On Communication Requirements for Control-by-Wire Applications

R. Johansson, Chalmers University College, Goteborg, Sweden

P. Johannessen, Volvo Car Corporation, Goteborg, Sweden

K. Forsberg, Chalmers University of Technology, Goteborg, Sweden

H. Sivencrona, SP Swedish National Testing and Research Institute, Boras, Sweden

J. Torin, Chalmers University of Technology, Goteborg, Sweden

Keywords: communication protocol, safety critical, system analysis

Abstract

Many control-by-wire applications are inherently safety critical. For distributed control systems, the communication subsystem as the backbone is a critical component. Thus it is vital that dependability requirements gathered from the application are well considered in the design of the communication component. However, dependability is costly and thus, it is important to carefully assess these requirements. With essential requirements from distributed control-by-wire applications in mind, we discuss the central role of the communication subsystem for system safety with focus on dependability and economy issues.

Applications of particular interest today are fly-, steer-, and brake-by-wire. From these applications we identify differences and similarities in e.g. fault-tolerance, intrinsic redundancy and production volume. Requirements on fault-tolerance states how faults should be tolerated before system failure. We acknowledge case specific redundancy, and exploit how it can be utilized to accomplish sufficiently high level of system safety. Production volume influence distribution between development and recurrent costs.

A common set of requirements for the communication sub-system have been established. We identify a set of features and properties that are the core requirements. This can serve as a foundation for any fault tolerant control-by-wire protocol definition. Finally, we compare this hypothetical protocol with four existing protocols intended for control-by-wire applications; FlexRay, SAFEbus, TTCAN, and TTP/C.

Introduction

Embedded systems are extensively used in closed-loop control systems. In particular we refer to drive-by-wire control systems as an automotive class of applications where there are no physical connections (mechanical, pneumatic or hydraulic) between the steering wheel, the pedals, and the wheels. Similarly, a fly-by-wire aircraft has no physical connections between the pilot stick and the aircraft's control surfaces. In this paper, we will refer to all such applications as *control-by-wire*.

A *drive-by-wire* application may be structured according to different levels of control [Ref1], see Figure 1. The *navigation* function selects the paths to be used to reach the *target* by using knowledge about present position (*localization*) and information about available *routes*. The output of the navigation function and environment information (weather, traffic conditions etc) constitutes the input to the *guidance* function which will for example pilot a vehicle with instructions in angle speed in yaw and acceleration in longitude direction. Many functions are handled with different degrees of autonomy from manual, to automatic localization, route path finding, navigation and guidance. There will be a variety of options due to the set of equipment hosted by the individual vehicles and to specific traffic conditions. Either the driver manually gives the order of angle speed and acceleration via the steering wheel or pedals, or the orders are given automatically by navigation and guidance equipment. The vehicle is expected to adaptively behave the same for a given set of commands despite disturbances from road and wind, different speed, and errors in the car equipment, etc. For vehicle behavior due to maneuvering commands, we use the term *vehicle dynamics* and the function that executes the maneuvering commands is called *vehicle dynamics control* (VDC) function.

