社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－OS－100 (10)
2005／8／3

Ethernet LAN を用いた分散リアルタイム制御機構における通信方式

兵頭和樹† 田胡和哉† 松下温†

近年，ロボットや自動車などの分野で制御技術の発展が著しい．これらの機械系は，多数のアクチュエータやセンサを，個々に実時間制御する必要があり，複数のプロセッサで実現される．また，配線等の理由で分散構成をとることが望ましい．一方，全体姿勢の制御や学習による最適化処理など，多数のモジュールの状況を総合して判断を下す必要がある．したがって，制御系全体では，きわめて複雑な実時間分散処理系が構成される．これに対応して，これらを制御するアルゴリズムを実際に実装するための基盤となるオペレーティングシステムの必要性が高まっている．本稿では，複雑な機械系の制御を目的としたオペレーティングシステム NOAH について述べるとともに，通信方式に焦点を充て，分散リアルタイム制御機構のための通信方式を検討する．

A Communication Mechanism on Ethernet LAN for Distributed Real-Time Control Systems

KAZUKI HYOUDOU,† KAZUYA TAGO† and YUTAKA MATSUSHITA†

A new approach toward building real-time distributed operating systems for controlling complicated mechanical systems such as humanoid robots is proposed in this paper. Robots are equipped with number of actuators and sensors, and these components are connected to local processors to attain better response and simpler physical structure. On the other hand, controlling robots requires globally integrated operation over all actuators, and this leads to need of new basis for building the distributed control system. We build control systems from a set of routines driven by the exchanged data among them. The design of NOAH system that is being build by the approach is stated.

1. はじめに

近年，ロボット，自動車，工作用機械等の分野において制御技術の発展が著しい．たとえば，2足歩行ロボットも多数製作され，歩行制御に関する知見も蓄積されつつある．これらの機械系は，多数のアクチュエータやセンサを持つこと，それら個々の制御にリアルタイム性が要求されることなどにより複数のプロセッサで実現される必要がある．また，配線などの物理的制約から，少数の制御装置による集中的な制御方法ではなく，制御装置とセンサやアクチュエータを 1 つのモジュールとし，それら個々のモジュールがネットワークによって結合された分散構成をとることが望ましい．これにより，各モジュール間は一般的なネットワークケーブルのみで接続でき，配線の簡略化およびインタフェースの統一などが可能となる．一方において，ロボットの全体姿勢の制御や，学習による最適化処理では，多数のアクチュエータの状況やセンサの入力を総合して判断を下す必要がある．したがって，制御系全体では，きわめて複雑な分散リアルタイム処理系が構成されることになる．これに対して，このような制御アルゴリズムを実際に実装するための基盤となるオペレーティングシステムの必要性が高まっている．オペレーティングシステムの要素技術や全体構成の方法を検討，実現することにより，制御ソフトウェアの開発の容易化，より複雑かつ高速な制御系の実現，ハードウェアコストの軽減等を図ることができる．

本稿では，我々が提案している，複雑な機械系の制御を目的としたオペレーティングシステム NOAH[2] について述べるとともに，通信方式に焦点を充て，分散リアルタイム制御のための通信方式を検討する．本研究では，特に Ethernet LAN を用いて分散リアルタイム制御機構を構成することを考える．制御機構における通信にはリアルタイム性が要求されるため，従来は専用の通信媒体が利用されることが多かったが，コスト等の観点から汎用の通信媒体を利用することの利点は大きい．

† 東京工科大学コンピュータサイエンス学部
Department of Computer Science, Tokyo University of Technology

本稿では，リアルタイム通信のための特別なプロトコルを設計するのではなく，ネットワークの状態を管理しパケット消失を防ぐことでリアルタイム性を確保する方式を提案する．利用者によってあらかじめ設定された通信計画に基づいてネットワーク状態を管理することで，パケット消失の発生を防ぐとともに効率の良い流量制御が可能となる．

2. NOAH の概要

NOAH は，分散リアルタイム処理系を構成する複雑な機械系の制御を目的としたオペレーティングシステムである．NOAH では，リアルタイム制御プログラムを，相互に同期をとりながら実行されるプロセス (タスク) の集合ではなく，データの流れによって駆動される関数 (以下，コールバック関数) の集合として記述する．コールバック関数は，実行の途中で同期命令によって中断することがなく，連続して実行され，単位となる処理を完了する．また，我々は分散環境でこのような論理構造のプログラムを実行するのに適したアプリケーションプロトコルを設計し，提案する方式に適用する．

