Chapitre 10

Les réseaux temps réel embarqués dans les véhicules

10.1. Contexte et contraintes

Dans les années à venir, la plupart des prestations offertes sur un véhicule automobile seront assurées par un système embarqué reposant sur des technologies numériques. La réalisation de ces prestations passe donc par la spécification de composants logiciels et par leur implémentation sur une architecture matérielle support. Initialement, chaque nouvelle prestation était réalisée sous forme d'un nœud (désigné sous le terme anglais d'*Electronic Control Unit* ou ECU) autonome, à savoir un microcontrôleur supportant le logiciel contrôlant l'ensemble des capteurs et des actionneurs. Rapidement, cette technique a montré ses limites. En effet, la nature même de certaines prestations comme le freinage, la direction, l'éclairage, la gestion des portières, le diagnostic, etc., impose de plus en plus une distribution des fonctions les réalisant sur un ensemble de calculateurs. De plus, il est actuellement difficile de considérer une fonction indépendamment des autres ; par exemple, la vitesse véhicule qui est élaborée notamment par le système de contrôle du moteur est nécessaire à la réalisation d'autres fonctions comme le contrôle de la suspension, la vitesse de balayage des essuie-glaces, etc. C'est ainsi que, actuellement, sur un véhicule haut de gamme, il peut y avoir jusqu'à 2500 informations, ou signaux, échangés entre des fonctions réparties sur environ 70 ECUs [ALB 04]. Dés lors, pour des raisons de coût, de poids, d'encombrement, de complexité de câblage, il est devenu rapidement impossible de supporter ces échanges par des connexions point à point et l'utilisation de réseaux locaux embarqués est apparu comme une solution incontournable. Dans le milieu des années 1980, l'équipementier Robert Bosch

---

Chapitre rédigé par Françoise SIMONOT-LION et Nicolas NAVET.

développe le réseau CAN (*Controller Area Network*), lequel a été initialement utilisé dans des automobiles Mercedes au début des années 1990. Si CAN [ISO 94a, ISO 94b] reste, à l'heure actuelle, le réseau le plus employé, d'autres réseaux à destination de l'automobile ont vu le jour, soit en concurrence de ce premier produit, soit pour satisfaire d'autres exigences. En effet, l'ensemble des systèmes embarqués à bord d'un véhicule peut être divisé classiquement en quatre domaines principaux, « contrôle moteur », « châssis », « habitacle », « télématique et multimédia », qui n'imposent pas les mêmes propriétés de performances et de sûreté. En particulier, pour ce qui est des réseaux, les critères de choix sont la bande passante offerte, les performances, le coût, la sûreté de fonctionnement. Par exemple, certains réseaux garantissent un comportement déterministe qui s'avère nécessaire pour des applications critiques comme le contrôle de la direction ou du freinage ; dans cette catégorie, on trouve les réseaux dont l'accès au médium est de type TDMA (*Time Division Multiple Access*) comme TTP/C [TTT 03] ou FlexRay [FLE 04, MIL 05] ainsi que certains protocoles développés au-dessus de CAN. D'autres protocoles sont mieux adaptés à des transmissions d'informations courtes, pour un nombre de nœud limité et sans besoin critique de performance ; il s'agit par exemple des réseaux de capteurs/actionneurs LIN [LIN 03, RAJ 05] ou TTP/A [KOP 02]. Enfin, certains sous-systèmes induisent des flux d'informations importants (flux multimédia, en particulier) pour lesquels les paramètres de qualité de service sont primordiaux, et ils sont alors déployés sur des réseaux à haut débit tels que MOST [MOS 04] ou IDB-1394 qui est la version adaptée à l'automobile de la norme IEEE 1394. En 1994, la SAE (*Society for Automotive Engineers*) a spécifié une classification des réseaux embarqués dans un véhicule sur la base de la vitesse de transmission offerte et des fonctions distribuées sur le réseau. Les réseaux de classe A offre une vitesse inférieure à 10kb/s et sont utilisés pour transmettre des données élémentaires en utilisant une technologie bon marché. Des exemples de tels réseaux sont LIN et TTP/A. Relèvent de la classe B les réseaux de vitesse moyenne qui peuvent assurer entre 10kb/s et 125 kb/s. Enfin, la classe C regroupe les réseaux qui supportent des applications temps réel critiques nécessitant plus de 125 kb/s. Au-delà de 1Mb/s, les réseaux sont souvent dits de classe D.

Les deux premiers domaines évoqués plus haut, le « contrôle moteur » et le « châssis », relèvent particulièrement du temps réel critique. La fonctionnalité du premier est de contrôler le moteur et la transmission en fonction des demandes du conducteur et de l'état du véhicule et de son environnement. Il est réalisé par des lois de contrôle complexes et des périodes d'échantillonnage pouvant être, suivant la vitesse de rotation du moteur, de l'ordre de quelques millisecondes. L'implantation de ces lois de commande se fait sous forme d'un ensemble de tâches dont l'ordonnançabilité doit être soigneusement garantie. De plus de nombreux échanges avec les fonctions d'autres domaines sont nécessaires.

Le châssis, quant à lui, est le domaine par excellence de la sécurité d'un véhicule; il regroupe, les fonctions qui, à partir des sollicitations du conducteur (freinage, direction), de l'état de la route, de la force et de la direction du vent, etc., et de l'état du véhicule (lacet, tangage, roulis), contrôlent les équipements, comme l'ASC (*Automatic Stability Control*), l'ABS (*Antilock Braking System*), l'ESP (*Electronic Stability Program*), le 4WD (*Four Wheel Drive*). Là encore, ce domaine repose sur des caractéristiques de lois de contrôle complexes et des contraintes de temps strictes, caractéristiques auxquelles s'ajoute une plus forte distribution des fonctions. De plus, l'avènement futur de systèmes « tout électriques » (*X-by-Wire*), c'est-à-dire de systèmes pour lesquels toute transmission mécanique ou hydraulique est remplacée par une transmission numérique, intensifiera encore plus le recours à des réseaux de communication. Dans ce contexte, la sécurité du véhicule passe notamment par la fiabilité de l'architecture de communication et, en particulier, par la garantie de propriétés temporelles sur les échanges d'informations (échéances sur les transmissions, fraîcheur des données consommées) et d'intégrité (tolérance sur les pertes d'informations par suite de surcharge ou d'altération du réseau).

« Contrôle moteur » et « châssis » regroupent donc des applications soumises à des contraintes de temps strict ; leur impact sur les performances ainsi que sur la sécurité d'un véhicule prônent pour l'utilisation de réseaux à accès synchrone de classe C. D'une part, ce type de réseaux offre une souplesse pour l'intégration de systèmes conçus par différents équipementiers et, d'autre part, ils assurent un déterminisme dans le fonctionnement global des communications (borne déterministe sur les temps de réponse, par exemple)

Affichages sur le tableau de bord, contrôle des essuie-glaces, des phares, des portières, des vitres, de la climatisation, etc. constituent le domaine « habitacle » (*Body*). Dans ce contexte on tend vers un nombre de plus en plus important des équipements contrôlés ainsi que vers une sophistication toujours accrue des fonctions de contrôle. Cette multiplicité de fonctions génère parallèlement de nombreux échanges inter ou extra domaine. Toutefois la quantité d'informations transmises ainsi que la période d'émission n'est pas la même pour tous les échanges. En particulier, si on considère une fonction comme la gestion des portières, son implantation peut être réalisée sous la forme d'un système mécatronique intégrant lui-même son propre réseau de capteurs/actionneurs pour des transmissions d'informations de faible taille et ne nécessitant pas de débit élevé. C'est ainsi que ce domaine utilise de plus en plus des réseaux de classe A comme LIN ou TTP/A qui relèvent d'un fonctionnement où une station maître plus puissante consulte, selon les besoins, des stations constituées de capteurs et d'actionneurs à faible capacité de calcul et de communication. La station maître, elle, communique avec les autres sous-systèmes grâce à un réseau de débit plus élevé, par exemple CAN. Plus généralement, ce domaine est caractérisé d'une part par les contraintes propres aux systèmes réactifs (garantie de la cohérence dans l'ordre des événements, temps de

réponse borné, etc.) et, d'autre part, par des architectures de communication hiérarchisées.

Le dernier domaine est dédié aux applications multimédia et télématiques et regroupe des fonctions telles que la téléphonie main libre, la radio, la lecture de CD, DVD, l'aide à la navigation, les jeux, le télé-diagnostic du véhicule, etc. Les flux de données importants générés par ces fonctions, que ce soit à l'intérieur du véhicule que en liaison avec le monde extérieur, se caractérisent par un besoin important de bande passante, le respect d'une certaine qualité de service, l'intégrité et la confidentialité des informations. Pour satisfaire ces besoins, des réseaux de classe D sont nécessaires. Dans ce contexte, le problème ne se pose plus en termes de temps réel mais, plutôt en terme de qualité de service acceptable par les utilisateurs, conducteurs et passagers, et réalisable à un coût raisonnable.

Enfin, de plus en plus se révèle le besoin d'assurer d'une part, un certain degré de flexibilité dans la configuration des véhicules et, d'autre part la portabilité et l'intégration de composants développés séparément. Le développement de l'électronique embarquée dans un véhicule s'appuie, de fait, sur un processus complexe partagé entre plusieurs acteurs, notamment les constructeurs et équipementiers de rang 1. Il s'agit alors non seulement d'intégrer des composants mais aussi de garantir que cette intégration préservera bien les propriétés requises de performances et de sûreté de fonctionnement. Classiquement, un moyen pour atteindre ces objectifs à moindre coût est de fournir un intergiciel qui offre, d'une part, des services communs, sous une qualité de service mesurable et contrôlable et, d'autre part, une interface commune aux composants logiciels applicatifs. L'impact d'un tel intergiciel dans la sûreté des systèmes n'est pas anodin et, une de ses vocations premières étant de masquer le réseau de communication, il apparaît alors comme un acteur incontournable de celle-ci.

