Abstract - Automotive Electric and Electronic systems come into wide use as requirement for more and complex functionality in vehicle is increase. Automotive E/E systems are based on networked ECUs, which each ECU is connected to each other through network, and distributed control system. As the ECUs are increase on the network, the development of automotive E/E system gets complicated. This paper considers the problems of automotive E/E system and proposes distributed service architecture, which can provide dependability and flexibility. Distributed service architecture separates service management from service processing, simplifies service layer and service node and makes development phase easy. Depend on service clustering. Proposed architecture separates services into sub-network from network and provides distributed service management.

Keywords: In-Vehicle Network; Distributed Control Network; Fault-Tolerant Architecture; Dependable Distributed System;

1 Introduction

Electric, electronic and software are generally used in automotive system. Automotive electric/electronic system provides huge improvement of functionality, performance and product properties. But automotive industry does not have much experience in automotive E/E engineering; include software, embedded computer system and management. In fact, 49.2% of car break downs in Germany were due to E/E system failure in 2003 and also major recalls from automotive industry are based on problems of E/E system

One important point of automotive E/E system for automotive industry is the possibility to achieve competiveness through cost-efficient and dependable system. But development and modification of automotive E/E system is complicated and dangerous, even these are easier than mechanic/hydraulic system. This paper proposes distributed service architecture to solve these problems and to make automotive E/E system dependable.

Section 2 describes background for automotive E/E system. Section 3 describes the overview of distributed service architecture, section 4 presents service management, flexible implementation, fail-safety, on-line diagnosis and test. Section 5 describes experimental environment. Finally, the last section presents conclusion and future work.

2 Background

In early days, a vehicle has only a few Engine Control Units (ECU) and ECUs were not connected to each other and worked as stand-alone units, shown as fig. 1 a). It is simple enough to develop and modify, and barely need engineering efforts. Also, an ECU failure does not affect other ECUs.

As a development of electronics and embedded system technology, Electronic Control Unit (ECU) which is based on software, is used to implement automotive functionality. Automotive E/E system gets complex substantially to implement a lot of functions which demand for competitive strategy, customer requirement and law. The Mercedes S-Class included more than 50 ECUs from 1991 and the BMW 5 and 7 series have 70 networked ECUs from 2004.

In-Vehicle Network (IVN), which connects ECUs to ECUs, with point-to-point connections is the simplest topology and can make functionality easily, shown as fig. 1 b). But this topology has complex wiring which is hard to modify and extend. And as ECUs are increase, the wiring and connection points are increase exponentially. This is not good for development, production and maintenance of vehicle. The Mercedes S-class from 1991 has more than 3km of wiring.

Section 2 describes background for automotive E/E system. Section 3 describes the overview of distributed service architecture, section 4 presents service management, flexible implementation, fail-safety, on-line diagnosis and test. Section 5 describes experimental environment. Finally, the last section presents conclusion and future work.
connected with others through network and are able to share some information from sensor. Although networked E/E system has some advantages compared to point-to-point connection, still some limitation are remaining to provide flexibility and scalability. Sensors, actuators and other properties do not have their own network interface, so they have to be a part of some kind of ECUs.

Fully networked E/E system consists of smart sensors and smart actuators which have their own network interfaces thus they can connect to IVN independently and assure high flexibility and scalability. But with the increase in number of ECUs, sensors and actuators, the development of Automotive E/E system becomes complicated. Moreover, heavy traffic on IVN causes congestion, delay and emission of data, thus safety and dependability are not sure. This paper proposes distributed service architecture to solve problems, which are described above, and provide flexibility. Distributed service management points (SMP) maintain services hierarchically and provide redundant services to assure dependability. According to the requirements, SMP can consist of various services and can modify services easily and support on-line diagnosis and test to validate services.

3 Basic System Architecture

Proposed distributed service architecture consists of a network of SMPs and Service Control Point (SCP). The functionality of SMP is a service management of system with service state messages and relays the service requests. Each SMP is connected with other SMPs through network but SMPs do not have any information over the other SMPs, shown as fig. 2. With service index table, SMP can only lookup available services and SMPs which manages services.

