

Membership Middleware for Dependable and Cost-Effective X-by-Wire Systems

Kohei Sakurai

Hitachi Europe GmbH

Masahiro Matsubara and Masatoshi Hoshino

Hitachi, Ltd.

Copyright © 2008 SAE International

ABSTRACT

Balancing between dependability and cost-effectiveness is essential to promote X-by-Wire systems in the next decade. To achieve this goal, we have so far proposed a network centric architecture based on a concept of autonomous decentralized systems, where if one node fails, the remaining normal nodes autonomously execute a backup control to maintain the system's functionality, as well as a membership middleware indispensable to this architecture to ensure the consistency of the node status information among all nodes. In this work, we implemented membership middleware on a hardware and software platform equivalent to one assumed to be used in actual X-by-Wire systems. This paper describes the implementation details and performance evaluation result, and shows that membership middleware and a real-time critical application can coexist within one microcontroller.

INTRODUCTION

Advances in automotive electronics will make more sophisticated vehicle control systems possible. Among these systems, X-by-Wire systems, where driving, steering and braking are electronically controlled, are expected to enhance vehicle driving performance and safety. Because acceleration, steering and braking control have great influence on vehicle safety, X-by-Wire systems are required to be dependable. On the other hand, the cost for dependability cannot be too high to promote X-by-Wire systems in the future. Therefore, our goal is to balance these competing demands: dependability and cost-effectiveness.

To achieve this goal, we have so far proposed a network centric architecture based on a concept of autonomous decentralized systems [1][2]. The electronic platform for X-by-Wire systems will steadily shift toward a network centric architecture and this trend is also shown in [3]. In our proposed architecture, all nodes related to control, including smart sensors and actuators, share various data through the network, and each node autonomously

gets or broadcasts the necessary data from or to the network. If a certain node stops operating because of some fault, the remaining normal nodes autonomously execute a backup control to maintain at least minimal necessary system functionality using the shared data. Therefore the proposed architecture can tolerate the existence of the failed node and does not require high cost fail-operational nodes with triple or more redundant architecture.

However, because there is no master node for communication monitoring in this architecture, in order to execute any autonomous backup control, we have a technical challenge of how to accurately identify the failed node and ensure consistency of the information about which node has failed among all remaining normal nodes. Our approach to solve this challenge is a membership middleware which monitors node status by voting on the node status data exchanged among all nodes [4].

The middleware approach to implement the membership functionality, unlike the implementation with communication controller hardware in TTP (Time-Triggered Protocol), has also been proposed recently in sources such as [5]. However, as far as we know, there are no related previous works which implement membership middleware on a hardware and software platform equivalent to one assumed to be used in actual X-by-Wire systems to evaluate the required overhead such as CPU computation power, and whether the membership middleware and applications which have to satisfy severe real-time requirements can coexist within one microcontroller. Therefore, in this work, we have implemented the membership middleware on an embedded microcontroller with OSEK OS and a three-phase brushless motor control application which is a typical real-time critical application for X-by-Wire systems. We then developed a prototype FlexRay-based brake-by-wire system to evaluate the above-mentioned issues.

Following a review of our proposed network centric architecture and membership middleware, this paper will

describe the implementation details, including a technique to synchronize FlexRay communication and OSEK tasks. Finally, evaluation results using the developed prototype brake-by-wire system will be discussed.

DEPENDABLE AND COST-EFFECTIVE ARCHITECTURE

A COMPARISON OF ECU CENTRIC ARCHITECTURE AND NETWORK CENTRIC ARCHITECTURE

To implement X-by-Wire systems satisfying both dependability and cost-effectiveness, we have so far proposed a network centric architecture based on a concept of autonomous decentralized systems [1][2]. Our proposed network centric architecture and conventional ECU centric architecture are compared in Figure 1. For highly dependable systems, they should have not only a normal vehicle control function, but also a backup control function executed in case of an ECU failure, and a node status monitoring function (membership function) to identify the failed ECU. In the ECU centric architecture, all these functions are centralized in the integrated ECU, which means that this architecture is essentially equivalent to a server-client architecture. An actuator ECU (client) only executes an actuator control as the integrated ECU (server) orders. In this architecture, a failure in the integrated ECU will lead to a system failure due to the centralization of functions. To avoid this situation, the integrated ECU should be fail-operational, i.e. it should keep operating even if one or possibly multiple faults have occurred in the ECU. Triple or more

redundant architecture is commonly used for fail-operational ECUs, but this solution tends to increase the ECU cost, and consequently the system cost.

The proposed network centric architecture is effective at eliminating ECU redundancy and thus reducing the system cost. In this architecture, although the normal vehicle control function is centralized in the integrated ECU, the backup control function and node status monitoring function are decentralized in all ECUs. All ECUs share various data through the network, and each ECU autonomously gets or broadcasts the data from or to the network. So each ECU, including a sensor/actuator ECU, can monitor the status of other ECUs and execute the backup control according to which ECU has failed. Therefore, the network centric architecture can tolerate the existence of the failed ECU and does not require the high cost fail-operational ECUs. Although sensor/actuator ECUs in the network centric architecture are required to be more intelligent compared to those in the ECU centric architecture, the increase in CPU computation power can be reduced, for example by limiting the backup control function to a minimum necessary function for safe vehicle operation (a basic backup control function).