これらの方策により，サーボ系等の，繰り返し計算を行うアプリケーションプログラムを多数相互に通信で結合した系を構築するにあたり，従来のプロセスを用いる方法に比較して，オーバヘッドの削減，アプリケーションプログラムの記述の容易化，大域的な同期の実現の容易化などを図ることができるようになる．

本章では，NOAH が対象とするシステム，NOAH に必要とされる機能と性能について述べる．また，それらに基づく NOAH の設計についても述べる．

2.1 対象となるシステム

はじめに，ここで対象とするシステムのハードウェア上の条件つについて述べる．NOAH は，図 1 に示すようなシステムに適用することを想定している．

対象となるシステムでは，アクチュエータの各々に個別に制御装置を付加し，これを統一的な方法でネットワークに接続できるようにする．装置内部のネットワークは，専用の通信線を用いるのではなく，Ethernet 等の LAN において一般に利用されている通信線を用いる．

ロボットを例に説明すると，通常，モータがアクチュエータとなる．さらに，モータの各々に制御装置を付加し，モータ，センサと一体構造のアクチュエータとすることによって，ネットワークに直接接続できるようになる．このような，制御機構を持ったアクチュエータをネットワークトアクチュエータ (Networked Actuator) とよぶことにする．従来，アクチュエータやセンサの接続信号は個々まちまちで，統一的な方法でシステムを構成することが困難であった．ネットワークトアクチュエータ構造を用いることにより，どのようなアクチュエータやセンサであっても，統一的な方法でシステムに組み込むことが可能になる．これは，汎用 PC の周辺機器の接続を USB によって統一化することによって，飛躍的に接続が容易になったことと類似である．ネットワークトアクチュエータ構成を実現するための技術課題として，単一のプロセッサで複数のモータを制御する，より集中的なハードウェア構成のシステムと比較して，さらに高度な分散リアルタイム制御機構が必要となることがあげられる．

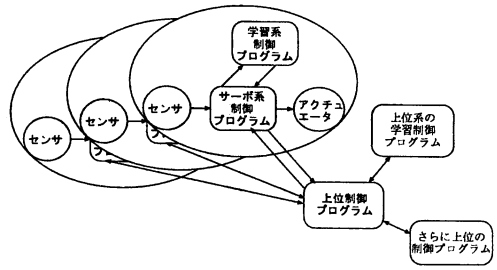

図 1 対象となる制御系の例
Fig. 1 Example of a target control system.

2.2 必要とされる機能と性能

複雑な機械の制御機構向きオペレーティングシステムに求められる特性として，

(1) 他の制御プログラムとの情報交換のオーバヘッドによって，制御周期を満たせなくなることがないこと，
(2) 多くのプロセッサにある制御プログラムからの情報を統合して制御に反映するためのプログラムが容易に構築できること，
(3) 大域的な時間同期，すなわち，多くのプロセッサにある多数の制御プログラム間での動作時間の同期が正確に行えること，
(4) ハードウェアを用いて実現された制御系と容易に連係できること，

が挙げられる．(1) は，制御プログラム間の連携に関する end-to-end のリアルタイム性の要請であり，さらに，

(1.1) アプリケーションソフトウェア自体の通信処理が効率良く記述できること，
(1.2) システムの通信処理が効率良く実現されており，アプリケーションソフトウェアに与える影響が少ないこと，

(1.3) スケジューリングが正確に行われること，

に分割される．リアルタイム性の本来の定義は，デッドラインのスケジューリングが正確に行えることにあるが，実際の制御系に適用するためには，これに止まらず，オーバヘッド自体が小さい必要がある．

多入力の制御系を実現するにおいて注意すべき点は，API のオーバヘッドと情報伝達のスキューである．たとえば，それぞれ個別のプロセッサに接続された多数のセンサからの情報を入力とする制御プログラムを実装する場合に，制御プログラムが個々のセンサ情報を個別の API を用いて順番に受信したのでは，オーバヘッドが大きくなるとともに，異なるセンサからの情報間で時間のずれが大きくなってしまう．同一時刻に関する情報を同時に複数受信できる機構が必要となる．

