Communication Protocols and Failure Semantics in a Material-Driven Intelligent Manufacturing Systems

ANDERS ADLEMO[1] and SVEN-ARNE ANDRÉASSON[2]

[1]Department of Computer Engineering and [2]Department of Computing Science
Chalmers University of Technology, S - 412 96 Göteborg, SWEDEN.

The paper describes a series of protocols that increases the ability to automatically cope with failures in a computerized manufacturing system. In order to achieve better automatic failure solutions the behavior of a subsystem must be more precisely defined regarding the possible failure modes. This is called to give the subsystem a stronger failure semantics. Then, regarding the systems specific failure semantics, an algorithm to cope with the corresponding failures can be given. In this paper material-driven systems, so called Push systems, are considered. To get more precise failure resolution protocols with one or two acknowledgment messages from the receiver are defined. A protocol with acknowledgment messages from an intelligent material transportation network is also described.

Keywords: Protocols, Failure Semantics, Fault Tolerance, Intelligent Manufacturing Systems, Reliability, Push Systems.

1 Introduction

To improve manufacturing flexibility in modern manufacturing facilities, more and more sophisticated integrated computing systems are used. By doing so the manufacturing facilities become more vulnerable to failures since the systems become more complex. In order to make the systems reliable, even in the presence of failures, fault tolerance must be introduced into the systems (Adlemo and Andréasson, 1992; Adlemo *et al.*, 1993a; Adlemo and Andréasson, 1993b; Adlemo *et al.*, 1993c; Adlemo and Andréasson, 1993d; Chintamaneni *et al.*, 1988; Elkhattabi *et al.*, 1992 Graham *et al.*, 1992; Harhalakis *et al.*, 1992; Holloway and Krogh, 1990; Holloway and Krogh, 1992; Lee and Tsai, 1992; Shieh *et al.*, 1990). By this is meant that the systems must be able to cope with failures in an automated manner.

In order to define the algorithms for the fault tolerance, there must be a definition of the expected failure behavior of the subsystems. This definition is called the failure semantics of a subsystem. We have described basic protocols for a simple Client-Server System, also called a Pull System, to achieve a more detailed failure semantics (Adlemo and Andréasson, 1994). Such a Client-Server System protocol can for instance be the protocol between two manufacturing cells or the protocol between manufacturing sections. In this paper protocols for material-driven systems, also known as Push Systems, are described.

Much of the efforts to improve the availability of modern production systems have been spent on obtaining effective, fault tolerant computer programs. For example, statistics show that as much as 90% of the source code in an FMS work-cell deals with error checking and error handling (Lee, 1989). One of the reasons for this high amount of error handling code is that the system designer has not clearly defined the failure semantics, or failure behavior, for the production system from the very beginning. Instead, more fault tolerance than otherwise would be necessary is introduced in the source code.

1.1 Failure Semantics in Computerized Systems

In a fault tolerant system it is necessary to extend the specification of its subsystems to include not only familiar failure free semantics, but also possible failure behavior, or failure semantics (Cristian, 1991). A system is said to have *strong* failure semantics when the types of possible faults are easy to detect and distinguish from each other. A system is said to have *weak* failure semantics when the possible types of faults are hard to detect and distinguish from each other. In general, the stronger the failure semantics specified for a

system, the more expensive it is to build a system that implements the semantics. However, a system with stronger failure semantics leads to simpler and more efficient fault tolerant mechanisms for higher system levels. The table below illustrates the advantages and disadvantages of strong and weak failure semantics.

Failure semantics	Advantages	Disadvantages
Strong	Cheap error handling	Expensive semantics implementation
Weak	Cheap semantics implementation	Expensive error handling

In (Cristian, 1991) failure semantics for the time aspect of a client/server response is defined. This leads to four types of failure semantics: crash failure semantics, omission failure semantics, timing failure semantics and arbitrary failure semantics.

