Clock Pulse Modeling and Simulation of Push and Pull Processes in Logistics

Carlo Simon, Stefan Haag, Lara Zakfeld

{simon, haag, lara.zakfeld}@hs-worms.de

Hochschule Worms
University of Applied Sciences

October 2020

The Theory of Constraints

According to the Theory of Constraints, revealing and eliminating bottlenecks in a manufacturing environment is the main target for expanding production and throughput

Finding bottlenecks in real-world applications is no trivial task

Further, to remove a bottleneck, investments must be made - sometimes substantially - so knowledge of impending implications would be beneficiary

Fluctuations in Stocks

Bottlenecks can typically be associated with storage buffers running full

If a working place is a bottleneck, the associated incoming buffer is heavily utilized, leading to a backlog upstream

Often, there are lower utilization rates on the downstream side of the bottleneck and, thus, lower stocks

In manufacturing environments, corresponding circumstances may not necessarily be obvious

Simulate to Find the Bottlenecks

- Simulating an appropriate model can help unearthing the real causes of bottlenecks
- Simulations are helpful in examining whether an investment may yield the desired outcome
- Simulatable Models could objectify decisions on reorganizations
- By usage of the Process-Simulation.Center (P-S.C) - a novel, web-based Petri net modeling and simulation environment - complex real-world processes in logistics can be modeled following a clock pulse spotted approach
- This enables the observation of augmentation and depletion of stocks during run-time and the detection of bottlenecks
- For ease of presentation, a teaching laboratory for students in logistics is used as small example

Petri nets

Reisig (2013): Understanding Petri Nets
Genrich and Lautenbach (1981): System Modelling with High-Level Petri Nets
Jensen (1992): Coloured Petri-Nets
Montali and Rivkin (2019): From DB-nets to Coloured Petri Nets with Priorities

Timed Petri nets

Merlin (1974): The Time-Petri-Net and the Recoverability of Processes
Ramchandani (1974): Analysis of Asynchronous Concurrent Systems by Timed Petri Nets
Sifakis (1977): Use of petri nets for performance evalutation
König and Quäck (1988): Petri-Netze in der Steuerungs- und Digitaltechnik
Hanisch (1992): Petri-Netze in der Verfahrenstechnik
Ghezzi et al. (1991): A unified high-level petri net formalism for time-critical systems
Hanisch et al. (1998): Timestamp Nets in Technical Applications
Simon (2001): Developing Software Controllers with Petri Nets and a Logic of Actions

Alternative Modeling Approaches

Knoeppel (1915): Installing Efficiency Methods
Ohno (1988): Toyota Production System
BPMI (2004): BPMN 1.0 - Business Process Model and Notation
OMG (2011): BPMN 2.0 - Business Process Model and Notation

The *Box Game*

The *Box Game* has been developed at the University of Applied Sciences Worms

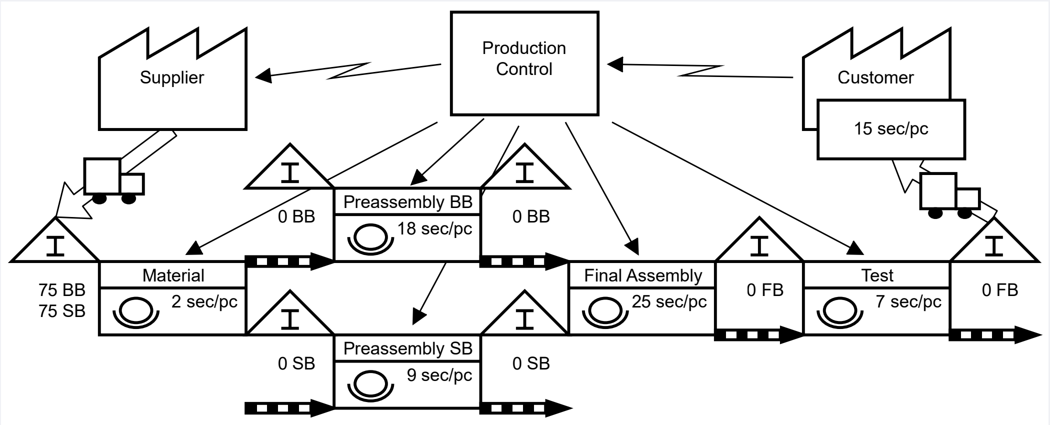
Implemented as a teaching laboratory, it aims at imparting knowledge in logistics

Focus lies on the strategic level, not on scheduling or problems concerning mechanical production

The *Box Game* has almost non-existing requirements:

- Five tables are arranged as working and storage places
- Standard positions like buffers are marked with adhesive tape
- Cardboard boxes are used as workpieces

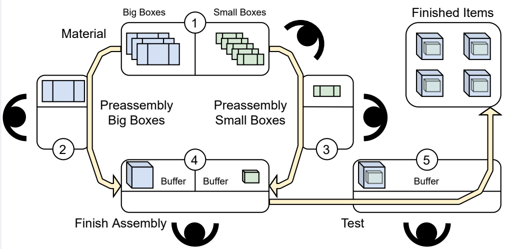
The game is easily transferable to assembly workstations



