A Factory Simulation System Based on Cloud Services and Portable Scheduling Intelligence

Toly Chen
Department of Industrial Engineering and Systems Management
Feng Chia University, Taichung, Taiwan
Email: tolychen@ms37.hinet.net

Tzu-Yun Lin
Department of Industrial Engineering and Systems Management
Feng Chia University, Taichung, Taiwan
Tel: (+886) 962-066-896, Email: jennycrta@gmail.com

Abstract. Cloud-based factory simulation systems have become technically affordable solutions for the effective and efficient simulation of manufacturing systems. However, a simulation cloud (SC) may be incapable of implementing uncommon sequencing and scheduling methods. In addition, ensuring the logical correctness of the scheduling method adopted on an SC is difficult. To resolve these problems, the concept of scheduling intelligence, a special type of humanized intelligence, is proposed in this study. Scheduling intelligence accounts for the subjective beliefs and judgment in scheduling jobs in a factory and can be encapsulated into extensible markup language (XML) files based on the standard data structure defined on a cloud service provider (CSP). Then, an SC uses a dedicated XML parser to convert the XML files into modules to be called by the main program of the factory simulation system adopted on the SC. Two examples are provided to illustrate the practicability of the proposed methodology.

Keywords: Cloud manufacturing, cloud computing, humanized computing, scheduling, intelligence.

1. INTRODUCTION

Simulation has been applied to improve the performances of factories by, for example, finding a gain-optimal policy that minimizes the long-term inventory costs (Mahadevan et al., 1997), comparing the performances of different scheduling methods (Chen and Lin, 2009), enhancing the product and production engineering process (Kühn, 2006), and translating six-sigma philosophy into competitive solutions (Miller, 1994). However, simulating a factory requires a lot of expertise, takes much time and effort, and is a long-standing task. As a result, not many factories have successfully built, let along practicably applied, their own simulation models. Facing these problems, factories are seeking for alternatives, such as cloud-based simulation services, which constitute a motive for this study.

Cloud manufacturing (CM) is the application of cloud computing in the manufacturing sector to enabling on-demand network access to a shared pool of manufacturing resources including virtualized machines (Wu et al., 2007; Colombo et al., 2013), information systems, services, and manufacturing knowledge (Chen et al., 2014). Most of the existing cloud manufacturing system were configured as client-server systems. A client reads the web service description language (WSDL) file (in extensible markup language, XML) on a server to know what manufacturing services are available on the server. The WSDL file instructs a client how to call a manufacturing service, the parameters that should be inputted when making a call, and the data structure of the data exchanged. A factory is a client to a cloud service provider (CSP), while a service cloud can be either a client or a server to a CSP. A well-established infrastructure, the operating strategies of a CSP, and the sequencing of jobs on a SC are some critical issues to the effectiveness of a cloud manufacturing system. This study is devoted to the last issue. Specifically, a factory simulation system used by a SC is to be developed, which supports the functionality of a SC and therefore can be called a cloudlet according to the definition by Ferreira et al. (2013). However, this topic has rarely been investigated in the past.

The main problem associated with the existing methods is – the data structures of a factory model in different simulation systems are not the same, which cause a problem if the factory is to be simulated collaboratively by a number of SCs that use heterogeneous modelling logics (Pinedo, 2008) and database management systems. To solve this problem, in this study, a standard format (in extensible markup language (XML)) (Chen and Lin, 2015) is defined for modelling a factory online. In addition, the output report from a SC is also generated in XML to facilitate the subsequent aggregation operation. Finally, an experimental simulation system is established to illustrate the applicability of the proposed methodology.

On the other hand, only a few scheduling rules, such as first in first out (FIFO), earliest due date (EDD), and shortest processing time (SPT), are commonly built in the existing scheduling systems. To apply an advanced or tailored scheduling method, a user must write program codes with the programming language specified by the simulation system, which is inconvenient and error-prone and reduces the user's willingness to use the simulation system. To solve this problem, the concept of scheduling intelligence is proposed. Scheduling intelligence is a type of humanized intelligence that accounts for subjective beliefs and judgment in scheduling the jobs in a factory. Based on the standard data structure of a factory model, scheduling intelligence can be realized by coding an XML for describing a scheduling method that can then be exchanged between a CSP and an SC or between two SCs. In the past, a scheduling method was either described using pseudocodes or converted into the corresponding structured query language (SQL) operations. However, pseudocodes are usually not standardized, and SQL operations have security risks such as SQL intrusion or injection (Pinzón et al., 2013). Therefore, this study uses XML.