Dans la suite de ce chapitre, nous présenterons, dans la section 10.2., quelques réseaux illustratifs de deux techniques d'accès au médium différentes, réseaux qui sont utilisés pour supporter les échanges entre fonctions au sein de systèmes embarqués temps réel critiques. Nous décrirons, en section 10.3., les problèmes inhérents aux intergiciels embarqués ainsi que certaines solutions actuellement proposées. Enfin, en section 10.4., nous montrerons comment la sûreté d'un système dépend des services fournis par l'architecture de communication embarquée avant de conclure en section 10.5.

10.2. Réseaux embarqués

10.2.1. *Réseaux à accès guidé par le temps* versus *réseaux à accès guidé par les événements*

Si l'introduction de réseaux de communication dans les automobiles a favorisé l'accroissement du nombre de fonctions embarquées qui peuvent dès lors partager facilement des données, leur impact sur la sûreté du véhicule est prépondérant tant en mode de fonctionnement nominal pour lequel ils introduisent déjà des délais entre production et consommation d'informations que lors des situations dégradées, comme par exemple lorsque des perturbations électromagnétiques importantes corrompent les trames circulant sur les réseaux. Une analyse des réseaux de communication embarqués doit donc toujours prendre en compte l'objectif de sûreté des systèmes.

Les réseaux embarqués relèvent de deux paradigmes principaux : leur comportement est dirigé par les événements ou par le temps. Un comportement guidé par les événements signifie que les messages sont transmis au contrôleur de communication à l'occurrence d'un événement reconnu comme, par exemple, la fermeture d'une portière, la demande d'appel de phare, etc. Dans ce cas, le système pourrait prendre en compte cet événement dès qu'il se produit. Le protocole doit alors proposer une stratégie qui garantit l'accès au réseau pour la transmission de cette information assez rapidement pour que l'information véhiculée ne soit pas caduque lors de sa consommation. Cette stratégie est très efficace en termes d'utilisation de bande passante. Elle permet d'ajouter facilement de nouvelles fonctions nécessitant de nouveaux nœuds et/ou la transmission de nouveaux messages. Néanmoins, même si cela reste possible, la vérification que le temps de réponse de tout message transmis respecte une échéance n'est pas aisée dans le cas général et la détection d'un nœud défaillant par les autres nœuds n'est pas directement assurée par ces types de protocoles. Le deuxième paradigme est le comportement guidé par le temps. Dans ce cas, les informations sont transmises au sein de trames sur le réseau et toute trame est émise à des instants précis et dans un intervalle de temps, appelé *slot*, prédéfini hors-ligne. L'enchaînement des *slots* sur le réseau suit donc un format rigide qui se répète régulièrement. La stratégie pour partager l'accès de cette manière est dite *Time Division Multiple Access*. En raison de cet ordonnancement statique des trames, le comportement du réseau est entièrement prévisible et il est immédiat de vérifier que les contraintes de temps exprimées sur les informations transmises sont respectées. La régularité ou périodicité de transmission d'un nœud joue le rôle d'un battement de cœur pour toutes les autres stations. Il est alors facile de détecter une station défaillante lorsqu'une trame n'est pas transmise dans le *slot* qui lui est réservé. De plus, ce temps de détection est borné par une valeur qu'il est aisé de calculer. Si cette stratégie s'accompagne d'un comportement complètement déterministe, elle a, par

contre, tendance à surcharger l'utilisation du bus. En effet, le rafraîchissement de l'information n'est pas forcément synchrone avec la période de transmission de la trame contenant cette information et la même valeur peut être transmise plusieurs fois de suite. De plus, comme la séquence des *slots* est prédéfinie, toute demande d'extension du système conduisant nécessairement à l'adjonction de nouvelles trames voire de nouveaux nœuds est complexe et demande, d'une part, de construire un nouvel ordonnancement des trames et donc une nouvelle séquence de *slots* et, d'autre part, de configurer tous les nœuds afin qu'ils prennent en compte ce nouveau comportement. Plusieurs études comparatives ont été réalisées sur ces deux approches. Le lecteur intéressé peut consulter notamment [ALB 04], [FER 02] et [KOO 02].

Dans la suite de cette section, nous présentons des réseaux obéissant à l'un ou l'autre paradigme. CAN s'avère particulièrement représentatif des réseaux à comportement guidé par les événements, il fait l'objet du paragraphe 10.2.2 tandis que TTP/C, décrit au paragraphe 10.2.3, est un réseau strictement guidé par le temps. Certaines propositions tentent de concilier les avantages des deux approches en minimisant leurs inconvénients respectifs. Il s'agit notamment de TTCAN [ISO 00], FTT-CAN [FER 02] et FlexRay [FLE 04, MIL 05]. Nous nous attarderons uniquement sur ce dernier réseau en raison de ses aptitudes à concurrencer les approches strictement synchrones dans le secteur de l'automobile.

10.2.2. *CAN (*Controller Area Network*)*

Pour garantir que les propriétés sous lesquelles les lois de commande ont été élaborées (retard borné, fraîcheur des données consommées) sont préservées lors d'une implantation distribuée, il est indispensable de disposer d'un protocole d'accès au medium qui permette d'assurer une borne sur le temps de réponse des trames (temps entre leur émission et la fin de leur transmission). Un des moyens pour atteindre ce but est d'allouer une priorité à chaque trame et d'utiliser une politique d'accès de type « priorité fixe non préemptive ». Le lecteur trouvera dans [TIN 95] et [NAV 01] les moyens de calculer les bornes sur les temps de réponse pour un ensemble fini de trames récurrentes, le cas échéant en présence de certaines formes d'erreurs affectant la transmission. Plusieurs réseaux sont représentatifs de cette classe d'accès. Le premier, CAN, est devenu un standard de fait en Europe. VAN [ISO 94c] reprend les caractéristiques de CAN mais intègre des fonctionnalités intéressantes ; il a été employé pour les applications du domaine habitacle dans un véhicule de série de PSA Peugeot-Citroën mais n'a pas réussi à pénétrer le marché international et est actuellement abandonné. Enfin, le réseau J1850 [SAE 96] a été adopté par les constructeurs américains pour des applications non critiques ; la tendance actuelle est de le remplacer par CAN ou un réseau de capteurs/actionneurs

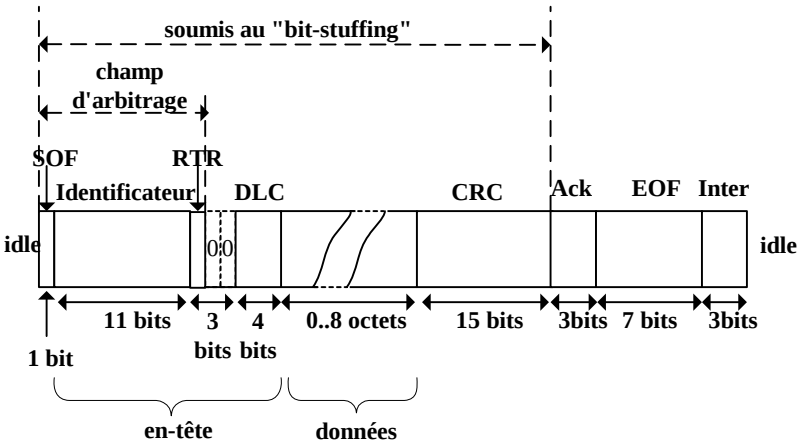
comme LIN. Dans la suite de cette section, nous ne détaillons que le protocole CAN.

CAN a été normalisé à l'ISO en 1994 [ISO 94a, ISO 94b]. Il s'agit d'un bus à diffusion selon une technique CSMA/CR (*Carrier Sense Multiple Access with Collision Resolution*). L'accès au médium se fait en fonction de la priorité des trames et en appliquant un arbitrage non destructif, comme nous le verrons plus loin dans cette section. Ne sont normalisées dans CAN que la couche liaison de données et la couche physique. La sous-couche LLC est chargée du filtrage des trames en réception, du recouvrement d'erreurs, du transfert de données et du transfert de demande de données tandis que la sous-couche MAC assure l'encapsulation/désencapsulation des données, l'insertion de bits de synchronisation (*stuffing*), l'arbitrage de bus, la détection et le signalement d'erreurs et l'acquittement de trames reçues. Le rôle de la couche physique est la mise en forme physique du bit ainsi que la synchronisation. Certaines des fonctions présentées sont spécifiquement détaillées ci-dessous. Conforme à la classe C SAE, dans sa version haut débit (250 Kbits à 1Mbits sur une longueur de 250 à 30 mètres), il peut alors être intégré, d'une part, dans les systèmes des domaines nécessitant des performances temps réel (contrôle moteur et châssis) et, sa conformité à la classe B SAE, dans sa version bas débit (10 à 125 Kbits sur une longueur de 5000 à 500 mètres) le rend apte à supporter les échanges du domaine habitable.

Chaque trame CAN est nommée par un identificateur unique (*identifier*) qui fixe la priorité de la trame. La longueur de cet identificateur est de 11 bits dans la version standard de CAN (CAN 2.0A) et de 29 bits dans sa version étendue (CAN 2.0B). La version standard qui permet de distinguer 2^{11} identificateurs est de fait la seule utilisée dans l'automobile.

Comme CAN utilise une représentation NRZ (sans retour à zéro), le protocole applique sur une partie de la trame un mécanisme d'insertion de bits (*bit-stuffing*) de longueur 5. Pour que chaque nœud maintienne un intervalle de temps constant entre la transmission de deux bits successifs sur le bus, il doit se resynchroniser régulièrement sur un front du signal.

Aussi, sur CAN, un bit supplémentaire de valeur 0 (respectivement, de valeur 1) est inséré à l'émission sur le bus, après chaque séquence de 5 bits de valeur 1 (respectivement, de longueur 0). Un mécanisme inverse est opéré par tout nœud récepteur.



