サービス利用状況の変化に対する適応支援機構

永田智大[†] 西尾信彦[†,††] 徳田英幸[†,†††]

本論文では，ユビキタスコンピューティング環境において利用者が移動した際，ネットワークの切断・接続によるサービスの利用状況の変化に対して動的に適応できるよう支援する機構の設計・実装について述べる．サービスの利用中断時に利用者がサービスとの事前処理を正確に行い，再開時に継続したサービス利用を行えることを実現する．中断のための事前処理が失敗した場合に備え，失敗時の処理も行えるようにする．また，サービス利用中断時でも利用者やサービスが自身だけで行える処理を実行できるような機能を提供し，中断時でも効率的なサービス利用を利用者が行えるようにする．これらの機能を利用することで利用者は移動しながら周辺のサービスを効率的に利用できる．本論文が提案する ASDS（Adaptive Service Disconnect/Reconnect Support）は無線 LAN の S/N 比に閾値を設定し，ネットワークの切断，接続を検知する．また，S/N 比の変化の度合いからネットワーク切断の予見を行い，切断のための事前処理を実行できる．ASDS は切断，接続に必要な処理を適切なタイミングで動的に実行する．本論文では ASDS の設計・実装を行い，実際に敷設されている無線 LAN ネットワーク網での実証実験を行った．また，応用アプリケーションである MP3 プレイヤを用いた評価を行い，ASDS の有効性を示した．

Design and Implementation of Adaptive Service Disconnect/Reconnect Support Mechanism

Tomohiro Nagata,[†] Nobuhiko Nishio[†,††] and Hideyuki Tokuda[†,†††]

In this paper, we describe the design and implementation of support mechanism adapting to the change of service availability. The service availability changes when the user migrates, causing network disconnection and re-connection. This mechanism supports continuous use of service by correctly suspending upon disconnection, and resuming on re-connection. The mechanism also supports recovery in case of failure to suspend. While disconnected, the mechanism supports disconnected operation by both the user and the service. Through this mechanism, the user can use the services around him effectively while migrating. We propose ASDS (Adaptive Service Disconnect/Reconnect Support) which detects the network disconnection and re-connection, defining the threshold of the wireless LAN's signal noise ratio. Using the change of the signal noise ratio, ASDS also predicts the network disconnection to execute the suspending. ASDS can execute the suspend and resume at the appropriate timings. We design and implement ASDS, evaluating in the actual wireless network environment. We also describe the effectiveness of ASDS evaluating with a MP3 player.

1. はじめに

近年，ユビキタスコンピューティング環境[1)]に関する研究が多くの研究機関で行われている．ユビキタスコンピューティング環境では，従来のように利用者がコンピュータの前にいてネットワークを利用した情報取得やコミュニケーションを行うのではなく，利用者の周辺に存在する様々な機器を利用して利用者の様々な要求を実行できる．そのため，利用者は自身の要求を実行するためにコンピュータの前に移動する，もしくはコンピュータを持ち歩く必要がない．周辺にある機器が協調動作し，利用者の要求を実行できるような

† 慶應義塾大学大学院政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 科学技術振興事業団，さきがけ研究 21,「協調と制御」領域 "Intelligent Cooperation and Control", PRESTO, Japan Science and Technology Corporation (JST)
††† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

本研究の利用中断に対する処理の実行に関しては文部科学省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ネットワーク基盤技術の研究」の，サービスとの同期に関しては IPA 未踏ソフトウェア創造事業「Mobiquitous Environment：適応型次世代ユビキタス基盤環境の構築」の支援による．

環境を提供する．

本研究では，ユビキタスコンピューティング環境の中もしくは複数の環境の間を利用者が移動しながらでも，周辺の機器やアプリケーションと協調動作して利用者の要求を処理できる新たな基盤環境 “Mobiquitous 環境”（Mobiquitous とは Mobile と Ubiquitous をあわせた造語）を構築することを目標とする．本研究では，周辺の機器や資源をユビキタスホストと呼び，その上で動作するアプリケーションをサービスプロバイダと呼ぶ（以後，サービスと呼ぶ）．Mobiquitous 環境では，無線ネットワークに接続できる PDA やウェアラブルコンピュータ [2]（以後，モバイルホストと呼ぶ）を利用者は携帯し，その上で動作する Mobiquitous アプリケーション（以後，アプリケーションと呼ぶ）内のモジュール（クライアントモジュール．以後単にクライアント呼ぶ）が周辺のサービスと協調動作をし，利用者の様々な要求を実行する．利用者が移動すると，利用するサービスを切り替えたり，サービスの利用中断・再開を正常に行ったりすることで，継続的なサービスの利用を実現する．

そこで，本論文では，この Mobiquitous 環境を実現する基盤技術として，サービスの利用状況の変化に適応することを支援するミドルウエアである Adaptive Service Disconnect/Reconnect Support（ASDS）を構築することを目的とする．利用者が移動し，サービスから物理的に離れてしまう，またはネットワークから切断されることにより，サービスを利用不可能になる場合がある．逆にサービスに近づく，ネットワークに接続できることで，利用可能になる．このようなサービスの利用状況の変化に対し，サービスを利用するクライアントが適応動作して，継続的なサービス利用を行うことを支援する．

本システムでは，「サービス利用中断・再開に対する処理の実行支援」，「サービスとの同期支援」，「ネットワーク切断の予見」の 3 点を実現する．アプリケーションは利用者の移動によって生じるネットワーク切断の予見を行うことで，サービス利用中断に必要な事前処理を切断前に正常に行える．また，サービスとの同期を行えることで，サービス利用中断中でもサービスやアプリケーション自身での処理を行うことも可能になる．これらを実現することにより，サービスとアプリケーションのステートを同期でき，サービス利用を正常に再開できる．

本論文では，ASDS の設計，実装を行い，評価，実証実験を行うことで ASDS の利便性を示す．たとえば，本システムを利用すると「ネットワーク切断中でもコンテンツサーバ上の音楽データを聞けるアプリケーション」や「移動する先で部屋のライトをつけていくフォローミーライト」，「サービス利用中断時のセンサデータをも後から取得できる状況監視アプリケーション」などを構築できる．

次章では，関連研究について述べ，従来のユビキタスコンピューティング環境の問題点を示し，サービスの利用方法について考察しながら本論文の目的について述べる．3 章では本論文が提案する ASDS の動作概要および設計について，4 章では実装について言及する．5 章では ASDS の評価を行う．

