The Development of PC-Based Highly Model-Mixed Engine Dispatching Systems

Chung-Hsien Kuo
Graduate Student
Robotics Laboratory, Dept. of Mechanical Engineering
National Taiwan University
Taipei, TAIWAN 10674
R.O.C.

Han-Pang Huang
Professor
Robotics Laboratory, Dept. of Mechanical Engineering
National Taiwan University
Taipei, TAIWAN 10674
R.O.C.

Kuang C. Wei
Sr. Tech. Specialist
Research Laboratory,
Ford Motor Co.
20000 Rotunda
Dr. Dearborn, MI 48121
U.S.A.

Steve S. H. Tang
Manager
Head Office and Plant
Ford Lio-Ho Motor Co.
705 Chung Hwa Rd. sec.
1 Chung Li, TAIWAN
R.O.C.

Abstract – **This paper aims to propose the design and implementation of a real-time model-mixed engine dispatching system . The proposed engine dispatching system (EDS) is applied to the engine plant of Ford Lio-Ho Motor Company. The production system of Ford Lio-Ho Engine Plant (FLHEP) is a highly model-mixed production system. It attemps to achieve the goals of more flexibility, higher efficiency, lower cost and less shop floor space. In order to handle such a model-mixed production system, an efficient real-time dispatcher, EDS, is required. The EDS is a client-server configuration. The EDS server handles the real-time engine dispatching, operator's loading balance, integrating distributed data and security control. Besides, the EDS server provides the performance measure, graphical user interface, user's classification, production information and dispatching information. On the other hand, the EDS server broadcasts the information to all EDS clients. In this manner, the production information can be synchronized and the engines can be assembled in parallel in FLHEP. The objectives of EDS are to eliminate the bottlenecks of the original system, balance operator's loading, improve product's quality, integrate distributed production data and provide efficient dispatching rules. Since the proposed EDS is constructed by using the modular design and object-oriented approach, it is easy to integrate, extend, test and debug.**

I. INTRODUCTION

Recently, the information technology is getting more important and more popular, especially on the internet and intranet. Efficient application of information technology on production system can improve the competency of enterprise. In this paper, the proposed engine dispatching system (EDS) integrates the information technology and dispatching algorithm to improve the performance of the original system. In addition, reasonable loading is important to the operators, and it directly affects the product's quality. Thus, the EDS also considers the operator's loading balance.

The production system of Ford Lio-Ho Engine Plant (FLHEP) is a highly model-mixed production system. Such an efficient flexible assembly system (FAS) is adaptive to the fast transition of market requirements [13]. Although the objective of this paper focuses on EDS, the procedures of analysis and modeling can not be ignored. In fact, all methodologies in developing EDS are derived from system model and system analysis. The Colored Timed Petri Net (CTPN) [6, 7, 8] is used for modeling this FAS. The CTPN is especially suitable because it can describe the concurrent and asynchronous discrete event dynamical systems (DEDSs). The readers can refer to [7] for detailed description of CTPN models of the flexible assembly system (FAS).

After analyzing the production system of FLHEP, some mismatches are found:

1. Although highly model-mixed production is flexible and efficient, the unbalancing loading of the operators and the mis-decking between the engine and the car body may occur.
2. Real-time dispatching rules are required to solve the decking problems.
3. Distributed and different databases, such as mainframe production database and PC-based scheduling databases, are located at various machines. These databases must be integrated to improve information share and decision support of the real-time dispatcher.
4. Client-server architecture [2] is necessary to provide distributed service. In this manner, the production information of all sub-lines and mainlines of FLHEP can be synchronized and the engines can be assembled in parallel.
5. An intelligent and efficient dispatcher that can resolve these mismatches is necessary. Besides, this dispatcher must be easily implemented and integrated, and it has the properties of low cost, continuous improvement and security control.

Therefore, an EDS will be developed in terms of Microsoft VC++ [9, 11] to satisfy all requirements. Furthermore, the EDS is constructed based on object-oriented approach and modular design [10]. Thus, all functions and modules can be tested individually, and the efforts of implementation and debugging may be reduced. Since the EDS users include managers, engineers and operators, the functions of security control and user's classification shall distinguish the users' priorities so that the production information and dispatcher's setting can be protected. Once the EDS server is constructed, the EDS client can be easily created by using open database connectivity (ODBC) [12] or directly accessing the remote database of the EDS server.