また，このような系における同期は，イベントの伝達に使用される．たとえば，二足で歩行するロボットでは，足が地面に接地している場合には機械系全体が閉リンク系をなし，地面から足が離れている場合は開リンク系をなすために，制御アルゴリズムが大きく異なる．そのため，足の接地ごとにそのイベントを関連する関節の制御系すべてに伝達し，同期して制御モードを切替える必要がある．この切替えがスムーズに行われるためには，複数の制御アプリケーションへのイベント伝達のばらつきが少ない方がよい．以下，このばらつき時間の平均を，同期スキューと呼ぶ．ここで求められる特性の一つは，同期スキューが小さいことである．たとえばロボットの制御において，1 ミリ秒間隔での制御ループの実行を実現するためには，最低でも一桁小さい 100 マイクロ秒程度の同期スキューが必要になると考えられる．

全体として，ここで述べた性質を持つソフトウェアシステムを，2.1 で述べた条件のハードウェア上で実現することが NOAH の設計目標となる．

2.3 NOAH の設計

図 2 に，典型的なシステムの構成例を示す．この図においては，管理ノードがルータの役割を兼ねている．管理ノードは汎用プロセッサと Linux オペレーティングシステムで構成され，NOAH と同等の機能を持つソフトウェアが Java を用いて記述されたミドルウェアの形式で実行される．対象とするネットワークは単一のセグメントを構成しており，単一のブロードキャストパケットをすべてのノードが受信できることを想定している．

NOAH では，制御システムをデータ駆動型の構造を用いて構成する．制御プログラム間で交換される個々のデータに，システムが認識できるメタ情報を付加し

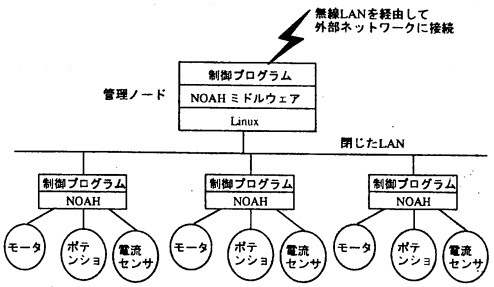

図 2 対象システムの構造例

Fig. 2 Example of a construction of target system.

てセグメントとして統一的に扱い，制御プログラムはスケジュールされるたびに実行が完了する関数として記述する．制御プログラムがスケジュールされる際，実行に必要なセグメントをすべて揃えて渡し，実行の途中で通信を行わない．制御プログラムを繰り返し実行する際には，個々のセグメントは制御プログラムが陽に解放するまで同一のもが使用される．

2.3.1 セグメント

セグメントは，64k バイトを上限とするユーザデータ領域であり，セグメント ID，サイズ，生成時刻，実行終了指定時間がメタ情報として付加されている．セグメント ID は 32 ビットの番号で，システム全体で一意に決められている．実行終了指定時間 (デッドライン) は，このセグメントを用いて行われるコールバック関数の処理が終了する時間を指定している．

ある時点で 1 つのコールバック関数によって生成されたセグメントの内容を，セグメントイメージと呼ぶ．セグメントイメージは，セグメント ID と生成時刻によって識別され，生成されて以後，その内容が書き換えられることはない．

2.3.2 コールバック関数

制御プログラムは，コールバック関数として実行される．コールバック関数は，コールバック関数の識別子，起動時に必要とされるセグメント ID のリスト，起動条件によって定義される．

同一のセグメントは，NOAH がネットワークを経由して自動的に転送することにより，それを必要とする，システム中の任意のノード上のコールバック関数間で共有することができる．すなわち，同一のセグメントイメージの複製が複数のノード上に存在する．

コールバック関数は，あらかじめ指定したセグメント ID のセグメントイメージが新たに生成されることによって起動される．コールバック関数が起動時に必要とするセグメントの ID の中から，1 個以上の ID

を起動条件としておき，そのIDを持つセグメントイメージがいずれかのノードで新規に生成され，アクセス可能になった時点でコールバック関数を起動する．

2.3.3 スケジューリング

NOAHは，特別なセグメントとして，タイマセグメントを，指定された周期で定期的に生成する機能を持つ．これによってコールバック関数を定期的に実行することができる．

制御プログラムの実行過程における，相互排除，シグナリング等のプログラム同期も，セグメントを用いることによって実現される．コールバック関数は，ノンプリエンプティブに実行されるので，実際には小粒度の相互排除を陽に記述する必要性は少ない．