2. 本研究の目的と課題

本章では，最初に本論文の関連研究について述べ，それらの長短所について言及する．それをふまえ，本研究の目的について述べる．

2.1 関連研究

これまでのユビキタスコンピューティング環境では，ある場所における周辺の機器との協調動作を実現することが目的であった．Sun Microsystems, Inc. の Jini [3] や Microsoft 社の Universal Plug and Play [4] は周辺の機器を計算機から利用するためのシステムアーキテクチャを提供している．AT&T 研究所の Active Bat [5] やカーネギーメロン大学の AURA プロジェクト [6] は特殊なデバイスを利用者が装着し，そのデバイスから周辺の機器を制御する．また，慶應義塾大学の VNA [7] は利用者の周辺の情報家電の機能を仮想的なオブジェクトとして見たて，それらを組み合わせて利用する．U.C. Berkeley 校の Ninja プロジェクト [8] や ICEBERG プロジェクト [9] は周辺の機器を用いてネットワーク上のコンテンツを利用することが目的である．これらの多くは利用者が移動先の場所で周辺にある機器を協調利用することに焦点が当てられている．そのため，利用者の移動に対する適応を実現するにはアプリケーションごとになんらかの処理を行わなければならない．

このような中で，最近，慶應義塾大学の Smart Space Lab と東京大学の STONE room の 2 つのユビキタスコンピューティング環境の相互接続実験が開始された [10]．これまでのユビキタスコンピューティングでは離れたユビキタスコンピューティング環境を利用者が移動することに関してあまり考察されていなかった．この 2 つのユビキタスコンピューティング環境が接続され，実証実験を容易に行えるようになった．

複数のユビキタスコンピューティング環境を利用者が移動する場合，これまでのモバイルコンピュー

ティング技術をそのまま利用すると様々な問題が生じる．TCP/IP 層で計算機の移動を支援するものとして Mobile IP [11] や I-TCP [12]，TCP-R [13] がある．また，アプリケーション層での移動支援を行うものとして Rover [14] や MSOCKS [15] などがある．これら関連研究では，IP 層からセッション層において移動の透過性を統一的に保証しているため，アプリケーションを修正する必要がない．その反面，個々のアプリケーションが必要とする移動時の処理を行えない．たとえば，アプリケーションが用意するタイマのタイムアウトの回避を関連研究で行うことは難しい．

また，関連研究の多くが，移動する前に通信していたサービスと移動後に通信するサービスが同じものであると想定する．しかし，Mobiquitous 環境では移動する前と後で利用するサービスが異なる場合が多く，クライアントが有するステートの同期がより重要となる．にもかかわらず，これら関連研究では，不可能な移動前後の事前処理が難しい．そのため，利用するサービスによって移動に対する対応方法が異なり，これらを柔軟に支援する方法が必要となる．

2.2 目　　的

本節では，まずサービスの分類を行い，利用者の移動によって生じるサービス利用の中断・再開に必要な要素技術について考察し，考察結果をもとに本論文の目的を設定する．

2.2.1 サービスの分類

Mobiquitous 環境では，利用者の要求を実行する際に利用するサービスとして，様々なものが考えられる．**図 1** に Mobiquitous 環境で利用されるサービスの分類を示す．

サービスは 3 つの指標によって分類される．第 1 に，サービスはクライアントとの間にステートを共有するかどうかで分類される．ここでいうステートとは，クライアントまたはサービス自身の状態や処理の実行状況のことを指す．たとえば，MP3 のデータをどこまで再生したか，といったサービスの処理状況や，電灯サービスが on の状態であるなど，サービスやクライアントがどのような処理を行っていたか，その状態を指す．そのため，サービスとクライアントの処理が進むに従い，両者のステートも変化する．ステートの共有とは，サービスとクライアントのステートが一貫性を保つ必要がある場合を指す．サービスとクライアントが有するステートがお互いに独立している場合，一貫性を考える必要ない．しかし，片方のステートがもう片方のステートに影響を与える場合，両者のステートの一貫性，つまりステートの共有が重要になる．

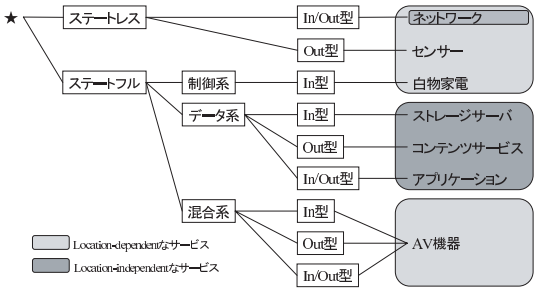

図 1 サービスの分類
Fig. 1 Service categories.

ステートレスとして分類されたサービスは，サービス，クライアントが保持するステートが独立しており，サービスとクライアントのステートはお互いに影響されない．ステートフルとして分類されたサービスは，サービスとクライアントのステートが相互に影響しあい，一貫性が保たれないと，サービスの処理に異常を発生させる可能性がある．つまりサービスとクライアントのステートがお互いに共有されることが重要になる．

第 2 にサービスとクライアントとの間でやりとりされるデータの種類によって分類される．制御系とは，サービスを制御するためにコマンドやイベントが送受信されるサービスを指す．データ系では，マルチメディアデータや，オブジェクトのバイトコードなどがサービスとクライアントの間で送受信される．最後の混合系は，サービスを制御するための制御系と，マルチメディアデータなどのデータ系の両方がサービスとクライアントの間で送受信される．

最後に主にサービスがデータを受信するのか，送信するのかによって分類される．この分類をみると，多くのサービスがクライアントとの間にステートを共有することが分かる．

この分類に基づくと，ネットワークやセンサはデータを単に送受信するため，それ自身にステートが存在せず，ステートレスなサービスとして分類できる．それ以外のサービスはサービス側に処理をすすめることで変化が生じるステートが存在するため，ステートフルなサービスである．

白物家電の多くはもともと自身が有するスィッチやリモコンによって制御可能なため，制御系のデータがやりとりされると考えられる．逆に CPU サーバやコンテンツサービス，アプリケーションサービスはメディアデータやプログラムの一部などが送受信されるため，データ系と分類した．AV 機器は白物家電と同様，制御系のデータによって機器が制御される．しかし，そ

の制御によって，サービスとクライアントとの間にメディアデータの送受信が発生するため，混合系として分類できる．

また，多くのサービスの場合，自身の存在する物理的位置が重要になる．サービスが物理的にどの場所に存在するか，その点が問題となるサービスで，location dependent なサービスといえる．逆に location independent なサービスもあり，CPU サーバやコンテンツサービスなど，物理的な位置は関係ないサービスがこれに含まれる．Mobiquitous 環境では，利用者のいる位置から近くに存在するサービスを利用する場合が多い．それらのサービスは location dependent なサービスであり，利用者から物理的に近くなければ，サービスを利用できない．

これらのことをふまえると，利用者が移動しても継続的なサービスの利用を行うには，ステートを共有しつつ利用するサービスを切り替えることが重要となる．利用しているサービスから物理的に離れ，別のサービスへと近づく場合，そのまま遠く離れたサービスを利用しても意味がない場合が多い．現在利用しているサービスの処理を一度中断させ，サービスのステートを次に利用するサービスで復元し，継続的なサービス利用を実現する．

利用するサービスを切り替えるときに，利用者がネットワークから切断される場合がある．つねに利用者がネットワークに接続できれば，サービス間のステートの移行は比較的容易に行える．しかし，今利用しているサービスから次のサービスへ切り替える際にネットワークから切断されていると，ステート移行のための通信ができない．ネットワークから切断されている間に利用していたサービスがつねに利用されたままになったり，異常終了してステートを次のサービスに移行できなかったりする場合が発生する．

