An integrated knowledge based assembly control system for automobile manufacturing

B. Fredtag
Technische Universitat Munchen, Institut fur Informatik
Lehrstuhl Prof. Bayer
Orleansstr. 34, D-8000 Munchen 80, W.-Germany

B. Huber, W. Womann
BMW AG - Abt. FI-121
Postfach 40 02 40, D-8000 Munchen 40, W.-Germany

Abstract

In this paper a knowledge based assembly control system for automobile manufacturing is presented which has been implemented for the BMW manufacturing facilities in Munich. The system is fully integrated in the process control system. Its main purpose is to support the assembly supervisor in taking appropriate corrective actions in case of abnormal process conditions. The distinctive feature of the system is its capability to validate actions proposed by a rule based troubleshooting component through a one to one simulation model of the real manufacturing process. Thus effectiveness of actions at the first attempt is ensured and predictive process control is supported. The approach proposed in this paper can be generalized to a domain independent scheme for manufacturing process control.

1. Introduction

In this paper a knowledge based assembly control system for automobile manufacturing is presented which has been implemented for the BMW manufacturing facilities in Munich. The system is fully integrated in the process control system. Its main purpose is to support the assembly supervisor in taking appropriate corrective actions in case of abnormal process conditions.

Current process control is restricted to reactive diagnosis of abnormal or faulty conditions that occur a significant time before they are detected. In order to reach 100% reliability, today's more and more automated high throughput manufacturing processes require the ability to predict the need for corrective actions before a severe fault is actually encountered. As a second requirement the effectiveness of the corrective actions on the first attempt is inevitable [Cahill and Demers, 1988].

Diagnostic expert systems have a long tradition in AI starting from shallow modeling rule based systems, such as MYCIN [Hayes-Roth et al., 1982]. Due to the lack of an explicit domain model they suffer from the expertise cutoff at the edge [Koton, 1985], low robustness and their relative inflexibility. Deep modeling techniques, such as qualitative reasoning [Bobrow and Hayes, 1984], have been proposed to overcome the drawbacks of shallow systems. The applicability of deep reasoning is limited by it complexity and low efficiency [Koton, 1985]. A combination of heuristic and causal knowledge has been advocated by several authors [Fink, 1985, Torasso and Console, 1987]. [Simmons and Davis, 1987] propose a very attractive solution. Their Generate, Test and Debug (GTD) applies reasoning from a causal model in order to modify a hypothesis or select another one. This approach, however, requires an explicit model of the cause-effect relationships of process conditions and corrective actions, which again limits its applicability in the domain of manufacturing process control.

Recently, in [Cahill and Demers, 1988] a knowledge based manufacturing control system has been presented which is directly linked to the process control system. [Meyer, 1987] also proposes a general methodology for manufacturing control systems. Both proposals do neither offer a means for solution validation, nor do they allow for process state prediction.

This paper proposes a new architecture for manufacturing control systems in domains where explicit causal knowledge is not available or too expensive to acquire. The architecture is based on a combination of heuristic associative rules and an explicit factory model. The main features of our system are

- Actionable Solutions, i.e. proposal of directly executable corrective actions
- Effective Solutions, i.e. hypothetic test of corrective actions by a simulation model
- Predictive Fault Handling
- Process State Prediction based on real process data
- Integration, i.e. direct data link to the process control system
- Real Time Capability

The remaining parts of the paper are organized as follows. Section 2 sketches the problem domain, Section 3 gives a system overview, Sections 4 and 5 present the major system components in more detail. Solution validation is discussed in Section 6 while Section 7 contains some information on the implementation of the running system. Finally, in Section 8 conclusions are drawn and an outlook is given.

2. Problem Domain: Automobile Manufacturing

The automobile manufacturing process has as sub-processes the *Raw Assembly,* the *Coloring* and[1] the *Final Assembly (Figure 1).* Partially assembled cars pass the *Buffering Section* on their way from the Coloring Section to the Final Assembly. The Buffering Section can, on an abstract level, be described as a network of stations, e.g. a local storage, and connectors.