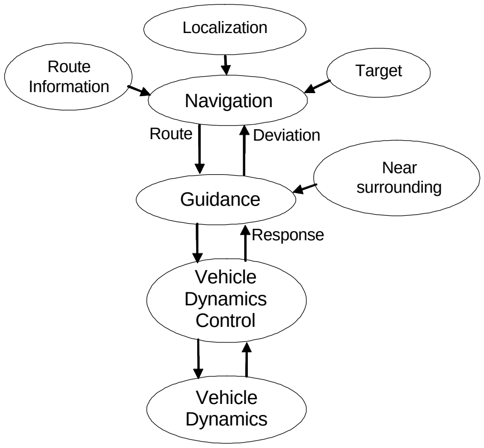

Figure 1: Different control levels in a by-wire application

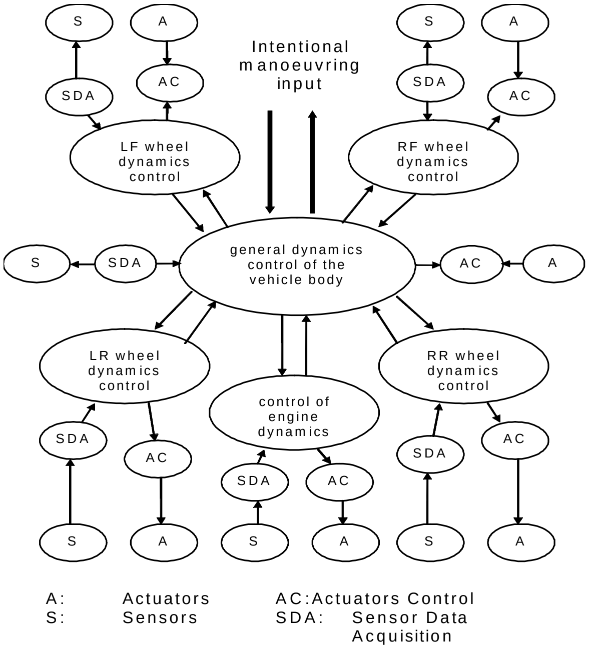
S
A
SDA
AC
Intentional manoeuvring input
LF wheel dynamics control
RF wheel dynamics control
general dynamics control of the vehicle body
LR wheel dynamics control
RR wheel dynamics control
control of engine dynamics
A: Actuators
S: Sensors
AC:Actuators Control
SDA: Sensor Data Acquisition

Figure 2: Vehicle Dynamics Control, four wheel vehicle

The functional partitioning of VDC functions in automotives may be illustrated as shown in Figure 2. The following six functions constitute the vehicle dynamics main functions; general dynamics control of the vehicle body (global), control of the engine dynamics (local), independent control of the four wheels (local). Furthermore both the global function and each local function have support functions maintaining local sensors data acquisition and controlling local actuators. Figure 2 also applies to a train vehicle model excluding the, steering functionality [Ref2]. The same model can also be extended to cover a fly-by-wire system, for example as a future system suggested for the Swedish military aircraft JAS39 Gripen [Ref3] in which the wheel modeling is replaced by seven control surfaces for the aircraft. From the functional partitioning we observe that some kind of functional distribution might be suitable for the implementation of the model. This is so for several reasons; the VDC is a safety critical application and fault tolerance must be built-in. Fault tolerance is often achieved using redundant units. By placing the controller units near their respective physical objects, the number of connections and the cable lengths can be reduced.

The VDC model for control-by-wire applications imposes important non-functional requirements on the system. *Dependability* is a system property. When handling error situations, we use the following dependability attributes; *Reliability*, *Maintainability*, *Availability*, *Safety* and *Security*. In our case we concentrate on *reliability* and *safety* aspects. In [Ref4] dependability in automotive systems is discussed. The dependability requirement for a failure mode with fatal consequences in a steer-by-wire system, i.e. the *safety*, is proposed to be 10^{-8} as the highest tolerable probability of failure per hour at system level. In order to verify that such a requirement is satisfied, we have to calculate the *reliability* of the system.

Real time behavior: Time requirements may range from soft to hard/critical depending on the task, as well as the current surrounding environmental conditions. An issue that becomes important in distributed control systems is the influence of randomly varying delays in control algorithms [Ref5]. Control algorithms rely on periodically sampled input. If delays and variations, i.e. *jitter* are in the magnitude with other timing properties in the vehicle dynamics then the control laws are not valid and unpredictable results may occur.

Economy: Production volume influences distribution between development costs and recurrent cost. This would justify significant differences e.g. in a flight control design versus an automotive steer control. On the other hand, as safety requirements are similar through the different control-by-wire applications it seems reasonable that lower volume applications (sea, air, railway etc) should be able to benefit from solutions designed for high volume applications such as in the automotive industry.

Objectives: We wish to share our experiences from three different cases and discuss communication protocols in control-by-wire applications. With this paper we aim to contribute to the ongoing discussion and encourage an application-focused viewpoint, rather than a protocol-focused viewpoint, to save cost and decrease system complexity.

Related Work: [Ref6] treats real-time communication efficiency and reliability in general. In [Ref7] two avionics (Honeywell's SAFEbus, NASA's SPIDER), and two automotive bus (TTTech's TTP/C, Flexray) architectures are compared from the perspectives of fault hypotheses, mechanisms, services, and assurance.