2.2.2 本研究の目的

前項のサービスの分類をふまえ，本論文では以下の 3 つを目的とし，これらを実現することで本研究の目標である Mobiquitous 環境の基盤技術を実現する．

利用中断に対する処理の実行

クライアントがサービス利用の中断を行う前に，サービスとのステートを交換し，また中断されたくない処理を実行完了できることを支援し，これらの処理が正常に行われるようにする．

サービスの多くがクライアントとの間にステートを共有する．クライアントがサービス利用を中断する際に，中断のための事前処理を正常に行わないと，サービスとのステートが不一致となる場合が考えられる．また，クライアントとサービスが重要な通信を行っている場合，通信中にサービス利用を中断されると問題が生じる場合も考えられる．

その結果，サービス利用の中断・再開を行うには，利用中断時にサービスとクライアントのステートの一貫性を保つべく，中断のための処理を正常に行う必要が生じる．

サービスとの同期

サービスとの同期とは，クライアントとサービスとのステートを共有することを意味する．その場合，クライアントやサービスの一方のステートが変化した場合，他方にそのステートが変化したことを通知して，一貫性を保つことが必要である．

サービスとの同期を正常に行えることにより，サービス利用の中断をしている間，サービスだけで，もしくはクライアントだけで自身のステートの変化に影響しない部分の処理を進めることができる場合もある．クライアントやサービスの処理の中でステートの変化に関係のない処理が存在する場合，サービスとの同期をとることなく，その処理を進められる．

これにより，モバイルホストがネットワークから切断されていても一部の処理をサービスは実行できる．しかし，通常クライアントがサービス利用可能な状況にいるかいないかをサービスが把握できないため，クライアントとの同期をとった処理を行えない．

ネットワーク切断の予見検知

クライアントが上記 2 つのことを実現できるよう，クライアントはネットワーク切断によるサービス利用中断を事前に予見する．

クライアントは無線 LAN によるネットワーク接続されることが考えられる．そのため，利用者の移動によってネットワーク接続が知らないうちに切断・再接続されることが生じる．ネットワーク切断によるサービス利用中断の場合，ネットワークが切断された段階で，上記 2 つの処理を行おうとしても無理である．クライアントとサービスが通信できないため，中断のための事前処理もサービス利用不可能な状況にいることサービスに伝えることを行えない．

そのため，ネットワーク切断を予見し，実際にネットワークが切断される前に上記 2 つの処理を行うことが必要になる．

3. システムの設計

本章では，本論文で提案する ASDS のシステム概要についてまず述べ，その後，設計について述べる．

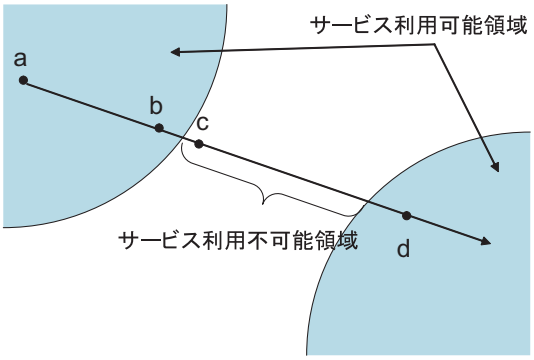

図 2　ASDS が想定するシナリオ
Fig. 2　ASDS's scenario.

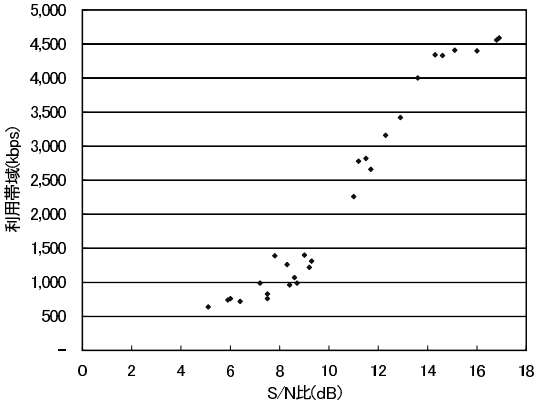

図 3　S/N 比と転送速度の関係
Fig. 3　SNR and throughput.

3.1　システムの概要

ASDS は，サービスの利用状況の変化に動的に支援することを支援するための基盤ミドルウエアである．**図 2** に利用者が利用可能領域（図中の a 地点）から不可能領域（c 地点）へ，そして再び可能領域（d 地点）へ移動しながら，サービスの利用を行う，ASDS が想定するシナリオを示す．

ASDS では，第 1 にサービス利用可能状況のうちに（b 地点），サービス利用不可能になる可能性があることを予見してクライアントにその旨を通知する．サービスが利用不可能な状況になったことをクライアントが検知するには，実際にサービス利用不可能な状況になる（c 地点）のを待たずして行える．そのため，ネットワーク切断によってサービス利用不可能になる直前（b 地点）でサービス利用中断の事前処理を行える．

第 2 にクライアントがサービス利用可能状況にいるか不可能状況にいるかをサービスに動的に通知するための機能を提供し，クライアントとサービスが同期をとれるようにする．クライアントがサービス利用不可能な状況（c 地点〜d 地点）でも，サービスは処理を部分的に進めることが可能な場合が考えられ城．たとえば，Lazy Evaluation を利用した Disconnected Operation があげられる．サービスがクライアントの有するステートを参照しようとした際，実際にサービスがその参照した値を扱わない限り，サービスの処理を進め，値を扱うときにようやく処理をブロックされる．別の例として，ネットワーク切断時に，クライアントのステートと独立した処理をサービスが先に行うこともあげられる．

第 3 にクライアントがサービス利用不可能状況から可能状況になった（d 地点）ことを動的に検知し，サービス利用再開のための処理を実行する．クライアントがネットワークへ接続できることを検知して，次に利用するサービスへステートを移行し，またサービスへ利用可能状況になったことを通知する．それにより，別のサービスに切り替えて継続的な利用を実現し，サービスの処理を再開できる．

3.2　ASDS の設計

本節では，本論文が提案する ASDS の設計について述べる．ASDS は 3 つの機能を提供し，利用者が移動しながらでもサービスを適応的に利用できる環境を提供する．以下では，ASDS が提供する 3 つの機能についてそれぞれ設計を行う．

3.2.1　無線 LAN の切断予見・接続検知方法

利用者が移動した際に無線 LAN の S/N 比と基地局の設置場所に関する情報によって，ネットワーク接続が切断されることを予見するための方法について述べる．ASDS では無線 LAN の S/N 比とその変化の度合い，および周辺の基地局の設置情報を利用する．