SOF : début de trame
RTR : trame de données (0), de requête (1)
DLC : longueur des données
CRC : code cyclique redondant
Ack : champ d'acquittement
EOF : fin de trame
Inter : 3 bits d'inter-émission

Figure 10.1.*Fomat de la trame CAN*

La couche physique de CAN doit être capable de réaliser un « et logique ». La valeur de bit dominant est 0, celle de bit récessif est 1. Ceci signifie que, si au moins un nœud transmet un bit sur le niveau 0, alors, quelque soit le niveau du bit transmis, le cas échéant, par d'autres nœuds au même moment, le niveau résultant du bus sera 0. Le support physique décrit par la norme est la paire torsadée mais il a été envisagé également un support monofilaire ou fibre optique, une technologie courant porteur, infra-rouge ou radio.

Le format de la trame CAN (dans sa version standard 2.0A) est donné à la figure 10.1 La trame peut transporter jusqu'à 8 octets de données utiles. Sa taille maximale incluant tous les champs de gestion, une donnée de 8 octets et le nombre maximum possible de bits de resynchronisation, est de 135 bits.

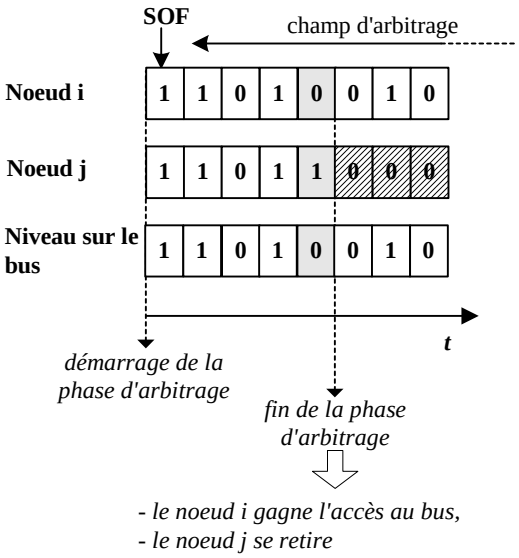


Figure 10.2. *Exemple d'arbitrage pour l'accès au bus sur CAN*

Tout nœud qui veut accéder au bus peut démarrer une transmission quand le bus est oisif. Les conflits entre les nœuds qui tentent de démarrer concurremment une transmission sont résolus par une technique d'arbitrage qui repose sur la notion de priorité de trame. Cette priorité est en fait obtenue par le champ formé de l'identificateur de la trame et du bit RTR. Cette technique, qui utilise le « et logique » fourni par la couche physique, est dite non destructive car elle permet toujours à une trame, la plus prioritaire, d'accéder au bus. Si un nœud transmet un bit récessif alors qu'un autre nœud transmet simultanément un bit dominant, le niveau résultant sur le bus sera dominant, le nœud transmettant le bit récessif observera alors sur le bus un niveau différent de celui qu'il a demandé, arrêtera en conséquence la transmission et attendra que le bus soit à nouveau oisif. Notons que pour que ceci soit possible, il faut que le signal se propage jusqu'au nœud le plus éloigné sur le bus et revienne avant la transmission du bit suivant. L'ordre de transmission sur le bus va du bit de plus fort poids au bit de plus faible poids ; aussi, le nœud qui transmet la trame dont l'identificateur a la plus petite valeur gagne l'accès au bus. L'ordre de priorité des trames est donc l'inverse de l'ordre des valeurs des identificateurs. Ce mécanisme est illustré à la figure 10.2 où deux nœuds émettent simultanément, le nœud 1 se retirant après transmission de cinquième bit de la trame. Un champ (*Ack*) est réservé dans la trame pour le mécanisme d'acquittement. Néanmoins, ce service permet au nœud émetteur de savoir qu'au moins un noeud a reçu la trame correctement, mais il ne confirme pas la réception

par un destinataire spécifique. Ceci peut poser des problèmes pour implanter des mécanismes de détection d'erreurs.

La détection d'erreur sur CAN repose sur plusieurs mécanismes comme la comparaison entre CRC calculé et CRC transmis, la réception d'une suite de 6 bits consécutifs égaux dans le champ soumis à *stuffing* ou la vérification de la structure de la trame. Toute station détectant une erreur signale celle-ci en envoyant une trame spécifique, dite trame d'erreur (6 bits consécutifs dominants). La trame corrompue pourra alors être à nouveau émise. Par l'évolution, dépendant du fait qu'une émission ou une réception s'est déroulée correctement ou non, de compteurs sur chaque nœud, CAN permet de confiner les erreurs dues à des défaillances permanentes du contrôleur ou du bus physique.

10.2.3. *TTP/C (*Time Triggered Protocol - Class C*)*

Le protocole TTP/C est en fait le cœur d'un concept plus large, appelé TTA (*Time-Triggered Architecture*) qui sont des architectures de systèmes dont le comportement (tâches et émission de trames) est strictement guidé par le temps. TTP/C fournit, entre autres, une base de temps stable à toutes les activités s'exécutant sur chacun des nœuds connectés. Cette approche rigoureusement synchrone a été développée et abondamment étudiée à l'université technique de Vienne, Autriche, dans l'équipe du Professeur H. Kopetz et des contrôleurs de communication implantant le protocole ainsi que des outils de configuration sont disponibles auprès de la compagnie TTTech (http://www.tttech.com). La spécification du protocole fait apparaître des caractéristiques et services très orientés sur la sûreté. Par exemple, la couche physique de TTP/C impose la redondance du bus physique, chacun des bus transmettant une copie de la même donnée. De plus, les contrôleurs de communication doivent s'adjoindre un équipement dit « gardien de bus », dont le rôle est de surveiller que le nœud ne transmet pas en dehors de sa spécification (par exemple, transmission d'une trame au mauvais instant ou transmission d'une trame trop longue). La technologie utilisée pour implanter ce gardien de bus doit éviter au maximum des modes de défaillance communs avec ceux du nœud ou du contrôleur (alimentation séparée, horloge distincte, etc.). TTP/C inclut un mécanisme dit de *membership* qui permet à tout nœud d'avoir une vision des nœuds défaillants et des nœuds sains. Enfin, le protocole offre un support aux changements de modes de marche ; ceux-ci décrits préalablement hors ligne sont ainsi mis en œuvre en un temps borné et connu. TTP/C supporte une topologie de bus ou d'étoile. Dans le cas d'une implantation en étoile, les gardiens de bus sont intégrés au coupleur et permettent un évitement d'erreurs qui serait impossible par chaque gardien de bus local. Néanmoins, comme le coupleur est un point critique en cas de défaillance, on a recours à une redondance du coupleur (*dual star*), ce qui accroît drastiquement le nombre de câbles.

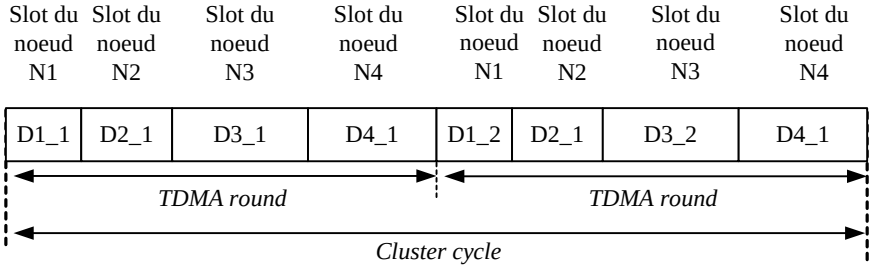


Figure 10.3. *Exemple de configuration d'un* TDMA cluster cycle

La politique d'accès au médium de communication proposée par TTP/C est de type TDMA (*Time Division Multiple Access*). Les nœuds ont accès au bus pendant un *slot* conformément à un ordre établi hors ligne. Un *TDMA round* est une suite de *slots* de longueur prédéfinie, chaque *slot* étant réservé pour un et un seul nœud. Un *cluster cycle* est une séquence prédéfinie de *TDMA round* qui peuvent différer uniquement par les données transmises à l'intérieur de chaque trame dans le *slot* réservé au nœud. Le *cluster cycle* se répète indéfiniment. Ainsi, de façon déterministe, chaque nœud sait à quel instant, il doit émettre et quelle donnée il doit transmettre. Un exemple d'ordonnancement est donné à la figure 10.3 (le *cluster cycle* est composé d'une suite de deux *TDMA rounds*; dans chaque *TDMA round*, les *slots* sont définis et successivement attribués aux nœuds *N1*, *N2*, *N3* et *N4* ; les nœuds *N1* et *N3* ne transmettent pas le même message dans les deux *TDMA rounds* alors que les deux autres nœuds transmettent toujours le même message.

L'accès au réseau repose sur une base de temps stable qui est assurée par un algorithme de synchronisation d'horloge efficace. Tout contrôleur de communication dispose de la liste des *slots* (MEDL, *Message Descriptor List*) qui lui sont réservés dans le *cluster cycle* et, pour chaque *slot*, du message qu'il doit contenir. Une mémoire à double accès, nommée CNI, *Communication Network Interface*, contient les messages émis par l'application et reçus du réseau. Aussi le contrôleur sait quand il doit émettre et quel message il doit aller chercher dans le CNI pour constituer la trame à cet instant. De manière symétrique, il sait également quand il doit recevoir une trame et ce qu'il doit en faire (voir figure 10.4). La liste des messages (MEDL) est définie statiquement à la configuration du contrôleur de communication. Ceci est un frein à l'évolutivité des systèmes puisque toute adjonction de nœud ou modification du comportement du réseau amène à reconfigurer tous les contrôleurs de communication participant à un cycle.