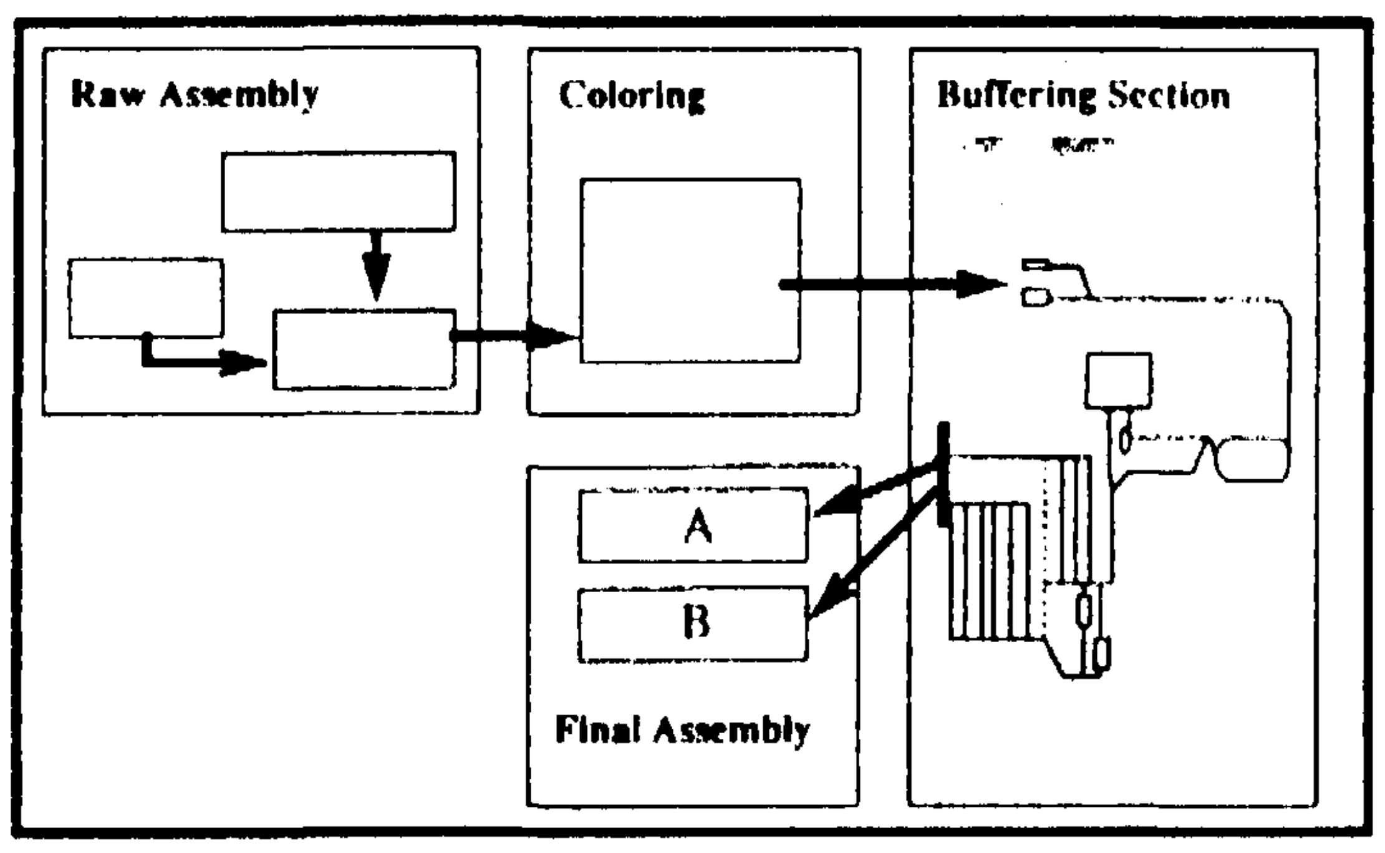

Figure 1 Automobile Manufacturing Process

The purpose of the Buffering Section is to compensate for irregularities of the preceding subprocesses and to arrange a new best mix. Constraints are imposed on the car sequence by the production planning, the requirements of the Final Assembly and the requirements from within the Buffering Section itself.

The car flow through the Buffering Section is normally controlled by the *Process Control System (PCS)* which is guided by a control database. This control scheme guarantees an optimal throughput as long as no faults occur. A human expert, the *Assembly Supervisor,* is responsible for the correct function of the Buffering Section, particularly in case of abnormal conditions.

