分散リアルタイムOS技術

慶應義塾大学環境情報学部／慶應義塾大学大学院政策・メディア研究科
徳田 英幸 hxt@ht.sfc.keio.ac.jp
慶應義塾大学大学院政策・メディア研究科
西尾 信彦 vino@mkg.sfc.keio.ac.jp
慶應義塾大学大学院政策・メディア研究科
永田 智大 ngt@ht.sfc.keio.ac.jp
慶應義塾大学大学院政策・メディア研究科
間 博人 haru@ht.sfc.keio.ac.jp

◆リアルタイムOSの重要性

一般的に，商用システムで広く使われている“リアルタイムOS”という言葉は，単に“処理スピードが速い”，“ROM化できるほど小さいOS”等といったことを意味する場合が多い．一方，研究コミュニティにおいては，システムの性能や動的特性が「どれだけ解析・予測可能か」という点[1]に重点が置かれ，アドホックなOS技術ではなく，より科学的なリアルタイムOS技術の確立をめざしている．

分散リアルタイムOSは，さまざまな情報システムを構築していく上で，ますます重要になっている．まず電話網，航空管制システム，交通制御システムといった社会インフラ系システムでの利用が挙げられる．また，自動組立工場などにおけるロボット制御や機器監視システムなど制御系システムにおける需要がある．この他に，PC上でのテレコンファレンスシステムやビデオ，音声，画像なども取り扱えるマルチメディアシステムなどいろいろな応用分野で使われてきている．

このような分散リアルタイムシステム上で実行されているリアルタイムタスクの持つ時間的制約を満足することを保証することは非常に難しく，システム構築を著しく複雑にするだけでなく，ソフトウェアコンポーネントの再利用やシステム全体の再構成を難しいものにしている．多くの場合，システム設計者は統一された開発方法や解析手法に欠き，アドホックな解決方法をとって分散リアルタイムシステムを構築してきた．

80年代後半から90年代にかけて，米国ではOffice of Naval Researchが中心となって重点研究課題としてリアルタイム処理技術を取り上げた．その結果，多くの研究グループによりスケジューリング理論，リアルタイムスケジューリング手法，リアルタイムOS技術，リアルタイム通信プロトコル，プログラム解析技術などが研究開発されてきた．

本稿では，分散リアルタイムOSの基礎的な課題，高度な分散リアルタイムシステムを構築していくためのリアルタイムOS技術の最新動向を解説する．また現在進行している研究プロジェクトの研究成果について報告し，今後の技術的な課題などについて考察する．

◆分散リアルタイムOSの基礎的な課題と事例

ここでは，まずリアルタイムシステムの基礎的な課題について解説し，いつくかの代表的な分散リアルタイムシステムの事例における問題点を概説する．

《基礎的な課題》

1）ハードvs.ソフトリアルタイムタスク

リアルタイムシステムにおけるソフトウェアの実行は，そのプログラムの論理的な正当性だけでなく，時間的制約を満たすことを保証しなければならない．一般に，リアルタイムタスクの時間的制約は，ハードデッドラインとソフトデットラインに分けられる．さまざまな定義

が存在するが，ここでは value function[2] を使って定義する．あるタスクの時間的制約が満たされない場合に，システムにとってその効用がマイナス無限大になる時，その時間的制約を“ハードデッドライン”といい，そのタスクを“ハードリアルタイムタスク”という．また，時間制約が満たされない場合に，その効用が緩やかに減少していく場合，その時間的制約を“ソフトデッドライン”といい，タスクを“ソフトリアルタイムタスク”という．

また，システム構築において，これらの時間制約がどのような経緯から導出されているかといった問題や，時間的制約がソースプログラム上に記述されないといったプログラミング上の問題もある．

2）スケジューリング問題

タスクスケジューリング問題は，与えられたタスクセットが限られた計算資源のもとで，その時間的制約をすべて満たせるようにタスクの実行順序を制御することである．一方，分散リアルタイムシステムでは，各ホスト上のタスクスケジューリングだけでなく，ネットワーク上のメッセージ・スケジューリングや，データベースのトランザクション・スケジューリング等が関係する．分散リアルタイムシステムにおける問題は，こうしたいろいろなドメインでのスケジューリングを統一的に行われなければならない点である．

あるタスクが，ネットワークを介して他のタスクとメッセージを交換しながら，仕事を遂行している場合，そのネットワークワイドの仕事のデッドラインを満たすようスケジューリングを制御することを，特に“end-to-end スケジューリング問題”という．