S/N 比の利用

無線 LAN は，無線 LAN の基地局から利用者が離れるに従い，S/N 比が低下していく．そして S/N 比が低下しすぎると基地局との通信が不可能になる．S/N 比は以下のような計算式で算出できる．SNR は S/N 比の値を，S は信号電力を表し，N は雑音電力を表す．S/N 比の単位は（dB）である．

$$SNR = 10\log_{10}\left(\frac{S(W)}{N(W)}\right) \qquad (1)$$

この特性を利用し，S/N 比の閾値を定め，その閾値より低下したら，ネットワーク接続が切断しそうであると決める．逆に閾値を超えたら，ネットワークに接続したと判断する．S/N 比の閾値を定めるために，S/N 比と転送速度の関係を実際に NetPerf[16] を利用して測定した．その結果を**図 3** に示す．

この結果をもとに，ASDS では切断の基本閾値を 8,

接続の基本閾値を 16 に定める．S/N 比の値が接続の閾値よりも高ければネットワークに接続できる．逆に切断の閾値より低ければ切断されたと判断する．それぞれの閾値を用意することで，S/N 比のブレによる急激な変化を防ぐ．

また，無線 LAN の基地局が設置されている環境によって，閾値を動的に変更できる．障害物が多い所では，S/N 比が 15〜12 から急激に 0 へ減少し，切断されることがある．次項で述べる基地局の設置情報に含まれる閾値情報を利用し，その基地局の環境で最適な閾値を利用して切断，接続の検知を行う．

さらに，S/N 比の閾値とともに，S/N 比の変化にも着目する．S/N 比の変化をもとに，現在，利用者が無線 LAN の基地局から離れていってるのか，近づいていっているのか，それとも止まっているのかを判断する．S/N 比が減少していれば離れており，増加していれば近づき，変化していなければ止まっていることになる．S/N 比だけでは事前処理をする間に基地局から離れてしまい，ネットワークが切断される可能性がある．そのため，変化の度合いからある程度先のことを予測することが重要となる．

ASDS では，S/N 比の過去の変化の度合いを監視し，減少傾向でかつ S/N 比の閾値に達しそうな場合に，ネットワークの切断が発生しそうだと判断する．逆に増加傾向にありかつ S/N 比の閾値に達した場合にネットワークに接続したと判断する．

基地局の設置および情報の利用

ASDS では，無線 LAN の S/N 比によってネットワークの接続，切断を検知する．そのため，無線 LAN の基地局の設置場所は，隣接する無線基地局とセルの一部が重なるように設置する．もし，隣接するセルが離れている場合，セル間をローミングしながら移動すると，頻繁な切断，接続が発生する．

また，ASDS では無線 LAN の切断予見・接続検知方法として，基地局の設置情報も利用し，より正確な予見，検知を行う．ASDS が利用する基地局の設置情報としてその基地局周辺で用いる切断検知の閾値があげられる．

この情報はクライアントがネットワーク接続時に基地局から取得する．基地局がネットワーク管理者によって入力されたこれらの情報を保持する．基地局周辺に障害物が多くあると，S/N 比の減少が急に生じる場合がある．そのため，障害物が周辺に多く存在する基地局の場合，閾値の値を高めに設定する必要がある．

3.2.2 サービス利用中断・再開の処理

サービス利用の中断・再開に必要な処理として，4 つのことがあげられる．

- 利用中断に対する事前処理
- 利用中断によるエラー処理
- 利用再開に対する処理
- 利用中断中の処理

サービスとクライアントとがステートの共有を行う場合，この 4 つの処理を通じて行う．まず，利用中断の事前処理によって，サービスとクライアントがお互いのステートの交換を行い，ステートのスナップショットを取得する．もし，事前処理を行う前に実際に利用不可能になり，スナップショットを取得することに失敗した場合，エラー処理を実行する．利用中断中はお互いのステートのスナップショットを利用するなどして，相手のステートに影響を与えない部分の処理を行ったり，処理をブロックしたりできる．再び，サービス利用可能になると，サービスとクライアントがお互いのステートを交換しあい，ステートの一貫性をとる．

本項では，これら 4 つの処理の実行支援について述べていく．

中断・再開のための事前処理の実行支援

利用中断に対する事前処理にはサービスとクライアントとのステートやりとりや中断時の処理に関するやりとりなどが考えられる．中断不可能な処理を行う間にネットワーク切断が生じそうになると，利用者に注意を促す処理もこれに含まれる．

また，事前処理や中断不可能処理を行う間にネットワークが切断された場合，致命的なエラーが発生することが想像できる．これらのエラーに対してクライアントがエラーからの復旧処理や例外処理を行うことが必要となる．

中断した後に，ネットワークに接続されサービスが利用可能になるとサービス利用再開の処理も当然必要になる．サービスとの接続を再生成し，切断中の処理結果をやりとりすることにより，サービスとクライアントとのステートの一貫性をとれる．

これらの処理はクライアントとサービスの種類によって内容が異なる．本論文では，ASDS が統一的な処理を行うのではなく，クライアントがその種類に応じてプログラムを記述できるほうが有益であると考える．統一的に処理を行うことで，クライアントになんらかの制限を加えたり，利用するサービスに特化した処理ができなかったりすることが考えられるためである．重要なのは，それらの処理を適したタイミングで実行できるような仕組みを ASDS が提供することである．

そこで，クライアントは起動時に，利用中断に対す

る事前処理やエラー処理，再開に対する処理を登録する．ASDS はクライアントのサービス利用状況の変化に応じて登録された処理を実行する．前項で述べたネットワークの切断予見・接続検知機能を利用し，ネットワークの切断によってサービスが利用できなくなりそうだと，切断のための事前処理を実行し，ネットワークに接続したことでサービスが利用できるようになると再開のための処理を実行する．また，中断不可能な処理を実行している間にネットワークが切断された場合，エラー処理を実行する．

利用中断中のサービス・クライアントの処理実行支援

ASDS では，クライアントが利用を中断している間，サービスは処理できるところは処理を進め，クライアントと同期しなければならない処理はその処理をブロックできるようにする．これによって，クライアントがサービス利用を中断した間でもクライアントとサービスは同期をとる必要のない処理を行え，効率的なサービス利用を実現できる．

サービスがクライアントと同期をとらなければならない場合は，サービスがイニシエータからなんらかのデータを受け取ったり，処理結果をクライアントに送信したりするときであると考えられる．サービスが処理を終了して次の処理要求をクライアントから受け取る場合，クライアントからのコマンドを待たなければならない．同様にクライアントがサービスからの処理結果を必要とする場合，結果を受信するまで処理を中断していなければならない．

そこで，クライアントがサービスと通信を行う際，クライアントがサービス利用不可能だと，自身の処理を中断できるようにする．クライアントとサービスがクライアントのサービス利用状況を管理し，両者がデータを送受信する際に，クライアントの状況を確認する．クライアントがサービス利用可能ならばデータを送受信し，もし不可能なら状況が変化するまで処理を中断する．

その際，クライアントが長い間，利用を中断すると他の利用者に影響がでるため，処理の中断期間に期限を設定できるようにする．クライアントが利用中断した結果，サービスの処理もブロックされると，それ以外の処理を行えない場合が考えられる．また，利用を中断している間，サービスはクライアントとのステートをずっと保持し続けなければいけない場合も生じる．これらは他の利用者にもサービスにも効率的でない．これらを避けるべく，サービスは中断期間に制限を設定できることが重要である．

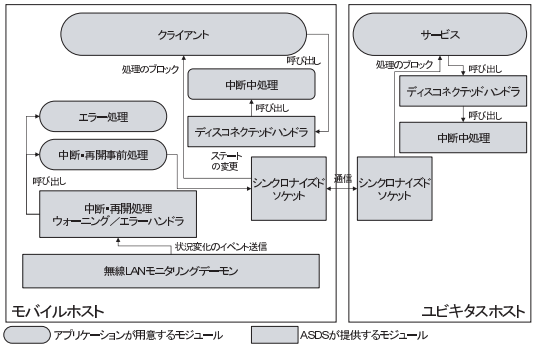

図 4 ASDS：システム構成図
Fig. 4 ASDS: system architecture.

