한국시뮬레이션학회 '00추계학술대회 논문집 2000.11.11 아주대학교

Virtual FMS Architecture for FMS Prototyping

Byoungkyu, Choi , Beumchul, Park, Donghwan, Hwang
VMS Lab., Department of Industrial Engineering
Korea Advanced Institute of Science & Technology (KAIST)
373-1 Kusong-dong, Yusong-gu, Taejon, Korea
bkchoi@vmslab.kaist.ac.kr, pbc@vmslab.kaist.ac.kr, snobbery@vmslab.kaist.ac.kr

Abstract

Proposed in the paper is a V-FMS (Virtual Flexible Manufacturing System) model to be used as a prototyping tool for FMS design. The proposed V-FMS framework follows an object-oriented modeling (OOM) paradigm and is based on a set of user requirements for FMS prototyping. The V-FMS model consists of four types of object: *virtual device*, *transfer handler*, *state manager* and *flow controller*. A virtual device model, which corresponds to a static model in OOM, consists of two parts, *shell* and *core*, for reusability. A transfer handler corresponds to a functional model of OOM and it stores low level device commands required to perform job flow operations between giving and taking devices. The state manager and the flow controller constitute a dynamic model of OOM. The proposed V-FMS model has been implemented for a couple of linear type FMS-lines

Key Words : virtual FMS, virtual factory, FMS prototyping, FMS simulation.

1. Introduction

If manufacturers are to remain competitive in the continuously changing market place, they must not only continue to improve their products, but also strive to continuously improve production systems. Thus an efficient prototyping environment for production systems is crucial, resulted in the concept of virtual factory (VF). Virtual factory (VF) has various definitions for different viewpoints and perspectives. The concept of VF may be described as a model executing manufacturing processes in computers as well as the real world5,6.

Onosato *et al.* (1993) proposed the architecture of virtual manufacturing systems consisting of virtual physical system and virtual informational system. Iwata *et al.* (1995) presented the architecture of virtual manufacturing systems for modeling and

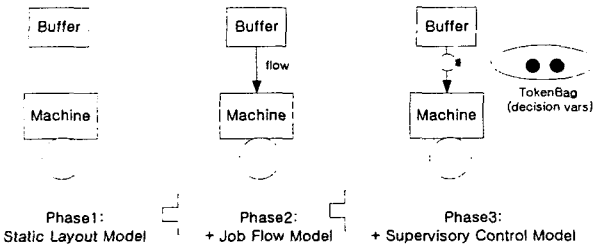
distributed simulation. Some researchers have employed conventional modeling technologies for the modeling and simulation of a virtual factory: colored timed Perti net(Huang 1999), an analytic form(Lin 1999), and DEVS(Hwang 2000). As with the object-oriented modeling paradigm of Rumbaugh *et al.* (1991), the JR-net modeling framework(Choi *et al.* 1996) was developed to cope with the drawbacks of formal modeling tools (Petri net, Event Graph, ACD), such as modeling difficulty and near-intractability.

To summarize from the previous work on the topic of VF, some aimed at developing the overall concept and architecture of VF, some focused on modeling scheme of VF. However, these schemes are more conceptual than realistic, for their results are either too coarse or somewhat fragmentary. Obviously, we can get many benefits from VF, but it is still difficult and time-consuming to build.

In this paper we present a virtual FMS (V-FMS) model, VF of an FMS, considering the following requirements; 1) Layout design, 2) Manual operation for layout evaluation, 3) Flow control logic programming, 4) Simulation for performance analysis, 5) Planning for implementation of a real FMS, 6) Fidelity for high-quality simulation results, 7) Easy to maintain (reusability of the model), 8) Modeling power and 9) Easy to build.

2. Approach to Develop V-FMS Model

Before explaining our approach, let us introduce a reference model of an FMS for the common view of an FMS. We employ the three-phase modeling framework(Choi *et al.* 1996) consists of three sub-models: static layout model (object model), job flow model (functional model), and supervisory control model (dynamic model). Figure 1 shows the three-phase modeling procedure; the static layout model is constructed by positioning 2D device symbols, the job flow model is described by defining flows (arrows) between devices and the supervisory control model defines the firing condition of the flows according to the system state (decision variables).