Control-by-Wire in distributed systems

Synchronization of distributed nodes: A Control system where information processing is distributed among several computing elements, *nodes,* generally requires synchronization, i.e. any computation should be based upon a common view of the system state. Sensor and actuator values should be consistent within the cluster of nodes that uses or computes these values. Consistency can be achieved by introducing *membership in value and time domain*. Membership is accomplished for example by periodic message passing where messages include both time stamps and sufficient information for the receiver to verify that the sending node is working as specified. Time synchronization requires a *global time* with granularity to a specified extent. Global time can be implemented with the aid of a centralized fault tolerant clock, or as distributed clocks synchronized with a fault tolerant clock synchronization algorithm. A distributed clock means that the system also must adapt to higher requirement (fault assumption, fault containment regions, etc.). There are many different clock synchronization algorithms see for example [Ref8]

Real Time Requirements: There are special *real-time requirements* emerging from control-by-wire applications. Sensors and actuators related to the same control object may be connected to different nodes in distributed systems. Delays are introduced in the control loops, mainly as a consequence of communication delays. These might be minimized to get a high control performance. The time between sampling and actuation for a given control loop is generally required to be constant in consecutive samplings even if this means it cannot be minimal. This is because the control laws are designed with specified delay compensations. A *varying control delay* invalidates the compensation, and causes reduced performance. Reduced performance, and even instability, is also caused by variations in the sampling frequency, *jitter*. Periodic processes for sampling and actuation must therefore be forced to execute with a fixed period.

Communication system topologies: A distributed computer system consists of a set of interconnected nodes. Topologies for the interconnection can be a physical broadcast bus, a star-coupled system, a ring system, or any combination of them. Redundant channels are often used for protocols that wish to achieve fault tolerance against more bus failures. If these redundant channels are combined with redundant nodes, it is possible to increase the reliability of the system.

Any bus architecture that is intended for control-by-wire applications, with requirements for extremely low failure rates, must be validated and verified for its purpose. A fault hypothesis specifies the type of faults and arrival rates that the bus should tolerate. Assurance will include massive testing of the actual implementations, as well as extensive reviews of its design and assumptions.

Dependability Requirements: Analysis [Ref9] of US ground vehicle usage shows that even if a highly improbable safety-critical failure rate of 10^{-9} failures/hr is achieved, the exposure of 200 million vehicles may result in a safety-critical system failure at least once in a week assuming some billion kilometers traveled. This puts high requirements on how to handle different types of faults, and still maintain a lowest level of service. For high integrity systems the probability of a catastrophic failure must be less than 10^{-9} per hour. Evidence to support that a design can achieve such a failure rate is required by the aerospace certification authorities. An automotive certification authority does not yet exist. However, studies indicate that the predicted failure rate of the emerging automotive drive-by-wire systems will need to be in the same order of magnitude. High safety requirements and the fact that no single point of failure can be accepted in the design implies that fault tolerance by means of redundant hardware units is required.

Consensus for distributed applications: Membership is considered to be a prerequisite for application consensus to avoid inconsistencies where the system could become partitioned i.e. some nodes in the system have their opinion about system state while other nodes form sub-clusters with diverging opinions. An example of this is when one sensor value is read differently by two entities in the system, and one process gets one value while another process gets another value. Reasons for this could be so-called slightly-out-of-specification in value domain. Nodes operating after strict timing constraints must have accurate time. Badly synchronized nodes can lead to inconsistencies in the distributed system and also lead to partitioning. The methods to handle these scenarios vary but all safety critical systems must handle them correctly.

Standard communication protocols comparisons

For distributed control systems, the communication subsystem as the backbone is a critical component. There are currently significant ongoing efforts to establish standards for future communication protocols, particularly for fly- and drive-by-wire. The protocols presented here represent such efforts; FlexRay, SAFEbus, TT-CAN, and TTP/C.

FlexRay: FlexRay [Ref10] has emerged from research at BMW, Daimler-Chrysler, Motorola, and Philips et. al. and is intended to provide a flexible communication concept. Message passing mechanisms support time-triggered and event-triggered messages. The protocol provides fault-tolerant clock synchronization via a global time base and collision-free bus access. FlexRay relies on mini slotting, taking advantage of synchronous and asynchronous message sending. The communication is designed for several different topologies. Further, the protocol supports guaranteed message latency; message oriented addressing via identifiers and scalable system fault-tolerance via either single or dual channels. Independent Bus Guardians in the physical layers provides functionality for efficient error containment.