II. A HIGHLY MODEL-MIXED PRODUCTION SYSTEM

A model-mixed production system is flexible and efficient. Owing to the high flexibility and efficiency, the model-mixed production system is more complex than the fixed type production. Fundamentally, a model-mixed

0-7803-4192-9/97/$10.00 © 1997 IEEE

assembly line can assemble multiple types of product on the same assembly line to accommodate rapid products and demand changes. Flexible fixtures, flexible tooling and skilled human operators are key elements of the highly model-mixed assembly line. Besides, timely delivery of material is crucial in keeping the line running at the desired speed.

A highly model-mixed production system has the following properties [7]:

1. Flexibility.
2. Production efficiency with low cost.
3. Meeting customers' need and delivering the products in a timely fashion.
4. Ability to quickly adapt to production condition changes.
5. Low work in process (WIP) and low inventory.
6. Ability to cope with rapid market and demand changes.
7. Low floor space requirement.

Since the production of model-mixed production system is complex and varying with time, thus the following factors should also be considered:

1. Training.
2. A real-time dispatcher.
3. Highly stable and reliable equipments.
4. Loading balance of operators.
5. Efficient dispatching rules.
6. Integration of production information.
7. Concurrent control and synchronization.

To examine its structure and functionality, we decompose the FAS hierarchy into five levels as shown in "Fig. 1".

1. Production level: It is the highest level in an FAS. It handles the system operations including master production schedule (MPS), system resource management, database, maintenance schedule and system performance measurement. One example system on this level is the vehicle information system (VIS) in an automobile factory.
2. Plant process level: This is the control center of a plant. It coordinates and controls all main assembly lines in the plant.
3. Main assembly line level: It is the level of the main assembly lines below the plant process level.
4. Sub-assembly line level: It is the level of the sub-assembly lines under the main assembly line level.
5. Machine level: It consists of machining, inspection and material handling equipments.

In this paper, we focus on a particular automobile FAS. The material flow of this system is shown in "Fig. 2", where EDU and PBS denote the engine dress-up and the painted body storage area, respectively. The engine assembly line (EAL) starts from the Block-In station and ends at Engine Buy-Off station right before the EDU line.

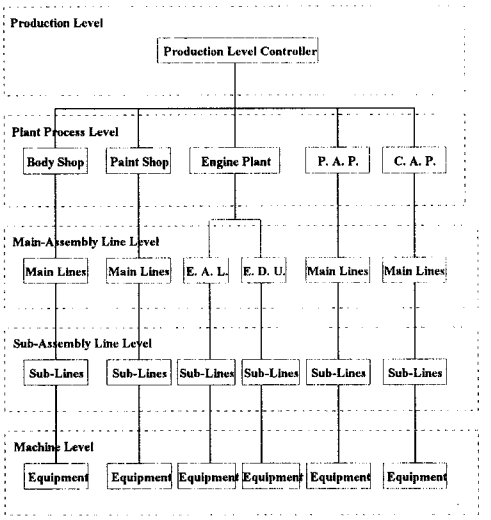

Fig. 1. Hierarchical structure of the FAS

III. DEVELOPMENT OF A BALANCED BUDGET WORK STANDARD FOR A SMALL BATCH PRODUCTION

In order to provide reasonable loading of operators and high performance of model-mixed production system, a methodology of balanced budget work standard (BWS) for a small batch production is developed to control EDS. Being flexible and dexterous, human operators are key elements in a highly model-mixed production system. The performance of such a system and the quality of products tie closely with the operator's performance. Overloaded operators may degrade the quality of products. On the other hand, underloaded operators directly affect the throughput of the system. In a model-mixed production system, different product types often have different budget work standard (BWS, it is the standard time measurement for an operator to assemble one product completely). It is crucial to select a proper sequence of model mixes so that budget work standard can be "balanced" as close as possible. For example, the assembly time of a series of processes for an operator is 2.795, 1.487, 1.487 and 2.795 hrs. (hours), respectively. Then a job command sequence of 2.795, 1.487, 2.795 and 1.487 hrs. would be better, or more "balanced" than the command sequence of 1.487, 1.487, 2.795 and 2.795 hrs for the operator since the operator may not be very busy or relaxed in some interval of time.