3）スジュラービリティ解析問題

従来のスケジューラビリティ解析問題は，あるスケジューリング方法に関して，与えられたタスクセットが，計算資源の制約下で，時間的制約を満たすことが可能かを判定する問題である．あるいはタスクセット内のどれか 1 つのタスクが，デッドラインを満たせなくなるまでに，どれだけ CPU 負荷をかけられるかといった，単一資源に関するブレークダウン・ユーティライゼーション（breakdown utilization）を決する問題であった．初期のスケジューラビリティ解析手法は，タスクセットが相互に依存せず，タスク切替えのオーバーヘッドがゼロと仮定しているケースに関して多く解析されていた．

しかし，分散リアルタイムシステムでは，タスク間の実行順序関係は，データやメッセージに依存する．メッセージ通信なども，共通のネットワークを介して行われるので，メッセージのスケジューラビリティ解析なども含めた end-to-end での解析が必要になっている．

4）リアルタイム同期問題

リアルタイムタスク間での同期問題は，一般の並行プロセスにおける同期問題より，さらに厳しい時間制約が課せられているため，従来の同期方法では，いくつかの不都合が起きてしまう．

たとえば，資源の相互排除を保証するための単純なクリティカルセクション問題がある．クリティカルセクションに入ろうとしているプロセスは，いずれも最終的に必ずクリティカルセクションに入れるといったスタベーションフリー（starvation free）の保証ではなく，すべてのリアルタイムタスクがデッドラインを満たすよう同期を保証する必要がある．

従来のセマフォアに代表される同期操作では，クリティカルセクションに入れなかったタスクは，セマフォアのキューで待たなければならない．一般には，並行プロセス間でのスタベーションが起こることを防ぐために，FIFO の順にキューイングされている．しかし，このような方式では，緊急度の高いリアルタイムタスクも低いタスクも同等にキュー内で待たされてしまうので，優先度逆転現象（Prioritiy Inversion）が起き，高いタスクがデッドラインを満たせなくなる場合が生じてしまう．

5）リアルタイム通信問題

リアルタイム通信における問題は，単にメッセージを送信先のタスクに実時間で転送することではなく，各メッセージの持っている時間的制約を満たすようにメッセージ転送をスケジュールできるかどうかが問題である．データリンク層の下位副層にあたり，フレームの送受信方法を規定するメディア・アクセスプロトコルのレベルから，トランスポートレベルまで，時間的制約を満たすように，各プロトコルレベルにおいて協調制御される必要がある．

たとえば，トランスポートレベルのプロトコルでは，メッセージに対して優先順序を指定することができたとしても，実際のメディア・アクセスプロトコルがイーサネットのプロトコルのようでは，最悪転送待ち時間を限定することができないので，リアルタイム通信に使うことは難しい．また，IEEE802.5 のトークンリングのようなアクセスプロトコルでは，フレームごとに 8 レベルの優先順序を指定することはできるが，同一ホスト内にお

いて，優先順序の低いフレームが，それより高いフレームの転送をブロックしてしまうという優先度逆転現象が起きてしまう．

《分散リアルタイムシステムの事例》

以下に組込み型分散リアルタイムシステムにおける2つの例についてみる．たとえば，**図-1**のようなNASAが今後打ち上げるスペースステーションなどの航行制御システムにおいては，各プロセッサ内のタスクのデッドラインに基づくスケジューリングだけでは不十分である．各ノードが接続しているネットワーク上でのメッセージ・スケジューリングや，航行用に使われているリアルタイムデータベースへの問合せ，書込み，変更等を行う各トランザクションもデッドラインを満たすようにスケジュールする必要がある．end-to-endでのスケジューラビリティを解析するためには，各スケジューリング・ドメインが他のドメインと統一的にスケジュールすることが必須である．また，システム構成の動的変更や，負荷の変化に対しても，いままでのリアルタイムタスクがデッドラインを継続して満たすことができるかどうかの解析や予測ができなければならない．