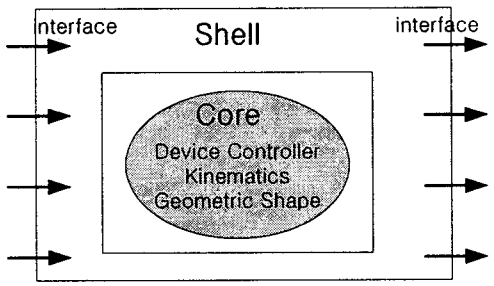


[Fig.1 : Three sub-models modeling procedure]

Based on the three-phase modeling framework, we suggest a V-FMS model satisfying the requirements mentioned in the previous section and concrete enough to be implemented. The following are the important issues for designing the three sub-models, and our approaches.

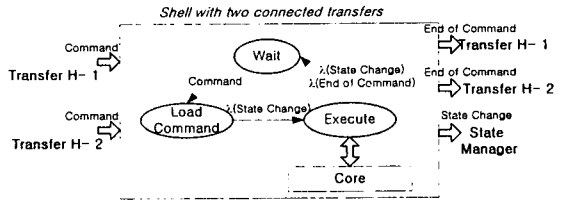
2.1 Static layout model(*virtual device*)

The static layout model consists of the set of manufacturing devices having their positions represented in the layout. To represent a manufacturing device, we employ the concept of a *virtual device*; a model imitating the behaviors of the real device. For the reusability, the *virtual device* needs to be split into two parts, shell and core; the shell part may be instanced (changed) according to the FMS configuration and the core part undertakes inherent properties of the device, such as device controlling (to execute device level commands), kinematics and geometric shape. The rough concept of the *virtual device* is shown in figure 2.

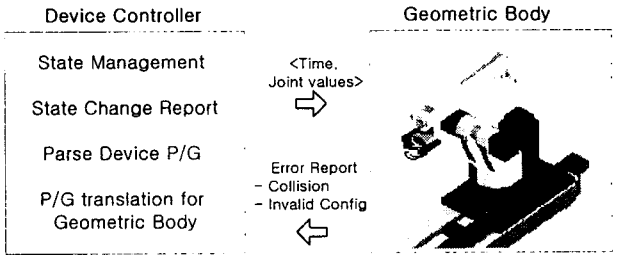


[Fig.2 : Outline of the *virtual device* model]

It is the main mission of the shell part to offer interface with other objects (*transfer handler*, *state manager*) and the shell part is represented as an atomic model containing the core part. The state transition diagram of the shell part is shown in figure 3, and an example of the core part is figure 4.



[Fig.3 : Shell part of a *virtual device*]

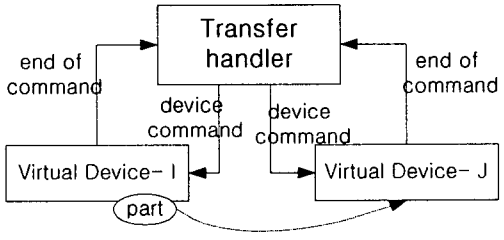


[Fig.4 : Core part of *a virtual device*]

2.2 Job flow model(*transfer handler*)

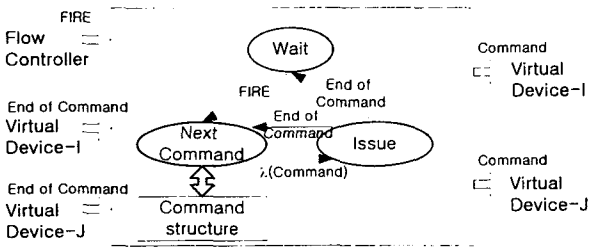
The job flow model is considered as a set of arrows defining the transfers of parts and of auxiliary resources between devices. Conventional approaches considered the flow as nothing more than an arrow between giving and taking devices, which may damage the fidelity of the model.

We define the *transfer handler* as a set of device level commands including their relationships (pre/post, synchronization). The concept of the transfer handler is shown in figure 5.