2 A Push Manufacturing System

The classifications of failure semantics made in (Cristian, 1991) and (Adlemo and Andréasson, 1994) are so called *client-server systems* or *pull systems* (Sartori, 1988). This means that a client makes a request to a server and waits for an answer. The behavior of, and response from, the server under certain faulty situations defines the failure semantics exhibited by the server, that is, in turn, perceived by the client. This paper, however, addresses manufacturing systems that are pure push systems. A simple manufacturing system that is built as a pure push system with manufacturing cells and AGVs is illustrated in figure 1.

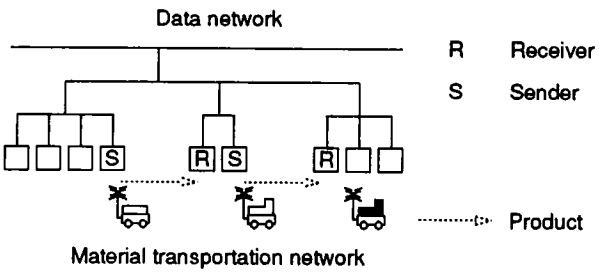


Figure 1: A manufacturing system (push system).

2.1 Simple Push System

This section presents protocols for a simple Material-Driven Push System where a sender S sends a product P to a receiver R using a material transportation network MTN after each completed task at the sender S. The protocol descriptions start with a very simple system without acknowledgment messages, then continues with a system with one acknowledgment message per request and a system with two acknowledgment messages per request.

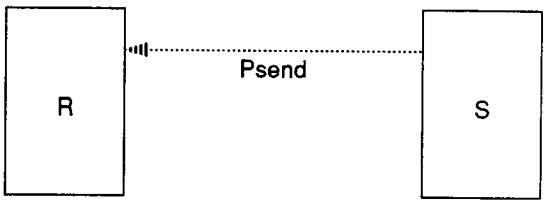
The failure effects and failure semantics is given from the senders point of view. The reason for defining a failure semantics is to give the sender a foundation for how to cope with a given failure.

The following failures might occur:

- there might be a material transportation network failure,
- the product might be sent to a receiver with a full input buffer,
- the product might be sent to a non-functioning receiver,
- the receiver might fail during the continuing task.

2.1.1 System without acknowledgment messages

When there are no acknowledgment messages, it is assumed that the product will arrive to the correct receiver and that there are no full buffers in the system. The system protocol is described in figure 2.



R	Receiver
S	Sender
Psend	S sends the product P

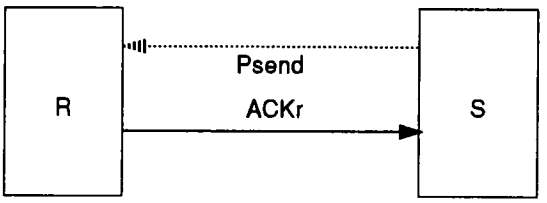
Figure 2: Material-Driven interaction without acknowledgments

Consequently, it is impossible for the sender to take appropriate actions due to a failure when using this simple protocol since the sender has no way to find out what happens.

2.1.2 System with one acknowledgment message per request

To make it possible to allow for an error recovery, an acknowledgment message is added to the protocol. When the receiver R gets the product P, it sends an acknowledgment

message, ACKr, to the sender S. This protocol is described in figure 3.



R	Receiver
S	Sender
Psend	S sends the product P
ACKr	Reception acknowledgment

Figure 3: Material-Driven interaction with one acknowledgment message.

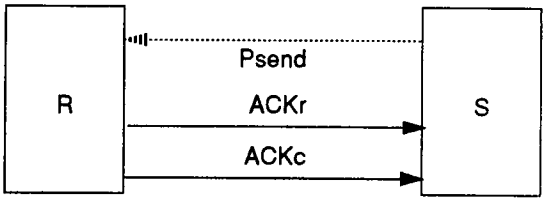
If there is no ACKr message there is a failure. There might be a material transportation network failure, a data network failure or the receiver is not working. The different failure behavior, as seen from the sender, is given in the following table.

Psend	ACKr
	no ACKr

2.1.3 System with two acknowledgment messages per task

To further improve the efficiency of error recovery another acknowledgment message is added to the protocol. When the corresponding product is treated and shipped to the next server, another acknowledgment message, ACKc, is returned to the sender S. In this way the original sending server S can check its following server R. This scenario is described in figure 4.