もう1つの例は，ネットワークでつながれたPCやワークステーションでビデオ，音声，画像などのマルチメディアデータを多重処理するマルチメディア系システムである．たとえば，UNIX上で音声データやMIDIデータの処理中に他のユーザ・タスクの起動によってプロセッサが横取りされてしまうと，ビデオや音の再生が正しく行うことができなくなってしまう．このような状況下では，リアルタイムシステムで使われているいろいろな資源の管理技術やデッドラインに基づくタスクのスケジューリング機能が不可欠である．従来のタイムシェアリングシステムで使用されているスケジューリング機能では，周期的なアクティビティの制御が難しいだけでなく，一時的な過負荷な状況（transient overload）の下では，どのアクティビティがデッドラインを満足できなくなるかなどの予測がまったくできず，実時間の制約を持つマルチメディアサーバなどの多重処理が著しく困難となってしまっている．

また，話している人の唇の動きと音声を同期が取れないと，リップ・シンク（lip-synching）問題が起きる．

これは，ネットワークを介して2つの独立した通信チャネルから送られてくる音声とビデオデータを同一時間軸上で同期して再生できないことにより生じる．また，アプリケーションからの音声やビデオ転送に対するサービスの質（Quality of Service, QOS）に対する要求をもとに，それに伴うデータ転送に関する時間制約（遅延（delay）やジッタリング（jittering）の上限）やスループットに対する制約を満足できるようどのようにネットワークやプロセッサの資源を管理するかといった問題も残されている．

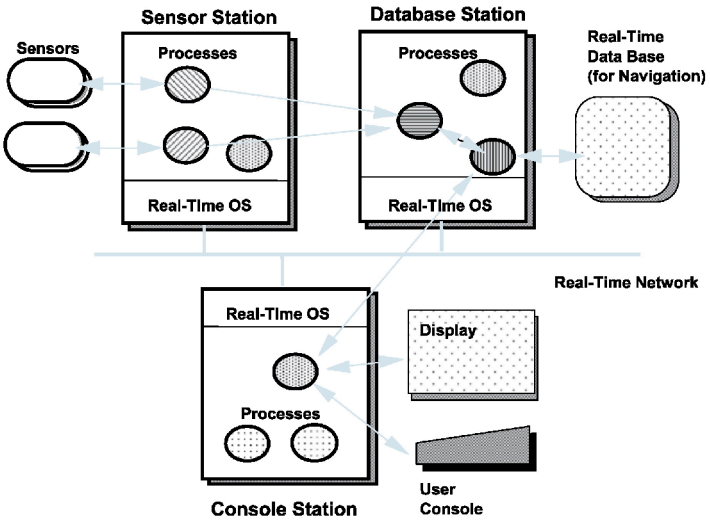

図-1　組込み型分散リアルタイムシステムの例

◆リアルタイムOS技術の最新動向

《時間制約の記述》

リアルタイムOS技術の発展は時間制約を持つ処理をいかに記述するかという観点からみると興味深い．当初Cyclic ExecutionのようなOSを用いないシステムであった時代は，時間制約が明示的にプログラムに記述されていなかった．その後，プリエンプティブなタスクスケジューラが導入され，時間制約の記述のために周期的タスクやデッドラインハンドラといった概念が導入された．次の周期までをデッドラインとして周期的に起床するタスクを，デッドラインミス時に起動されるハンドラとペアで記述する．このときのスケジューリングポリシーには，スケジュール可能性解析のためにRate MonotonicやEarliest Deadline Firstが適用され，その後この枠組みの中で，非周期タスクサーバなどが導入さ

れていった．近年では，組込みシステムなどのハードな時間制約だけではなく，マルチメディア処理のような比較的ソフトな時間制約を記述するためにデッドラインミスに関する評価関数を導入したり，QoS（サービスの品質）を多段階に記述することによって処理品質を動的に制御できるような方式が導入されるようになってきている．一方，より簡便な時間制約の記述として資源予約方式の研究も盛んである．資源予約はCPUやネットワークといった資源の種類によりさまざまな方式があるが，代表的なCPU資源予約では，C（処理時間）/T（補充周期）を与えて周期起床機構を流用して実装している．以下では，この資源予約の近年の研究トピックについて我々が特に関心を持っている点にふれる．

《資源予約方式の最新研究トピック》

もともとCPUのリアルタイムスケジューリングを簡易に記述できるものとして始まった資源予約も，CPU以外の資源も予約できるようにしたり，リアルタイムアプリケーションの開発支援を目指すといったかたちで発展をとげている．本節では，近年のトピックとしてより統合化された資源予約システムを提供するために今後解決していくべき4つの課題について説明する．

・複数種類の資源の統合

これまでは，CPUやネットワーク帯域といった単一の資源を独立に予約するものであったが，これらには依存関係が存在するものがあり，独立には扱えないことがある．大量のデータ転送はプロトコルスタックでの処理が重くCPU資源も相応して必要となる．

