Problems Facing Group Membership Specifications for X-by-Wire Protocols

Elizabeth Latronico
Electrical and Computer Engineering Department
Carnegie Mellon University
Pittsburgh, PA, USA
beth@cmu.edu

Abstract

A guarantee is only as good as its assumptions. Three group membership areas defy firm, practical guarantees: transient faults, reintegration, and large groups. We show why these areas are problematic, discuss current approaches, and illustrate alternative approaches that may lead to a more dependable system. Ultimately, designs should be tested against a comprehensive fault model, and a probabilistic analysis cannot be avoided.

1. Introduction

A group communication service can play a number of vital roles in the dependability of a network protocol. A group communication service is often the cornerstone of a Byzantine fault tolerance policy. Nodes may need to agree on members of the group in order to accomplish a certain function, such as clock synchronization. Nodes may need to agree on the contents of a message. There are numerous useful guarantees a group communication service can provide [Chockler01]. Is there a practical, sufficient set of these guarantees for automotive X-by-Wire protocols? First, a designer must examine the requirements.

Cost and high dependability are two factors that shape the requirements of automotive X-by-Wire systems. X-by-Wire systems are "...safety-related fault-tolerant electronic systems in vehicles...", where the X stands for a safety-related application (such as braking or steering), typically to be implemented without mechanical backup [Krug97]. 'Cost components' for an X-by-Wire protocol include nodes used for redundancy and items that impact bandwidth, such as overhead per message and the number of extra messages sent to ensure group properties. The acceptable failure rate for critical system functions will be on the order of 10^{-9} failures/hour [PALBUS01].

Node diversity also significantly impacts the requirements. In contrast with typical client/server systems, functionality often cannot be migrated to other nodes as many nodes require physical connections with sensors and actuators to perform a task. Further, nodes have different needs. Some may be safety-critical, while others non-critical. Also, a node as a data consumer may require only a small subset of messages, while nodes participating in clock synchronization will need information from the entire group.

How can a designer know if all requirements are satisfied? Any set of guarantees comes with a set of assumptions. Section 2 discusses the guarantee design paradigm. Section 3 presents three fundamental problem areas. Section 4 presents a point solution to a portion of these areas, and Section 5 summarizes conclusions.

2. The Guarantee Design Paradigm

There are two approaches to dependable systems design. One approach is probabilistic - a design is created and analysis techniques show that the probability of failure is acceptable. To reason about these systems, accurate probability estimates are required. Unfortunately, probabilities are difficult to estimate with the degree of precision necessary for safety-critical systems.

The second approach is to define a set of guarantees for the system, and relegate probabilities to the assumptions. Guarantees are proven to hold as long as a specified set of assumptions hold. While easier to reason about, a designer must now examine the probability that the assumptions will hold. Common assumptions include, but are not limited to: limits on the number of faults that can occur within a certain number of rounds, an assumption that the network is non-partitionable, limits on when a node may reintegrate, and the assumption that a majority of good nodes exists. Clearly, these assumptions are not expected to hold for every imaginable fault scenario. The designer must show that the probability of pernicious faults occurring is 'reasonably low', and the dependability of the system is still acceptable.

Unfortunately, we are back to the same problem that a probabilistic approach has - 'reasonably low' for safety-critical systems means an acceptable failure rate on the order of 10^{-9} failures/hour. A comprehensive fault model for fail-operational safety-critical systems will never meet all the assumptions. Stated otherwise, the dependability requirement is so high that odd or rarely occurring faults can't be ignored. One additional complication is that safety analysis may depend on criticality. Faults or fault combinations with severe consequences or low controllability must be handled or ameliorated regardless of the probability. An example is the requirement of no single point of failure, regardless of the probability of that failure [MISRA95].

3. Fundamental Research Problems

Three problems frustrate our ability to create efficient, reliable services with firm guarantees. Since dependability is a system property, these problems can be attacked at different levels of the system. One approach is to have the protocol assume total responsibility. This may be inefficient, impacting cost. A second approach is to delegate responsibility to the application. This affects the ability to assure adequate dependability of the protocol. Another alternative is to develop partial, area specific solutions. The difficulty here is creating a beneficial solution that does not not violate other guarantees . Essentially, each approach involves trade-offs among different dependability attributes.