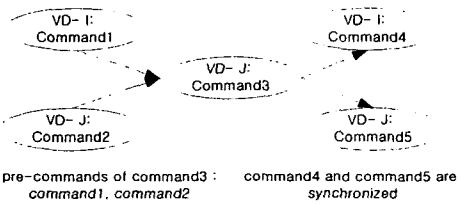


[Fig.5 : Outline of the *transfer hander* model]

To represent the relationships between required commands, we employ a simple structure, as shown in figure 6. With the command structure, the *transfer handler* selects a next command to be issued, when it gets the feedback, end of command, from the *virtual devices*. Figure 6 shows the state transition diagram of the *transfer handler* including the command structure.



(a) State transition diagam of the Transfer Model

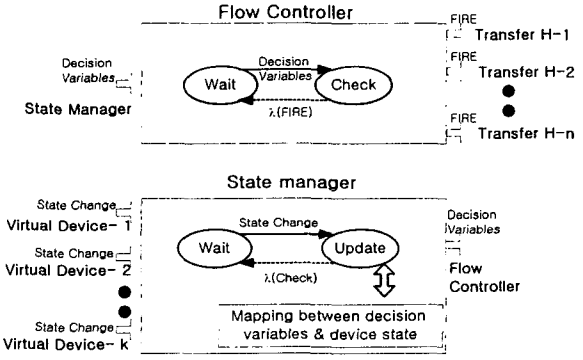


(b) Example of the command structure

[Fig. 6 : *Transfer handler* & command structure]

2.3 Supervisory control model(*state manager, flow controller*)

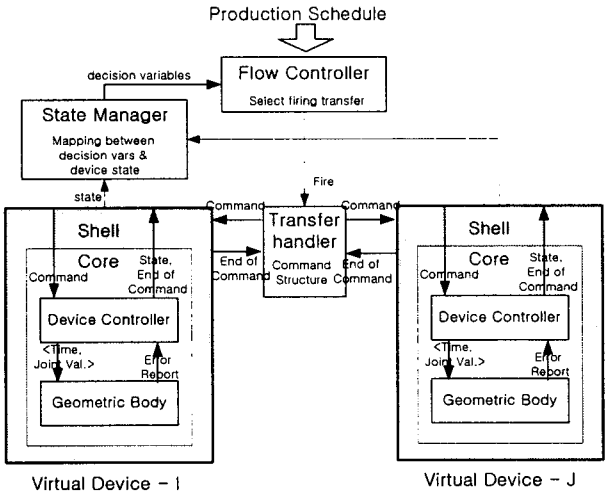
The supervisory control model has the flow control logic to select firing transfers according to decision variables reflecting the system state. Traditionally, the main purpose of FMS modeling has been to analyze the performance of the flow control logic. There was no need to consider about the mapping between the decision variables and real sensors monitoring system state. However, in order to support the requirement, planning for the implementation of a real FMS, we need to consider the mapping between the decision variables and the sensors. We should note that one of the most delicate processes for the implementation of a real FMS is the design of sensors. For the supervisory control model, we employ two objects (*flow controller* and *state manager*), the *flow controller* makes decisions according to the decision variables and the *state manager* stores the mapping between decision variables and the state of devices (device level sensors), as shown in figure 7.



[Fig. 7 : *State manager* and *flow controller*]

As a result, the proposed V-FMS model consists of four objects, *virtual device*, *transfer handler*, *state manager*, and *flow controller*. We employ Zeigler's DEVS formalism to implement these objects, because the semantics of the formalism is highly compatible with object-oriented specifications for simulation

models. The overall architecture of the proposed V-FMS model is given in figure 8.



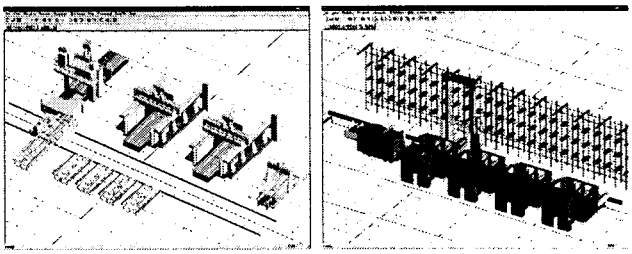
[Fig. 8 : Proposed V-FMS model]

3. Examples & Illustrations

We have been implementing a software system, named V-FMS, for the efficient FMS prototyping with the proposed V-FMS model. For the implementation, we used C++ language in Visual C++6.0, OpenGL for the graphical rendering, GKDEVS for simulation engine, and I_COLLIDE for collision detection.