4. ASDS の実装

本章では，3 章で述べた設計に基づき，ASDS の実装方法について述べる．まずシステムの概要について記述し，その後，詳細について述べる．なお，本実装は FreeBSD 4.6-RC 上で MELCO 社製の無線 LAN カード WLI-PCM-L11 を用いて C 言語で実装した．

本研究では PDA などをモバイルクライアントホストとして想定している．すでに Linux は Compaq 社の iPaq などの PDA 上で動作しており，NetBSD の移植も現在行われている．そのため，FreeBSD 上で本プロトタイプ実装を行ったことは妥当であると考える．

4.1 システム概要

ASDS は，無線 LAN モニタリングデーモン，中断・再開処理ウォーニング/エラーハンドラ，ディスコネクテッドハンドラ，およびシンクロナイズドソケットの 4 つのモジュールを提供する．アプリケーションおよびサービスをプログラムする場合，利用するサービスに合わせて中断・再開事前処理，中断中処理，エラー処理を用意する．**図 4** に ASDS のシステム構成図を示す．

無線 LAN モニタリングデーモンは無線 LAN の S/N 比の変化を監視する．利用者が移動し，S/N 比が低くなると，ネットワークの切断を予見し，S/N 比が閾値を超えると，ネットワークが切断されたと判断する．ネットワークの切断を予見，切断，または再接続を検知した場合，各クライアントの中断・再開処理ウォーニング/エラーハンドラを呼ぶ．

中断・再開処理ウォーニング/エラーハンドラはクライアントのサービス利用状況を管理し，必要に応じてクライアントが用意するコールバック関数を呼ぶ．無線 LAN モニタリングデーモンから通知を受けた中断・再開処理ウォーニングハンドラは，通知の内容に従い，事前にクライアントから登録されたコールバッ

ク関数を呼び，クライアントが用意した中断・再開事前処理もしくはエラー処理を実行させる．また，利用中断中にアプリケーションやサービスが相手のステートを参照する場合，ディスコネクテッドハンドラが呼ばれ，中断中処理が実行される．なお，このディスコネクテッドハンドラを用いた同期対応は，本論文ではまだ実装されておらず，今後の課題となっている．

シンクロナイズドソケットはサービスやクライアントをプログラムする際，事前，中断中，再開処理のためのコールバック関数を支援するために利用される．クライアントの利用状況に応じてクライアントとサービスのソケットの処理をブロックする．クライアントの利用状況はクライアントや利用者が明示的に変更する．

4.2 無線 LAN モニタリングデーモン

無線 LAN モニタリングデーモンはシステムコールを用いて無線 LAN の信号電力と雑音電力を取得して S/N 比を計算し，その変化の直線回帰を求めてネットワークが切断されることを予見する．

無線 LAN モニタリングデーモンは 200 msec ごとに信号電力と雑音電力を取得する．FreeBSD では，無線 LAN の信号電力（dBm）と雑音電力（dBm）を `ioctl` システムコールを通じて取得できる．信号電力と雑音電力は無線 LAN の基地局からデータを受信するごとに更新される．また，基地局からは 100 msec〜200 msec の周期的な間隔でビーコンが発信される．100 msec ごとに取得する信号電力と雑音電力から S/N 比を計算する．

無線 LAN モニタリングデーモンは過去 3 秒間の S/N 比の値から直線回帰を求め，およそ何秒後にネットワークが切断されるかを計算する．回帰直線の傾きを a，その切片を b と表す．また時間 i の S/N 比の値を SNR_i，3 秒間の平均を $\overline{SNR}$，時間 i の値を T_i，3 秒間の平均時間を $\overline{T}$ と表すと以下の計算式で示せる．

$$a = \frac{\sum_{i=0}^{n}(SNR_i - \overline{SNR})(T_i - \overline{T})}{\sum_{i=0}^{n}(SNR_i - \overline{SNR})^2} \quad (2)$$

$$b = \overline{T} - a\overline{SNR} \quad (3)$$

クライアントは起動時に切断予見時間の何秒前に次節で述べるウォーニング/ハンドラを呼んでほしいか指定する．その指定された時間よりも短い未来で切断されることが予見された場合，ウォーニング/エラーハンドラにイベントを送信し，クライアントは事前処理を行ったり，利用者への注意を促したりする．

逆に切断と接続の検知は純粋に S/N 比が閾値を超えたことで判断する．切断を検知すると，ウォーニング/エラーハンドラにイベントを送信する．また接続を検知するとウォーニング/エラーハンドラにイベントを送信する．なお，イベントの送信にはイベント配送メカニズムである環境情報サーバ [17] を拡張して利用する．

4.3 中断・再開処理ウォーニング/エラーハンドラ/ディスコネクテッドハンドラ

サービスの利用中断・再開をアプリケーションが正常に行えるよう，ネットワークの切断予見・切断・再接続時にクライアントやサービスのコールバック関数を呼ぶ．これにより，クライアントやサービスがサービス利用状況の変化に動的に適応することを支援する．

クライアントやサービスは `setCallBackFunction` 関数を用いて事前処理や事後処理，エラー処理や中断中処理のためのコールバック関数を設定する．サービスは中断中処理のみを設定できる．**図 5** に setCallBackFunction の関数宣言を示す．

Type には，これから第 2 引数が示すコールバック関数が事前処理，事後処理，エラー処理，中断中処理のどれかを示す．各コールバック関数は `int (*FunctionName)(void *Arg)` に従い，記述されなければならない．これにより，クライアント，サービスが必要とする各処理をプログラム作成者が自由に用意できる．**図 6** にクライアントのサービス利用状況の状態遷移の様子を示す．

現在，クライアントがサービス利用可能な状況だとする．環境情報サーバを通じて無線 LAN モニタリングデーモンから切断予見のイベントをウォーニング/エラーハンドラが受けると，利用中断の事前処理を行わせ，サービス利用不可能な状況に変わる．事前処理を行わずサービスを利用し続けて切断のイベントを受けるとエラー処理が実行され，その後，サービス利用不可能な状況に変わる．クライアントがサービス利用不可能な状況にある場合，無線 LAN モニタリングデーモンから接続検知のイベントを受信すると，接続に必要な事前処理が実行され，サービス利用可能な状況に変わる．

```
int setCallBackFuntion
(int Type, int (*FunctionName)(void *Arg));
```

図 5 setCallBackFuntion() 関数
Fig. 5 Definition of setCallBackFunction().