Value stream diagram of the *box game*

The Rules of the Game

- Deliver unfolded boxes from the warehouse (1) to the preassemblies (2) & (3)
- Fold and close big (2) and small (2) boxes
- Pass the boxes to the final assembly (4)
- Open the big box and insert the small one
  Then, close and seal the big box
- Pass the box to the quality assurance (5) where a shake (as acoustic check) is performed
- Put the finished box on the outgoing storage



Spatial organization of the *box game*

The Clock Pulses

A clock pulses once for each discrete time step - in this case every second - initiating a reprocessing of the system's state

Thus, a real time observation of object flows and storage utilization becomes possible

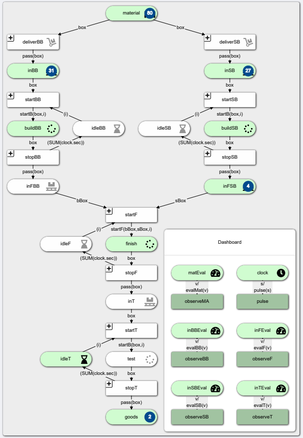
This allows for discovering bottlenecks or other flow-related problems via visual clues during run-time

The presented models simulate the *box game* and implement two different logistics strategies:

- Push processes guide workpieces through production as fast as possible
- Pull processes initiate production only when there actually is a demand

Both models have the same setup - initial stocks and a batch size of one - and also look quite similar

Push Model



- Places (ovals) serve as storage,
- transitions (rectangles) as activities,
- and arcs transport objects
- using a lot size of 3

In the Dashboard,

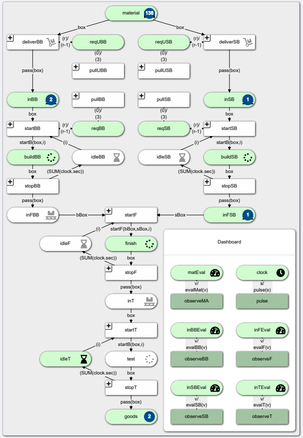
- the *clock* serves as pulse generator,
- and the observers sum up costs associated with their storage places

Petri net model as depicted in the P-S.C

Further elements and conditions implement pull principles

Only when *req*-places indicate a demand, the corresponding transitions can fire

The information concerning pull requests leads to a higher computation effort, but not to additional computation steps



Pull Model

Petri net model as depicted in the P-S.C

An Alternative Modeling Approach

Clock pulse models are very illustrative: raise and discharge of stocks can be followed in ”real-time”

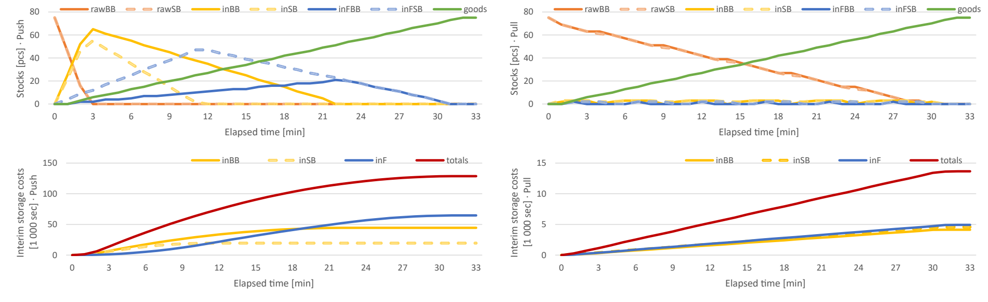
However, such models rely on computing every single time-step for correct visualization
In the presented case, each second has to be processed for almost 33 minutes

An alternative are event triggered models where system state changes initiate computations

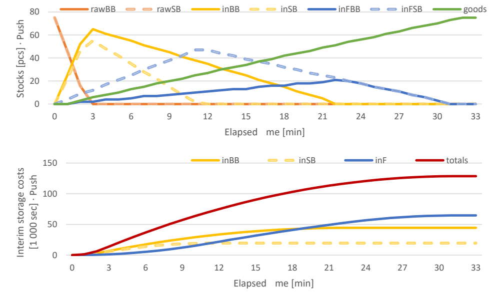
Using an iMac (Quad-Core Intel Core i7 @ 4 GHz, 16 GB RAM) and Chrome, corresponding models behave as follows:

- Clock pulse models (push & pull): 8 234 ms
- Event triggered model (push) 315 ms & (pull) 923 ms (difference due to additional pull computations)

As a downside, visualization of event triggered simulations may have to be generated in a separate step since the ”real-time view” is not directly available



Stocks and associated costs on relevant storage places during simulation in P-S.C



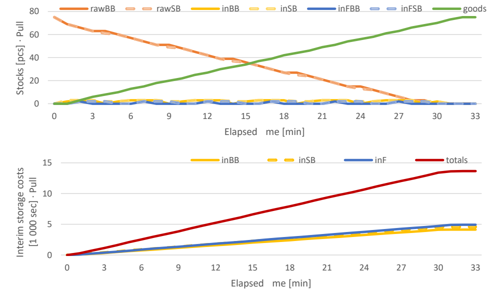
In the push model . . .