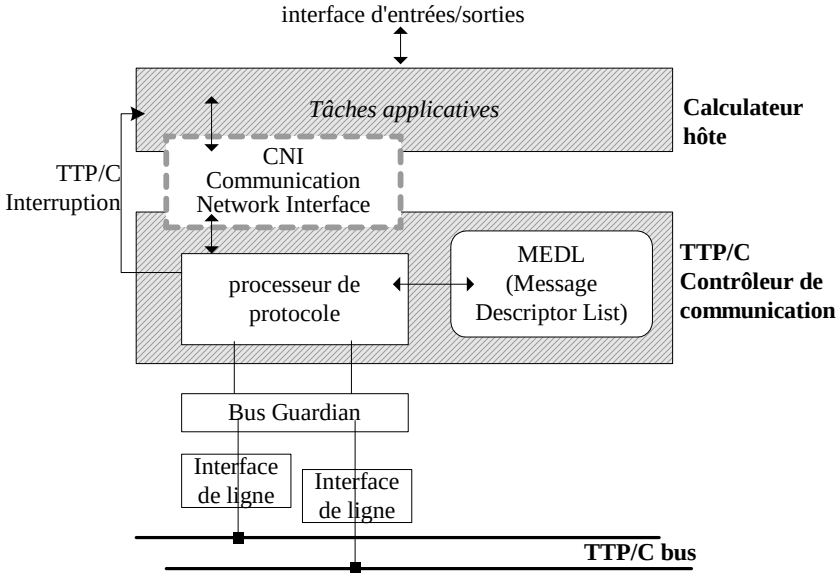


Figure10.4. *Architecture d'un nœud TTP/C*

Une trame est composée d'un champ indiquant si un changement de mode de marche est demandé par l'émetteur, du champ de données et d'un CRC. La charge utile de la trame est au maximum de 240 octets. A l'heure actuelle, le format exact de la trame n'est pas diffusé publiquement. Trois types de trames sont définis :

– la trame de « démarrage à froid »,

– la trame « à C-State explicite »,

– la trame « à C-State implicite ».

Le premier type de trame n'est utilisé qu'à l'initialisation du réseau et les autres le sont pour transporter les données émises par les nœuds. Le *C-State* est une information structurée qui indique l'état du contrôleur de communication du nœud émetteur. Il comprend : la valeur courante de son horloge, le *slot* affecté au nœud émetteur, le mode courant du contrôleur, une demande éventuelle de changement de mode de marche, le vecteur de *membership*, c'est-à-dire la liste des nœuds vus sains à cet instant par le contrôleur. Dans les trames à « C-State explicite », le « C-State » du contrôleur est intégré à la trame et est dans la portée du CRC transmis. Les trames à « C-State implicite » ne contiennent pas cette information. Néanmoins, le CRC transmis est calculé sur la base de la donnée intégrée à la trame et du « C-State » ; ainsi, les nœuds récepteurs peuvent vérifier que le nœud émetteur est sain.

La conception d'applications tolérantes aux fautes est facilitée par certains services implantés par TTP/C, en particulier, par le service dit d'évitement de « cliques » (ensemble de nœuds qui n'ont pas la même vision de l'état du système) et celui d'acquittement qui repose sur un algorithme d'exploitation des vecteurs de *membership* transmis. Ces algorithmes ont été formellement prouvés [BAU 00, PFE 00] sous des hypothèses restrictives d'occurrences de fautes ; par exemple l'hypothèse qui stipule que la distance minimum entre deux occurrences de fautes doit être supérieure à deux *TDMA rounds* peut ne pas se vérifier lorsque les fautes arrivent en rafales. Lorsque la situation courante n'est plus conforme aux hypothèses, les nœuds mettent en place un mode dégradé local et tentent ensuite de rejoindre le groupe des stations saines, ou *cluster*, en mode normal. Enfin, toujours pour accroître la sûreté des systèmes, TTP/C introduit la notion de redondance de nœuds. Cette redondance peut se faire au sein d'unité tolérante aux fautes (FTU, *Fault Tolerant Unit*) dans laquelle les nœuds de l'unité possèdent tous un *slot* sur le réseau et transmettent donc, à chaque *TDMA round*, un répliqua de la même donnée qu'ils élaborent séparément. Une autre possibilité consiste à mettre en place une redondance passive par l'introduction de nœuds fantômes. Ceux-ci, en cas de défaillance du nœud principal, prennent sa place et émettent dans les mêmes *slots*. La gestion de la redondance n'est pas assurée par le protocole mais il est néanmoins important d'intégrer cette connaissance lors de la configuration des cycles et la génération des listes de messages (MEDL) locales.

10.2.4. *FlexRay*

Ce protocole [FLE 04] est proposé par un consortium dont les membres fondateurs sont BMW, Bosch, Daimler Chrysler, General Motors, Motorola, Philips et Volkswagen et qui inclut actuellement les acteurs majeurs de l'industrie automobile.

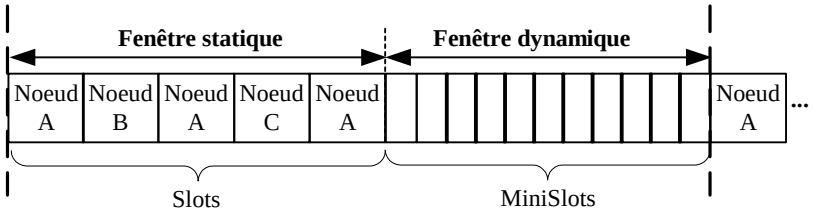


Figure 10.5. *Un exemple de cycle FlexRay*

La topologie d'un réseau FlexRay peut être un bus, une étoile simple une étoile multiple ou relever d'une combinaison de ces solutions. La redondance du support physique pas plus que l'existence d'un gardien de bus n'est imposée par le protocole. La stratégie d'accès au médium de communication repose sur un cycle de communication qui se répète périodiquement et qui enchaîne une fenêtre statique à accès strictement TDMA et une fenêtre dynamique à accès TDMA flexible (voir figure 10.5). La fenêtre statique est divisée en un nombre fixe de *slots* de taille constante (le nombre de *slots* est limité à 2047). Chaque nœud se voit allouer statiquement un ou plusieurs *slots* pendant lesquels il peut transmettre ses messages (l'exemple de la figure 10.5 montre un cycle partagé par trois nœuds A, B, C et composé de cinq *slots* ; le nœud A émet dans les premier, troisième et cinquième *slots* ; les nœuds B et C émettent respectivement dans les *slots* 2 et 4). La fenêtre dynamique permet de réaliser un comportement guidé par les événements. La base de temps est le *minislot*.

Chaque nœud possède un nombre prédéfini de *minislots* qui ne sont pas nécessairement contigus et ne peut démarrer une transmission qu'à l'intérieur de l'un d'eux. Si le nœud ne transmet rien pendant l'intervalle correspondant, le réseau reste oisif pendant ce temps et, à la fin de l'intervalle, le droit de parole est accordé au nœud possédant le *minislot* suivant. De plus, s'il y a redondance du médium, la transmission peut être différente sur les deux canaux. Ce principe est illustré à la figure 10.6. Sur le premier canal, le nœud correspondant au *minislot n* ne transmet pas d'information alors que sur le canal 2, une trame est émise. Sur le canal 1, dés la fin du *minislot n*, le nœud correspondant au *minislot n+1* transmet une trame, etc. On peut remarquer notamment que les trames envoyées par le nœud correspondant au *minislot n+4* ne seront pas reçues simultanément. Cette stratégie peut, le cas échéant, rendre complexe la gestion de répliquas.

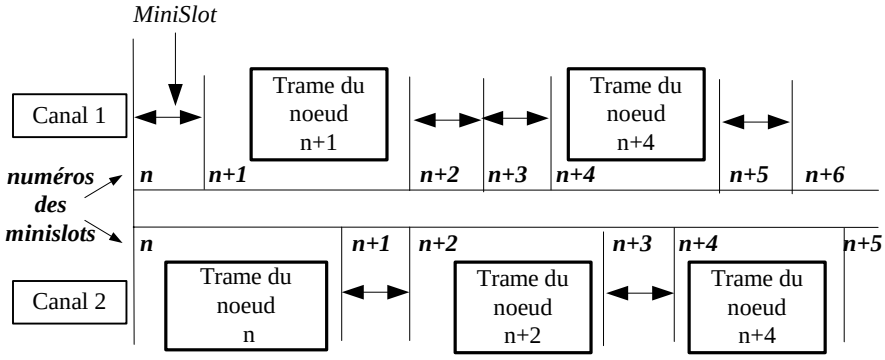


Figure 10.6. *Exemple de transmission dans la fenêtre dynamique*

La trame FlexRay est formée :

– d'une en-tête de cinq octets composée de la taille de l'information utile et d'un identificateur de trame ; ce dernier point est intéressant car il facilite la portabilité de tâches applicatives qui émettent des données sur le réseau sans avoir besoin de reconfigurer les nœuds récepteurs de ces données et sous réserve que la trame n'intègre pas d'informations issues d'autres tâches ;

– de l'information utile (limitée à 254 octets) ;

– d'un CRC de 24 bits.

FlexRay apparaît donc comme plus flexible que TTP/C et induisant une consommation moindre de bande passante puisque la transmission sur événement est possible dans la fenêtre dynamique. Sur le plan de la sûreté de fonctionnement, la fenêtre statique permet la transmission de données pour lesquelles des propriétés temporelles doivent être garanties de manière déterministe. Cependant, il faut noter que FlexRay ne spécifie pas au niveau protocolaire d'algorithme de *membership* ou de mécanismes de gestion des modes de marche. Si ces services doivent être assurés, il faut alors les implanter au-dessus des couches protocolaires FlexRay ce qui peut s'avérer plus complexe et moins performant. Enfin, FlexRay souffre encore de sa jeunesse et, contrairement à TTP/C, peu de travaux ont tenté de prouver formellement le protocole.