Figure 2 shows an example of our proposed network centric architecture applied to the brake-by-wire system. This system consists of an integrated ECU, four brake ECUs, and a brake pedal ECU. Each ECU is designed to be fail-silent, i.e. it does not interfere with communication among other ECUs and shuts off the power supply of the braking actuator after detecting a fault in its own ECU.

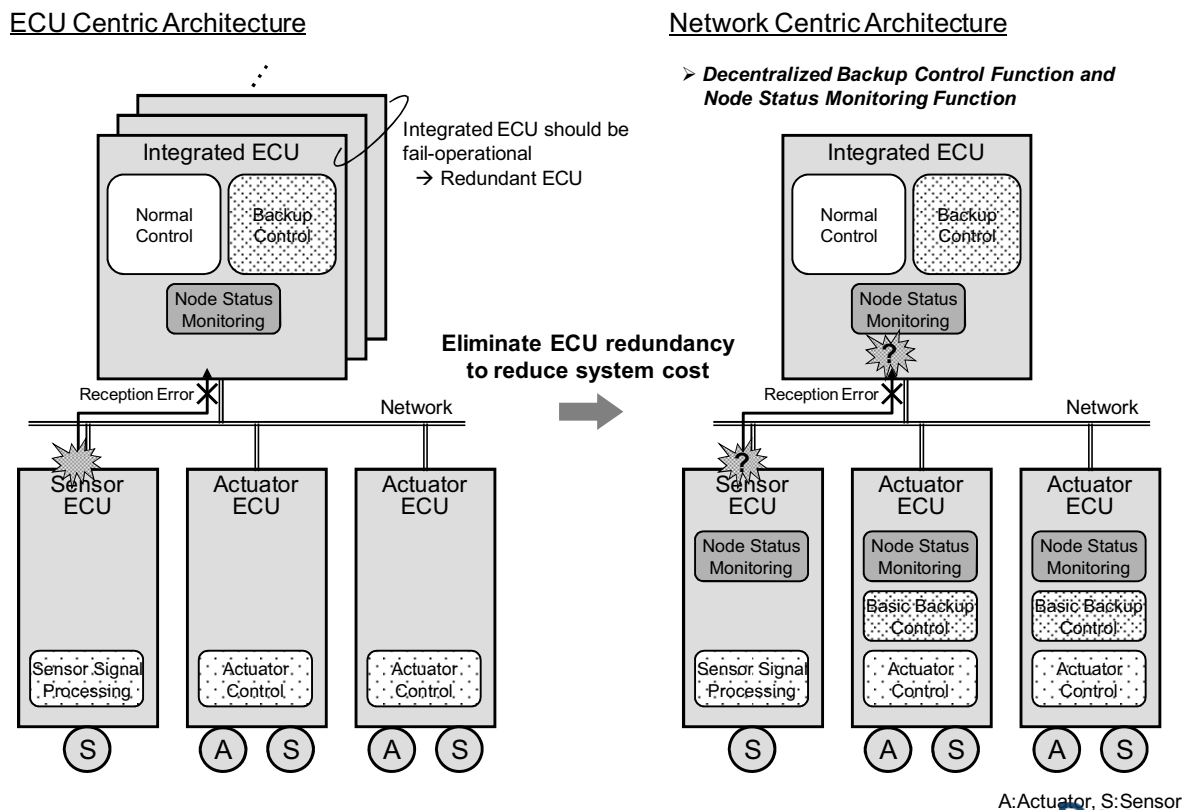


Figure 1. Comparison of ECU Centric and Network Centric Architecture

These ECUs share the information required for brake control throughout the network.

When the system operates normally, as indicated in Figure 2 (a), the integrated ECU gets the data S1 put by the brake pedal ECU on the network and calculates the target braking force values, B1 to B4, for four brake ECUs. Each brake ECU controls a braking actuator so that the actual braking force becomes the target value. As shown in Figure 2 (b), if the integrated ECU should fail, a sophisticated braking control function would be suspended. However, the vehicle can maintain minimal necessary braking functionality for safe vehicle operation thanks to the autonomous backup function, where each brake ECU determines the integrated ECU has failed, gets the shared brake pedal position value S1, and independently calculates the target braking force with the basic backup control logic integrated in each brake ECU.