セグメントのメタ情報として，コールバック関数の実行を完了させたい時刻を指定するとができる．これは，複数のノードにあるコールバック関数の実行の時間同期をとる上で，重要な機能である．

3. 通信方式の検討

本章では，NOAHの対象となるシステムの通信部分に焦点を充て，どのような通信が要求されるかを考察し，分散リアルタイム制御のための通信フレームワークを検討する．

3.1 対象となるシステム

NOAHが対象とするシステムでは，センサやアクチュエータの個々に制御装置を付加した，ネットワークアクチュエータ(NA)を想定している．システム全体が協調して動作し複雑な運動を実現するためには，ネットワークを介して各NAの状態や各種センサ情報を共有する必要がある．具体的には，フィードバック制御に必要な情報をネットワークを介して各NAに供給することを考えている．また，2.3で述べたように，対象となるネットワークは単一のセグメントを構成しており，単一のブロードキャストパケットをすべてのノードが受信できることを想定している．

装置内部のネットワークにはEthernet LANを利用することを考える．制御機器を相互接続する通信媒体として専用の規格が用いられてきたが，コスト等の観点から汎用の通信媒体を利用することの利点は大きい．専用の通信媒体の最も重要な機能として，ハードウェアレベルでのエラー検出，訂正機能が挙げられる．これに対して，インターネットプロトコルでは通信媒体によるエラー検出を仮定していない．代わりに，TCPプロトコルにみられるように，受信側から送信側にACKパケットを返送することで信頼性のある通信を実現している．しかしながら，ACKパケットを返送する方式では，エラーの検出，パケット再送の遅延のためにend-to-endの平均遅延が大きくなりやすいこと，マルチキャストを実現する際の受信タイミングのずれが大きくなることなどの問題がある．ここでは通信媒体の利用法を検討することによって，ハードウェアレベルでのエラー検出，訂正機能を持たない汎用の通信媒体を，分散リアルタイム制御に適用できるようにするフレームワークを考える．

3.2 要求される通信機能

これまでに述べてきた制御システムに必要とされる通信として，

(1) フィードバック制御に要求される通信
(2) イベント，例外通知に要求される通信
(3) バースト転送に要求される通信

が挙げられる．

サーボ系の制御では，周期的に制御プログラムを実行し，センサ等から得られる実際の制御量を基に微調整を行うフィードバック制御が行われる．(1)は，このフィードバック制御に必要なデータを，他のNAやセンサから周期的に得るための通信である．これらのデータは，緊急度は高くなくデータサイズも小さいが，制御周期を守って通信が行われる必要がある．また，これらのデータは周期的に与えられ更新され続けるので，フィードバック制御に致命的な問題を生じない限りはある一定の割合でデータが消失しても良いと考えられる．

一方，接触，転倒や妨害の発生などの突発的事象や機械の故障等を知らせる，イベント，例外通知の緊急度はかなり高い．これらのデータはサイズは最小であるが，可能な限り速く確実に届けられる緊急パケットとして定義される必要がある．

また，NOAHでは各NAに制御プログラムを動的にロードすること，制御結果などを基に学習制御を行うことを想定している．このためには，制御プログラムのバイナリコードや各NAが保持する制御ログを効率良く転送するためのバースト転送が必要である．これらのデータは，サイズは大きいが緊急度は低い．この通信にはリアルタイム性は要求されないが，比較的高いスループット，データ順序の保存，データ到着の信頼性が要求される．

実現する通信フレームワークでは，この3つのパターンの通信が互いを乱さないように共存して実現される必要がある．バースト転送によって，緊急パケットが阻害されたり，フィードバック制御のデータが制御周期を守れなくなってはならない．緊急パケットはどのような場合にも即時に送信されなければならない．

表 1 実験環境
Table 1 An experimental environment.