10.3. Intergiciels embarqués dans l'automobile

Ainsi que nous l'avons vu précédemment, les différents réseaux n'offrent pas les mêmes services et, plus important, pas les mêmes caractéristiques vis-à-vis de la sûreté de fonctionnement. D'un autre côté, les applications sont très dépendantes des services qui leurs sont directement fournis, que ce soit en termes d'interfaces mises à leur disposition qu'en terme de propriétés qu'elles peuvent attendre. Uniformiser, d'une part les interfaces d'appel des services et, d'autre part, d'homogénéiser l'offre de services ainsi que la qualité et la sûreté de ces services est le rôle classique d'un intergiciel (*middleware*). Plus précisément, les principales fonctionnalités que l'on attend d'un intergiciel sont :

– masquer la distribution c'est-à-dire de fournir la même interface et les mêmes services de communication entre composants locaux, distants, voire appartenant à d'autres domaine ;

– masquer l'hétérogénéité des plateformes (protocoles, processeurs, entrées-sorties) ;

– fournir des services de haut niveau comme la gestion de la redondance (gestion des envois et des réceptions de répliquas), la gestion des modes de marche, etc. ; pour ceci, un intergiciel a en charge de gérer d'un coté, les signaux émis ou reçus au niveau applicatif et, d'un autre coté, les trames transmises sur le réseau et

transportant un ou plusieurs de ces signaux ; ce service est particulièrement important car il se doit de préserver les propriétés de fraîcheur exprimées au niveau de chaque signaux par leurs consommateur tout en minimisant l'occupation du médium ;

– garantir les performances et la sûreté exigée de l'application et, le cas échéant, réaliser des services qui ne sont pas fournis par les couches sous-jacentes ; par exemple, si la distance de Hamming du CRC fourni par le protocole utilisé est insuffisante, l'intergiciel peut réserver une partie de la trame dédiée aux données utiles pour y insérer un CRC supplémentaire ; un autre cas de service réalisé par un intergiciel pour garantir un niveau de sûreté est de fournir une information d'état pour toute donnée reçue sur un nœud ; enfin, un intergiciel pourrait offrir un service de gestion d'une configuration adaptative qui, en fonction de l'état courant du système (par exemple, en cas de perturbations électromagnétiques), redéfinit en ligne les priorités et règles de transmission des messages sur CAN.

Plusieurs propositions spécifient un intergiciel dédié à l'automobile. Certaines sont issues de groupes de travail comme le projet Eureka ITEA EAST-EEA (www.east-eea.net) ou plus récemment le projet AUTOSAR (www.autosar.org). En fait, pour l'instant, les seuls résultats disponibles ont été obtenus par le consortium OSEK/VDX. Il s'agit de OSEK/VDX Communication et de OSEK/VDX Fault Tolerant Communication. Ils accompagnent et s'appuient sur la spécification d'un système d'exploitation embarqué dans l'automobile, OSEK/VDX OS. Nous en donnons brièvement les principales caractéristiques ci-dessous. Notons que, indépendamment de ce contexte, les principes développés dans [CAS 99] sur Volcano ont donné lieu à une diffusion commerciale pour concevoir les services de communication d'un intergiciel développé au-dessus de LIN ou de CAN.

10.3.2. *OSEK/VDX Communication*

Cet intergiciel [OSE 04] spécifie une interface et les mécanismes de mise en œuvre pour assurer la communication entre composants logiciels applicatifs indépendamment de leur localisation. Il s'exécute au-dessus d'une couche protocolaire de niveau transport et nécessite des services d'un système d'exploitation conformes à OSEK/VDX OS (gestion des tâches, des interruptions et des événements).

Au niveau applicatif sont échangés des signaux, ou « messages », entre une tâche productrice et des tâches consommatrices tandis qu'au niveau de la couche gérant la communication sur le réseau, on considère des I-PDU (*Interaction layer Protocol Data Unit*) qui incluent les messages.

Le rôle de l'intergiciel est d'une part, de mettre (respectivement, extraire) les messages dans les I-PDU et, d'autre part, d'envoyer (respectivement, recevoir) les I-PDU. Sur chaque nœud récepteur et, pour chaque message applicatif, est défini la gestion du message en réception, dans une file FIFO (premier entré, premier servi) de taille bornée ou dans un *buffer* à écrasement (une seule place). Un algorithme de filtrage est associé au message. Gestion de la file et exécution du filtre sont à la charge de l'intergiciel. Enfin, afin de masquer l'hétérogénéité des processeurs, des fonctions de *Byte Ordering* fournissent un ordre homogène des bits pour chaque octet transmis.

La manière dont les messages sont insérés dans les I-PDU est définie statiquement hors ligne. Cette information, mise à disposition de l'intergiciel lors de son initialisation, lui permet d'exécuter les services de paquetage des messages dans les I-PDU et de dépaquetage des I-PDU pour en extraire les messages.

L'application a à sa disposition deux primitives principales : initialiser un message avec la valeur d'une expression (par exemple, *SendMessage(m,x)* signifie le message *m* qui sera transmis aura la valeur courante de la variable *x*), initialiser une variable avec la valeur d'un message extrait d'une trame. Chaque message est caractérisé par une politique qui lie l'initialisation du message à l'envoi de la I-PDU dans laquelle il s'insert.

Deux modes sont disponibles : la I-PDU doit être transmise sur le réseau à chaque fois que le message est initialisé par la requête *SendMessage* (*Triggered Transfert Property*) ou non (*Pending Transfert Property*).

Par ailleurs, le mode de transmission de chaque I-PDU est lui même caractérisé : il est « direct » lorsque l'I-PDU contient au moins un message de type *Triggered Transfert Property* et, dans ce cas, la I-PDU sera transmise dés qu'un tel message sera initialisé ; le mode est « périodique » lorsque la I-PDU est envoyée périodiquement ; enfin, le mode est dit « mixte » lorsque la I-PDU est au moins transmise périodiquement et, le cas échéant, lorsqu'un message de type *Triggered Transfert Property* est initialisé.

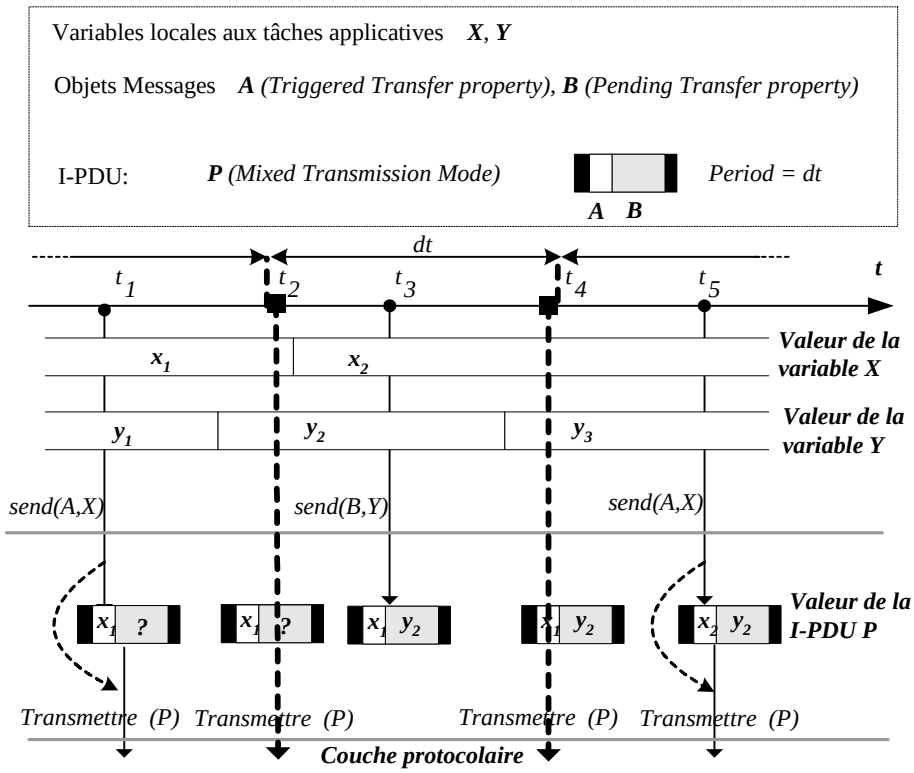


Figure 10.7. *Exemple de mode de transmission géré par OSEK/VDX Communication*

Ces mécanismes sont illustrés dans la figure 10.7. Sur cet exemple, le nœud est émetteur de deux signaux ou messages (*A* et *B*), ces messages sont initialisés, au cours du scénario, sur la base de la valeur courante de variables (*X* et *Y*). Les deux messages sont inclus dans la même I-PDU de nom *P*. Le mode d'émission de *P* est mixte et la période minimale entre deux envois de *P* est *dt*. Le message *A* est de type *Triggered Transfert Property*, tandis que *B* est de type *Pending Transfert Property*. A l'instant t_1, une requête *SendMessage(A,X)* est émise par l'application ; cette requête provoque l'envoi de *P* à la couche protocolaire sous-jacente avec la valeur, x_1 qui vient d'être affectée au message *A*.

A l'instant t_2, une période *dt* est échue et *P* est transmis automatiquement au contrôleur de communication, avec la même valeur que précédemment. A t_3, une

requête *SendMessage(B,Y)* issue de l'application met à jour la valeur de *B*, avec y_2, dans la I-PDU *P*, mais ne provoque pas l'envoi de *P*. En t_4, une nouvelle période est échue et *P* est transmise à la couche protocolaire sous-jacente avec les valeurs x_1 pour *A* et y_2 pour *B*. Lorsque les échanges se font entre applications localisées sur le même nœud, il n'y a pas constitution de I-PDU et le mode de transmission est systématiquement direct entre l'objet message émis et l'objet message reçu.

La transmission des messages est non bloquante pour l'application. Pour que les tâches applicatives puissent connaître l'état d'une transmission ou d'une réception, OSEK/VDX offre un mécanisme de notification. Il est significatif de noter qu'un mécanisme reposant sur des chiens de garde permet à l'intergiciel de maintenir une variable d'état qui fournit à l'application les moyens de détecter des fautes temporelles comme le dépassement d'une échéance relative entre une requête de transmission et la fin réussie de la transmission.

De même, une application peut détecter l'absence de réception au bout de la période spécifiée pour une I-PDU de type périodique. Enfin, un mécanisme est disponible pour éviter à un nœud d'émettre en dehors de sa spécification (par exemple, en raison d'un disfonctionnement de l'application) en imposant un délai minimum configurable entre deux émissions ce qui contribue à une fonction de gardien de bus lorsque le protocole sous-jacent n'offre pas ce service.