・多階層のQoS表現の統合

資源の予約量の表現はシステムレベルではメモリ容量やネットワークバンド幅などを用いるが，ミドルウェアではフレームレートや画像のサイズを用いることもあり，アプリケーションではさらに抽象的な表現を用いることが普通である．これらの表現の間では翻訳が必要となるが，同じ品質でもプラットフォームに依存して一意には翻訳できないのが普通である．

・複数の実行主体の統合

近年のソフトウェア構成では単一のスレッドの挙動を保証するだけでは十分ではない．1つのアプリケーションがサーバとクライアントや，プロキシやエージェントなどといった複数の実行主体の協調動作によって実現するのが主流である．

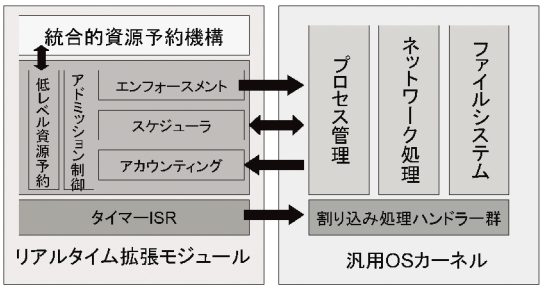

図-2　Linux/RKのシステム概要

・分散資源予約の統合

前項のように複数の実行主体は通常分散しており，そのホストごとに異なった性能や機能であるため，たとえばend-to-endでの性能保証をするためには何らかの統合フレームワークに従っての制御が必要となる．

◆コモディティOS上での分散リアルタイム技術

我々はソフトおよびハードリアルタイムデータを通信できる分散リアルタイムネットワークを実現するためのOS技術について研究開発している．

《概　要》

ハードリアルタイムタスクのデータ通信機構では，デッドラインをミスしたかどうかの“1”か“0”といった判定が最も重要であった．一方，さまざまなQoSパラメタを持ったソフトリアルタイムメッセージ通信を可能にするには，どのレベルまでQoSを満たすことができるかといった満足レベルを動的に制御できるメッセージ処理機構がOSで必要となる．そこで，以下の3つの課題を設定し，これらの課題を解決している．