The Assembly Supervisor continuously observes a number of process parameters, monitored by the PCS, e.g. the filling level of a station. He or she determines the system status by comparison of actual values and admissible ranges of each process parameter and identifies the appropriate corrective actions to be taken. Control actions are actually taken by changing the corresponding control records in the PCS database.

The most common faults result from abnormal input conditions, e.g. input flow is too slow. Therefore a fault diagnosis in terms of detecting the device that causes the fault does not make sense, because no repair is possible from within the Buffering Section. Fortunately, the time an abnormal input condition needs to cause a failure at the output to the final assembly allows to take corrective actions and prevent a severe fault. The assembly supervisor applies a troubleshooting strategy that in general relies on the fact that local abnormalities within the Buffering Section can be detected and compensated for a sufficient time before they propagate to the output to the Final Assembly. Small local irregularities are tolerated in order to achieve the overall objective [Cahill and Demers, 1988].

[1] By the word "car" we mean "car body without motor, gear etc."

The early detection of abnormal conditions and thus the predictive control of the Buffering Section by a human expert is limited by the great number of parameters and their rather complex interrelationships. The system described in this paper is designed to assist the Assembly Supervisor in controlling the Buffering Section.

3. System Overview

The overall architecture of our system is shown in *Figure 2.* The high modularity supports various combinations of components for different tasks.

The Buffering Section is structurally modeled by the *Factory Model,* which serves as both, the basic data structure for the entire assembly control system and a means for the real time simulation of the Buffering Section at the car flow level.

The overall control scheme of our assembly control system is an observe-reason-act cycle as frequently described in the AI literature. The control graph is shown in *Figure 3.*

Process parameters are directly taken from the process control system (PCS) via the data link and used to (re-) initialize the parameters of the Static Model. Subsequently either a process simulation can be started by activating the dynamic part of the Factory Model or the *Troubleshooting Component* can be invoked. The latter consists of three subcomponents, the *Status Determination,* the *Tactics Selection* and the *Solution Generation.* The Troubleshooting Component derives a qualitative status description from the process parameters, selects appropriate tactics and finally proposes a set of corrective actions to be taken in the actual process state as represented by the actual parameter values of the Static Model. The proposed actions can directly be executed either by allowing the Dynamic Model to update the parameters of the Static Model accordingly or by actually updating the control records of the PCS.

The user communicates with the system via a convenient mouse driven graphic user interface.

4. The Factory Model

The Factory Model consists of the *Static Model* and the *Dynamic Model.* The former represents the entities of the Buffering Section and their structural interrelationships, e.g. cars with their characteristics, stations and connectors, as well as the entities of the control system, e.g. control records and process parameters. It can be viewed as a materialization of the conceptual schema underlying the entire system and thus provides the basic data structure of every system component. The Static Model holds a snapshot image of the real manufacturing process at a given time.

The *Dynamic Model* models the time dependent behavior of the Buffering Section at the car flow level. It describes the movement of cars through the network of stations and connectors, and it models the process control system. A quantitative simulation of the manufacturing process is achieved by continuously updating the corresponding parameters of the static part of the Factory Model. The simulation is more than one order of magnitude faster than the real process so that it can be used for process state prediction as well as for the validation of corrective actions.