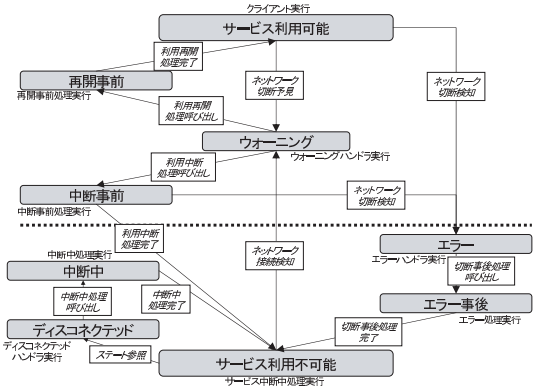

図 6 サービス利用状況の状態遷移図
Fig. 6 Initiator's service availability state transition diagram.

```
int prefetchData (struct *State);
    // 事前処理の例
int readData (struct *Data);
    // 中断中処理の例
int getBufferedData (struct *NewState);
    // 再開処理の例
int resetState (void *NULL);
    // エラー処理の例
```

図 7 各コールバック関数例
Fig. 7 Examples of callback function.

各ハンドラおよびコールバック関数の動作例として，5.3 節で利用する MP3 ストリーミングクライアントを見てみる．MP3 ストリーミングクライアントが用意した各コールバック関数を**図 7** に示す．これらの処理をすることで，MP3 ストリーミングクライアントは，ASDS を利用しない場合に比べ，長時間の利用中断を可能にし，サービス利用中断中でも長時間継続した音楽再生が可能となる．

サービス利用中断前に呼ばれる `prefetchData()` 関数は，サービスが有するデータを利用中断中に参照できるよう，中断前にできるかぎり取得する．

`prefetchData()` 関数は利用中断中に呼ばれ，中断前にあらかじめ取得したデータを読み込める．利用再開時に呼ばれる `getBufferedData()` 関数は，すでに取得したデータの続きをサービスから取得できるようにサービスにステートを渡す．`resetState()` 関数は，中断前にデータを取得できなかった場合に，ユーザにエラーメッセージを提示する．

4.4 シンクロナイズドソケット

シンクロナイズドソケットは，サービスやクライアントをプログラムする際，事前，中断中，再開処理のためのコールバック関数作成を支援するために利用される．クライアントからのサービス利用中断・再開の処理によって，クライアントとサービスとのコネクションを動的に生成し，利用を中断する間はそのコネクションへの読み込み，書き込みをブロックする．

クライアントは `setServiceAvailabilityState` 関数を呼ぶことで，シンクロナイズドソケットをコントロールできる．クライアント，サービス両者はサービス利用状況を表す `ServiceAvailabilityState` フラグを管理し，このフラグがオンだとクライアントがサービス利用可能状況に，オフだと不可能状況にいることを示す．フラグがオフのときにクライアントやサービスがソケットに読み書きしにいくと処理がブロックされる．利用可能な状況に変化すると，ブロックが解除されサービス利用が再開される．

`setServiceAvailabilityState` 関数は，切断のための事前処理や接続のための処理の際に呼ばれることを想定する．なぜなら，これらの処理の際に行うことで，サービス利用状況の状態変化と一致できる．中断・再開処理時に関数が呼ばれるようにすることで，動的にしンクロナイズドソケットを制御できる．

また，`setSynchronizeTimeOut` 関数を利用することで，ブロックのタイムアウト時間を設定する．クライアントがサービス利用中断した際でも，サービスの資源は利用され続ける．もし，クライアントが長い間利用を中断すると占有されたサービスの資源は有益に利用されているとはいえない．この関数を利用してタイムアウト時間を設定することで，サービスの無駄な占有や計算資源の浪費を防げる．

5. 評　　価

本章では，ASDS の評価について述べる．本論文では，まず 1 つの無線 LAN の基地局を利用したホットスポットなどを想定した単純な環境で ASDS が正確に動作するか検証した．次に実際の無線 LAN ネットワーク網において ASDS が動作することを示す．無線的に理想的な環境と，日常的に利用されている複雑な環境での実験を行うことにより，一般的な結論を導く．また，応用アプリケーションとして，ASDS を利用した MP3 プレイヤを実際に実装し，ASDS を使わない他のケースよりも有効であったことを示す．

5.1 ネットワーク切断予見の動作検証

ASDS の評価として，S/N 比の変化を監視して実際に切断予見の処理，切断時のエラー処理，接続時の処理が行えるか，動作検証する．今回の動作検証では，単純に障害物がない理想的な環境である場所で，無線 LAN の基地局 1 つに対し，利用者が移動して ASDS

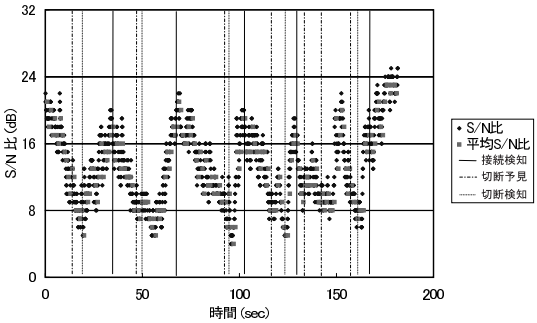

図 8 S/N 比の変化と各種処理
Fig. 8 SNR and each performances.

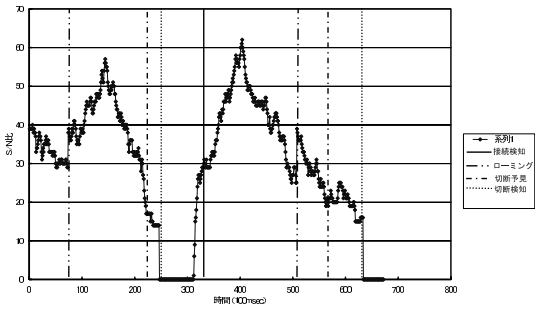

図 9 無線ネットワーク網における S/N 比の変化と各種処理
Fig. 9 SNR and each performances in wireless campus network.