1）リアルタイムネットワークのためのOS支援技術：リアルタイム通信を支援するOS技術を研究開発する．

2）ネットワークプロトコル技術：リアルタイム性を保証・支援する通信プロトコル，通信機構を研究開発する．

3）分散リアルタイム応用システム技術：リアルタイム技術を利用した応用システムソフトウェアを研究開発する．

さらに，これらの技術をコモディティOSであるLinux上で実現することで，研究成果を他の研究者や利用者に容易に利用してもらい，世間への普及を目指す．

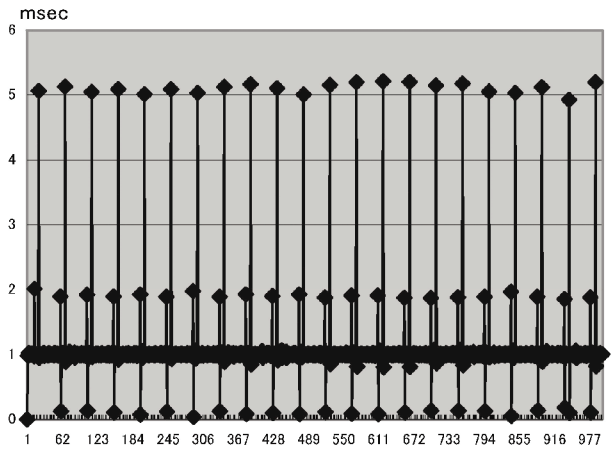

図-3　/proc アクセスに周期的に妨害される 1msec 起床スレッド

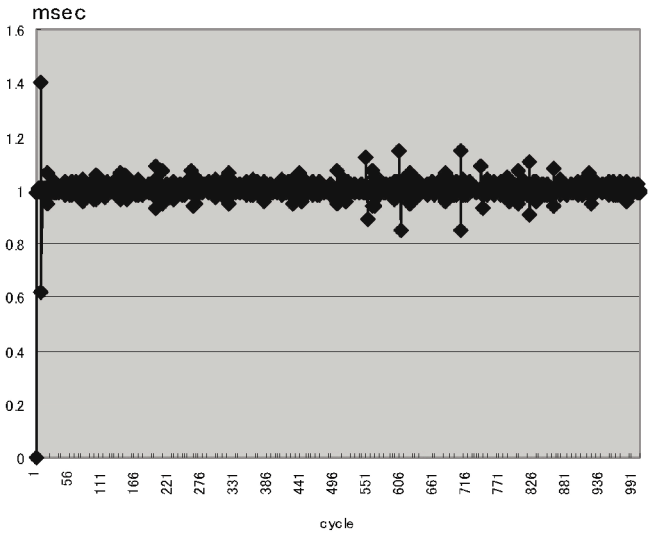

図-4　改善された 1msec 起床スレッド

《Linux 上でのリアルタイム機能の実装》

これまで Real-Time Mach マイクロカーネル[3] に統合的資源予約機構を導入してきた．しかしながら近年のコモディティ OS である Linux や FreeBSD の進歩に，その OS パーソナリティサーバを追従させていくことは困難である．そこで今回，コモディティ OS である Linux に Real-Time Mach で実現してきたリアルタイム機能を拡張実装することにした．

リアルタイム性を拡張する方式として CMU の Linux/RK[4] の用いた方式をベースとしてオリジナルの Linux カーネルのソースプログラムに極力手を入れないで，拡張部分はカーネルモジュールとして実装した．**図-2** にその全体構成を示す．

統合的資源予約機構は，iReserve と呼ばれる統合的な資源予約機構を実装している．予約できる資源は現在はプロセッサのみだが，Real-Time Mach では実装されていたネットワーク転送性能の予約なども導入していく予定である．

ここで実装されている iReserve 機構では，生成される各予約オブジェクトごとに QoS プロファイルハンドラが装備されて，このハンドラによってさまざまな機能が実現されている．

QoS プロファイル機能は，予約オブジェクト生成後にバインドされるスレッドの資源消費を QoS プロファイルハンドラが監視することによって実現している．プロファイル対象となるスレッドは，当初資源予約なしで走行しそのときにプロファイルが実施され，システムの許容範囲であればそのプロファイルされた処理に対して必要となる資源が適切に予約される．プロファイル時は予約なしでの実行であるために，他の予約つきで実行している処理に影響を与えることはない．

今回，実装されているリアルタイム拡張の基本性能としてまず 1msec 周期でのスレッド起床を行った．ここでの 1msec という周期はロボット制御で必要とされる仕様から適用したものである．現在，このような周期的な起床をするプログラムは（i）自分を周期起床するスレッドに変更する API，（ii）その開始時間まで待つ API，（iii）次の周期の起床時刻がくるまでブロックする API により明示的に記述することによって実現している．

以下の実験結果は，システムに負荷を与えた状況での 1msec 周期起床がどのような挙動になるかを調べたものである．周期起床プログラムを周期起床スレッドに変換する API によってリアルタイムスレッドに昇格したスレッドが通常の Linux のスレッドより優先してスケジュールされるので，ユーザレベルでの他のスレッドに影響されることはほとんどない．しかし，Linux のようなコモディティ OS ではカーネル内の処理がいわゆるジャイアントロックを持っており，それを多用するようなアプリケーションが存在すると，たとえユーザレベルの処理が優先されていても，時間制約充足に悪影響を与えることがある．その例としては，proc ファイルシス