The arrival of an ACKr message presents a possibility to exclude material network failures. If there is no ACKc message after the arrival of the corresponding ACKr message there might be a server failure. However, there might occur a data network failure during the second acknowledgment message.



R	Receiver
S	Sender
Psend	S sends the product P
ACKr	Reception acknowledgment
ACKc	Completion acknowledgment

Figure 4: Material-Driven interaction with two acknowledgment messages.

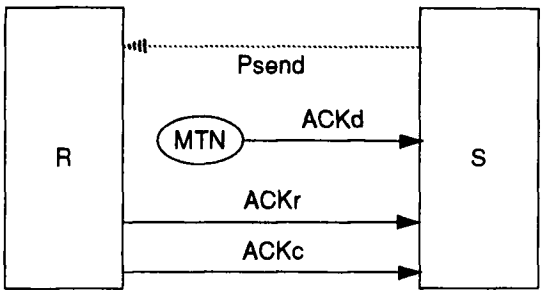
The following table gives the different results that can be obtained from one single product sending, Psend.

Psend	ACKr	ACKc
		no ACKc
	no ACKr	ACKc
		no ACKc

When dealing with different types of failures (one per each row in the table except the first row), all or a part of these can be taken care of. The types of failures that are taken care of gives the failure semantics expected from the receiver R. Other failures will lead to a system crash, since they are not taken care of.

2.2 Push System with "intelligent" material transport system

In order to give a more detailed failure semantics for the system as seen by the sender, an acknowledgment message from the material transport network is introduced. This implies that there is a material transport network that is connected to the data network (at certain times or always). A message ACKd is sent to the sender when the product is delivered to the receiver. In this way there is a possibility to distinguish between material transportation network failure and receiving server failure (figure 5).



R	Receiver
S	Sender
MTN	Material Transportation Network
Psend	S sends the product P
ACKd	Delivery acknowledgment
ACKr	Reception acknowledgment
ACKc	Completion acknowledgment

Figure 5: Material-Driven interaction with material transport network acknowledgment messages.

The following table gives the different results that can be obtained from one single product sending, Psend.

Psend	ACKd	ACKr	ACKc	1
			no ACKc	2.
		no ACKr	ACKc	3.
			no ACKc	4.
	no ACKd	ACKr	ACKc	5.
			no ACKc	6.
		no ACKr	ACKc	7.
			no ACKc	8.

The different failure modes are described below:

1. This case describes the normal behaviour when everything is working as it should.

2. In this case there is both a delivery acknowledgement from the material transport network and a reception acknowledgement from the receiver. But there is no completion acknowledgement from the receiver. This case implies a server failure by the receiver. However, there might have occurred a data network failure instead.

3. The product is delivered but there is no reception acknowledgement from the receiver. However there is later a completion acknowledgement from the receiver. This implies that there was a shorter failure in the data network.

4. The product is delivered but there is no response from the receiver. Either the receiver is not working, or the data network has had a longer breakdown.

5. There is no delivery acknowledgement from the material transportation network but instead a reception acknowledgement from the receiver. In this case there has been communication problems with the data network from the material transportation network.

6. There is no delivery acknowledgement from the material transportation network but instead a reception acknowledgement from the receiver. Then no completion acknowledgement is received from the receiver. This indicates that there is problems in the data network. However, there might be a combination of a single data network failure and receiver service failure.

7. There is no delivery acknowledgement from the material transportation network and no reception acknowledgement from the receiver. However, later a completion acknowledgement is received from the receiver. This indicates that there has been a longer failure in the data network.

8. No response at all. Either a material transportation network failure or a data network breakdown.

2.2.1 Using a negative acknowledgment message from the material transportation network

To get a possibility to distinguish material transportation network failure from the case that the buffer is full at the receiver, a Buffer Full at Receiver Message can be introduced.