が動作するかを確認する．1 台のノート型 PC 上にクライアントのサンプルアプリケーションを実行させ，そのクライアントは，切断を予測した推定時間から 5 秒前に通知するように設定した．この値は，クライアントによってサービス利用中断の事前処理に必要な時間がかわるため，ASDS が一意に決定できる値でないためである．利用者はクライアントが動作するノート PC を持ちながら実際に移動し，その変化に適応する．**図 8** にその際の S/N 比の変化と各種処理が実行された時間を示す．

まず，S/N 比が激しく変動しているのが分かる．単純に閾値を設定し，切断・接続を検知すると，閾値付近では頻繁に切断・接続が発生してしまう．クライアントがそのつどサービス利用状況の変化に適応していてはオーバヘッドが高いうえに，利用者にとっても不便である．ASDS を利用することで，S/N 比の変動のブレを吸収し，利便性の高い切断・接続検知機能を提供できる．

また，切断予見に関しても，実際に切断される前に予見を行えている．125 秒あたりからの連続的な切断予見は実験者が切断予見に基づいて一度戻り，また移動を行うことをし，閾値付近を行き来したためである．この際も S/N 比が減少傾向にある際に切断を予見し，それ以外は予見をしていない．

このように，取得できる S/N 比にブレが発生しているにもかかわらず，ネットワークの切断予見・検知，接続検知を行えることは，ユビキタスコンピューティング環境を移動する利用者にとってとても重要なことである．

5.2 実証実験

ASDS の実証実験として，実際に利用されている無線 LAN ネットワーク網において ASDS がネットワークの切断・接続の検知を行えることを示す．本論文では，慶應義塾大学湘南藤沢キャンパスに敷設されたキャンパス内無線ネットワーク網を利用する．このキャンパス内無線ネットワーク網はキャンパス中に基地局を設置し，学生が所有するノート PC がネットワークに接続するために利用されている．前節の理想的な環境と本節の複雑な環境のもとで実験を行うことで，一般的な結論を導く．

実証実験では，ノート PC 上にクライアントを実行させ，切断を予見した推定時刻から 2 秒前にクライアントに通知するように設定した．利用者はクライアントが動作するノート PC を持ちながら実際にキャンパス内を移動し，利用する無線 LAN の基地局の間をローミングしながら，その変化に適応する．**図 9** にその際の S/N 比の変化と各種処理が実行された時間を示す．

約 9 秒および 50 秒付近で急激に S/N 比が増加している．これはローミングが発生したためである．約 25 秒付近でネットワークから切断されるが，ASDS は約 23 秒付近で切断の予見に成功している．約 30 秒付近で再びネットワークに接続し，約 63 秒付近で再びネットワークから切断される．その際も約 56 秒付近で切断の予見に成功している．

このように，ASDS は既存の無線ネットワーク網においてもその機能を発揮できる．ASDS が一般的な無線ネットワーク網においてネットワークの切断・接続の検知および切断の予見を行えることは Mobiquitous 環境を実現するうえでとても有意義なことであると考える．

5.3 MP3 プレイヤアプリケーションを用いた実験評価

ASDS を用いた応用評価として，MP3 をストリーミングするサービスを利用し，サービス利用中断・再開が生じた場合の ASDS の優位性を示す．本評価では，ASDS を利用して利用中断前に事前処理を行った場合（ケース 1），ステートが存在する場合（ケース

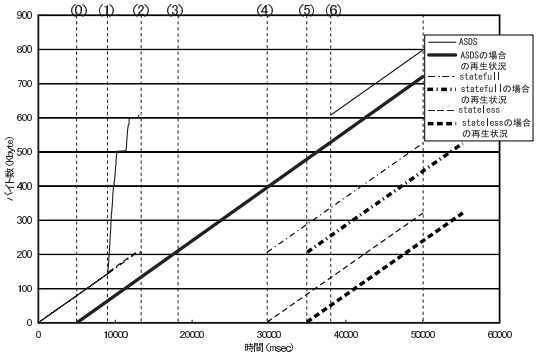

図 10 MP3 プレイヤの受信量と再生の関係
Fig. 10 MP3 player evaluation result.

2)，サービスとクライアントの間でステートが存在しない場合（ケース 3）との比較を行う．

本評価では，クライアントは MP3 ストリーミングサービスから 128 Kbps で 44.1 KHz の MP3 データを 128 Kbps のコンスタントビットレート（以下 CBR とする）で受信する．クライアントは 5 秒分のバッファリングを行い,（0）で示される時刻から受信した MP3 を再生する．その際，クライアントは一度，サービス利用可能な状況から不可能な状況になり，しばらくした後，再び利用可能な状況へ置かれる．**図 10** に評価結果を示す．

まず，すべてのケースにおいて，最初からサービス利用可能な状況にある.（2）で示される時刻に向かうにつれ，サービス利用不可能な状況に近づく．ケース 2 と 3 はサービス利用不可能な状況になるまで，同じ動作を行う．ケース 1 だけは，サービス利用不可能な状況になることを（1）の時刻に予見し，CBR で受信していたストリーミングをベストエフォートで受信するように処理内容を変更し，できる限り，データをバッファリングする．

（2）の時刻になると，ケース 2 と 3 はデータを受信できなくなるため,（3）の時刻，つまりバッファリングした 5 秒分しか再生できない．ケース 1 はケース 2, 3 と比べ，事前処理としてできる限りのデータをバッファリングしたため，データを受信しなくても，長時間，音楽を再生し続けられる．受信せずに MP3 を再生できる時間は，ASDS が中断を予測した将来の時刻から何秒前にクライアントに通知するかによって変わる．再生可能な時間を T（秒）とし，クライアントが予見の通知を受けた時刻から中断するまでの時間を P（秒），そのときの転送速度を $S(P)$（Kbps）とする．また，受信する MP3 のビットレートを R（Kbps）とすると，T と P の関係を次の式で表せる．

$$T = f(P) \tag{4}$$

$$T = \frac{\sum_{i=0}^{P} S(i)}{R} \tag{5}$$

この式をもとに，今回のケース 3 の場合を計算すると，データを受信できなくても最大 25 秒間，音楽を再生し続けられる．

（4）の時刻になると，すべてのケースにおいて，サービスを利用できるようになる．ケース 3 は，サービスとクライアントとの間でステートが共有されていないため，再びデータの最初から受信を開始し，5 秒間のバッファリングを行い,（5）の時刻で音楽の先頭から再生を開始する．ケース 2 の場合，ステートの共有が行われているため,（2）の時刻までに受信したデータの続きを受信し，5 秒間のバッファリングの後,（3）の時刻までに再生した分の続きから再生を開始する．ケース 1 の場合，事前処理でバッファリングしたデータが基本値の 5 秒よりも多いため，クライアントは再接続した時点である（5）の時刻では，まだデータを受信せず音楽の再生を行う．クライアントはバッファリングしたデータサイズからデータ受信の再開時刻を計算し,（6）の時刻になってようやくデータの受信を開始する．

このように，ASDS によって，サービスとクライアントとのステートの共有，およびサービス利用不可能な状況になることの予見を実現することは，Mobiquitous 環境において利用者に有益であることが示された．

6. ま と め

本論文では，ユビキタスコンピューティング環境での利用者の移動にともなうサービスの利用状況の変化，特にネットワークの切断・接続による変化に対し，クライアントが動的に適応できることを支援するための ASDS を提案し，設計・実装を行い，評価を行った．

第 1 に，ASDS はネットワーク切断予見を実現し，無線 LAN 利用時の曖昧なサービス利用中断・再開を正確に検知できる．無線 LAN の S/N 比と転送性能の実測値をもとに閾値を設定し，閾値を超えたことで切断・接続を検知する．また，S/N 比の直線回帰を求め，閾値を超える時間を計算し，ネットワークの切断を予見する．