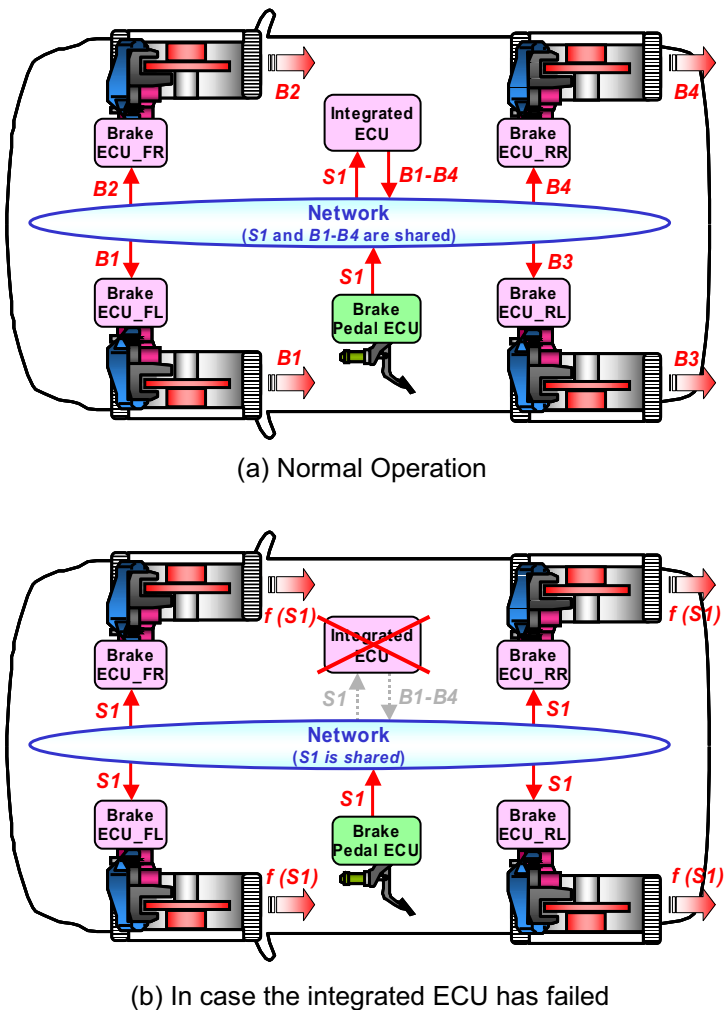


Figure 2. Network Centric Brake-by-Wire System

A TECHNICAL CHALLENGE IN NETWORK CENTRIC ARCHITECTURE

On the other hand, because there is no master node for communication monitoring in the network centric architecture, to execute any autonomous backup control, we have a technical challenge of how to identify the failed node accurately and ensure consistency of the information about which nodes are available among all remaining normal nodes to avoid a disagreement in the control mode. For example, as shown in Figure 1, when the integrated ECU fails to receive data from a certain sensor/actuator ECU, the integrated ECU cannot determine whether the sender (sensor/actuator ECU) or receiver (integrated ECU) is faulty.

MEMBERSHIP FUNCTIONALITY IN NETWORK CENTRIC ARCHITECTURE

Membership functionality will solve the above-mentioned problems in our network centric architecture. It provides information on the availability of all the other nodes by exchanging a local view of the status of other nodes among all nodes.

We have proposed a process sequence of the membership functionality in FlexRay systems as shown in Figure 3 [4]. FlexRay communication is scheduled so that every node sends a frame at least one time in one communication cycle and receives frames from all the other nodes. In Communication Cycle i, each node evaluates other nodes' status locally by receiving and evaluating frames sent by other nodes. In this communication cycle, even if a certain node detects a fault when some node sends a frame, the node cannot identify whether the fault has occurred in the sender node or its own node because each node evaluates the node status by itself. In the following communication cycle, Cycle i+1, the local view of other nodes' status evaluated in Cycle i is exchanged among all nodes, and each node determines node status by voting on the exchanged node status data.

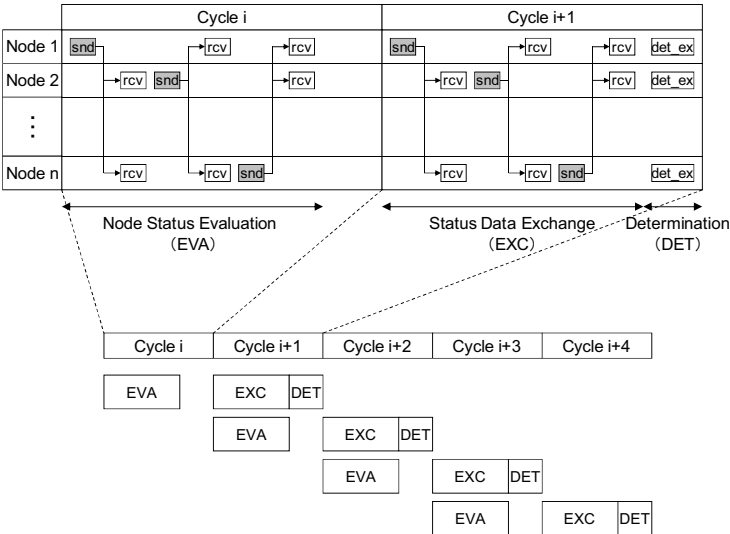


Figure 3. Process Sequence of the Membership Functionality in FlexRay Systems