SMP manages service states of SCPs and relays service request from SCP to specific SMP which can process requested service. Each SMP checks state of other SMPs with service state message, maintains service index table which describes service information. Each SMP has its SCPs which are connected to sub-network. Depend on functional and strategic requirements of automotive E/E system, the composition of SMPs and SCPs, which called service clustering, can vary considerably. In most cases, service cluster and services are defined in system analysis documents already.

If service is processed within a SMP, it called in-bound service. If service is not processed within a SMP, SMP has to relay service request to other SMPs, it called out-bound service, shown as fig. 3. In case of in-bound service, requested services do not make any data on main network, only on sub-network which separated from main network.

![Figure 3 Basic composition of SMP and SCPs](image)

SCP consists of ECUs, sensors, actuators and other properties. SCP processes relayed service request from SMP with its own resources. When some kind of event happened, SCP sends service request to SMP simply and processes relayed service request from SMP. If SCP needs some service it sends a service request to the SMP to which it is directly connected. If the requested service is available within the same SMP, in-bound service, it is relayed to SCP, which can process the service, on the sub network. On the other hand, if the service is unavailable, out-bound service, the SMP relays service request to another SMP on main network. In case of no SMP on the network is able to process the service, the SMP replies with a service fail message. Also, service requests and service replies are define in system analysis documents. The entire architecture is shown as fig. 4.

![Figure 4 Overview of Distributed Service Architecture](image)

Logging SMP can collect all of service request, service state and service failed messages. Logging SMP is connected with SCPs which are related to log services and provide on-line diagnosis with various logging data services.
Depend on service clustering, proposed distributed service architecture separates sub-network from main network. The separation of services can reduce numbers of connection points and disperse network traffic. Comparing state messages of SMPs, the proposed architecture can detect faults of SMPs and can operate in fail-operational mode with redundant SMPs. Additionally, SMPs can isolate fault of SCPs and redundant SCPs can cover faults of other SCPs.

4 Distributed Service Architecture

This section describes service management of proposed architecture and flexible configuration in implementation level. Also, describe reliability and construction of on-line diagnosis and test.

4.1 Service Management

Distributed service architecture adopts point-to-point service architecture in general distributed systems. Proposed architecture manages services and cooperates without central supervisor and central arbiter [1][9]. Because of single point failure problem, complexity and flexibility, central control supervisor is not good for distributed control system. Automotive E/E system is a kind of real-time control system which is based on a set of control units with software and interfaces interconnected on network. Depend on system requirement, allocation of control units on the network will be various as well as allocation of interfaces and software on the control units. But services have to be finished within deadline, which defined in the system requirement, although system has different design. We assume that proposed architecture just has simple layers, which consist of single SMP layer and single SCP layer, to assure service result within deadline and to provide simplicity for flexible reconfiguration. To manage services on the distributed service architecture, each SMP has 3 information tables, service index table, service page table and service list table.

Table 1 Service Index Table

<table>
<thead>
<tr>
<th>Service List ID</th>
<th>Service Page ID</th>
<th>Service Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>List ID 1</td>
<td>Page ID 1</td>
<td>T</td>
</tr>
<tr>
<td>List ID 2</td>
<td>Page ID 1</td>
<td>T</td>
</tr>
<tr>
<td>List ID 3</td>
<td>Page ID 1</td>
<td>T</td>
</tr>
<tr>
<td>List ID 9</td>
<td>Page ID 2</td>
<td>T</td>
</tr>
<tr>
<td>List ID 10</td>
<td>Page ID 2</td>
<td>T</td>
</tr>
<tr>
<td>List ID 11</td>
<td>Page ID 9</td>
<td>F</td>
</tr>
</tbody>
</table>

According to service clustering and design policy, service page table is defined statically, shown as Table 2. SMPs have to share same information of service page table, also. For dependability and simplicity, service page table separate from service index table. Because service index table indicates service state only, SMP can recognize service state easily without lookup of all of SMPs, if service needs to be moved to another SMP, service page table will be modified only.