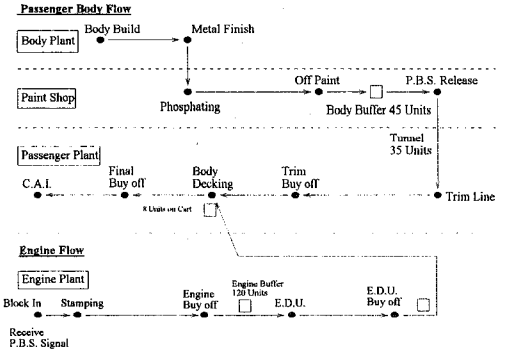

Fig. 2. Material flow of an Automobile FAS

In the original system, the dispatching command is performed manually by a human scheduler in order to balance the budget work standards. Such process is subjective and unstable. The personal experience and attitude affect the dispatching results heavily. Thus, there is a strong need to develop a systematic algorithm to generate a balanced budget work standard for a small batch production.

The following parameters are defined to illustrate the development of a balanced budget work standard for a small batch:

1. Weighed mean budget work standard:

$$\overline{BWS} = \frac{\sum_{i=1}^{k} W_i BWS_i}{n}$$

where n is the total number of products in the daily schedule; k is the total number of product types in the daily schedule; BWS_i is the budget work standard of product type i; W_i is the number of products for product type i in the daily schedule.

2. Let M be the number of products in a small batch. The manager selects M out of S products from the sample space. The operator can reset the loading after assembling M products. Namely, the operator can reset the overloading value to zero after assembling M products.
3. Let S be the number of products in the sample space $S \geq M$. The goal of dispatching is to select M out of S products to form a small batch and satisfy the balanced budget work standard.

A direct search algorithm for selecting M products to achieve a balanced budget work standard is developed as follows:

1. Select M products arbitrarily from S products. The number of possible selection combinations will be C_M^S.
2. Sum up the process time for each combination, i.e. $SUM(j)$, where j is from 1 to C_M^S.
3. Calculate the deviation of these process time from the mean budget work standard for each combination, $Error(j)$, defined as:

   $Error(j) = abs(SUM(j) - BWS \times M)$

   where $abs(.)$ denotes the absolute value.
4. If $Error(j)$ equals to zero, this combination, j, meets the balanced budget work standard.
5. If $Error(j)$ is not equal to zero for all combinations, choose the minimum one as $Error(j)$. This value will be recorded and added to the next batch computation to compensate the previous loading.
6. To improve the convergence and reduce the computation time, we can define an acceptable deviation range. If $Error(j)$ falls within this range, this combination is considered to meet the balanced budget work standard requirement. If none of the combinations meets the specified range, we can relax this range until one of $Error(j)$ satisfies the range. Again, we may record this value and add it to the next batch computation to compensate the previous loading.
7. Once an acceptable combination is found, the same product type should not be used subsequently. In addition, the first product's type in this combination shall not agree with the last product's type in the previous combination.
8. The values of M, S, acceptable deviation range and the range relaxation are determined by taking into account both the product's assembly budget work standard and the desired system performance.

IV. MODULAR DESIGN

The proposed EDS is constructed in terms of modular design and object-oriented approach. The programs of EDS are coded by using Microsoft Visual C++. Therefore, the EDS can provide strong computation and graphical user interface [5].