2. THE PROPOSED METHODOLOGY

2.1 Defining a Factory Simulation Model in XML

Extensible Markup Language (XML) is an open standard that defines a set of rules for encoding documents in a format which is both human-readable and machine-readable (World Wide Web Consortium, 2008). XML is designed for exchanging data on the Internet. In addition, it is straightforward to import or export XML files into and from common databases. For these reasons, defining a factory simulation model in XML is a viable strategy for overcoming the incompatibility problem among various simulation systems. In the literature, Chen and Lin (2015) have taken this treatment. In their methodology, the data structures of the XMLs for modelling a factory were defined in document type definition (DTD) or XML schema definition (XSD) files.

In the proposed methodology, a factory simulation model is composed of two XML files – jobs.XML, and scenario.XML. The XSDs for defining the XML files are shown in Figure 1. In addition, another XSD for defining the output report from a SC is given in Figure 2, in which the name of SC is a required field for discriminating the output reports from different SCs.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="job">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="job_no" type="xs:string"/>
      <xs:element name="release_time" type="xs:date"/>
      <xs:element name="processing_time" type="xs:decimal"/>
      <xs:element name="due_date" type="xs:date"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

(jobs.XSD)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="scenario">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="scenario_no" type="xs:string"/>
      <xs:element name="dispatching_rule" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

(scenario.XSD)

Figure 1: The XSDs for defining the XMLs.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="output">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="job_no" type="xs:string"/>
      <xs:element name="SC" type="xs:string"/>
      <xs:element name="start_time" type="xs:date"/>
      <xs:element name="completion_time" type="xs:date"/>
      <xs:element name="lateness" type="xs:decimal"/>
    </xs:seque>
  </xs:complexType>
</xs:element>