各ノードの仕様	
CPU	Intel Celeron 1.0 GHz
メモリ	SDR-SDRAM 128 MByte
NIC	Realtek RTL8139 搭載 Fast Ethernet
ネットワーク構成	
ノード数	4
スイッチ	コレガ CG-SW08GTP 8 ポート ギガスイッチ

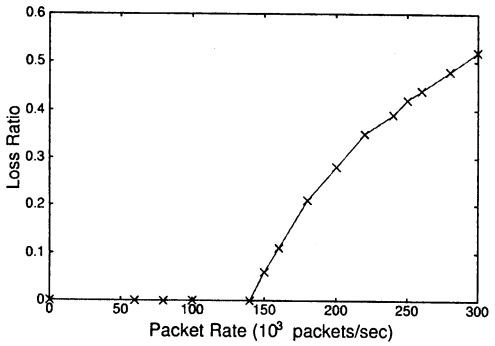

図 3 パケット数対ロス率 (パケットサイズ = 64 バイト)
Fig. 3 Packet rate vs. loss ratio (packet size = 64 bytes).

3.3 検 討 案

本節では，Ethernet LAN 等の汎用の通信媒体を用いて，前節で述べた要求を満たす通信フレームワークを実現するための方策を検討する．

3.3.1 基 本 方 針

3.1 で述べたように，リアルタイム通信に利用される専用の通信媒体の主な機能として，ハードウェアレベルでのエラー検出，訂正機能が挙げられる．データに冗長性を持たせ，受信データにエラーが検出された際にも受信側でのエラー訂正を行うことで可能な限り再送処理を排除し，高いリアルタイム性を実現しているのである．このことはデータの再送が発生した時点で通信のリアルタイム性を確保することが困難になることを示している．

一方，3.2 で述べたように，フィードバック制御では，周期的に新しいデータが到着し常に値が更新されるため，制御周期に遅れて到着したデータは無意味であり，致命的な問題が発生しない限りはデータが消失してもかまわない．このことは本質的に消失したデータの再送の必要がなく，データ消失の割合だけが問題になることを示している．

以上より，本研究ではリアルタイム通信のための特別なプロトコルを設計するのではなく，ネットワークの状態を管理しパケット消失を起こさないことでリアルタイム性を確保する方式を提案する．この方式には，通信プロトコルとしては UDP/IP などの従来のプロトコルをそのまま利用できるという利点がある．

図 3 は，Ethernet LAN におけるパケット消失の割合を調べるために行った予備実験の結果である．実験環境は表 1 の通りである．実験では，定期的に指定サイズのパケットを送信するプログラムを用いて，多対 1 や 1 対 1 などの通信パターンで通信線レベルでのパケット消失を調べた．パケット流量とパケット消失の割合の関係は通信パターンによらず一定であり，スイッチが処理しきれないパケットを破棄していると考えられる．ここで重要な点は，パケット流量が一定以下の時にパケット消失の割合が 0 になることである．このことは，汎用の通信媒体を用いても，ネットワーク状態の管理を工夫することで，分散リアルタイム制御に要求される通信を実現できることを示している．

なお，予備実験の結果は，パケットサイズを変更した際も，パケット消失が発生しはじめる時の流量こそ異るが，グラフの傾向は同様であった．また，その時の使用帯域は，パケットサイズ 64 バイトの時で約 70Mbps，512 バイトの時で 90Mbps であった．パケット消失の原因がネットワーク帯域の制限のみによるのではないということは注目すべき点である．これはネットワーク状態の管理が，使用帯域に基づく流量制限だけでは不十分であることを示している．

3.3.2 管理されたネットワーク

ネットワークの状態を管理する最も単純な方法としては，使用可能帯域をノード数で分割し，ラウンドロビンなどで各ノードが一定間隔毎に送信権を得るようにすることで流量制限する方法がある．しかしながら，この方法は通信パターンの偏りを無視しており，状況によってはかなりの無駄が生じることになる．

通信パターンを動的に検出し，実行時に最適化を行う方法も考えられるが，雑多な環境で通信パターンを事前に知ることが困難な通常の LAN と異り，ハードウェア構成が固定される制御システムの内部ネットワークでは，制御プログラムの持つ通信パターンも変化が少なく，通信パターンを事前に知ることはさほど難しくないと考えられる．本研究では，利用者があらかじめ通信パターンを記述し，実行時にはその設定された通信計画に基づいてネットワークを管理する方式を提案する．このように，通信媒体のハードウェア構成が固定され，利用者の通信計画に基づいてネットワーク状態を管理することで，パケット消失の発生を防ぐとともに効率の良い流量制御が可能となる．この状態を管理されたネットワーク (Managed Network) と呼ぶことにする．管理されたネットワークでは，利