The schematic architecture of the EDS is given in "Fig. 3". The function of each module is described as follows:

1. Production information integration: the open database connectivity (ODBC) is chosen to integrate the distributed and various databases. In the proposed system, there exist several kinds of database developed by different departments. These databases are stored and managed by different machines and data systems. Since the cooperation of production in different plants shall be concurrent and synchronized, the distributed data must be integrated. In the parent system, the major production information is built in a vehicle information system (VIS), which is a database stored in a mainframe UniSys A15 machine. On the other hand, the daily, monthly schedules and engine BWS data are developed by using Microsoft Excel in different departments and stored in personal computers. These databases must be integrated to support decision making of the EDS dispatcher.
2. Security control and user's classification: Security is important since all the process parameters must be well protected. Thus, in addition to the module of database integration and management and the module of EDS dispatcher kernel, the module of user's classification must be also involved. The priority setting can be used to distinguish the right of the users. The management of the users focuses on user account management. It includes the functions of adding, modifying, and removing users. In the EDS, password is encoded to a specified form by using the technology of cryptography [2]. It is very difficult to be decoded by any other users. The priority numbers range from 1 to 999, and number 999 is used for the administrator only. Generally, the priority levels are divided into three levels: administrator and managers, engineers and supervisors of operators, and operators. If necessary, the production information in the network can be encrypted and decrypted by using RSA cipher [2], and the production

information can be greatly protected. The priority table is shown in "Table 1".

3. Real-time EDS dispatcher kernel: as described before, the analysis and modeling [4] of the proposed FAS are important. The complete CTPN models are simulated on an G2 real-time expert system [3]. The EDS extracts some important modules and functions which are important and easy to implement from the real-time simulator [7]. The objectives of EDS dispatcher kernel are: (1) applying methodologies of model-based and loading balance to dispatching; (2) providing reasonable sequences of engine blocks to be assembled on engine assembly line (EAL); (3) improving the quality of engine block assembly on EAL; (4) attempting to find reasonable units per day (UPD) that the operator can accept and system performance is not affected. In EDS, ODBC is used to define the input and output interfaces of EDS dispatcher. The outputs of EDS are stored in Access database by using ODBC. The block diagram of EDS dispatcher kernel is shown in "Fig. 4". Besides, the EDS can inspect the errors of the transactions of files or messages. Upon detecting the occurrences of these errors, the EDS can handle them properly.
4. Performance presentations and reports: The reports and performance presentations are encapsulated in a module. In this module, graphical user interfaces are used to represent the performance measurements. Since all of the input and output items are stored in EDS database by using ODBC, the data sources of this module are the data items stored in EDS database. All designs in this module are flexible and adaptive to the user's requirements.

From the discussions in the previous modules, the readers can find that each module can be constructed and tested independently. The most important thing is to define clear interfaces so that these modules can be integrated easily. The modular design is beneficial to the implementation of the proposed EDS. Besides, such approaches are easy to extend, debug and modify.

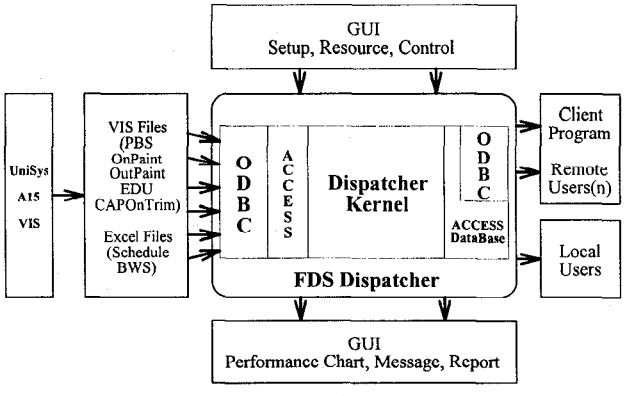

Fig. 3 EDS modular block diagram

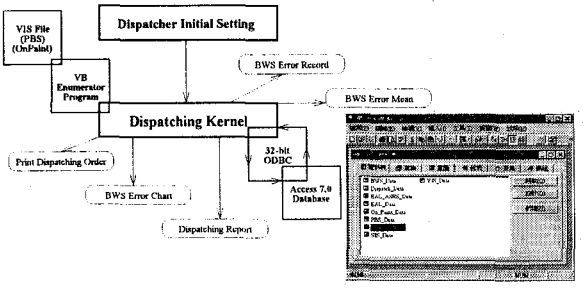

Fig. 4 EDS dispatcher kernel block diagram

Table 1 EDS priority table

Priority Number	Title	User Account	Modify Parameter	Operate EDS
1 to 500	Operators	Impossible	Impossible	Possible
501 to 800	Engineers Supervisors of Operator	Impossible	Possible	Possible
801 to 998	Manager	Possible	Possible	Possible
999	Administrator	Possible	Possible	Possible