SAFEbus: SAFEbus [Ref11] was developed by Honeywell to serve as a communication backbone of the Boeing 777 Airplane Information Management System, which supports several critical functions. The protocol uses a bus topology where the bus interfaces (called Bus Interface Units) are duplicated, and the bus consists of four wires. Most of the functionality of SAFEbus is implemented in the BIUs, which performs clock synchronization, message scheduling, and transmission functions. Each BIU acts as its partner’s bus guardian by controlling its access to bus. Each BIU controls one pair of communication channels but is able to read all four channels. Each bus consists of two data lines and one clock line. This bus architecture must be considered to be the most mature protocol since it has been in use for many years. It is however, due to the massive redundancy, costly.

Time Triggered CAN (TTCAN): The CAN protocol has evolved from it’s initial release in 1986, known as *Standard CAN*, through the extended protocol definition in 1991, *Extended CAN*. It is currently under revision by ISO TC22/SC3/WG1/TF6 for a new specification where, in particularly, time triggered communication issues are addressed, *Time Triggered CAN* [Ref12]. TTCAN is a higher-layer protocol on top of the unchanged extended CAN protocol. TTCAN synchronizes the communication schedules of all CAN nodes in a network, and provides a global system time. With synchronized nodes a message can be transmitted at a prescheduled slot without the need for bus contention. Apart from the synchronized communication the TTCAN nodes operate according to the previously defined standard CAN protocol (ISO 11898-4). As in FlexRay, TTCAN thus supports a mixture of time-triggered and event-triggered operation.

The Time-Triggered Protocol (TTP): The TTP/C protocol was originally developed at the Technical University of Vienna within the MARS project. Since then, the protocol has matured and communication controllers are commercially available today. TTP/C [Ref13] provides four services; deterministic message sending, clock synchronization of global time, membership service, and clique avoidance. The protocol supports time-triggered communication and the local time in all nodes within the cluster is synchronized with the cluster’s global time. A message is considered valid as long as it is syntactically correct and on time. The nodes are responsible for the delivery of correct messages to the other nodes, and the data that the host wishes to send is not checked for correctness. Further, the membership service is a function that allows a communication system to maintain and establish a consistent view of the correct nodes in the cluster even in the presence of faults. The clique avoidance algorithm is designed to avoid membership sub-clustering.

Case Studies

Several standards and directives specify safety requirements and behavior of safety critical systems. This includes development processes of software and hardware to comply with specific industry areas. Avionics and railway systems are covered by a number of standards to ensure reliability and safety. These standards are not compatible with each other. For the automotive industry, there are still few international standards.

Fly -By-Wire: This case is based on the Flight Control System, FCS, for JAS 39 Gripen, a multi-role, combat aircraft. It has seven primary control surfaces that are controlled by the FCS, and in a future architecture all control is distributed to seven actuator nodes, placed by each primary control surface, pictured as boxes in Figure 3. The actuator nodes are connected via a communication bus to which also sensor and cockpit nodes are connected.

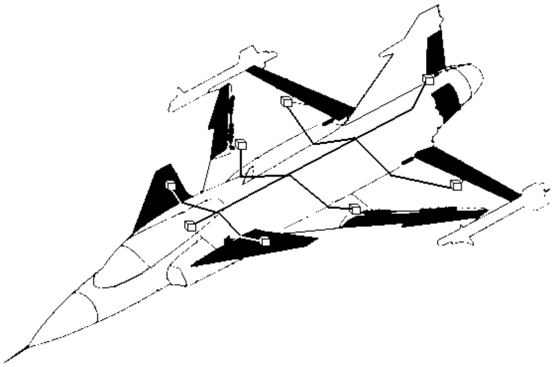

Figure 3: A future distributed flight control system for JAS 39 Gripen

For commercial aircrafts the Federal Aviation Administration, FAA, and Joint Aviation Authorities, JAA, organizations are responsible for certification. Requirements are stipulated in Federal Aviation Regulations, FAR, and Joint Aviation Regulations, JAR. For military aircrafts, it is traditionally the respective country's departments of defense who set the requirements.

The effect of a failure in the aircraft is rated using a 3-degree scale ranging from Level 1, loss of redundancy, to Level 3, loss of control. The safety requirement for the aircraft states that a) the probability for loss of control shall be less than $0.5 * 10^{-6}$ per flight hour, and b) no single point failure shall cause loss of control. To fulfill Req. a) the sensor nodes are duplicated, and to fulfill Req. b) the communication bus is duplicated. One actuator node for each control surface is sufficient due to two redundancy strategies. First, we utilize intrinsic redundancy since JAS 39 Gripen is well maneuverable with only six of its seven control surfaces, given that the surface of a failing actuator streamlines. Second, all software for control law computation is redundant located and executed in all actuator nodes, and by exchanging data each actuator can vote on seven results.