第 2 に，ASDS を利用することで，クライアントがサービス利用中断・再開に必要な事前処理を正常に行える．ウォーニング/エラー/ディスコネクテッドハンドラにクライアントが用意する処理を登録するだけで，

必要なタイミングで各処理を実行できる．また，利用中断の事前処理に失敗した場合に備え，エラー処理を適切に行うことも支援できる．

第3に，クライアントがサービス利用を中断している間でもクライアントおよびサービスは処理を進められる．シンクロナイズドソケットを利用することで，クライアントとサービスの通信が必要ある処理を行う場合のみ処理がブロックされる．クライアントがサービス利用可能な状況になるとブロックが解除される．

ASDSはこれら3点を実現することにより，利用者の移動によって変化するサービス利用可能状況にクライアントやサービスが動的に適応できるように支援する．本論文ではASDSの設計，実装を行い，評価を行った．

ASDSの評価として，本論文ではサンプルプログラムを用いて利用者が実際に移動した場合，ネットワーク切断の予見・検知，接続の検知を行えるかを調べた．最初に無線LANの基地局1つに対し，利用者が移動する場合を調べた．単純にS/N比の閾値だけで行う場合，取得できるS/N比のブレによって正確に変化を監視できない．直線回帰を利用することで，正確な切断の予見を行える．また，一般的な既存の無線ネットワーク網での実証実験を行い，ASDSがネットワーク接続・切断の検知および切断の予見を行えることを示した．また，実際にMP3プレイヤを実装して，ASDSを使用する場合としない場合とで動作の比較を行い，ASDSの有効性を示した．

今後の課題として，ディスコネクテッドハンドラの実装と，利用中断事前処理，中断中処理，再開事前処理，エラー処理のパターン化，テンプレートの作成の2つがあげられる．ディスコネクテッドハンドラは利用中断中にサービスのステートを参照する際に，中断中処理を実行する．これにより，利用中断中の同期対応が正常に行われ，ディスコネクテッドオペレーションを実現できる．また，本論文では，利用中断事前処理，中断中処理，エラー処理をサービスやクライアントを作成するプログラマに用意してもらう必要がある．これらの処理をパターン化，テンプレートを作成することで，サービス，クライアント作成の容易化，プログラマの負担軽減をできる．

参 考 文 献

1) Weiser, M.: Computer of the 21st Century, *Scientific American*, No.3, pp.94–104 (1991).
2) Mann, S.: Wearable Computing: A First Step Toward Personal Imaging, *IEEE Computer*, Vol.30, No.2, pp.25–32 (1997).
3) Sun Microsystems, Inc.: Jini Architecture Specification (1998). http://www.javasoft.com/products/jini/specs/jini-spec.pdf
4) Universal Plug and Play Forum: Universal Plug and Play (UPnP) (1999). http://www.upnp.org/
5) Harter, A., Hopper, A., Steggles, P., Ward, A. and Webster, P.: The Anatomy of a ContextAware Application, *The 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (*MOBICOM'99*), pp.59–68 (1999).
6) Sousa, J.P. and Garlan, D.: Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments, *The 3rd Working IEEE/IFIP Conference on Software Architecture* (2002).
7) Nakazawa, J., Okoshi, T., Mochizuki, M., Tobe, Y. and Tokuda, H.: VNA: An Object Model for Virtual Network Appliances, *IEEE International Conference on Consumer Electronics* (*ICCE2000*) (2000).
8) Gribble, S.D., Welsh, M., von Behren, R., Brewer, E.A., Culler, D., Borisov, N., Czerwinski, S., Gummadi, R., Hill, J., Joseph, A., Katz, R., Mao, Z., Ross, S. and Zhao, B.: The Ninja Architecture for Robust Internet-Scale Systems and Services, *Computer Networks*, Vol.35, No.4, pp.473–497 (2001).
9) Wang, H.J., Raman, B., nee Chuah, C., Biswas, R., Gummadi, R., Hohlt, B., Hong, X., Kiciman, E., Mao, Z., Shih, J.S., Subramanian, L., Zhao, B.Y., Joseph, A.D. and Katz, R.H.: ICEBERG: An Internet-core Network Architecture for Integrated Communications, *IEEE Personal Communications*, Vol.7, No.4, pp.10–19 (2000).
10) 東京大学青山・森川研究室，慶應義塾大学徳田研究室：近未来志向ユビキタスコンピューティング環境実現のためのインターネットスケール実験システムを本格稼働．http://www.mlab.t.u-tokyo.ac.jp/events/stonepress2002/
11) Perkins, C.: IP mobility support, Internet Request For Comments RFC 2002 (1996).
12) Bakre, A.V. and Badrinath, B.R.: I-TCP: Indirect TCP for Mobile Hosts, *International Conference on Distributed Computing Systems* (*ICDCS*), pp.136–143 (1995).
13) Funato, D., Yasuda, K. and Tokuda, H.: TCP-R: TCP mobility support for continuous operation, *International Conference on Network Protocol* (*ICNP*), pp.229–236 (1997).

14) Joseph, A.D., deLespinasse, A.F., Tauber, J.A., Gifford, D.K. and Kaashoek, M.F.: Rover: A Toolkit for Mobile Information Access, *The 15th ACM Symposium on Operating System and Principles* (*SOSP*), pp.156–171 (1995).
15) Maltz, D. and Bhagwat, P.: MSOCKS: An Architecture for Transport Layer Mobility, *The 17th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp.1037–1045 (1998).
16) Information Network Division Hewlett-Packard Company: A Network Performance Benchmark. http://www.netperf.org
17) 西尾信彦，徳田英幸：EISS：環境情報サーバスイートを用いたシステムの状況適応，情報処理学会コンピュータシステムシンポジウム，Vol.97, No.8, pp.1–8 (1997).

(平成 14 年 7 月 8 日受付)
(平成 14 年 12 月 3 日採録)

永田　智大

1998 年慶應義塾大学総合政策学部卒業．2000 年同大学院政策・メディア研究科修士課程修了．現在，同大学院政策・メディア研究科後期博士課程に在学中．ウェアラブル・ユビキタスコンピューティング，適応支援ミドルウェア等の研究に従事．

西尾　信彦（正会員）

1992 年東京大学大学院理学系研究科情報科学専攻博士課程所定単位取得後退学．1993 年より慶應義塾大学に勤務，同大学より博士（政策・メディア），現在，政策・メディア研究科助教授．連続メディア処理システムの研究開発，分散リアルタイムシステム，動的 QOS 制御に関する研究に従事．1994 年山下記念研究賞受賞．

徳田　英幸（正会員）

慶應義塾大学より工学修士．カナダ，ウォータールー大学より Ph.D.（Computer Science）．現在，慶應義塾大学常任理事，同大学環境情報学部教授．分散リアルタイムシステム，マルチメディアシステム，通信プロトコル，超並列・超分散システム，モバイルシステム等の研究に従事．IEEE，ACM，日本ソフトウェア科学会各会員．