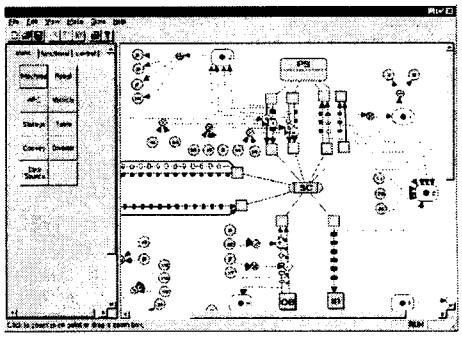
Figure 9 shows two linear type FMS-lines. The FMS line in figure 9-a is an automobile stamping-die machining line having three high-speed machining centers and a CMM, and the FMS line in figure 9-b is a mechanical parts machining line having four machining centers and a washing machine.

The modeling of the geometric body of the *virtual device* is performed interactively by the constructive solid geometry (CSG) modeling scheme. The modeling process of the *transfer handler* and the flow control logic requires well-designed GUI. As shown in figure 10, we employ the JR-net modeling scheme, 2D symbolic modeling GUI (Choi et al. 1996) for the efficient modeling of the *transfer handler* and flow control logic.



(a) (b)

[Fig. 9 : Two linear type FMS]



[Fig.10: Flow control logic modeling GUI]

4. Conclusions

In the paper we present a virtual FMS (V-FMS) model for linear type FMS lines based on a set of user requirements observed from a FMS prototyping process. Similar to the object-oriented modeling paradigm of the

Rumbaugh *et al.* (1991) and the JR-net modeling framework, the proposed V-FMS model consists of four objects; *virtual device* (static layout model); *transfer handler* (functional job flow model); *state manager* and *flow controller* (supervisory control model).

The proposed V-FMS model aims to serve as a prototyping tool for automated manufacturing systems design. However, the validity of the V-FMS architecture has been demonstrated only for linear type FMS lines having a single level control. For example, a FMS line having an AGV-net (having more than one AGV) may require a hierarchical control scheme, which in turn would require a hierarchical transfer-model as well as a multi-level flow-controller. A further investigation is needed along this line. In addition, a more structured implementation and testing of the proposed V-FMS model are needed.

Reference

1. B.K. Choi, Kwan H. Han, Tea Y. Park, Object-oriented graphical modeling of FMSs, *The international journal of Flexible Manufacturing Systems*, 8, 1996, 159-182
2. J. Rumbaugh, M. Blaha, W. Premerlani, W. Losenson, *Object-Oriented Modeling and Design*, Prentice Hall Inc., Englewood Cliffs, NJ, 1991.
3. B. P. Zeigler, *Multifacetted modeling and discrete event simulation*, Academic Press, Orland, 1984.
4. M.H. Hwang, 자동화 제조시스템용 시뮬레이터 개발을 위한 DEVS 형식론의 응용에 관한 연구, 박사논문, I.E dept, KAIST, Korea, 1999.
5. M. Onosato, K. Iwata, Development of a virtual manufacturing system by integrating product models and factory models, *CIRP*, 42(1), 1993, 475-478.
6. K. Iwata, M. Onosato, K. Teramoto, S. Osaki, A modeling and simulation architecture for virtual manufacturing systems, *CIRP*, 44(1), 1995, 399-402.
7. H. Huang, C. Yeh, Development of a virtual factory emulator based on three-tire architecture, *IEEE Int. Conf. On Robotics & Automation*, Detroit Michigan, May 1999, 2434-2439.
8. L. Ye, F. Lin, Virtual system simulation A step beyond the conventional simulation, *22nd Int. Conf. on Computer and Industrial Engineering*, 1997/12/20, 304-306
9. M. Lin, L. Fu, T. Shih, Virtual Factory A novel testbed for an advanced flexible manufacturing system, *IEEE Int. Conf. on Robotics & Automation*, Detroit Michigan, May 1999, 2422-2427