Because each approach presents trade-offs, an attempt to make a firm guarantee may not result in the most dependable system possible. In particular, if the probability of a best-effort design failing is less than the probability of a guarantee's assumptions failing, the best-effort design would be superior. Also, providing a firm guarantee for one attribute might be too inefficient with respect to other attributes. Here, the probability of system failure may be greater with the guarantee than without.

Our intent is not to argue for or against guarantees. Rather, we want to show that all design choices involving guarantees should be clearly stated. All design assumptions should be tested with a comprehensive fault model, and any rules component designers must follow should be clearly stated and practical. Next we examine the three problem areas in detail, describe the current approach, and examine potential alternatives.

3.1. Transient Faults

Transient faults (for example, noise on the network corrupting a message) are difficult to deal with as they cause a system to be inherently asynchronous. There could be infinite delay between the sending and reception of a message. Transient faults violate the Byzantine Generals assumption of reliable messengers. Waiting for an infinite time is not an option for practical systems, so timeouts can be used to discriminate a transient fault from a permanent fault by selecting a appropriate Δt. Faults with duration less than Δt are considered transient, while others are considered permanent. However, there is no value of Δt that is guaranteed to separate all transient faults from all permanent faults, since transient faults occur in actual time (which can be thought of as having a real number domain).

A common way to make guarantees about a system with transient faults is to set Δt equal to zero, thus treating all faults as permanent faults. Through this approach, protocols accept all responsibility for handling transient faults. A main advantage is that the system is guaranteed to handle all transient faults that conform to the assumptions.

A disadvantage of this approach is decreased availability. A transient fault will be attributed to one or more nodes. These nodes will lose membership, which is particularly concerning if a node provides safety-critical functionality. It is generally thought that transient errors outnumber permanent errors by a factor of 10. This does not mean that treating transient faults as permanent will cause the system to fail, but it does indicate a major inefficiency. In addition, since all faults are considered 'permanent', how can a permanently faulty node be reintegrated?

Any alternative approach must balance the availability gain from better detection against the risk of misidentification. Alternatives include carefully selecting a different Δt, or using additional data to diagnose faults.

3.2. Reintegration

A generic node reintegration policy is problematic to define at the protocol layer. Aside from the permanent fault recovery problem, the issue of internal state complicates reintegration. The protocol itself cannot know if the application uses only explicit external state, or if it also uses internal state. Even if internal state is forbidden, the problem of incidental state remains. Namely, Poledna notes three inescapable sources of non-determinism: inaccuracy of real-world abstraction, impossibility of exact agreement, and intention and missing coordination [Poledna94].

Current X-by-Wire protocols define the minimum waiting time, but the remainder of the reintegration policy is left to the application. This choice makes sense as the protocol cannot know for sure if the application has hidden state, and imposing design restrictions may prove unacceptable. Also, there are certain fault combinations that are not tolerated. The group communication service must achieve consensus before a new node may reintegrate. After a single fault, consensus can always be achieved within two rounds [Pfeifer00]. With multiple faults in this time frame, however, consensus is not guaranteed to be achieved.

Unfortunately, if no restrictions are placed on the state an application can have, a dependability assessment at the protocol level must make the pessimistic assumption that reintegration is not possible. This is a major limitation, since a large number of redundant components would be needed in order to tolerate component failures.

One way to tackle this problem is that the protocol could stipulate rules the application must follow so that all state is externally viewable. Rules should include the types and times data is made available to other applications. With periodic networks, most state is broadcast already and available at predictable times [Kopetz03]. This does not eliminate the possibility of incidental internal state, and there is no guarantee a design rule will be followed correctly. The risk of violating design rules must be balanced against the cost of redundancy required otherwise.

3.3. Fault Tolerance Scalability

In a single group design, all nodes belong to one group. A single group design is straightforward and eliminates the overhead of coordinating multiple groups. Services such as clock synchronization would be difficult to implement without a single group, as nodes need to agree on the subset of nodes used as time providers.

However, fault tolerance does not scale neatly with group size. For permanent faults, the node failure rate remains constant as long as permanent processor and link faults are independent. However, for transient faults, the node failure rate will **increase** as the number of nodes increases. For example, consider noise on the bus, represented by a Bit Error Rate. A system with ten messages of the same size per fixed round length will encounter fewer faults than a system with hundreds of messages of the same size per fixed round length. Even if bandwidth is the same, this remains true as a less loaded network can use 'larger' bit sizes, decreasing the probability of noise corruption.