Service list table has information of SCPs which correspond to services, shown as Table 3. Each SMP has its own service list table, it is local information.
Table 2 Service Page Table

<table>
<thead>
<tr>
<th>Page ID</th>
<th>Primary SMP</th>
<th>Secondary SMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page ID 1</td>
<td>SMP1</td>
<td>SMP4</td>
</tr>
<tr>
<td>Page ID 2</td>
<td>SMP2</td>
<td>SMP5</td>
</tr>
<tr>
<td>Page ID 9</td>
<td>SMP3</td>
<td>SMP2</td>
</tr>
</tbody>
</table>

Table 3 Service List Table

<table>
<thead>
<tr>
<th>List ID</th>
<th>Primary SCP</th>
<th>Secondary SCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>List ID 1</td>
<td>SCP1</td>
<td>SCP4</td>
</tr>
<tr>
<td>List ID 2</td>
<td>SCP2</td>
<td>SCP5</td>
</tr>
<tr>
<td>List ID 3</td>
<td>SCP3</td>
<td>SCP1</td>
</tr>
</tbody>
</table>

When local SMP receives service request from SCP, local SMP finds service list ID of service request in service index table. If local SMP finds service list ID in service index table, it retrieves service page ID. Local SMP finds information of SMP in service page table with service page ID, shown as fig. 6. In case of in-bound service, local SMP finds information of SCP in service list table with service list ID. In case of out-bound service, local SMP relays service request to remote SMP. When remote SMP receives service request from local SMP, remote SMP finds information of SCP in service list table with service list ID, also. In case of both, SMP can find SCPs in service list table and relays service request to SCP.

Table 1 is example of service index table, 6 services and 3 pages are in service index table. Service List ID 1, 2 and 3 belong to Page ID 1 and all of these services are available. Service List ID 9 and 10 belong to Page ID 2, service is available, also. Service List ID 11 belongs to Page ID 9, service is not available. When SMP receives service request with List ID 9, it retrieves Page ID 2 from service page table. Page ID 2 consists of SMP 2 and SMP 5 in service page table. From Table 2, Primary SMP is SMP 2, SMP will relay service List ID 9 to SMP 2.

Table 3 is an example of service list table of SMP1. When SMP want to relay service List ID 2 to SCP, SMP just retrieves List ID 2 in service list table. List ID 2 has two SCPs, SCP 2 for primary SCP and SCP 5 for secondary SCP in service list table. SMP will relay service list ID 2 to SCP2.

Each SMP checks service state of SCPs periodically and manages service flag in service index table. Also, if an event occurs, SMP updates service flags and notify immediately. When SMP detects that service state of SCP is changed, SMP update service state and sends service state message to other SMPs..

4.2 Flexible Implementation

Proposed distributed service architecture provides flexible implementation. Depending on system requirement, main network and sub-network can be implemented with various digital control networks [2]. Generally, CAN, FlexRay, and MOST which have high bandwidth and dependability, are used for main network. Especially, FlexRay and FT-CAN, which have redundant communication channel, are good for safety and dependability but not for flexibility. Sub network often uses CAN and LIN, which are cheap and easy to use. SMP and SCP can construct various hardware and software architecture; also, SMP and SCP can be on same hardware (like a multi-processor, SoC) in implementation level. SCP can be implemented as various combinations of ECUs and sensors, actuators, depending on system requirement and restriction.

Figure 6 Relation of service tables

Table 1 is example of service index table, 6 services and 3 pages are in service index table. Service List ID 1, 2 and 3 belong to Page ID 1 and all of these services are available. Service List ID 9 and 10 belong to Page ID 2, service is available, also. Service List ID 11 belongs to Page ID 9, service is not available. When SMP receives service request with List ID 9, it retrieves Page ID 2 from service page table. Page ID 2 consists of SMP 2 and SMP 5 in service page table. From Table 2, Primary SMP is SMP 2, SMP will relay service List ID 9 to SMP 2.