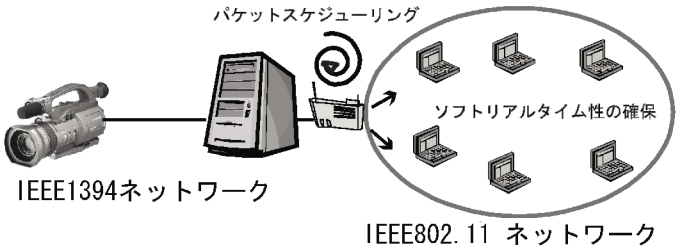

図 -5　IEEE 1394 と IEEE 802.11b からなる実験システム

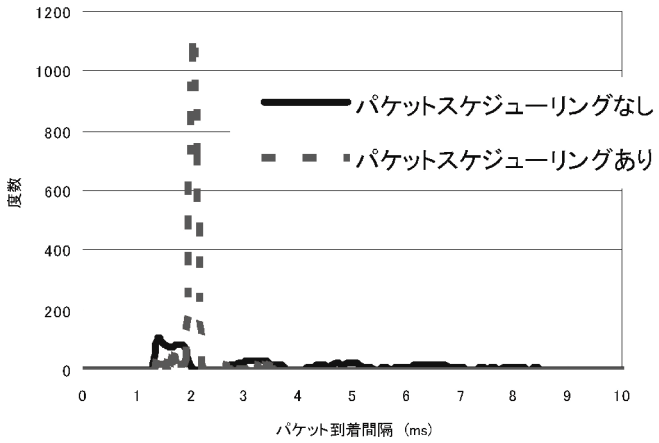

図 -6　リアルタイムトラフィックのパケット到着間隔の度数分布

テムを周期的にアクセスする top コマンドなどが考えられる．**図 -3** では，top コマンドをわざと 10msec 単位で CPU の負荷を proc ファイルシステムに見にいくようにして負荷を高めたときに周期起床スレッドが何 msec で次の周期に起床するかを縦軸，毎周期を横軸にプロットしたものである．

ただこのような状況も Linux カーネル内の処理レイテンシを縮める改良を施すことによって**図 -4** 程度には改善することができている．

現在，我々はこのリアルタイム拡張 Linux 上に Real-Time Mach で今までに実現されていた統合的資源予約機構を移植するための研究開発を行っている．QoS プロファイラを導入する目途はたっているが，Real-Time Mach で実現していたようなユーザレベルライブラリによる TCP/IP プロトコルの実装を用いたネットワーク転送性能の予約機構などを Linux で導入していくのは困難である．実現するためには，カーネル内のプロトコルスタックに手を入れていく必要があるであろう．

《無線 LAN システムにおけるソフトリアルタイム制御の応用》

分散リアルタイム OS 支援の一貫として行われている，無線 LAN によるソフトリアルタイムの応用について紹介する．ホームネットワークや，ウェアラブルネットワークでは，動画やセンシング情報などのリアルタイム性を要求するアプリケーション例が考えられている [6]．こうしたアプリケーションを実現する際に AV データの転送媒体として IEEE 1394 が期待される．その一方で，利用者の移動性を確保するために無線 LAN の利用が必要となる．こうした状況を踏まえ IEEE 1394 と IEEE 802.11b とがゲートウェイを介して接続された実験システム（**図 -5**）を構築しリアルタイム通信支援機構の研究開発をしている．

本実験システムは，DV（Digital Video）と他の種類のデータが混在したときに，DV ストリームをスムースに配送することを目指す．IEEE 1394 には，アイソクロナス（等時）転送と非同期転送の 2 種類の転送モードがある．アイソクロナス転送では，125μs 周期に 1 回の送信スロット利用が保証される．送信側および受信側のノードで OS によるリアルタイムサポートがあれば，アイソクロナス転送を予約することにより，両エンドのアプリケーション同士での時間制約を満たすことができる．しかし，途中にゲートウェイが入ることにより，一貫した時間制約制御が損なわれる．1 つの方法として，CSMA/CA の改良により 802.11b におけるリアルタイム性を実現する研究がある [7]．しかしながら，IEEE 1394 と IEEE 802.11b のゲートウェイの場合においては，広帯域 100Mbps もしくは 200Mbps から低帯域 11Mbps へと帯域が小さくなるので，ゲートウェイ内部で未送信パケットのキューによる遅延時間が増加する可能性がある．また，無線 LAN 特有の問題として，ゲートウェイと無線クライアントノード間の電波強度の大小に応じてパケット損率が変化して，無線 LAN 全体としての有効スループットが低下する可能性がある．そこで，伝送媒体の優先度に応じたパケットスケジューリング，および複数のクライアントノードが存在するときに，各クライアントとゲートウェイ間の無線リンクの電波強度を加味