Further, nodes that are data consumers do not benefit from larger groups (except for replicated nodes for fault-tolerance purposes). An algorithm that requires a certain number or majority of nodes 'working' will likely scale because adding nodes to the system will increase the probability of having enough 'working' nodes. However, this does not hold for consumers interested in particular nodes. If a receiving node needs data from four other nodes, it will still need data from those same four nodes regardless of the group size. Dependability strictly decreases with group size, because of greater exposure to transient faults.

Since smaller groups are smaller targets, we would like keep group size small without affecting guarantees. However, splitting the single group into multiple groups and then keeping these groups fully integrated with each other incurs overhead with no benefit. Strict subsets are usually not possible due to node data requirements. Transitive closure on nodes interested in another node's messages leads to a single group. The challenge remains to exploit multiple groupings without breaking single group guarantees.

4. Summary

Essentially, probabilities cannot be eliminated from a design, whether they occur in the design itself or in the assumptions. A comprehensive fault model is needed to determine if a design is adequate. We plan to take a population of guarantees from a Group Communication Specification survey by by Chockler, Keidar, and Vitenberg [Chockler01] and test these properties with a comprehensive automotive fault model. While precise probabilities are unlikely to be obtained, sensitivity analysis should provide valuable insight into design limits.

Point solutions look like a promising alternative to abandoning the guarantee approach entirely. We have presented one grouping scheme in [Latronico03] that improves dependability without altering the underlying group communication specification. By creating smaller virtual groups according to message period, this scheme lessens the effect of a transient fault. A node that sends messages in multiple virtual groups will only lose membership in one of these groups if a single message is corrupted. This scheme might also be used for fault classification - a node that is able to send m of n messages probably suffered a transient fault, compared to a node that successfully sends 0 of n messages.

5. Conclusions

All designs are inherently probabilistic. Realizing this helps us critically examine design choices for X-by-Wire protocols in three fundamentally problematic areas. Transient faults are currently treated as permanent faults for nearly perfect coverage, but reduced availability. Reintegration is left to the application, with no prohibition on internal state, but reintegration is not guaranteed to be possible. A single group system system is easier to construct and analyze, but fault tolerance may not scale.

Designs should be evaluated with a comprehensive fault model. If a design falls short of the required dependability, point solutions provide an attractive alternative to abandoning guarantees entirely. A guarantee is only as good as its assumptions; evaluating these assumptions is key to developing practical, dependable systems.

Acknowledgments

This work is supported in part by the General Motors Collaborative Research Laboratory at Carnegie Mellon University and by the United States Department of Defense (NDSEG/ONR). Thanks to Phil Koopman and Priya Narasimhan for their valuable input.

References

[Chockler01] Chockler, G., Keidar, I., and Vitenberg, R., Group Communication Specifications: A Comprehensive Survey, *ACM Computing Surveys*, Vol. 33, Number 4, Dec. 2001, pp. 427-469

[Kopetz03] Kopetz, H., Bauer, G., The Time-Triggered Architecture, *Proceedings of the IEEE*, Jan 2003.

[Krug97] Krug, M. et. al., Towards an Architecture for Safety Related Fault Tolerant Systems in Vehicles, *ESREL '97*, June 1997.

[Latronico03] Latronico, E. and Koopman, P., A Period-Based Group Membership Strategy For Nodes of TDMA Networks, *FeT 2003*, To Appear, July 2003.

[PALBUS01] Definitions, Version 2.0, PALBUS Task 10.1, SP Swedish National Testing and Research Institute, Apr. 2001.

[Pfeifer00] Pfeifer, H., Formal Verification of the TTP/C Group Membership Algorithm, *Proceedings of FORTE XII/PSTV X*X, Oct. 2000, pp. 3-18.

[Poledna94] Poledna, S., Replica Determinism in Fault-Tolerant Real-Time Systems, Dissertation, Technische Universität Wien, Institut für Technische Informatik, Apr. 1994.

[MISRA95] Integrity, MISRA Report 2, The Motor Industry Software Reliability Association, Feb. 1995.