Table 3 is an example of service list table of SMP1. When SMP want to relay service List ID 2 to SCP, SMP just retrieves List ID 2 in service list table. List ID 2 has two SCPs, SCP 2 for primary SCP and SCP 5 for secondary SCP in service list table. SMP will relay service list ID 2 to SCP2.

Each SMP checks service state of SCPs periodically and manages service flag in service index table. Also, if an event occurs, SMP updates service flags and notify immediately. When SMP detects that service state of SCP is changed, SMP update service state and sends service state message to other SMPs..

4.2 Flexible Implementation

Proposed distributed service architecture provides flexible implementation. Depending on system requirement, main network and sub-network can be implemented with various digital control networks [2]. Generally, CAN, FlexRay, and MOST which have high bandwidth and dependability, are used for main network. Especially, FlexRay and FT-CAN, which have redundant communication channel, are good for safety and dependability but not for flexibility. Sub network often uses CAN and LIN, which are cheap and easy to use. SMP and SCP can construct various hardware and software architecture; also, SMP and SCP can be on same hardware (like a multi-processor, SoC) in implementation level. SCP can be implemented as various combinations of ECUs and sensors, actuators, depending on system requirement and restriction.

Figure 6 Relation of service tables

Table 1 is example of service index table, 6 services and 3 pages are in service index table. Service List ID 1, 2 and 3 belong to Page ID 1 and all of these services are available. Service List ID 9 and 10 belong to Page ID 2, service is available, also. Service List ID 11 belongs to Page ID 9, service is not available. When SMP receives service request with List ID 9, it retrieves Page ID 2 from service page table. Page ID 2 consists of SMP 2 and SMP 5 in service page table. From Table 2, Primary SMP is SMP 2, SMP will relay service List ID 9 to SMP 2.

Table 3 is an example of service list table of SMP1. When SMP want to relay service List ID 2 to SCP, SMP just retrieves List ID 2 in service list table. List ID 2 has two SCPs, SCP 2 for primary SCP and SCP 5 for secondary SCP in service list table. SMP will relay service list ID 2 to SCP2.

Each SMP checks service state of SCPs periodically and manages service flag in service index table. Also, if an event occurs, SMP updates service flags and notify immediately. When SMP detects that service state of SCP is changed, SMP update service state and sends service state message to other SMPs..

4.2 Flexible Implementation

Proposed distributed service architecture provides flexible implementation. Depending on system requirement, main network and sub-network can be implemented with various digital control networks [2]. Generally, CAN, FlexRay, and MOST which have high bandwidth and dependability, are used for main network. Especially, FlexRay and FT-CAN, which have redundant communication channel, are good for safety and dependability but not for flexibility. Sub network often uses CAN and LIN, which are cheap and easy to use. SMP and SCP can construct various hardware and software architecture; also, SMP and SCP can be on same hardware (like a multi-processor, SoC) in implementation level. SCP can be implemented as various combinations of ECUs and sensors, actuators, depending on system requirement and restriction.

Figure 7 Flexible Implementation

Proposed architecture separates sub-network from main network. Because of this separation, numbers of ECUs, sensors and actuators are decreased on main network and network traffic is dispersed. Also, design verification and validation of automotive E/E system will be easier. Physical
restriction of network can be avoided, that’s why the physical link of network can be shortened. Traditionally, I VN consists of power-train, chassis, body and entertainment domains, each domain works independently. But, they need to be connected with each other to share information, e.g., Adaptive Cruise Control needs information and control of power-train and chassis domain. Infotainment system needs information of all of domains.

Depend on system requirements, SMPs may support various network and sub-network in implementation level, shown as Table 4. For example, CAN 2.0, which generally used in automotive E/E system, for main network, CAN 1.0 and LIN for sub-network are possible. Also, FlexRay, which based on time-triggered protocol, for main network, CAN and LIN for sub-network are possible.