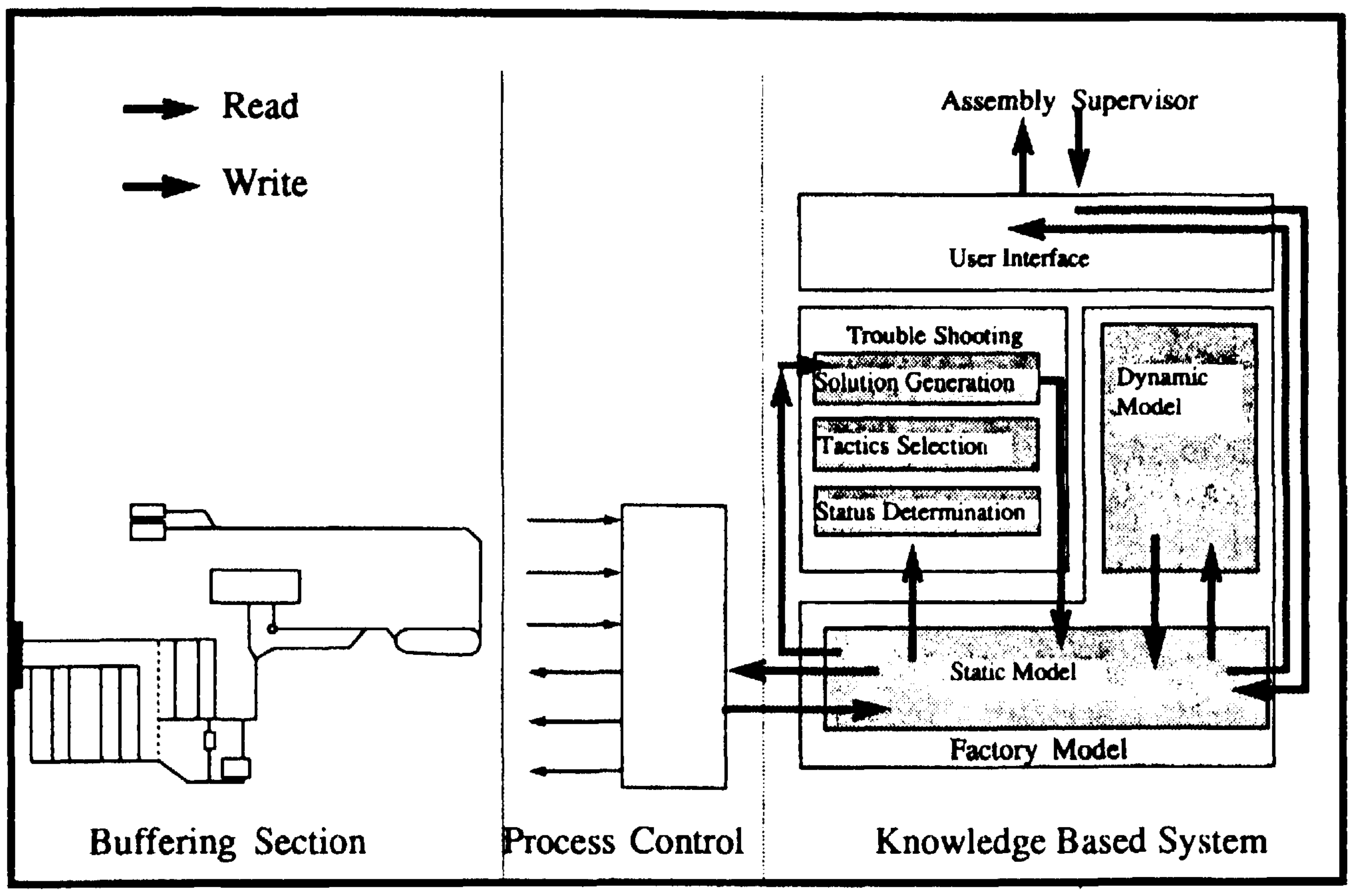

Figure 2 System Architecture

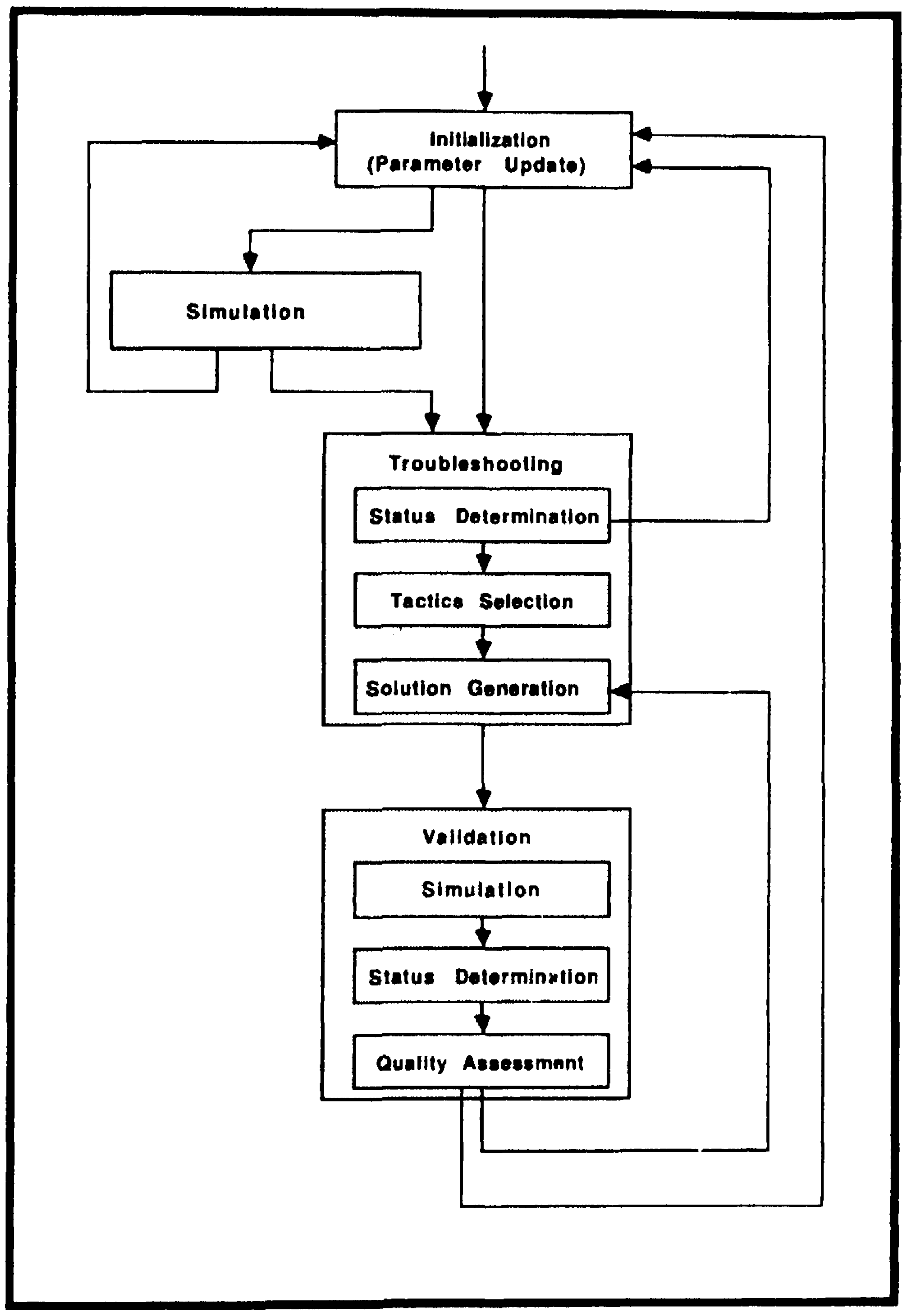

Figure 3 Control Graph

The Factory Model has a one to one modeling property of the manufacturing process as far as the car flow is concerned. However, there is no causal modeling of the manufacturing process itself. Only the phenomena are described.

5.1. Status Determination

The Status Determination component reads the quantitative process parameters from the static part of the Factory Model and converts them into qualitative state descriptions, e.g. "filling level of local storage is low". The status determination for a single process parameter is done on the basis of conversion tables which reflect the expert's heuristics. An overall process status is then derived by bottom-up aggregation of the single qualitative status descriptions.

The Status Determination component uses the overall process status to determine the next monitoring interval, i.e. the time interval until the next model parameter update by real process data, or time the simulation shall be run. In addition it decides whether to continue the troubleshooting and derive corrective actions or not.

5.2. Tactics Selection

The Tactics Selection component works on the qualitative status descriptions as determined by the Status Determination component. A tactic relates (a subset of) the qualitative status parameters to an abstract goal which is to be achieved by corrective actions. A goal itself defines a location within the Buffering Section and a guideline how to correct the abnormal conditions observed. Purpose of the tactic selection is to focus the search for corrective actions.

Normally, there are conflicting pairs of tactics. The priority of a tactic, i.e. the need for correcting a specific fault at a specific location, increases the closer the location of the fault is to the output of the Buffering Section (see Section 2). Due to the acyclic structure of the station graph representing the Buffering Section, a simple priority ordering of all possible tactics is induced. This fact is exploited for conflict resolution. Professional troubleshooters have a good command of experiential conflict resolution knowledge in "compiled form". They know which tactics fit together and which do not. The current version of our system makes use of this knowledge. The conflict resolution starts from the tactic with highest priority and excludes all conflicting tactics. This process is repeated until no conflicting pair is left.