V. DISPATCHING RULES FOR THE ASSEMBLY OF A NEW ENGINE

In the FAS under study, the techniques of concurrent control and synchronization are important since the production of this system is model-mixed production. The vehicle body and engine are built in different plants, and these two components are brought together to be assembled on the vehicle assembly line at a specified time. Due to the instability in the upstream process, the vehicle trim line sometimes takes less time than that in an engine assembly. Consequently, the corresponding vehicle body arrives at the Body Decking station without an engine to be assembled with. Instead of stopping the line and waiting for the missing engine, the vehicle body and corresponding engine will be assembled at a later time off-line. Such off-line assembly is inefficient and very costly. Therefore, it is necessary to develop efficient dispatching rules to improve system performance and ensure that the components will meet at the right assembly station at the desired time and be assembled together on-line.

To develop these dispatching rules, the processing time for different assembly stations must be measured first. Based on these measurements, several strategic points on the shop floor are selected to send out control messages. The messages to be sent out are stacked in a message queue. Based on proper dispatching rules and the balanced budget work standard, dispatching messages are sent to the engine assembly line to start the assembly of a new engine.

Four dispatching rules are developed as follows:

1. Dispatch of new engine block follows the Painted Body Storage (PBS) signal:

   The dispatching message queue is formed after receiving the PBS signal. The effectiveness of this rule depends on whether the quality in the paint shop is stable and acceptable. If the quality is stable and acceptable in the paint shop, then the engine will be assembled in time for the corresponding painted body.

2. Dispatch of new engine block follows the OnPaint Signal:

   The dispatching message queue is formed after receiving the OnPaint signal. This rule can improve the lead time (at least 2 hours) of assembling engines, but uncertain sequencing is a problem.

3. Dispatch of new engine block follows the OutPaint Signal:

   The dispatching message queue is formed after receiving the OutPaint signal. This rule can slightly improve lead time (about 15 minutes) of assembling engines, and sequencing is determined.

4. Dispatch of new engine block follows the In Line Vehicle Sequencing (ILVS) Signal:

   The dispatching message queue is formed after receiving the ILVS. This rule can properly improve lead time (about 40 minutes) of assembling engines, and the sequencing error is about 5 percents.

All of the dispatching rules will cope with the approach of balanced BWS for a small batch production to operate. Notice that no matter which dispatching rule is used, the command for EAL to assemble a new engine will only be issued once for each engine to avoid duplicated orders. Thus, the efficient shop floor control system [1] of FLHEP is desirable.

VI. PC-BASED IMPLEMENTATION

The proposed EDS is developed for the purposes of low cost, obvious benefits, easy implementation and continuous improvement. Therefore, an economical project for hardware only requires one PC, one industrial PC (IPC), three client printers (network support), hubs and cabling. The relationships among these devices are shown in "Fig. 5". The functions of these devices are described below:

1. PC: EDS dispatcher server.
2. IPC: EDS client. EAL shop floor operators can access the dispatched message from the EDS server automatically. In the future, it is possible to connect to the stamping machine.
3. Line printers (Network Support): It can be the printer client of EDS server to print the EAL dispatched message.
4. HUB: Support the network capability of proposed architecture.
5. LAN: EDS server must connect to the office LAN at EP office. EDS server can download the VIS files from the mainframe A15.

The operation systems of EDS are Microsoft NT server and workstation. The development tool of EDS is Microsoft VC++. The dispatched data and production information are stored in Microsoft Access database. The connectivity of application program and database is ODBC.

The initialization of EDS needs some information and settings to operate, and these data will be automatically started by the EDS. The setting of EDS will follow the data of previous run of EDS. Any change of settings on EDS will be recorded in database. The block diagram of initialization settings of EDS is shown in "Fig. 6".