10.3.3. *OSEK/VDX Fault Tolerant Communication*

OSEK/VDX Communication [OSE 01] est prévu plutôt pour s'exécuter sur une infrastructure de communication reposant sur des réseaux dont le comportement est guidé par les événements. Il apporte aux applications, ainsi que nous l'avons signalé ci-dessus, des services pour assurer la sûreté de fonctionnement du système, services qui ne sont pas fournis par le protocole sous-jacent. OSEK/VDX Fault Tolerant Communication (noté généralement, OSEK/VDX FTCom) offre, quant à lui, des services de haut niveau dans le cas où les protocoles sous-jacents sont FlexRay ou TTP/C.

Outre la gestion des messages applicatifs, il fournit des mécanismes de synchronisation d'horloges, la gestion de l'image des stations vivantes sur le réseau et des services d'initialisation du système.

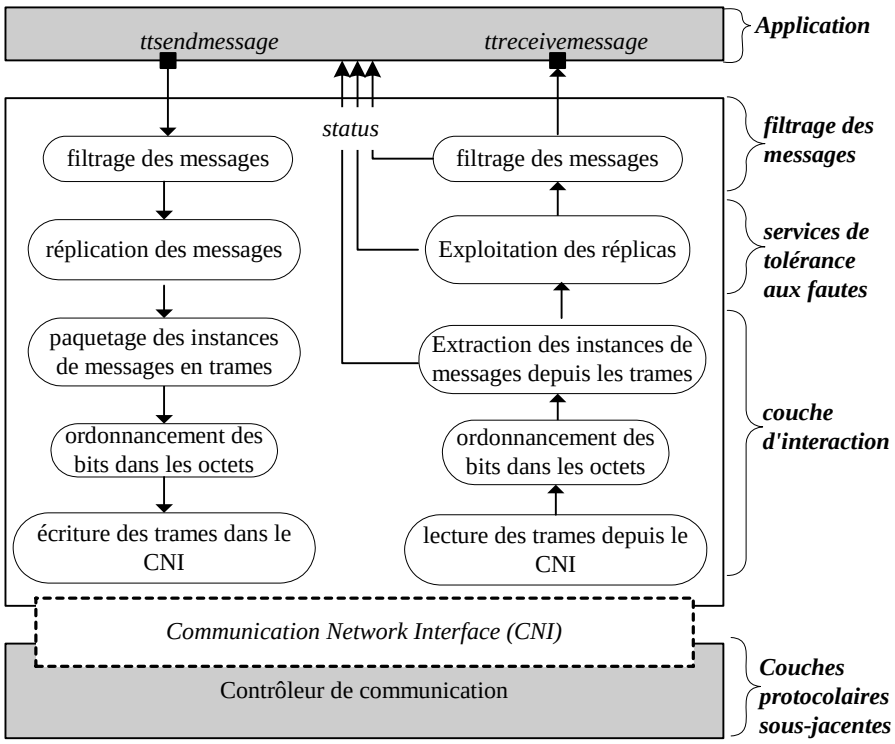


Figure 10.8. *Architecture des services de OSEK/VDX Fault Tolerant Communication*

OSEK/VDX FT Com est structuré en plusieurs couches (voir figure 10.8). La « couche d'interaction » se charge de la communication de et vers le CNI, c'est-à-dire la mémoire partagée par le micro-contrôleur et le contrôleur de communication (voir paragraphe 10.2.3.). Elle assure le paquetage et l'extraction des messages dans les trames en fonction de stratégies spécifiées hors ligne ainsi que l'ordre des bits selon les caractéristiques du processeur local. La couche de « tolérance aux fautes » fournit des services pour gérer la redondance lors des échanges. D'une part, une instance d'un message peut être transmise sur les deux canaux, et ce dans des trames de structures différentes. D'autre part, chaque nœud appartenant à la même unité FTU élabore une instance du même message ; chaque instance est transmise dans le *slot* réservé au nœud et le mécanisme précédent peut également s'appliquer. Par exemple, sur la figure 10.9., deux nœuds N1 et N2 composent une FTU. Ils élaborent tous deux une instance du message *m* et la transmettent sur les deux canaux, dans leurs *slots* réservés, selon deux stratégies de paquetage en trame

différentes. Quatre valeurs de *m* seront donc reçues sur les nœuds récepteurs : au sein des trames *N1_canal1*, *N1_canal2* transmises dans le *slot* réservé au nœud N1, *N2_canal1* et *N2_canal2* transmises dans le *slot* réservé au nœud N2. Une des fonctionnalités de OSEK/VDX FTCom est alors de ne présenter qu'une seule instance en appliquant un algorithme de sélection prédéfini hors ligne, appelé *Replica Determinate Agreement*. Plusieurs stratégies peuvent être déployées pour ce faire : utiliser n'importe quelle valeur parmi celles transmise (applicable uniquement si les calculateurs sont prouvés silencieux sur défaillance), appliquer un vote majoritaire, évaluer la moyenne, etc. Notons que ce service facilite la portabilité des applications réceptrices et les rend indépendantes du niveau et du type de redondance utilisée.

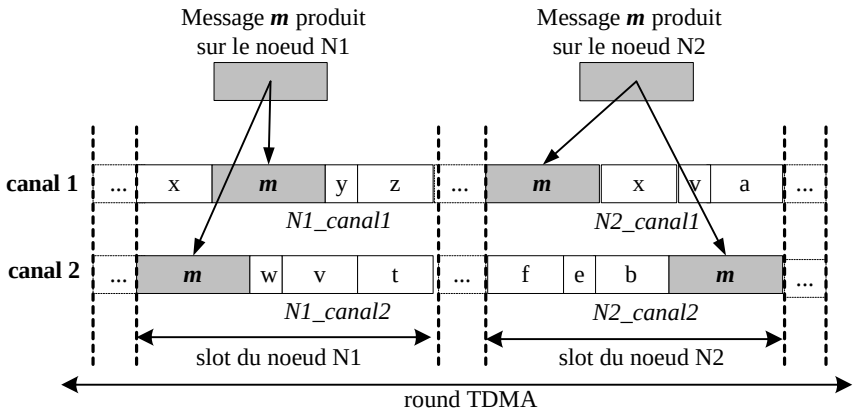


Figure 10.9. *Exemple d'utilisation de la redondance. N1 et N2 sont deux nœuds appartenant à la même FTU ; ils produisent tous deux une instance du message m, transmise sur les deux canaux.*

Enfin, lorsque le système d'exploitation gérant les ressources sur un nœud assure un comportement des tâches guidé par le temps, comme le fait OSEK/VDX OS Time (la spécification de ce système est disponible sur http://www.osek-vdx.org/), l'ordonnancement des tâches est entièrement défini hors ligne dans une table d'ordonnancement ou *dispatcher table*. Cette table donne sur une période, appelée le *dispatcher round* l'instant, compté à partir du début de la période, de l'activation de chaque tâche de l'application. OSEK/VDX FTCom fournit un service pour permettre au système d'exploitation de synchroniser le début de l'exploitation de la table d'ordonnancement sur un point particulier du *round* TDMA. Notons que ce service permet également de synchroniser les applications sur des nœuds différents. La figure 10.10. illustre ce service. L'exploitation de la table d'ordonnancement se fait après la réception de la trame située dans le *slot* s_3. Il est facile de voir sur cet

exemple que le *dispatcher round* doit être obligatoirement un multiple du *round* TDMA. Ce service de synchronisation nécessite que le système d'exploitation, le *round TDMA* et l'intergiciel soient configurés en cohérence statiquement au préalable.

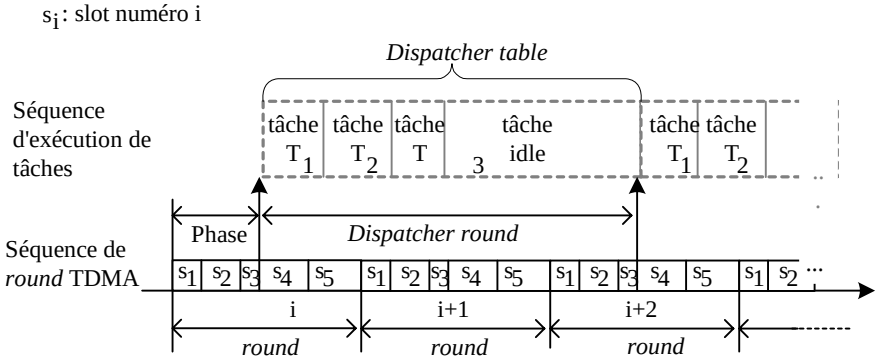


Figure 10.10. *Synchronisation des tâches assurée par OSEK/VDX FTCom*

10.3.3. *Configuration d'intergiciels*

Les intergiciels présentés ci-dessus nécessitent tous deux une configuration statique hors ligne des données, tables, *buffers*, etc., qui seront exploités par les différents services qu'ils fournissent et, en particulier, qui seront cohérentes avec les stratégies de paquetage de signaux en trames, de gestion de la redondance ou de filtrage choisis. Aussi, la génération de ces intergiciels doit inclure, d'une part, la spécification de ces données et tables et, d'autre part, la génération des composants logiciels réalisant ces services ainsi que la configuration des tâches les exécutant. Un souci majeur dans le contexte des systèmes embarqués dans l'automobile est d'obtenir un code optimisé en termes d'occupation processeur et mémoire ainsi qu'un motif de paquetage des trames optimisant, par exemple, la bande passante libre sur le réseau tout garantissant les propriétés de sûreté exigées par l'application, en particulier les propriétés temporelles exprimées au niveau des échanges de signaux entre applications distantes. La génération d'intergiciels sûrs et optimisés passe inévitablement par le paquetage optimal pour un réseau donné. Il s'agit, par exemple, sur CAN, de déterminer sur chaque nœud la répartition des signaux dans les trames émises par ce nœud et d'affecter, d'une part, une priorité et, d'autre part, une période de transmission à chaque trame définie sur le réseau. Ce problème est un problème d'optimisation (minimisation de la bande passante consommée, minimisation des temps de réponse des trames les moins prioritaires, etc.) sous

contraintes (respect des propriétés temporelles sur les signaux et donc sur les trames les transportant). Il peut être rapproché du problème de *bin packing* qui fait partie de la classe des problèmes NP-complets et demande donc le développement d'heuristiques (voir [NOR 00, SAN 03] pour une présentation d'heuristiques applicables à ce domaine). Notons que les produits issus de la proposition d'intergiciel Volcano [RAJ 02] proposent une fonction de paquetage de trames sous des heuristiques non publiques. Un problème similaire se pose sur un réseau à comportement guidé par le temps. Dans ce cas, le problème de la spécification d'un paquetage optimal s'ajoute à celui de la définition de la longueur du *round* et à celle de la spécification des *slots* réservés à chaque nœud et de leur taille. Dans ce cas, il s'agit également d'un problème d'optimisation sous contraintes où le critère à maximiser pourrait être la sûreté de la transmission de signaux sous des hypothèses d'occurrences de fautes pouvant affecter la communication ; la répartition des *slots* entre les différents nœuds, en particulier entre les nœuds redondants au sein d'une unité tolérante aux fautes (FTU) a un effet significatif sur la garantie qu'un signal intègre sera consommée sur un nœud récepteur (voir [GAU 03] pour plus de détails sur les algorithmes de configuration de *slots* dans un réseau de type TDMA.