<table>
<thead>
<tr>
<th>Example</th>
<th>Main Network</th>
<th>Sub-Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
<td>CAN 2.0</td>
<td>CAN 1.0, LIN</td>
</tr>
<tr>
<td>Example 2</td>
<td>FlexRay</td>
<td>CAN, LIN</td>
</tr>
<tr>
<td>Example 3</td>
<td>MOST</td>
<td>CAN, LIN</td>
</tr>
</tbody>
</table>

In case of CAN network, CAN use priority-based protocol, a kind of event-triggered, for multiple access control. To implement functions, priority assignment methods for CAN messages are needed in design phase. The design of priority assignment gets more and more complicated as the number of connection points, numbers of messages are increase. Low priority messages cannot be transferred through the network because of jitter and delay introduced after the implementation. To solve this problem, flexible priority of CAN messages are proposed [5], but this is not good for safety and reliability of the entire system. Moreover, as physical link runs long, transfer rate is going down, e.g., maximum transfer rate is up to 1Mbps at 40m but 250Kbps at 200m and 50Kbps at 1km.

In case of FlexRay, FlexRay use TTP (Time Triggered Protocol) for multiple access control, design of slot allocation of static segment is needed. According to static slot allocation, messages are transferred in each static segment. The rest of a time segment, message is transferred as a dynamic segment. Similar to CAN, as numbers of connection-points are increase, the design of slot allocation gets complicated exponentially. 74 parameters need to be configured with suitable value within predefined range. Possible configuration spans more than 10^24 for each design. Lack of flexibility and sub-optimal resource usage problem are still remaining [13]. In worst case, messages cannot be allocated in a static segment because the numbers of slots in a static segment are limited, e.g., 16 bytes payload, 3ms static segment cycle will be 93 slots at 10Mbps, 51 slots at 5Mbps and 27 slots at 2.5Mbps [14].

The Proposed architecture separates sub-network from main network, only SMPs are connected to main network, the number of connection-points will be decrease, shown as fig.8. As only out-bound services are transferred through the main network, network traffic will be decrease and design, validation and verification will be easier. Furthermore, as physical link is shortened, loss of transfer rate can be minimized.

### 4.3 Fail Safety

A fault-tolerant system has redundant components to assure dependability and safety generally [3][4][8]. The proposed architecture provides distributed fault tolerant architecture. Redundant SMP in service page table and redundant SCP in service list table minimize the failure of the services. To detect software faults, at least, two version of software are needed. Also, to detect hardware faults, more than two types of hardware are needed. Use of different version of software can detect software fault and hardware transient fault, shown as Table 5. And use of different types of hardware can detect hardware permanent fault.

<table>
<thead>
<tr>
<th>S/W</th>
<th>H/W Transient</th>
<th>H/W Permanent</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>O</td>
<td>X</td>
</tr>
<tr>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

**Figure 8 Comparison of service flow**

![Figure 8 Comparison of service flow](image)
Each SMP can manage primary SCP and secondary SCP in service list table for fault-tolerant system as well as SMP in service page table. Different software on SCP, for fault detection, is available. Depending on system requirements for safety and dependability, fault-tolerant SCP runs in dual mode. In first mode, all of SCPs can fully process services. When run in fail-operational mode, secondary SCP processes only degraded services.

4.4 On-line Diagnosis and Test

Logging SMP can collect all of state messages and service messages while logging SCPs process those messages on development phase and product phase. On-board computer, which can process simple diagnosis, can be a kind of logging SCP. Off-board interfaces, like OBD-II, for intensive diagnosis, can be a kind of logging SCP, shown as fig. 9[6].

Collection and analysis, statistics, data-mining of messages are processed by logging SCPs. As the composition of logging services, messages can be processed efficiently and systematically for a long time during development phase. Simple logging SCPs can be used during production phase for maintenance. As the increase of services, various SCPs can be used for on-line diagnosis. Flexible on-line diagnosis can be possible for domain characteristics of vehicle and requirement.