The real time demands is set by the inertia of the object to be controlled from which specific timing requirements at different levels (sampling, communication) are derived. Due to stability criteria and performance, the global control laws of Jas 39 Gripen are executed in 60 Hz, while the local loop is considerable faster. Each actuator node closes the inner loop locally, so the real time demands on the communication are set by the global control. Hence, all inter-node communication runs at 60 Hz.

The distributed FCS has twenty nodes attached to both buses, and here we will estimate the required bandwidth for control on one bus without regards to overhead due to communication protocol, coding etc. All sensor signals are sent on the bus, and all the adaptation and the control law computation are performed redundantly in all seven actuator nodes. To avoid scale factors, and in order to simplify programming, it is in this analysis assumed that all sensor signals are expressed in floating point notation with 24 bits mantissa, 8 bits exponent, giving 32 bits total. The actuators exchange the seven computed command words here assumed to be 16 bits (quite high resolution for a D/A-converter), and a status word of another 16 bits. The estimation of required bandwidth is calculated and listed in Table 1.

Table 1 Fly-by-Wire communication bandwidth		
Advanced Air Data sensor		
15 Hz: Pressure static, Mach number, Altitude	3 x 32 bits x 15 Hz	10560
60 Hz: Angle of attack, Angle of sideslip	2 x 32 bits x 60 Hz	
The sensors are duplicated	x 2	
Angular Rate Gyro sensor		
60 Hz: Pitch, Roll and Yaw	3 x 32 bits x 60 Hz	11520
The sensors are duplicated	x 2	
Accelerator Sensor		
60 Hz: Acceleration in z- and y-axle	2 x 32 bits x 60 Hz	7680
The sensors are duplicated	x 2	
Cockpit node		
60 Hz: Pilot command for Pitch, Roll and Pedal.	3 x 32+16 bits x 60 Hz	13440
Assume 16 bits for discrete signals		
The sensors are duplicated	x 2	
Interconnection node		
60 Hz: Acceleration x-axle, Pitch, Roll and Aircraft weight	4 x 32 bit x 60 Hz	7680
Actuator nodes		
60 Hz: Seven command words (16 bits) and one status word (16 bits) from all seven nodes	(7 x 16 bits + 16 bits) x 7 x 60 Hz	53760
Secondary control surfaces and Engine		
60 Hz: Assumed 32 bits from these four nodes	4 x 32 bits x 60 Hz	7680
Resulting bandwidth		112.320 bits/s

<u>Steer-by-Wire:</u> Currently steering with no mechanical connection between the steering wheel and the road wheels, i.e. steer-by-wire, is not permitted for commercial cars. However, The Economic and Social Council of The United Nations is currently working on new regulations for steering equipment. These new regulations will permit steer-by-wire systems.

This case is based on two steer-by-wire concept studies [Ref14,15]. The FAR car is a small drive-by-wire car in scale 1:5 that was developed at The Royal Institute of Technology together with Volvo Car Corporation during 2003. Lulea University of Technology and Volvo Car Corporation developed the SIRIUS 2001 car together in 2001. The network topologies of the cars are very similar. Both have six nodes that are connected with a broadcast bus. The FAR car uses TTCAN and the Sirius car uses TTP/C. A sketch of the two cars' steering system is show in figure 4.