</xs:schema>
```

(output.XSD)

Figure 2: The XSD for defining the output report.

Based on the XSDs, the simulation model of a factory and the accompanying scenario are both defined in XML. The corresponding output report is also generated in XML.

2.2 Scheduling Intelligence

Traditionally, scheduling intelligence is realized in the following ways:

(1) Writing the pseudo codes for implementing a scheduling intelligence. However, pseudo codes are usually not standardized.

(2) Drawing a flow chart that can be converted into program codes for implementing a scheduling intelligence. However, it is not easy to draw a logically correct flow chart that can be smoothly converted into the required program codes.

(3) Developing a module to perform the required database operations, set operations, and calculations. However, the data structures, databases, and programming languages adopted in different simulation systems are not the same, causing difficulties in performing scheduling intelligence in exactly the same way. Consequently, transferring scheduling intelligence from one system to another becomes challenging.

Cloud manufacturing provides some solutions to these problems:

(1) Data transferred between the participants of a cloud manufacturing system are usually encoded in the form of standardized XML.

(2) The data structure of a simulation model is defined by the DTD or XSD files on a CSP. To serve the CSP, an SC must obey or adapt itself to the standard data structure.

With a standardized data format and structure, the operations required for implementing scheduling intelligence can be defined in a standard manner. Thus, scheduling intelligence can be easily transmitted from one participant to another and across different systems and platforms; in other words, it becomes portable.

The proposed methodology involves the following three major parts, as illustrated in Fig. 3:

(1) The steps required for implementing scheduling intelligence, which are defined in XML files based on the standard data structure defined on a CSP.

(2) The XSDs for defining the XML files.

(3) A standard terminology table.

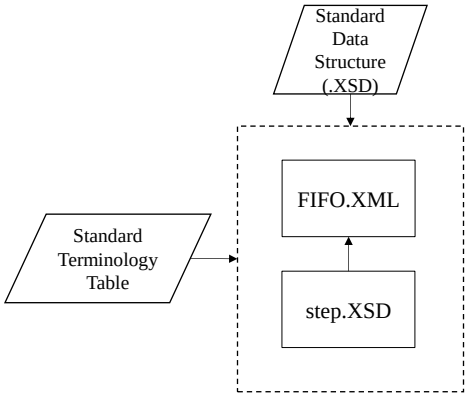

Figure 3: Major components of the proposed methodology.

3. AN EXPERIMENTAL SIMULATION SYSTEM

An experimental system has been established to illustrate the applicability of the proposed methodology. Visual Studio 2015 was applied to develop the experimental simulation system on a PC with Intel Core i7-4770 CPU 3.40 GHz and 8 GB RAM. The program codes were provided in Appendix. There were two modes of running the experimental simulation system. In the manual mode, the system administrator manually controlled the execution of each step, while in the automatic mode, the system continuously scanned the system database to search for any simulation task that has not been fulfilled (see Figure 4). If there are more than one unhandled task, the one with the earliest granted time would be simulated first.

simulation-tasks

task-no	granted-time	CSP	files-URL	simulated	按一下以新增
1	2015/10/14 下午 12:03:55	CSP BBB	http://www.CSPBBB.com/tasks/5g7Wsi6	☑	
2	2015/10/14 下午 12:05:32	CSP AAA	http://www.CSPAAA.com/models/XML/nl	☐	
3	2015/10/14 下午 12:06:01	CSP BBB	http://www.CSPBBB.com/tasks/6h87ujb	☐	
(新增)				☐	

Figure 4: The system database is scanned for unhandled tasks.

The factory data and simulation scenario were prepared by an engineer of the simulated factory and uploaded on to a CSP. Both of the two files were in XML. Then, the data of jobs and the scenario were imported from the CSP based on the URL provided by the CSP, and were saved into the system database of the SC if necessary. In the experiment, the system database was built using Microsoft Access 2013. Subsequently, the dispatching rule specified in the scenario

was applied to sequence jobs in the factory. Dispatching rules built in the experimental system included first in first out (FIFO), shortest processing time (SPT), earliest due date (EDD), and critical ratio (CR). Let us take the FIFO dispatching rule as an example. The XML file for implementing FIFO is shown in Figure 5, which is defined on the basis of the standard data structure, the XSD file, and the standard terminology table.

```
<?xml version="1.0" encoding="UTF-8"?>
<dataroot>
  <step>
    <step_name>sequencing</step_name>
    <input>ALL</input>
    <sort_by>release_time</sort_by>
    <output>S1</output>
  </step>
  <step>
    <step_name>scheduling</step_name>
    <input>S1</input>
    <condition>release_time&gt;SYSTEM_TIME</condition>
    <satisfied>SYSTEM_TIME=release_time</satisfied>
    <calculation>start_time=SYSTEM_TIME</calculation>
    <calculation>completion_time=start_time+processing_time</calculation>
    <calculation>SYSTEM_TIME=completion_time</calculation>
  </step>