If service list and service configuration are not correct, service fail message will be returned. Input of test cases, e.g., service ID, service table configurations, and output, e.g., expecting service result, can be generated automatically from system design documents. When service configuration is changed for optional service and variation, test automation with test probes and test case is good for verification and validation.

5 Experimental Results

5.1 Experimental environment

Experimental environment, adaptive front lamp control system, consists of three SMPs and four SCPs, shown as fig. 11. Assume that two SCPs in drive shaft cluster detect degree of steering wheel and speed of vehicle and send control request to SMP. SCP, which detects degree of steering wheel, sends control requests of yaw of the front lamps. SCP, which detects speed, sends information of control speed. As the speed of vehicle, front lamps have to move fast or slow. SMP with two SCPs use CAN for a sub-network, because of sampling rate of data. SMP with yaw SCPs use LIN as a sub-network. Test SMP and test SCP is on same hardware and CAN use for main network.

Freescale MC9S12XF512 board is used for SMPs which have two interfaces of each LIN, CAN, Flexray. MC9S12XF512 board is working in stand-alone mode, shown as fig. 12. Software of SMP is developed with Freescale Cordwarrior IDE and BDM interface.
5.2 Experimental Scenarios

We design service list and service configuration of system. To manage service state, simple service flag message is developed. When SMP detects service change of SCPs, SMP sends service flag message to others. Only service flag message can change service state of service index table. If there is no corruption, all SMPs have same service state.

Service message consist of service request message and service fail message. It is possible that local SMP sends service request message to remote SMP before local SMP change service state with service flag message. In this case, remote SMP will reply with service fail message. When local SMP receives service fail message, Local SMP retrieve service state and try again. Out-bound service can be validated with CAN analyzer, shown as Fig. 13.

<table>
<thead>
<tr>
<th>Message type</th>
<th>CAN ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Flag</td>
<td>0xD8E00000</td>
</tr>
<tr>
<td>Service Notification</td>
<td>0xD9E00000</td>
</tr>
<tr>
<td>Service Interrogation</td>
<td>0xDAE00000</td>
</tr>
<tr>
<td>Service Fail</td>
<td>0xE8E00000</td>
</tr>
<tr>
<td>Service Req1(Left)</td>
<td>0xE9E00000</td>
</tr>
<tr>
<td>Service Req2(Right)</td>
<td>0xEAE00000</td>
</tr>
<tr>
<td>Service Req3(Up)</td>
<td>0xEBE00000</td>
</tr>
<tr>
<td>Service Req4(Down)</td>
<td>0xECE00000</td>
</tr>
<tr>
<td>Service Req5(Reset)</td>
<td>0xEDE00000</td>
</tr>
</tbody>
</table>

6 Conclusions

This paper proposed distributed service management to solve complexity of traditional in-vehicle network architecture and provide flexibility. The proposed architecture can use various control networks in implementation level. Because of separation of service management from processing, distributed SMPs manage own SCPs simply and just relay service requests. As the sub-network is separated from main network, design, verification and validation of vehicle network is easier than before. Redundant SMPs and SCPs improve safety and dependability of system. Proposed architecture can be used other distributed control area. For design and implementation of automotive E/E system, EAST-ADL (Architecture Description Language) and Automotive Open system Architecture(AUTOSAR) can be used [10][11][12].

Research about the framework to define and design services and system is needed. Requirement analysis, system configuration and formalized development process, which are suitable for de-facto of the automotive industry, will be added to our future work. Moreover, research on implementation platform for SCPs and SMPs with AUTOSAR complaint is also needed. Finally, Test interfaces and test platform for verification and validation are needed for safety and dependability, also.
7 ACKNOWLEDGEMENTS
This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the CITRC(Convergence Information Technology Research Center) support program (NIPA-2012-C6150-1202-0011) supervised by the NIPA(National IT Industry Promotion Agency) and the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation.

8 References