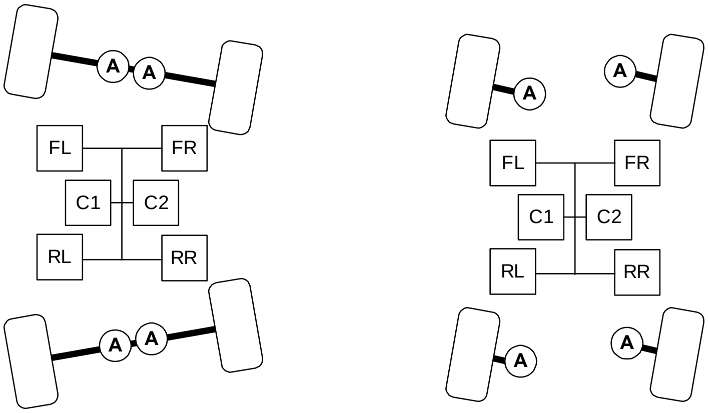

Figure 4: The steer-by-wire systems of the FAR (left) and SIRIUS (right) cars

The main difference between the two steer-by-wire subsystems lies in the actuator control. For the FAR car the wheels' axes are controlled by dual redundant actuators on each steer axis, each actuator can steer the wheels alone. In the SIRIUS car, single actuators individually control each wheel. Redundancy in the SIRIUS car for the steer system is achieved by inherent system redundancy.

The control system in both cars uses a global update frequency of 100 Hz for continuous values and 10 Hz for mode control. Both cars use a similar allocation pattern as for the fly-by-wire case, where sensor data is broadcasted on the communication bus. Table 2 shows the communication analysis.

Table 2 Steer-by-Wire communication bandwidth

Central node (C1 and C2)		
100 Hz: steering wheel angle (14 bits) 10 Hz: steering mode (2 bits) Sensor redundantly allocated to C1 and C2.	14 bits x 100 Hz 2 bits x 10 Hz x 2	2840
Wheel node (FL,FR,RL and RR)		
100 Hz: wheel speed and steer angle (12 bits) steer angle sensor duplicated	3 x 12 bits x 100 Hz	32400
100 Hz: Four command words (14 bits) and one status word (16 bits) from all four nodes	((4 x 14 bits) + 16 bits) x 4 x 100 Hz	
Resulting bandwidth		35.240 bits/s

Brake-by-Wire: This case is based upon the electro-mechanical brake system SAB WABCO EBC10 designed for railway vehicles such as Light Rail Vehicles (LRV) or trams. The brake system provides several functions; *Service braking* is frequently used by the driver in controlling the train. Service braking shall achieve specified levels of performance at any time. *Emergency braking* may be initiated by the driver or, in some cases, even by a passenger in the case of extreme hazards. Emergency braking shall achieve a specified level of performance, and a high level of integrity. *Security braking* is activated in the case of system failure within the ordinary brake system, thus it is a redundant back-up system. The security brake system is designed to apply maximum forces, so as to stop the car within the shortest possible distance. *Holding brake* maintains speed during downhill movements. *Parking brake* should be able to hold a defined load on a defined gradient for an infinite time. It is intended for use while the train is stabled. The parking brake should be designed to ensure that it automatically secures the train in the event of loss of emergency or service brake. *Wheel slide protection* is fitted to optimize braking performance and to provide protection against wheel set damage e.g. during braking in poor adhesion conditions. Such systems are furthermore designed to minimize the braking force so as to achieve minimal practical stopping distance.

A brake control system is a safety critical application. System failures may introduce accidents, severe damage and human injuries. See for example [Ref2,16,17] for detailed discussions of general requirements on computer based safety critical control applications. The use of electronic devices for safety critical train applications are subject to standardization, see for example [Ref18,19].

The brake system is a centralized brake-by-wire solution and consists of a Master Controller (Brake Control Unit), two distribution units and four physical brake actuators. Recently a distributed brake-by-wire solution has been proposed (see figure 5). This late approach is the objective for our case.

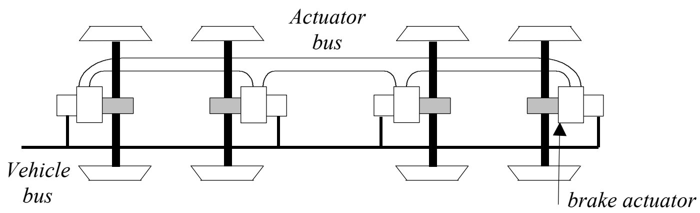

Figure 5: Brake system case

The effect of a failure in the system is rated in three categories; *critical failure* requires immediate stop of vehicle operation. The brake system is out of order or the security requirements can't be satisfied, the vehicle is not usable anymore. *Heavy failure*, the vehicle can stay in operation for the present, but be removed after the daily operation time, i.e. failure of a single brake actuator and the total braking force is sufficient due to distribution to the rest of the system. *Insignificant failure*, there is no need to stop operation; the failure is handled during the next scheduled maintenance. The safety requirement for the brake system (LRV) is stated as "mean driving power between two failures in km" (MDBF); for critical failure: $2*10^6$ km, for heavy failure: $1* 10^6$ km, and for insignificant failure: 666, 666 km.