3 Conclusions

In this paper was described different protocols that give different failure semantics, or failure behavior, for computerized manufacturing systems. By using one or two acknowledgment messages from the sender and zero or one acknowledgment message from the material transport network the failure semantics can be given more precisely, which will make it easier to take measures to cope with failures. In this way the systems can be made more reliable.

The protocols were given for a material-driven, system, a so called push system. For one of the protocols the failure modes and their causes were given.

4 References

Adlemo, A., and S.-A. Andréasson (1992). Models for fault tolerance in manufacturing systems. *Journal of Intelligent Manufacturing*, **3**, pp. 1-10.

Adlemo, A., S.-A. Andréasson, and M. I. Johansson (1993a). Fault tolerance strategies in an existing FMS installation. *Control Engineering Practice*, **1**, pp. 127-134.

Adlemo, A., and S.-A. Andréasson (1993b). Fault tolerance in partitioned manufacturing networks. *Journal of Systems Integration*, **3**, pp. 63-84.

Adlemo, A., S.-A. Andréasson, and M. I. Johansson (1993c). Information Accessibility and Reliability Improvement in an Automated Kitting System. *Proceedings of the 12th World Congress of the International Federation of Automatic Control, IFAC'93*, Sydney, Australia, vol. 3, pp. 99 - 106.

Adlemo, A., and S.-A. Andréasson (1993d). Failure semantics in intelligent manufacturing systems. *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*. Atlanta, U.S.A., vol. 2, pp. 166 - 173.

Adlemo, A., and S.-A. Andréasson (1994). Communication Protocols and Failure Semantics in Intelligent Manufacturing Systems. Will appear in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. San Diego, U.S.A. (May 1994).

Chintamaneni, P. R., P. Jalote, Y.-B. Shieh, and S. K. Tripathi (1988). On fault tolerance in manufacturing systems. *IEEE Network*, **2**, pp. 32-39.

Cristian, F. (1991). Understanding Fault Tolerant Distributed Systems. *Communications of the ACM*, vol. 34, no. 2, pp. 56 - 78.

Elkhattabi, S., D. Corbee and, J. C. Gentina (1992). Integration of Dependability in the Conception of FMS. *Preprints of the 7th IFAC/IFIP/IFORS/IMACS/ISPE Symposium on Information Control Problems in Manufacturing Technology, INCOM'92*, Toronto, Canada, pp. 249 - 254

Graham, J. H., J. Guan, and S. M. Alexander (1992). A hybrid CIM diagnosis system with learning capabilities. *Proceedings of the 3rd International Conference on Computer Integrated Manufacturing*. Troy, U.S.A. pp. 308-314.

Harhalakis, G., C. P. Lin, R. Nagi, and J. M. Proth (1992). Hierarchical decision making in computer integrated manufacturing systems. *Proceedings of the 3rd International Conference on Computer Integrated Manufacturing*. Troy, U.S.A. pp. 15-24.

Holloway, L. E., and B. H. Krogh (1990). Fault detection and diagnosis in manufacturing systems: a behavioral model approach. *Proceedings of the 2nd International Conference on Computer Integrated Manufacturing*. Troy, U.S.A. pp. 252-259.

Holloway, L. E., and B. H. Krogh (1992). On-line fault detection via trajectory encoding. *Preprints of the 7th IFAC/IFIP/IFORS/IMACS/ISPE Symposium on Information Control Problems in Manufacturing Technology, INCOM'92*. Toronto, Canada. pp. 643- 648.

Lee, J., and J. Tsai (1992). Fault detection in intelligent material handling. *Proceedings of the 3rd International Conference on Computer Integrated Manufacturing*. Troy, U.S.A. pp. 298-307.

Lee, M. (1989). *Intelligent Robotics*. Halsted, New York, USA.

Sartori, L. G. (1988). *Manufacturing Information Systems*. Addison-Wesley Publishers Ltd., Wokingham, UK.

Shieh, Y.-B., D. Ghosal, P. R. Chintamaneni and S. K. Tripathi (1990). Modeling of Hierarchical Distributed Systems with Fault Tolerance. *IEEE Transactions on Software Engineering*, vol. 16, no. 4, pp. 444 - 457.