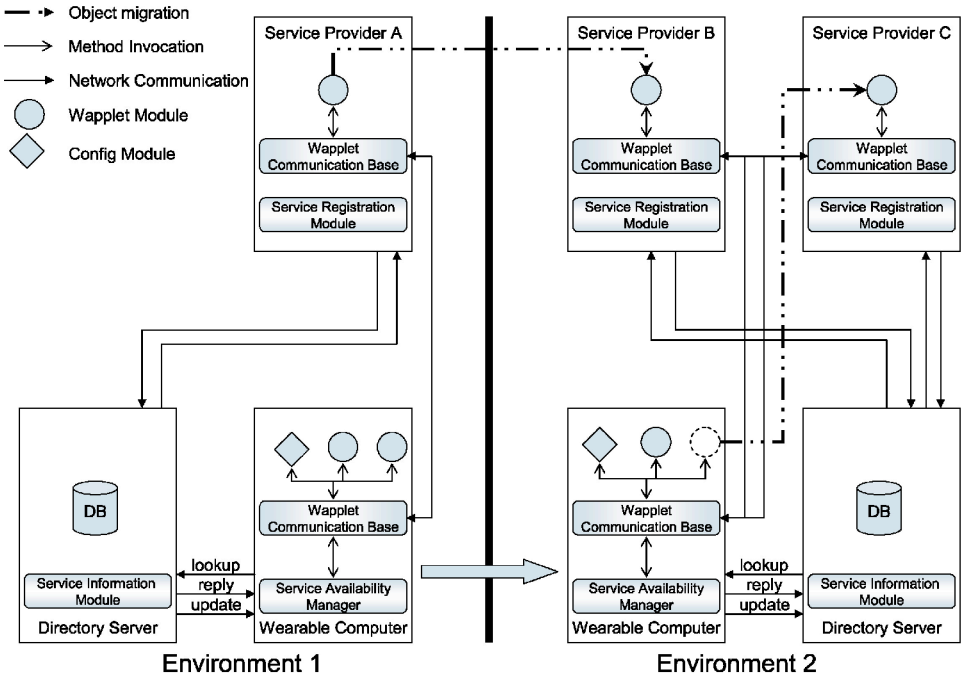

図 -7　リアルタイム応用ソフトウェア概要

したスケジューリング[8]を実現する．

次に，ゲートウェイにおけるパケットスケジューリングの効果を調べるために，IEEE 802.11b とゲートウェイからなるシステムにおいて単純にパケットの優先制御を行った実験結果について述べる．実験システムにおいて，2 台のホストからゲートウェイを介して IEEE802.11b 受信ホストへデータを送り続けるものとする．1 台のホストからは，2ms 周期の UDP（User Datagram Protocol）パケットを送り，これをリアルタイムトラフィックとする．もう 1 台のホストからは，ランダムにバックグラウンドトラフィックを発生させる．**図 -6** は，IEEE802.11b 受信ホストにおいて，リアルタイムトラフィックのパケット到着間隔の度数分布を示したものである．

ゲートウェイにおいてリアルタイムトラフィックの優先度を考慮したパケットスケジューリングを実行するとパケット到着間隔の揺らぎが抑制され，実行しないときには揺らぎが広がりソフトリアルタイム制御が行われている．この結果に示されるように単純なパケットスケジューリングを適用するだけでも，パケット間隔の揺らぎの抑制に効果がある．

《サービス利用へのソフトリアルタイムの応用》

これまで述べてきた分散リアルタイム OS 技術を応用，利用し，ユビキタス空間でのサービス利用システムを構築している．従来の PC に加え，身につけた PDA や WC（Wearable Computer），周辺に遍在する情報家電機器や各種センサ（これらをサービスと呼ぶ）などがネットワークに接続されたユビキタス空間内[9]で，利用者がそれらの機器を容易に利用できる．現在のユビキタス環境のように，AV 機器などの情報家電が多く存在する場合，マルチメディアデータを扱う頻度が多く，そのリアルタイム性をソフトに保証することが重要となる．たとえば，音楽を聞きたい場合，利用者の所有する PC にスピーカが装備されてなくとも，周辺にある AV 機器に装備されたスピーカを利用することで，音楽を聞ける．このようなサービス利用システムを構築するため，我々は Wapplet フレームワークと ASAMA システム[10]を前節でのべたリアルタイム機能の上に構築した．**図 -7** に Wapplet フレームワークと ASAMA システムを用いたアプリケーションの動作の様子を示す．Wapplet フレームワークは，Wapplet Communication Base, Wapplet Module, Config Module の 3 つから，ASAMA システムは，Service Availability Manager, Service Registration

Module, Service Information Module の 4 つから構成される.