For the control system a global update frequency of 50 Hz is to be used. This does not apply for wheel slide protection algorithms which requires considerably faster control. This control is closed in an inner loop and does not load inter-node communication. Thus, all signals (sensor and actuator values) are transmitted each 50 Hz cycle. The brake system has four nodes attached to a communication bus. Each node masters a single actuator but compute values for all actuators and there are two sensors attached to each node (Table 3).

Table 3 Brake-by-Wire communication bandwidth

Wheel node (four identical)		
50 Hz: Required disc pressure actuator	4 x 10 bits x 50 Hz	2000
50 Hz: applied pressure sensor The sensor is duplicated	4 x 10 bits x 50 Hz x 2	4000
50 Hz: rotational speed sensor The sensor is duplicated	4 x 10 bits x 50 Hz x 2	4000
Resulting bandwidth		10.000 bits/s

Case Studies Conclusions: In the cases demonstrated above we have established application-derived requirements for bus communication bandwidth. We obtain more realistic figures if we also consider the overhead imposed by a communication protocol. In our calculations we have numbers gathered from previous work and standards [Ref10,11,12,13]. The results are summarized in Table 4.

Table 4 Bandwidth consumption (bits/sec)

Application	Effective	TTCAN	FlexRay	SAFEbus	TTP/C
Fly-by-wire	112 320	235 680	187 680	148 320	156 000
Steer-by-wire	35 240	139 040	113 440	80 800	93 600
Brake-by-wire	10 000	19 200	19 200	12 800	19 200

Conclusions

Most control-by-wire applications are classified as safety-critical. Therefore, these systems have to be fault-tolerant. Fault tolerance requires some form of redundancy. However, introducing redundant units to achieve fault tolerance is an extra cost that should be avoided if possible. It is obvious that any inherent redundancy in the application should be exploited if possible. To exploit inherent redundancy, the nodes in a distributed system must have a consistent view in both functional and temporal domains. Together with system mode changes, this requires synchronization at application level, **application membership**.

If the control-by-wire approach is to be more than a handful of exclusive systems designed for space and aviation, they have to become much less expensive than today's solutions. Hence, the systems should use standard components as far as possible. High volume components like **COTS** are designed for this purpose and are not application specific. The communication protocol should benefit from being a COTS. Therefore it should only implement the **basic required functionality** and **limited autonomous behavior**.

Control by-wire applications are characterized by predetermined functional and temporal behavior based on control theory. The applications perform fast closed loops locally, and close slower loops globally. Preferably sensor and actuator values need to be communicated to allow distributed processing. Sizes of these values are predetermined and it becomes straightforward to calculate the total amount of data to be communicated. The result is that we can determine a periodicity and required bus bandwidth for all sensor and actuator values at design time. These requirements need to be met by the communication system in terms of **constant and limited delays** together with **sufficient data rate**.

From the previous discussions we conclude that for control-by-wire systems in general, and their communication protocols in particular, the following should be included or supported:

- ❑ The communication subsystem is vital to assure system safety.
- ❑ The control-by-wire system should not host any other functionality than vehicle dynamics control.
- ❑ Inherent redundancy should be used by the application for cost effective fault-tolerance.
- ❑ The communication system must guarantee a sufficient data rate, as well as constant and limited time delays.
- ❑ The communication system should be a COTS with basic required functionality and limited autonomous behavior.
- ❑ The communication protocol should not guarantee application consensus, it is best accomplished at the application level.

Acknowledgments: This work has been conducted within The Centre of Excellence CHARMEC (CHAlmers Railway MEChanics), The Swedish Automotive Research Program, and The National Aerospace Research Program, all within VINNOVA, the Swedish Agency for Innovation Systems.

References

[Ref1] Torin, J. et.al. Architecture of a Dependable Computer Network for Control of Safety and Real-time Critical Functions. Tech. Report 119, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 1991.

[Ref2] Johansson, R. Dependability characteristics and safety criteria for an embedded distributed brake control system in railway freight trains, Tech. Report 8, Chalmers Lindholmen University College, Goteborg, Sweden, August 2001.

[Ref3] Forsberg, K. Design Principles of Fly-By-Wire Architectures, PhD. Thesis, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 2003.