- the main warehouse gets emptied fast,
- interim buffers are built up,
- a massive bottleneck is visible at the buffers *inBB* and *inSB*,
- and interim storage costs accumulate to 128 325 [seconds]

Stocks and associated costs on relevant storage places during simulation in P-S.C

In the pull model . . .

- the main warehouse gets emptied steadily over the production time,
- interim buffers usage is low,
- there is no bottleneck visible,
- and interim storage costs accumulate to 13 643 [seconds]



Stocks and associated costs on relevant storage places during simulation in P-S.C

Clock Pulse versus Event Triggers

Use a clock pulse simulation if you either need a clock pulse visualization of the system's states or if your computer is fast enough for the few simulations that must be run for the modeled system

Use an event trigged simulation if simulation speed is necessary due the complexity of the modeled system, if you need fast answers in production, or if you need to compare a large number of variations of the production schedule or input data

Guidelines

1. Define data types for the different stocks and other data objects, and initialize the corresponding places in accordance with the starting condition
2. Augment the model by transitions for beginning and ending specific tasks like delivering raw materials, building or testing a box
3. Identify the next item to be taken and the moment this will occur
   This also allows for implementation of different prioritization strategies
4. Start with modeling the simpler push principle and augment this model by pull principles
5. Look for a proper visualization of the simulation results

Takeaways

- Development of clock pulse models seem preliminary
  for the development of event triggered models due to two main reasons:
  - Clock pulse models can be implemented in a more straightforward way
  - State changes, which are needed to evaluate the triggers in the event triggered models, can be derived from observing the clock pulse model
- Discovering state changes needs to be done manually at the moment,
  however, support by a tool would be beneficiary
- Modeling pull principles is hard, even if following the presented guidelines,
  hence, they should be refined to account for this
- Such a refinement should include handling of mixed push/pull systems

What's to be done next?

- Modeling and simulation with the aid of Petri nets is only limited by modelers' imagination and tool functionality, thus, such functionality has to be enhanced further
- Modelers need to create sophisticated and abstract models
  Hence, both tool support and modeling guidelines should be focused on
- Visualization capabilities should be integrated in the tools without need of external revision
  This includes both appropriate data models and display possibilities

BPMI (2004): BPMN 1.0 - Business Process Model and Notation. https://www.omg.org/spec/BPMN/ (last accessed 2020.09.20).

Genrich, H. J. and K. Lautenbach (1981): System Modelling with High-Level Petri Nets. *Theoretical Computer Science*, 13.

Ghezzi, C.; D. Mandrioli; S. Morasca; and M. Pezzè (1991): A Unified High-Level Petri Net Formalism for Time-Critical Systems. *IEEE Transactions On Software Engineering*, 17(2):160–172.

Hanisch, H.-M. (1992): *Petri-Netze in der Verfahrenstechnik*. Oldenbourg, München.

Hanisch, H.-M.; K. Lautenbach; C. Simon; and J. Thieme (1998): Timestamp Nets in Technical Applications. In: *IEEE International Workshop on Discrete Event Systems*. San Diego, CA.

Jensen, K. (1992): *Coloured Petri-Nets*. Springer, Berlin, 1st edn.

Knoeppel, C. E. (1915): *Installing Efficiency Methods*. The Engineering Magazine.

König, R. and L. Quäck (1988): *Petri-Netze in der Steuerungs- und Digitaltechnik*. Oldenbourg Verlag, München, Wien.

Merlin, P. (1974): *The Time-Petri-Net and the Recoverability of Processes*. Tech. rep., University California, Irvine.

Montali, M. and A. Rivkin (2019): From DB-nets to Coloured Petri Nets with Priorities (Extended Version). *CoRR*, abs/1904.00058.

Ohno, Taiichi (1988): *Toyota Production System*. Taylor & Francis, Milton Park, UK.

OMG (2011): BPMN 2.0 - Business Process Model and Notation. http://www.bpmn.org/ (last accessed 2020.09.20).

Ramchandani, C. (1974): *Analysis of Asynchronous Concurrent Systems by Timed Petri Nets*. Technical Report 120, MIT, Project MAC.

Reisig, W. (2013): *Understanding Petri Nets*. Springer, Berlin.

Sifakis, J. (1977): Use of Petri nets for performance evalutation. In: *Measuring, modelling and evalutating computer systems*, eds. H. Beilner and E. Gelenbe. North Holland Publ. Co., IFIP, pp. 75–93.

Simon, C. (2001): Developing Software Controllers with Petri Nets and a Logic of Actions. In: *IEEE International Conference on Robotics and Automation, ICRA 2001*. Seoul, Korea.