</dataroot>
```

Figure 5: XML file for implementing FIFO.

Although the simulated factory was small, the simulation results still successfully demonstrated the applicability of the proposed methodology. In addition, it becomes possible for a general-purpose computing cloud to join the collaborative simulation of a factory by incorporating the related scheduling intelligences, which not only reduces the technical burden on a computing cloud but is also conducive to the scalability of the cloud-based factory simulation system.

4. CONCLUSIONS

A standard format (in XML) is defined for modelling a factory online in this study. In this way, the factory model can be easily imported into various database management systems, which makes it suitable for cloud-based applications, such as simulating a factory by several SCs collaboratively. In addition, the output report from a SC is also generated in XML to facilitate the subsequent aggregation operation. To illustrate the applicability of the proposed methodology, an experimental simulation system, including a dedicated simulator that accepted and generated only XML files, has been established. The experimental simulation system was then applied to fulfill a factory simulation task passed from a CSP.

Subsequently, for implementing scheduling intelligence, defining a factory simulation model on a standard data structure defined on a CSP is a prerequisite. Subsequently, the steps required for implementing scheduling intelligence are defined in XML files based on the standard data structure. Finally, an SC uses a dedicated XML parser to convert the XML files into modules to be called by the main program of the factory simulation system adopted on the SC.

The successful experience on this small case not only supported the effectiveness of the proposed methodology, but also encouraged us to extend the capability of the experimental simulation system.

ACKNOWLEDGMENTS

This study is sponsored by Ministry of Science and Technology, Taiwan.

REFERENCES

Colombo, A. W., Bangemann, T., Karnouskos, S., Delsing, J., Stluka, P., Harrison, R., Jammes, F., and Lastra, J. L. (2013) *Industrial Cloud-Based Cyber-Physical Systems - The IMC-AESOP Approach.* Switzerland: Springer International Publishing.

Miller, D. J. (1994) The role of simulation in semiconductor logistics. *Winter Simulation Conference Proceedings*, pp. 885-891.

Lindskog, E., Berglund, J., Vallhagen, J., Berlin, R., and Johansson, B. (2012) Combining point cloud technologies with discrete event simulation. *Proceedings of the Winter Simulation Conference*, 281, 1-10.

Ferreira, L., Putnik, G., Cunha, M., Putnik, Z., Castro, H., Alves, C., Shah, V., and Varela, M. L. R. (2013). Cloudlet

architecture for dashboard in cloud and ubiquitous manufacturing. *Procedia CIRP*, **12**, 366-371.

Lamport, L., "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, **vol. 21**, no. 7, pp. 558-565, 1978.

Wu, L., Meng, X. and Liu, S. (2007) Service-oriented encapsulation of manufacturing resources. *IEEE International Conference on Services Computing*, pp. 727-728.

Held, M., and Karp, R. M. (1962) A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, **10**(1), 196-210.

Pinedo, M. L. (2008) *Scheduling: Theory, Algorithms, and Systems*, New York: Prentice Hall.

Brucker, P. (2001) *Scheduling Algorithms*. Berlin: Springer.

Mahadevan, S., Marchalleck, N., Das, T. K., and Gosavi, A. (1997) Self-improving factory simulation using continuous-time average-reward reinforcement learning. *Proceedings of the 14th International Conference on Machine Learning*, pp. 202-210.

SQLCourse.com (2015) What is SQL? *http://www.sqlcourse.com/intro.html*

Chen, T. (2003) A fuzzy back propagation network for output time prediction in a wafer fab. *Applied Soft Computing*, **2(3)**, 211-222.

Chen, T. (2013) An effective dispatching rule for bi-objective job scheduling in a wafer fabrication factory - considering the average cycle time and the maximum lateness. *International Journal of Advanced Manufacturing Technology*, **67(5-8)**, 1281-1295.

Chen, T. (2014) Strengthening the competitiveness and sustainability of a semiconductor manufacturer with cloud manufacturing. *Sustainability*, **6**, 251-268.

Chen, T., and Lin, C.-W. (2015) Estimating the simulation workload for factory simulation as a cloud service. *Journal of Intelligent Manufacturing*, in press.

Chen, T., and Lin, Y.-C. (2009) A fuzzy-neural fluctuation smoothing rule for scheduling jobs with various priorities a semiconductor manufacturing factory. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, **17(3)**, 397-417.

Chen, T., and Lin, Y.-C. (2015) A digital equipment identifier system. *Journal of Intelligent Manufacturing*, in press.

Chen, T., and Wang, Y.-C. (2015) Estimating simulation workload in cloud manufacturing using a classifying artificial neural network ensemble approach. *Robotics & Computer Integrated Manufacturing*, in press.

Chen, T., Wang, Y.-C., and Lin, Z. (2014) Predictive distant operation and virtual control of computer numerical control machines. *Journal of Intelligent Manufacturing*, in press.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009) *Introduction to Algorithms*, MIT Press.

Kühn, W. (2006) Digital factory: simulation enhancing the product and production engineering process. *Proceedings of the 38th Conference on Winter Simulation*, pp. 1899-1906.

World Wide Web Consortium (2008) Extensible markup language (XML) 1.0 (fifth edition). *http://www.w3.org/TR/REC-xml/*

Chi, X., Pepper, M. P., and Spedding, T. A. (2004) A web-based virtual factory and simulator for industrial statistics. *Winter Simulation Conference*, pp. 2103-2106.