makes DEVS a very suitable tool to test different improvement strategies for improving manufacturing systems.
Further analysis can identify the idle time of the machines to schedule maintenance and the response time to fulling the requests from the clients on different demand scenario and working conditions.

ACKNOWLEDGMENTS

This research was funded by NSERC.

REFERENCES

[1] E.M. Goldratt, and J. Cox. *The Goal*, Croton-on-Hudson. NY: North River Press Inc., 1984
[2] T. Ohno. *Toyota Production System: Beyond Large-Scale Production*. CRC Press, 1988
[3] J.P. Womack and D.T. Jones. *Lean Thinking-Banish Waste and Create Wealth in your Corporation*. Free Press: Simon and Schuster, London. 1996
[4] S. Chopra, and P. Meindl. *Supply Chain Management: Strategy, Planning, & Operation*. Pearson, 2007
[5] M. Al-Mashari., A. Al-Mudimigh, and M. Zairi. Enterprise resource planning: A taxonomy of critical factors. *European Journal of Operational Research*, vol. 146, no.2, pp. 352-364, 2003
[6] M. McClellan, *Applying manufacturing execution systems*. CRC Press. 1997
[7] E.M. Goldratt, and R.F. Fox. *The Race*, North River Press, Croton-on-Hudson, NY, 1986
[8] C. Pegels, and C. Watrous. Application of the theory of constraints to a bottleneck operation in a manufacturing plant. *Journal of Manufacturing Technology Management*, vol. 16, no.3, pp.302-311, 2005
[9] M. Leporis, and Z.A. Králová. "A simulation approach to production line bottleneck analysis". *International conference cybernetics and informatics, 2010,* pp. 13-22.
[10] A.A. Tako, and S. Robinson, S. The application of discrete event simulation and system dynamics in the logistics and supply chain context. *Decision Support Systems*, vol. 52, no.4, pp. 802 – 815, 2012
[11] S. Brailsford, and N.A. Hilton. A comparison of discrete event simulation and system dynamics for modelling healthcare systems. *Proceedings of ORAHS,* 2001
[12] B.P. Zeigler, H. Praehofer, and T.G. Kim. *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*, Academic Press, 2000
[13] H. Vangheluwe. DEVS as a common denominator for multi-formalism hybrid systems modelling. *IEEE International Symposium on Computer-Aided Control System Design*, 2000, pp. 129–134.
[14] G. Wainer. *Discrete-event modeling and simulation: a practitioner's approach*. CRC Press, Boca Raton, FL, USA, 2009
[15] P. Bocciarelli, A, D'Ambrogio, E. Paglia, A. Giglio, and V. Plinio. On the Performance Prediction Capabilities of the eBPMN-based Model-driven Method for Business Process Simulation. *In CIISE*, 2018, pp. 71-78
[16] H. Bazoun, J. Ribault, G. Zacharewicz, Y. Ducq, and H. Boyé, SLMTOOLBOX: enterprise service process modeling and simulation by coupling devs and services workflow. *International Journal of Simulation and Process Modelling*, vol 11, no 6, pp. 453-467, 2016

[17] H. Bazoun, G. Zacharewicz, Y. Ducq, and H. Boye. Business process simulation: transformation of BPMN 2.0 to DEVS models. *In Proceedings of the Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium*, 2014, pp. 13-16.
[18] L. Dávid, H. Vangheluwe, and Y. Van Tendeloo, Translating engineering workflow models to DEVS for performance evaluation. *Winter Simulation Conference*, 2018, pp. 616-627.
[19] T. Alix, and G. Zacharewicz. Product-service systems scenarios simulation based on G-DEVS/HLA: Generalized discrete event specification/high level architecture. *Computers in Industry*, vol. 63, no. 4, pp.370-378, 2012
[20] T. Alix, and G. Zacharewicz. G-DEVS based simulation of toy industry client behavior in PSS (WIP). *In Proceedings of the 2012 Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium*, 2012, p.24
[21] P. Viale, C. Frydman, and J. Pinaton. New methodology for modeling large scale manufacturing process: Using process mining methods and experts' knowledge. *9th IEEE/ACS International Conference on Computer Systems and Applications*, 2011, pp. 84-89
[22] P. Viale, C Frydman, and J. Pinaton, Constructing DEVS models based on experts' knowledge: application to STMicroelectronics' large scale manufacturing processes. *In Proceedings of the 2011 Symposium on Theory of Modeling and Simulation: DEVS Integrative M&S Symposium*, 2011, pp. 193-198
[23] G. Godding, H. Sarjoughian, and K. Kempf. Application of combined discrete-event simulation and optimization models in semiconductor enterprise manufacturing systems. *Winter Simulation Conference*, 2007, pp. 1729-1736.
[24] S. Gholami, H. Sarjoughian, G. Godding, D. Peters, and V. Chang, Developing composed simulation and optimization models using actual supply-demand network datasets. *Winter Simulation Conference*, 2014, pp. 2510-2521.
[25] H. Sarjoughian, D. Huang, G. Godding, K. Kempf, W. Wang, D. Rivera, and H. Mittelmann. Hybrid discrete event simulation with model predictive control for semiconductor supply-chain manufacturing. *Winter Simulation Conference,* 2005, pp. 256-266.
[26] H. Sarjoughian, J. Smith, G. Godding, and M. Muqsith, Model composability and execution across simulation, optimization, and forecast models. *In Proceedings of the Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium*, 2013, p. 30
[27] P. Pujo, M. Pedetti, and N. Giambiasi, Formal DEVS modelling and simulation of a flow-shop relocation method without interrupting the production. *Simulation Modelling Practice and Theory*, vol. 14, no. 7, pp. 817-842, 2006
[28] L. Rajaoarisoa, and M. Sayed-Mouchaweh, Adaptive online fault diagnosis of manufacturing systems based on DEVS formalism. *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 6825-6830, 2017
[29] H. Praehofer, and D. Pree. Visual modeling of DEVS-based multiformalism systems based on higraphs. *Winter Simulation Conference*. 1993, pp. 595-603.
[30] G. Wainer. CD++: A toolkit to develop DEVS models. *Software Practice and Experience* vol. 32, no. 12, pp. 1261-1306, 2002.
[31] X. Hu, and R. An. Modeling and Simulation of Manufacturing Systems in Unstable Environment, *Proceedings of the World Congress on Engineering*, 2011.

AUTHOR BIOGRAPHIES

CRISTINA RUIZ MARTIN is an Assistant Professor, teaching stream, at the Department of Systems and Computer Engineering at Carleton University. She received her PhD from University de Valladolid and Carleton University. *cristinaruizmartin@sce.carleton.ca.*

GABRIEL WAINER is a Professor at the Department of Systems and Computer Engineering at Carleton University. He received his Ph.D. from UBA/ Université Aix-Marseille-III, France. He is a fellow of SCS *gwainer@sce.carleton.ca.*