5.3. Solution Generation

The Solution Generation component derives a set of executable corrective actions from the qualitative status parameters. It is guided by the selected tactics. The Solution Generation component uses the troubleshooting knowledge of the Assembly Supervisor to derive corresponding corrective actions for each tactic which contribute to reaching the goal and which are consistent with the actual process state, i.e. the qualitative process status descriptions.

A pair of tactics does not always define mutually disjoint or nonconflicting action sets. In the current version of our system the conflict resolution is again done by representing the "compiled" troubleshooting expertise of the Assembly Supervisor.

Finally, the derived corrective actions are refined to directly executable actions.

6. Solution Validation

The main feature of our system is its capability to test the generated solutions with the help of the Factory Model. As the control graph of *Figure 3* shows, the proposed corrective actions can be fed into the simulation. The current system version supports user driven solution validation. After running the simulation for the desired time the process status can again be derived. The relative quality of any two solutions, i.e. sets of corrective actions, can be determined by comparing the associated process status. The decision which solution to prefer is left to the user. This does, of course, not meet the objectives of an assisting system, because an informed search strategy is replaced by an experienced user. The general idea is therefore to add a more sophisticated validation component to the system which supports the direct automatic solution quality assessment. To this end the following upgrades currently being incorporated into our system are required:

- Add a facility to compare the quality of two solutions
- Generate more than one solution at a time

The overall process status as derived by bottom up aggregation of single status descriptions is a rather coarse measure of the solution quality and might also be inadequate in other problem domains. The decision which solution to prefer does not only depend on how well the main controlling objective, i.e. optimal output conditions, are achieved. Minor objectives, e.g. a good mix of cars in the local storage, which do not affect the overall process status but do contribute to the midterm functionality of the Buffering Section, should be taken into account. The upgraded version will therefore provide for a more explicit status determination and solution quality assessment.

Currently the tactics are selected according to a fixed priority order (cf. Section 5.2). This scheme leads to the selection of only one tactic in most cases. But as pointed out above possible minor improvements should also be considered, thus requiring a more detailed dynamic determination of the tactics selection priority.

Instead of generating one solution at a time, corrective actions derived by the Solution Generation component will be grouped into a number of different consistent sets. In order to prevent combinatoric explosion, heuristics derived from the tactics priorities are applied in order to define which candidate action set should be processed next.

7. Implementation

The assembly control system has been implemented in KEE/SIMKIT and is currently running on a SUN 3/260. The fundamental design decisions underlying the system are

- Create an explicit Factory Model
- Use the Static Model the common basic data structure
- Explicitly represent the expert's strategy
- Strictly differentiate between declarative and procedural parts of knowledge
- Provide for high modularity

The program design follows classic software engineering guidelines. A detailed problem analysis including the major part of the knowledge acquisition has been completed prior to the system design and implementation.

The Static Model as the basic data structure has been designed using a slight modification of the Entity Relationship Approach [Chen, 1985]. For the design of the remaining components a frame language and associative rules have been used.

HW	SUN 3/260
	24 Mbyte Main Memory
	560 Mbyte Hard Disc
OS	SUN-Unix 4.2, Release 3.4
Appl. SW	SUN-COMMON-LISP V. 2.1.1
	KEE V. 3.1
	SIMKIT V. 1.3

Tabic 1: System Environment

KEE/SIMKIT has been chosen as the implementation language of our system. The Static Model has been implemented in an object oriented fashion by KEE frames. The Dynamic Model has been modeled as a set of methods attached to each station object and coded in CommonLisp. The experiential knowledge of the troubleshooting component has been implemented by separate KEE rule sets. Each part of the reasoning component has its own manager object explicitly representing its control strategy. The system is linked to the process control system via network. The process parameters are transmitted by file transfer. The user interface makes intensive use of KEE's and SIMKIT's graphics and window system.