10.4. Sûreté de fonctionnement et architectures de communication embarquées dans l'automobile

Garantir la sécurité des systèmes embarqués à logiciels prépondérants devient actuellement un challenge de première importance. La plupart des fonctions au sein d'une automobile sont assurées complètement ou partiellement par des systèmes implantés sous forme de composants logiciels déployés le plus souvent sur des architectures distribuées. Pour des raisons économiques et pour l'amélioration que cette technologie permet en termes de confort, qualité et flexibilité des prestations, personnalisation des véhicules, il n'y a à l'heure actuelle aucun retour en arrière raisonnablement envisageable et cette tendance s'accroîtra encore avec l'émergence prochaine des systèmes *X-by-Wire*. Cette explosion de l'électronique embarquée doit être confrontée à la confiance qui peut lui être accordée. Il s'agit donc de savoir prouver que la sécurité du véhicule est au moins égale, voire plus, à celle qui serait assurée par des technologies mécaniques et / ou hydrauliques. Ce problème n'est pas simple à résoudre. En effet, si depuis longtemps, on sait évaluer la fiabilité d'une pièce mécanique en mouvement, d'un dispositif électrique ainsi que de tout système intégrant ces types de composants, il n'en est pas de même pour les systèmes informatiques. Notons que d'autres domaines critiques (centrales nucléaires, transports ferroviaires, avionique) imposent des exigences de sécurité très strictes sur les systèmes intégrant des technologies logicielles et assurant des fonctions critiques. En particulier, ces systèmes relèvent de réglementations internationales et doivent passer sous les fourches caudines d'un rigoureux processus de certification dont le but est de garantir leur conformité à des normes spécifiques. Par exemple, la

norme RTCA/DO-178B [RTC 92] utilisée en avionique ou la norme EN 50128 [CEN 01] appliquée dans le domaine ferroviaire fournissent des directives très précises pour le développement des systèmes, imposant des contraintes sur le cycle de développement (étapes, processus de validation, utilisation de méthodes formelles, documentation, etc.) aussi bien que l'obligation d'appliquer certaines solutions comme le partitionnement des logiciels entre critiques et non critiques, le développement de composants en plusieurs versions, l'utilisation de redondance active, le recours systématique à la redondance matérielle. Ces normes, en raison, d'une part, de leur lourdeur induisant un accroissement considérable du cycle de développement et, d'autre part, du coût, du poids et de l'encombrement dus à une redondance matérielle massive, sont totalement incompatibles avec le modèle économique du secteur automobile.

Il n'existe pas, à l'heure où nous écrivons, de normes dédiées à la sécurité des systèmes embarqués dans un véhicule. Un premier effort vers la réglementation vient du groupe MISRA (*Motor Industry Software Reliability Association*) qui fédère les principaux acteurs anglais du domaine automobile et qui propose un modèle semi-formel pour un développement de systèmes embarqués, contraint par des exigences de sécurité. Signalons également que la norme IEC 61508 [CEI 98] qui s'applique aux systèmes électriques/électroniques/programmables est, à l'heure actuelle, un bon candidat pour servir de base à une réglementation dans l'industrie automobile. Notons que cette norme introduit quatre niveaux de criticité pour les systèmes (niveaux S.I.L., *Safety Integrity Level*) qui imposent pour chaque système une évaluation de la probabilité que ce système atteigne un état défaillant en une heure de fonctionnement ainsi que la preuve que cette valeur est inférieure au seuil spécifié dans la norme pour ce système (par exemple, cette probabilité doit être inférieure à 10^{-8} pour un système du niveau S.I.L.4, c'est-à-dire du niveau le plus critique).

Nous pouvons citer également les recommandations issues du projet européen, Brite Euram III, « *X-by-Wire, Safety Related Fault Tolerant Systems in Vehicle* » [BRI 98], dont l'objectif principal était de définir un cadre de travail pour les futurs concepteurs de systèmes *X-by-Wire*. Parmi les résultats de ce projet, on peut noter certaines recommandations quantitatives comme l'obligation pour une unité tolérante aux fautes (FTU) de comporter quatre nœuds (deux nœuds dans le cas de silence sur défaillance) exécutant un protocole d'accord tolérant aux fautes byzantines ou de prouver que la probabilité de rencontrer n'importe quel mode de défaillance critique pour la sécurité du conducteur ne doit pas excéder 5.10^{10} par heure de fonctionnement. Enfin, des préconisations internes à certains constructeurs automobiles avancent une obligation de prouver que la probabilité d'atteindre une défaillance critique par heure de fonctionnement doit être inférieure à 10^{-9} pour un système embarqué et 10^{-7} pour le véhicule. Atteindre et évaluer le niveau de sûreté qui sera exigé par une future réglementation passe inévitablement par deux moyens

complémentaires. Le premier consiste à appliquer un processus sûr de développement (spécifications formelles, validation du logiciel produit, technologie de test, etc.) et à mettre en œuvre des techniques de prévision de fautes qui, en particulier lors de la phase de conception de l'architecture opérationnelle, permettent une évaluation qualitative (identification des modes de défaillances, identification des causes de défaillances, etc.) et quantitative (approche probabiliste d'évaluation d'occurrence de défaillance). Notons enfin, que si certains éléments de solutions existent [NAV 01, SIM 05], le problème général de mise en œuvre de technique d'évaluation quantitative de la sûreté reste ouvert.

L'impact du système de communication sur la sûreté se situe à plusieurs niveaux. Les systèmes embarqués sont inévitablement soumis à un environnement agressif : particules α, pics de températures, interférences électromagnétiques (EMI, *ElectroMagnetic Interferences*). En particulier, les perturbations de type EMI, dues à l'environnement électrique interne au véhicule ou à la présence de radars d'aéroport, de relais de télétransmission, de la foudre, etc., ont un effet significatif sur le comportement des réseaux. Les moyens pour limiter leurs effets passent par le blindage du médium ou, plus efficacement le recours à la fibre optique. Cette dernière solution est, notamment en raison de son coût actuel, peu diffusable dans l'automobile. Un autre moyen consiste à dupliquer le médium de communication. Ceci néanmoins induit un câblage, un coût et un poids supplémentaire ainsi que la mise en œuvre de la gestion de la redondance des échanges. Enfin, toujours sur le même plan, afin de permettre la détection d'erreurs dans les transmissions, les protocoles intègrent systématiquement la mise en œuvre d'un code cyclique redondant ou CRC dont la distance de Hamming garantit un taux d'erreurs détectables. Si ce taux apparaît insuffisant pour les impératifs de sûreté imposés, il faut alors insérer dans les données utiles un CRC complémentaire géré au-dessus du protocole.

Un comportement d'application guidé par le temps permet naturellement une garantie des propriétés de sûreté qui s'expriment sous forme de propriétés temporelles (échéances sur les tâches, sur les messages, fraîcheur des données consommées, bornes sur les temps de réponse de bout en bout, borne sur les gigues d'émission, etc.) ; en effet, ainsi que nous l'avons montré au paragraphe 10.2.2, ce mode de conception assure le déterminisme de l'accès au médium de communication ainsi que la détection de nœuds défaillants. Ceci explique les nombreux travaux engagés sur la définition de protocoles guidés par le temps pour les applications critiques de l'automobile. De plus, garantir des propriétés d'interopérabilité temporelle à un ensemble de fonctions distantes qui coopèrent, passe par l'accès à une base de temps commune et stable. Pour ce faire, il est nécessaire de disposer de mécanismes de synchronisation des horloges locales dont la qualité et les performances doivent être soigneusement évaluées. Les protocoles guidés par le temps fournissent naturellement ce service. Un nœud qui, par suite

d'une défaillance locale, émet en dehors de sa spécification relève d'un comportement dit *babbling-idiot*; l'introduction d'un composant dit gardien de bus prévient une telle station de perturber le réseau. Idéalement, ce composant ne devrait avoir aucun mode de défaillance commun avec le nœud ; néanmoins, pour des raisons de coût, ceci est difficilement réalisable. Enfin, afin d'assurer que plusieurs fonctions distantes ont la même vision de l'état du système et de son environnement et appliquent ainsi des actions cohérentes, il faut garantir les propriétés de cohérence spatiale des informations échangées et pour ceci, disposer d'un mécanisme de diffusion atomique fiable. Ce service doit être complété par un service d'acquittement qui permet à un émetteur, aussi bien qu'à d'autres nœuds sur le réseau de détecter qu'une information n'a pas été reçue correctement par un nœud récepteur. Des algorithmes de consensus doivent, alors, généralement être implémentés au-dessus du protocoles (voir [RUS 02], [LIM 02]).