The process parameters of EDS can be changed when the login user has the corresponding priority. "Fig. 7" shows the dialog of process parameters setting. The dispatching rules of EDS can be changed when the login user has the corresponding priority. "Fig. 8" shows the dialog of dispatching rules setting. The EDS server is represented by using split views. The users of engineers and managers can pause, resume and stop the EDS, and they can load the VIS, BWS, schedule information manually. If the error of downloading VIS information occurs, the EDS will inform the users to check and handle this error automatically. "Fig. 9" shows the report of current dispatched data. The user account management system provides the functions of adding, removing and modifying users' data. The security control of this system is based on the user's classification and user's priority.

"Fig. 10" shows an example of client program that indicates the engine's code, sequence number and corresponding vehicle body number. The performance measurement and comparison are shown in the lower view of "Fig. 9". Here the value of BWS error is used to indicate the performance. The lower the BWS error is, the more balanced the operator loading will be. In this view, the comparison is obvious that the absolute value of BWS error (left chart) is much smaller than that in the original PBS error (right chart).

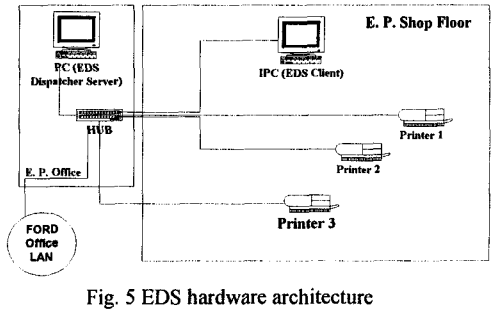

Fig. 5 EDS hardware architecture

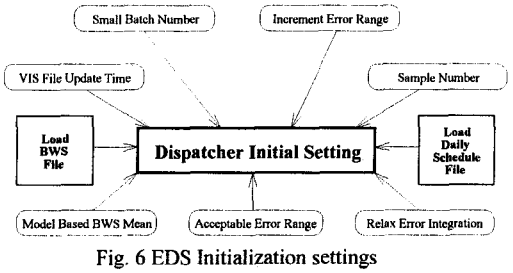

Fig. 6 EDS Initialization settings

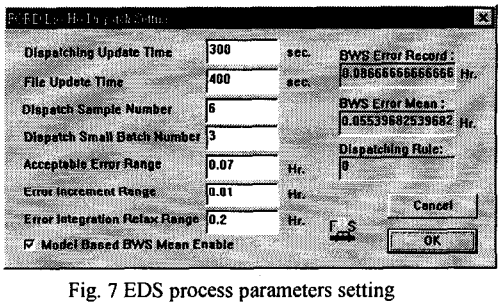

Fig. 7 EDS process parameters setting

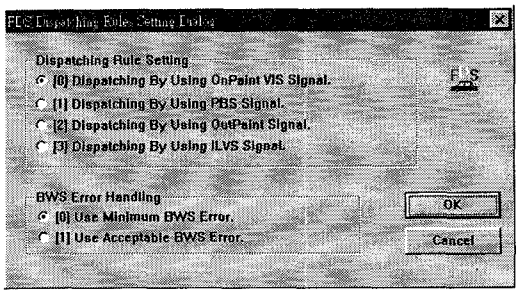

Fig. 8 EDS dispatching rules setting

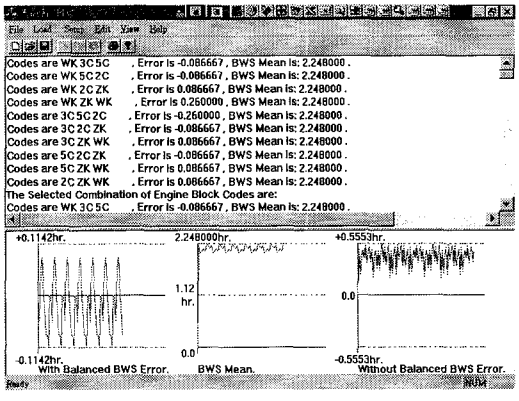

Fig. 9 EDS current dispatched vehicle data

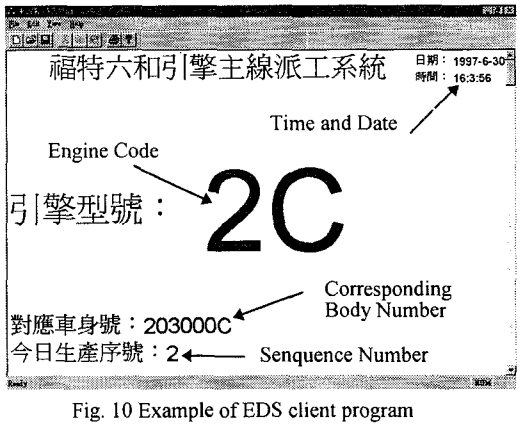

Fig. 10 Example of EDS client program

VII. CONCLUSIONS

The proposed EDS is a PC-based real-time dispatcher. This system aims to integrate distributed production information, eliminate the bottlenecks in the shop floor, balance the operator's loading, improve product's quality and integrate with other systems easily. In addition, the EDS have the properties of low cost, easy implementation, high availability, high reliability, client-server architecture, and security control. Therefore, an approach of balanced BWS for a small batch production is developed to cope with EDS to improve the system performance. Besides, the modular design and object-oriented programming make the EDS easy to debug, extend and modify.