The good performance of the assembly control system allows for solution validation and process state prediction under real time conditions. Details about the system environment can be found in *Table 1.*

8. Conclusion

A knowledge based assembly control system based on a structural factory model has been presented. The factory model serves as the common basic data structure and as a means for quantitative real time simulation of the manufacturing process. It thus allows to validate the solutions proposed by the troubleshooting component. The main system features are predictive fault handling, directly executable and effective solutions, integration and real time capability.

The most important preconditions for the applicability of our approach are

- Experiential troubleshooting knowledge is predominant and available
- Quantitative simulation is appropriate and managable
- Implicit structural knowledge suffices for status determination and tactics selection
- Tactics are not totally interdependent
- The time interval between the detection of an error and its effect is long enough to derive and take corrective actions

Our approach is applicable in problem domains where explicit causal troubleshooting knowledge is not available or the knowledge acquisition for causal modeling is too expensive. This is typically true for process troubleshooting which relies on shallow experiential knowledge whereas all available explicit control knowledge has been incorporated into the process control system (PCS). Structural process descriptions can be extracted from the computer based design of the manufacturing process. The system can also be used as a means for the offline development of process control strategies. The technique presented in this paper is therefore particularly well suited to computer integrated manufacturing (CIM) environments

Different from [Cahill and Demers 1988] our system does not attempt to explicitly backtrace a problem to its cause condition. The knowledge required to localize a problem is rather implicitly included in the status conversion tables and the tactics selection heuristics. Explicit reasoning from structure would not only further improve the Solution Validation component, but would also in general increase the flexibility and robustness of our system Klein and Finin 1987, Koton, 1985]. To this end structural knowledge as represented by the Static Model should be exploited in a future version.

The system is currently undergoing in site tests in order to complete its knowledge base and to gain information needed for the extension to a fully operational system.

Acknowledgement

The authors wish to thank the manufacturing control experts V. Holz and G. Huber for their cooperation during knowledge acquisition. Many thanks also to Prof. R. Bayer for supporting this work.

References

[Bobrow and Hayes, 1984] D. Bobrow and P. Hayes (Eds.). Special issue on qualitative reasoning. Artificial Intelligence, (24), 1984

[Cahill and Demers, 1988] C. Cahill and A. Demers. Using Knowledge Technology to gain competitive advantage in manufacturing through predictive control. In: Proc. Intelligent Manufacturing, Benjamin/Cummings Publ. Inc., 1988

[Chen, 1985] P. Chen (Ed.). The use of the ER concept in knowledge representation. IEEE CSPress, North Holland, Washington, 1985

[Fink, 1985] P.K. Fink. Control and integration of diverse knowledge in a diagnostic expert system. In: Proc. 9th International Joint Conference on Artificial Intelligence, Los Angeles, CA, 1985, p. 426

[Hayes-Roth et al., 1982] F. Hayes-Roth, D. Waterman and D. Lenat (Eds.). Building Expert Systems. McGraw Hill, 1982

[Huber, 1988] B. Huber. Ein wissensbasiertes System zur Simulation und Diagnose von Storungen in einem Teilbereich der KFZ-Montage. Diploma Thesis. Technische Universitat Munchen, May 1988

[Klein and Finin, 1987] D. Klein and T. Finin. What s in a deep model. In: Proc. 10th International Joint Conference on Artificial Intelligence, Milan, Italy, August 1987, p. 559

[Koton, 1985] P.A. Koton. Empirical and model-based reasoning in expert systems. In: Proc. 9th International Joint Conference on Artificial Intelligence, Los Angeles, CA, 1985, p. 1308

[Meyer, 1987] W. Meyer. ESPRIT 932: Knowledge based real-time supervision in computer integrated manufacturing (CIM). In: Wissensbasierte Systeme, W. Brauer and W. Wahlster (Ed.), Proc. 2. Internationaler GI Kongress, Munchen, Oct. 1987, p. 401

[Simmons and Davis, 1987] R. Simmons and R. Davis. Generate, test and debug: Combining associational rules and causal models. In: Proc. 10th International Joint Conference on Artificial Intelligence, Milan, Italy, August 1987, p. 1071

[Torasso and Console, 1987] P. Torasso and L. Console. Causal reasoning in diagnostic expert systems. In: Applications of Artificial Intelligence V, John F. Gilmore, Proc. SPIE 786, 1987, p. 598