Au-dessus de la couche protocolaire d'accès au médium de communication, un certain nombre de services offerts par certains protocoles ou certains intergiciels (voir sections 10.2 et 10.3) facilite la gestion de la tolérance aux fautes au niveau applicatif. Le premier qui est, sous certaines hypothèses (voir 10.3.2), assuré par TTP/C est le service de *membership* qui permet à tout nœud opérationnel sur le réseau d'avoir une vision commune de l'état de tous les autres nœuds. Lorsque la redondance est appliqué, des services, comme le *Message Agreement* fourni par l'intergiciel OSEK/VDX FTComm décharge l'application de la gestion des répliquas reçus. Enfin, la mise en œuvre locale de mécanismes assurant que le nœud n'émet pas sur le réseau en cas de défaillance locale garantit l'hypothèse dite de silence sur défaillance (*fail silence*) sous laquelle la correction et l'efficacité des mécanismes cités ci-dessus sont assurées. Enfin, la tolérance aux fautes qui peut être implantée au niveau de l'application passe généralement par une identification hors ligne des modes de marche en fonction de la présence de défaillances reconnues. Notons que cette notion de mode de marche s'étend également à des modes d'utilisation du véhicule (phase de calibration, téléchargement, arrêt, initialisation au démarrage, en marche, etc.). Il s'agit, dans tous les cas, d'exécuter en ligne à l'occurrence d'une défaillance ou à un changement demandé explicitement, une transition entre modes de marche afin d'atteindre un état connu et maîtrisable (recouvrement d'erreur par poursuite). Néanmoins, toute situation détectée localement doit être connue et partagée par tous les nœuds opérationnels du réseau. Notons que pour atteindre cet objectif, un protocole comme TTP/C (voir paragraphe 10.3.2.) inclut un service permettant une détection de demande de commutation de mode de marche par tous les nœuds à la fin du *slot* portant cette demande (prédictibilité du temps de détection).

En conclusion, tout réseau susceptible à être embarqué dans l'automobile doit être soigneusement étudié en fonction des propriétés de sûreté exigées du système. Dans ce contexte, certains protocoles offrent de base de puissants services ; cela

peut se faire au détriment de la bande passante utilisée (c'est le cas pour les réseaux guidés par le temps). Lorsque ces services ne sont pas offerts au niveau du réseau, ils doivent être, s'ils sont jugés nécessaires, au niveau des couches supérieures au prix d'une implantation qui peut se révéler moins performante. Leur intégration dans un intergiciel permet alors de garantir le niveau de sûreté requis tout en favorisant la portabilité de composants développés séparément.

10.5. Conclusion

Le nombre de données échangées entre des calculateurs au sein d'une automobile est toujours croissant. Ceci a incité au développement de plusieurs réseaux et protocoles pour ce domaine applicatif. Néanmoins, les systèmes de communication sont des équipements particulièrement sensibles et leurs caractéristiques doivent être soigneusement analysées avant de les intégrer. En effet, si une automobile se doit de garantir à son conducteur, ses passagers et son environnement un niveau de sécurité important, il n'est pas possible, contrairement à ce qui se pratique dans l'avionique, d'assurer cette sécurité au prix d'une redondance élevée et les concepteurs de systèmes embarqués ont toujours à faire face à des choix relevant de critères qui sont souvent antagonistes, à savoir, la sûreté, le coût et l'encombrement. De plus, l'impact des réseaux sur le processus de développement des systèmes est important car ils apparaissent souvent comme le noyau d'intégration entre des composants développés par différents acteurs. Ceci amènera dans les années à venir à développer des approches systèmes permettant une intégration totalement maîtrisée, c'est-à-dire assurant les propriétés globales de sûreté tout en préservant la propriété intellectuelle de chaque fournisseur et en définissant précisément les limites de leurs responsabilités.

10.6. Bibliographie

[ALB 04] A. Albert, « Comparison of Event-Triggered and Time-Triggered Concepts with Regards to Distributed Control Systems », *Embedded World 2004*, Nuremberg, Allemagne, février 2004.

[TTT 03] TTTech Computertechnik GmbH, Time-Triggered Protocol TTP/C, High-Level Specification Document, Protocol Version 1.1, http://www.tttech.com, 2003.

[FLE 04] FlexRay Consortium, FlexRay Communication System, Protocol Specification, Version 2.0, http://www.flexray.com , juin 2004.

[MIL 05] D. Millinger, R. Nossal, « FlexRay CommunicationTechnology », *The Industrial Communication Technology Handbook*, CRC Press, Taylor & Francis, éditeur R. Zurawski, ISBN 0-8493-3077-7, janvier 2005.

[ISO 94a] International Standard Organization, ISO 11519-2, Road Vehicles - Low Speed serial data communication - Part 2: Low Speed Controller Area Network, ISO, 1994.

[ISO 94b] International Standard Organization, ISO 11898, Road Vehicles - Interchange of Digital Information - Controller Area Network for high-speed Communication, ISO, 1994.

[LIN 03] LIN Consortium, LIN Specification Package, revision 2.0, http://www.lin-subbus.org/, septembre 2003.

[RAJ 05] A. Rajnák, « The LIN Standard », *The Industrial Communication Technology Handbook*, CRC Press, Taylor & Francis, Editeur R. Zurawski, ISBN 0-8493-3077-7, janvier 2005.

[KOP 02] H. Kopetz et al., Specification of the TTP/A Protocol, université technologique de Vienne, Autriche, septembre 2002.

[MOS 04] MOST Cooperation, MOST Specification Revision 2.3, http://www.mostnet.de, août 2004.

[FER 02] J. Ferreira, P. Pedreiras, L. Almeida, J.A. Fonseca, « The FTT-CAN protocol for flexibility in safety-critical systems », *IEEE Micro, Special Issue on Critical Embedded Automotive Networks*, vol. 22, n°. 4, p. 46–55, juillet-août 2002.

[KOO 02] P. Koopman, « Critical embedded automotive networks », *IEEE Micro, Special Issue on Critical Embedded Automotive Networks*, vol. 22, n°. 4, p. 14–18, juillet-août 2002.

[TIN 95] K. Tindell, A. Burns, A.J. Wellings, « Calculating controller area network (CAN) message response times », *Control Engineering Practice*, vol. 3, n°. 8, p. 1163–1169, 1995.

[NAV 01] N. Navet, Y.-Q. Song, « Validation of real-time in-vehicle applications », *Computers in Industry*, vol. 46, n°. 2, p. 107–122, novembre 2001.

[ISO 94c] International Standard Organization, ISO 11519-3, Road Vehicles - Low Speed serial data communication - Part 3: Vehicle area network (VAN), ISO, 1994.

[SAE 96] Society of Automotive Engineers, Class B data communications network interface - SAE J1850 standard, novembre 1996.

[ISO 00] International Standard Organization, 11898-4, Road Vehicles - Controller Area Network (CAN) - Part 4: Time-Triggered Communication, ISO, 2000.

[BAU 00] G. Bauer, M. Paulitsch, « An investigation of membership and clique avoidance in TTP/C », *Proceedings 19th IEEE Symposium on Reliable Distributed Systems*, Nuremberg, Allemagne, 2000.

[PFE 00] H. Pfeifer, « Formal verification of the TTP group membership algorithm », *Proceedings FORTE/PSTV 2000*, Pise, Italie, 2000.

[FLE 04] FlexRay Consortium, FlexRay Communication System, Protocol Specification, Version 2.0, http://www.flexray.com, juin 2004.

[CAS 99] L. Casparsson, A. Rajnak, K. Tindell, P. Malmberg, Volcano - a revolution in on-board communications, Technical Report 98-12-10, Volvo, 1999.

[OSE 04] OSEK Consortium, OSEK/VDX Communication, V.3.0.3, http://www.osek-vdx.org/, juillet 2004.

[OSE 01] OSEK Consortium, OSEK/VDX Fault-Tolerant Communication, V.1.0, http://www.osek-vdx.org/, juillet 2001.

[NOR 00] C. Norström, K. Sandström, M. Ahlmark, Frame packing in real-time communication, Technical Report 00-07-25, Mälardalen Real-Time Research Center, Suède, 2000.

[SAN 03] R. Santos Marques, N. Navet, F. Simonot-Lion, « Frame packing under real-time constraints », *Proceedings 5th IFAC International Conference on Fieldbus Systems and their Applications - FeT'2003*, p. 185–192, Aveiro, Portugal, juillet 2003.

[RAJ 02] A. Rajnák, M. Ramnefors, « The Volcano communication concept », *Proceedings Convergence 2002*, Detroit, Michigan, USA, 2002.

[GAU 03] B. Gaujal, N. Navet, « Optimal replica allocation for TTP/C based systems », *Proceedings 5th IFAC International Conference on Fieldbus Systems and their Applications - FeT'2003*, Aveiro, Portugal, juillet 2003. Version étendue sous le titre « Maximizing the Robustness of TDMA Networks with Applications to TTP/C » dans *Real-Time Systems*, vol. 31, n°.1-3, p. 5-31, décembre 2005.

[BRI 98] Brite-Euram, X-by-wire - safety related fault tolerant systems in vehicles. Technical report, Brite-EuRam III Program, octobre 1998.

[CEI 98] CEI, Sécurité fonctionnelle des systèmes électriques / électroniques / électroniques programmables relatifs à la sécurité - Partie 1 : Prescriptions générales. CEI/IEC 61508-1 :1998, 1998.

[CEN 01] CENELEC, Railway applications-safety related electronic systems for signalling, 2001.

[RTC 92] RTCA-Inc., DO-178B : Software Considerations in Airborne Systems and Equipment Certification, 1992.

[RUS 02] J. Rushby, « An overview of formal verification for the time-triggered architecture », *Proceedings Formal Techniques in Real-Time and Fault-Tolerant Systems*, p. 83–105, 2002.

[LIM 02] G. Lima, A. Burns, « Timing-independent safety on top of CAN », *Proceedings 1st International Workshop on Real-Time LANs in the Internet Age*, 2002.

[SIM 05] F. Simonot, F. Simonot-Lion, Y.-Q.Song, « Dependability Evaluation of Real-Time Applications Distributed on TDMA-Based Networks », *Proceedings 6th IFAC International Conference on Fieldbus Systems and their Applications, FeT'2005*, Puebla, Mexico, novembre 2005.