用者が通信計画をあらかじめ記述することで，使用するネットワークが必要な性能を満たせるかどうかを事前に検証することも可能になると考えられる．

管理されたネットワークを実現するためには，適用する制御システムが要求する通信性能や制御プログラムの通信パターンなどの通信計画を表現するための記述様式が必要となる．通信計画を記述するために必要なパラメータや様式については現在検討中であり，今後の課題である．

3.3.3 緊急パケット

緊急パケットを実現するために，前節で述べた管理されたネットワークの流量制御において，常にある一定量の緊急パケットのための流量的余裕を残すことを考える．緊急パケットの処理はその他のパケットと完全に別扱いとすることで，送信側においては即時送信を，受信側においては即時の処理を可能とする．

この実現法では，ネットワークの使用状況によって通信媒体上での通信遅延が大きくなってしまうことが考えられる．これに対しては，通信スケジューリング等から最大通信遅延を予測できると考えられる．常に最短の時間で緊急パケットが届けられるとは限らないが，利用者はこの予測値から対象となる制御システムに適用できるかを判断できる．また，必要によってはスイッチ機器にパケットの追い越し機能を付加するなど，最小限のハードウェア的変更で対応することも考えられる．

4. 関連研究

複雑な機械系を制御するためのシステムソフトウェアに関する要素技術の研究が多数行われている．ここでは，特にリアルタイム通信機構についてとりあげる．

リアルタイム通信の実現に関する研究としては，Tenet[1] や Loeser ら[4] の研究がある．Tenet では，下位から上位まですべての層がリアルタイム性を保証するプロトコルになっている．Loeser らは，スイッチによって接続された Ethernet 上でリアルタイム通信を実現しているが，ネットワーク層でリアルタイム通信プロトコルを使用している．NOAH の通信においては，下位層のプロトコルとしては UDP/IP を使用し，プロトコルスタック，通信機構の実装の工夫で管理されたネットワークを実現し，リアルタイム性を確保する点が上述の研究と異なる．

Publisher/Subscriber (P/S) モデルの通信[3),5)] は，あるノードが送信した情報が，それを必要とする複数のノードにおいて暗示的に共有されるという点で NOAH の情報共有方式と似ている．しかしながら，NOAH ではセグメントの概念による通信の隠蔽がなされている点，セグメントに時間制限やイベント等の情報が含まれる点が P/S モデルとは異なる．

5. おわりに

本稿では，我々が提案している，分散リアルタイム処理系を構成する複雑な機械系の制御を目的としたオペレーティングシステム NOAH について述べ，特に通信方式に焦点を充て，Ethernet LAN における分散リアルタイム制御システムのための通信フレームワークの実現法を検討した．リアルタイム通信のためのプロトコルを設計するのではなく，管理されたネットワークを実現することで分散リアルタイム制御に要求される通信を実現する方式を提案した．

現在，有効性を確認するために RT-Linux 上で提案した方式の実装を進めている．また，有効性を確認した後は，人型ロボットの制御系に適用し，実機による評価を行う予定である．

謝　辞

本研究の一部は，文部科学省科学研究費補助若手研究 (B) 課題番号 17700078 の支援により行った．

参考文献

1) Banerjea, A., Ferrari, D., Mah, B.A., Moran, M., Verma, D.C. and Zhang, H.: The Tenet Real-Time Protocol Suite: Design, Implementation and Experience, IEEE/ACM TON, Vol.4, No.1, pp.1–10 (1996).
2) 兵頭 和樹, 田胡 和哉, 伊藤 雅仁, 井上 亮文, 宇田 隆哉, 市村 哲, 星 徹, 松下 温: ロボット用オペレーティングシステム NOAH の構想, コンピュータシステム・シンポジウム論文集, pp. 113–120 (2004).
3) Kaiser, J. and Mock, M.: Implementing the Real-Time Publisher/Subscriber Model on the Control Area Network (CAN), *2nd Int'l Symposium on ObjectOriented Distributed Real-Time Computing Systems* (1999).
4) Loeser, J. and Haertig, H.: Low-latency Hard Real-Time Communication over Switched Ethernet, Proc. *16th Euromicro Conference on Real-Time Systems (ECRTS 04)* (2004).
5) Rajkumar, R., Gagliardi, M. and Sha, L.: The Real-Time Publisher/Subscriber Inter-Process Communication Model for Distributed Real-Time Systems: Design and Implementation, In *IEEE Real-time Technology and Applications Symposium* (1995).