Wapplet フレームワークは，利用者が身につけた WC を中心に，周辺にあるサービスを効果的に使用するためのアプリケーションフレームワークである．図 -7 に示したように，Wapplet フレームワークに基づいたアプリケーションは，Config モジュールと複数の Wapplet モジュール群から構成され，WC やサービス上に移動し，WC やサービスの負荷軽減，ネットワーク帯域の効率的な利用を実現する．これにより，マルチメディアデータの送受信や再生の際にソフトリアルタイムを保証することが容易となる．Wapplet フレームワークでは，モジュール全体やモジュールの一部のみといった段階的なモジュールの移送を行える．これにより，利用できるネットワークに応じたモジュール移送が実現される．さらに，モジュールの移送の際に必要となるモジュールの管理を WC 上で一括して行う．その際，各モジュールごとに WC の IP アドレスと動的に生成されるシリアル番号，およびモジュール名から構成される識別子を利用する．

ASAMA システムは，利用者の移動による利用可能なサービスの変化に動的に適応することを支援するシステムである．図 -7 に示したように，周辺で利用可能なサービスを把握するディレクトリサーバが存在し，WC はサービスを利用する際，ディレクトリサーバに問合せを行う．この際，サービスの種類によって問合せ方法が異なり，マルチメディアデータを扱うサービスの場合，問合せに MIME を利用する．これにより，代替可能なサービスをも検索できる．また，利用者の移動（動的変化）やサービスの提供開始，停止（受動的変化）による利用可能なサービスの変化を，ディレクトリサービスが管理し，以前，同種のサービスを検索した WC に動的に通知を行う．通知を受けた WC は現在利用しているサービスが利用不可能になると他のサービスを利用し，より品質の高いサービスが利用可能になれば，そのサービスを積極的に利用できる．利用するサービスを切り替える場合，スムースに行えるよう，ソフトリアルタイム性をきちんと保証しなければならない．そこで，前述したリアルタイム OS 技術やリアルタイムネットワーク技術を適応し，これらの問題を解決していかなければならない．

◆リアルタイム OS の今後の展開

従来のリアルタイム OS 技術だけでは，今後のユビキタスネットワーク社会で基盤を支えるさまざまな分散リアルタイムシステムには対応できないのが現状である．ロボットなどの制御系システムだけでなく，センサーネットワーク系，社会インフラ系，マルチメディア系システムなどさまざまな制約を持ったタスクを柔軟にかつ統合的にサポートできる OS 技術の開発が必須である．CPU 資源，メモリ資源，ネットワーク資源などを統合的に管理し，end-to-end での QOS 制御も包含したかたちの制御機構が望まれる．

謝辞　本研究の一部は科学技術庁科学技術振興調整費「人間支援のための分散リアルタイムオペレーティング基盤技術の研究」の下に行われています．

参考文献

1) Stankovic, J. A.: Misconceptions about Real-time Computing: A Serious Problem for Next-generation Systems, IEEE Computer, Vol.21, No.10 (Oct. 1988).
2) Locke, C. D., Jensen, E. D. and Tokuda, H.: A Time-Driven Scheduling Model for Real-Time Operating Systems, Proc. of 6th IEEE Real-Time Systems Symposium, IEEE (Dec. 1985).
3) Tokuda, H., Nakajima, T. and Rao, P.: Real-Time Mach: Towards a Predictable Real-Time System, In Proceedings of USENIX Mach Workshop (Oct. 1990).
4) Rajkumar, R. et al.: Resource Kernels: A Resource-Centric Approach to Real-time and Multimedia Operating Systems, SPIE Conf. on Multimedia Computing and Networking (Jan. 1998).
5) 西尾信彦，徳田英幸：QOS プロファイルハンドラつき資源予約機構の設計と実装，情報処理学会論文誌，Vol.42, No.6, pp.1570-1579 (June 2001).
6) 蔵田武志，鈴木琢治：ウェアラブル機器によるリアルタイムセンシング技術，情報処理，Vol.44, No.1, pp.40-45 (Jan. 2003).
7) Sobrinho, J. L. and Krishnakumar, A. S.: Real-time Traffic over the IEEE 802.11 MAC, Bell Labs Technical Journal, Vol.1, No.2, pp.172-187 (Autumn 1996).
8) 間　博人，戸辺義人，徳田英幸：無線 LAN におけるチャネル状態依存スケジューリングの実験評価，情報処理学会論文誌，Vol.43, No.12 (Dec. 2002).
9) Czerwinski, S. E., Zhao, B. Y., Hodes, T., Joseph, A. D. and Katz, R.: An Architecture for a Secure Service Discovery Service, The 5th Annual ACM/IEEE International Conference on Mobile Computing and Networkings, pp.24-35 (Autumn 1999).
10) 永田智大，西尾信彦，徳田英幸：ASAMA：適応的なサービス利用管理機構，情報処理学会論文誌，Vol.42, No.6, pp.1557-1569 (June 2001).

（平成 14 年 12 月 11 日受付）