Based on the EDS, the production information can be integrated, and the connections between the mainframe and the PC can be created. Notice that the modeling and analysis of the system are very important since the results of system analysis are the basis of EDS.

VIII. ACKNOWLEDGEMENT

This work was partially supported by the National Science Council, Taiwan, R.O.C., under Grant NSC85-2212-E-002-073, the University Research Program from Ford Motor Company, and Ford Lio-Ho Motor Co. (FLH).

IX. REFERENCES

[1] A. Bauer, R. Bowden, J. Browse, J. Duggan and G. Lyons, *Shop Floor Control System - from Design to Implementation* , London:Chapman & Hall, 1991.

[2] G. Coulouris, J. Dollimore and T. Kindberg, *Distributed Systems - Concepts and Design* , England:Addison-Wesley, 1994.

[3] *G2 Reference Manual Version 4.0* , Gensym, 1995.

[4] H.P. Huang and P.C. Chang, "Specification, Modeling and Control of a Flexible Manufacturing Cell," *Int. J. Production Research,* Vol. 30, No. 11, pp. 2515-2543, 1992.

[5] H.P. Huang and Y.H. Tseng, "Modeling and Graphic Simulator for Integrated Manufacturing Systems," *Intelligent Automation and Soft Computing*, Vol. 1, pp. 183-186, 1994.

[6] P. Huber, K. Jensen and R.M. Shapiro, "Hierarchies in Coloured Petri Nets," *Advanced in Petri Nets 1990, Lecture Notes in Computer Science*, Springer-Verlag, pp. 313-341, 1990.

[7] C.H. Kuo and H.P. Huang, K.C. Wei and S. Tang, "Dispatching and Simulation for Highly Model-Mixed Automotive Plants," *Proc. Int. Conference on Automation Technology*, Taiwan, Vol. 1, pp. 423-430, 1996.

[8] C.H. Kuo and H.P. Huang, "Colored Timed Petri Net Based Statistical Process Control and Fault Diagnosis to Flexible Manufacturing Systems," *IEEE Int. Conf. on Robotics and Automation*, Albuquerque, New Mexico, pp. 2741-2746, April, 1997.

[9] D.J. Kruglinski, *Inside Visual C++ - The Standard Reference for Programming with Microsoft Visual C++ Version 4* , Washington:Microsoft, 1996.

[10] S.S. Lu and H.P. Huang, "Modularization and Properties of Flexible Manufacturing Systems," in *Advances in Factories of the Future, CIM and Robotics* (edited by M. Cotsaftis and F. Vernadat), Amsterdam: Elsevier, pp. 289-298, 1993.

[11] R.C. Martin, *Designing Object-Oriented C ++ Applications* , Englewood Cliffs, N.J.:Prentice-Hall Inc., 1995.

[12] *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide* , Microsoft, 1994.

[13] H.P. Tang, J.K.L. Ho and H.Y. Tam, "Reconfigurable Data-Modules for the Reconfiguration of a Flexible Assembly System," *Intelligent Automation and Soft Computing,* Vol. 1, pp. 231-236, 1994.