[Ref4] Torin, J. Dependability in Complex Automotive Systems - Requirements, Directions and Drivers, Tech. Report 128, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 1992.

[Ref5] Nilsson, J. Analysis and design of Real-Time Systems with Random Delays, Licenciate Thesis TFRT-3215, Lunds University of Technology, 1996.

[Ref6] Claesson, V. Efficient and Reliable Communication in Distributed Embedded Systems, PhD. Thesis, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 2002.

[Ref7] Rushby, J. A Comparison of Bus Architectures for Safety-Critical Embedded Systems, CSL Technical Report, SRI International, September 2001.

[Ref8] Lonn, H. Synchronization and Communication Results in Safety-Critical Real-Time Systems, PhD. Thesis, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 1999.

[Ref9] Koopman, P. et.al. Toward Middleware Fault-Injection for Automotive Networks, Proc.of 28th International Symposium on Fault Tolerant Computing Systems, Munich, Germany, June 1998.

[Ref10] Fuehrer, T. et.al. FlexRay – The Communication System for Future Control Systems in Vehicles, SAE World Congress, Paper No. 2003-01-0110, March 2003.

[Ref11] Hoyme, K. Driscoll, K. SAFEbus™, IEEE Aerospace and Electronic Systems Magazine, 8(3):33-39, March 1993.

[Ref12] ISO Standard 11896. Road Vehicles – Controller Area Network (CAN) – Part 4: Time Triggered communication; Working Draft ISO 11898-4.

[Ref13] TTP Specification, www.ttagroup.org.

[Ref14] Backstrom, J. et.al. Project FAR – Project Report, Tech. Report TRITA MMK 2003:28, Division of Mechatronics, Royal Institute of Technology, Stockholm, Sweden, 2003.

[Ref15] Johannessen, P. SIRIUS 2001 – A University Drive-by-Wire Project, Tech. Report 01-14, Dept. of Computer Eng., Chalmers University of Technology, Goteborg, Sweden, 2001.

[Ref16] EN 50126:1999, Railway applications, The specification and demonstration of dependability, reliability, availability, maintainability and safety (RAMS).

[Ref17] prEN 13452:1999E, Railway applications – Braking – Mass transit brake system

[Ref18] EN 50155:1995, Railway applications, Electronic equipment used on rolling stock.

[Ref19] EN 50125-1:1999, Railway applications, Environmental conditions for equipment, Part 1.

Biographies

Roger Johansson, Chalmers University College, Dept. of Electrical and Comp. Eng., SE-412 72 Goteborg, Sweden. Phone: +46 31 772 57 29, Fax: +46 31 772 57 31, E-mail: roger@chl.chalmers.se.

Roger is currently doing applied research at Chalmers University College. His current research focuses on a brake-by-wire application for railway vehicles.

Per Johannessen, Volvo Car Corporation, Dept. 94221, ELIN, SE-405 31 Goteborg, Sweden. Phone: +46 31 59 55 56, Fax: +46 31 59 66 12, E-mail: pjohann1@volvocars.com

Per is currently pursuing his Ph.D. in computer engineering at Chalmers University of Technology. His research focus is on the design process of architectures for safety critical drive-by-wire systems.

Kristina Forsberg , Chalmers University of Technology, Dept. of Computer Engineering, SE-412 96 Goteborg, Sweden. Phone: +46 31 772 10 00, Fax: +46 31 772 36 63, E-mail: stinaa@ce.chalmers.se.

Kristina is working together with Saab AB Gripen and has gained experiences from the Swedish military aircraft JAS 39 Gripen. Her research focus is on future fly-by-wire systems.

Hakan Sivencrona, SP Swedish National Testing and Research Institute, ELP, SE-501 15 Boras, Sweden. Phone: +46 33 16 56 80, Fax: +46 33 12 50 38, E-mail: hakan.sivencrona@sp.se.

Hakan is currently pursuing his Ph.D. with a focus on validation of communication protocols for safety critical systems.

Jan Torin, Chalmers University of Technology, Dept. of Computer Engineering, SE-412 96 Goteborg, Sweden. Phone: +46 31 772 10 00, Fax: +46 31 772 36 63, E-mail: torin@ce.chalmers.se.

Dr Torin is professor emeritus at Chalmers University of Technology and has for 30 years been working with dependable computer systems. He has been the general chair of the DSN2001 conference in Goteborg.