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動的に適応できるように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．

Design and Implementation of Adaptive Service Disconnect 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 user the services around him effectively while migrating.

1. はじめに

近年，ユビキタスコンピューティング環境[14]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したり，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環境を提供する．

このような背景として，ネットワークに接続できる様々な機器が市場にでてきたことが挙げられる．すでに，IEEE 1394 や USB を利用してネットワークや PC に接続できる TV やビデオレコーダといった AV 機器が市販されている．また，Web カメラや位置センサなどもイーサネットやシリアルケーブルによってネットワークに接続できる．さらに，電灯線などを利用した白物家電に関する研究も進んでいる．

また，他にも無線を利用したネットワークにより，様々な場所でネットワークに接続できることが挙げられる．IEEE 802.11b や PHS による無線ネットワークの普及により，利用者は移動しながらでもノート PC や PDA，ウェアラブルコンピュータ[7]を利用してネットワークに接続できる．

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 Environment” (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，周辺の機器やその上で動作するアプリケーションをサービスと呼ぶ．Mobiquitous

† 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
†† 科学技術進行事業団，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

Environment 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を携帯し，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・再開を正常に行うことで，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 Environment 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アプリケーション (本論文ではイニシエータと呼ぶ) 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．イニシエータ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，切断前にサービス利用中断に必要な事前処理を正常に行える．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イニシエータ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，サービスとイニシエータのステートを同期できるため，サービス利用を正常に再開できる．

本論文では，ASDS の設計，プロトタイプ実装を行い，プロトタイプの実証実験を行うことで ASDS の利便性を示す．

例えば，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「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」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との違いを示す．第 3 章では，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．第 4 章では，本論文が提案する ASDS の動作概要について述べ，第 5，6 章では ASDS の設計・実装について言及する．第 7 では ASDS の評価を行う．

2. 関 連 研 究

本章では，ユビキタスコンピューティング環境に関する先行研究についてまず述べる．その後，従来のモバイルコンピューティングに関する技術をそのままユビキタスコンピューティング環境で利用する際の問題点を挙げる．

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[11] や Microsoft 社の Universal Plug and Play[12] 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[4] やカーネギーメロン大学の AURA プロジェクト [10] 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[8] 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，それらを組合せて利用する．U.C. Berkeley 校の Ninja プロジェクト [3] や ICEBERG プロジェクト [13] 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毎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，先日，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 [16]．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[9] や I-TCP[1]，TCP-R[2] 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[5] や MSOCKS[6] などがある．これらはアプリケーションやミドルウエアが用意するタイマのタイムアウトなどに対応できない．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，これらを柔軟的に支援する方法が必要となる．

3. 目　　的

本章では，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．

Mobiquitous Environment 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 Environment で利用されるサービスの分類を示す．

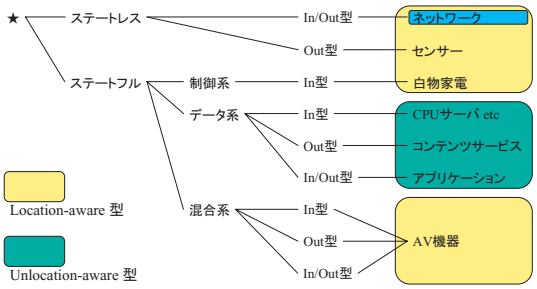

図 1　サービスの分類
Fig. 1　Service Categories

サービスは 3 つの指標によって分類される．第 1 に，サービスはイニシエータとの間にステートを共有するかどうかで分類される．第 2 にサービスとイニシエータ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．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，多くのサービスがイニシエータとの間にステートを共有することがわかる．

また，多くのサービスが，利用者のいる位置から近くに存在しなければ利用する意味がないものが多い．CPU サーバやコンテンツサーバはネットワーク的な距離を考える必要があるが，物理的に近くに存在する必要はない．しかし，情報家電やセンサといった多くのサービスは，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．そこで，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 Environment の基盤技術を実現する．

利用中断に対する処理の実行

イニシエータ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たり，中断されたくない処理を行えることを支援し，これらの処理が正常に行われるようにする．

サービスの多くが利用するイニシエータとの間にステートを有する．イニシエータ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する場合が考えられる．また，イニシエータ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

イニシエータとサービスが同期を取って処理を行えるよう，イニシエータがサービス利用不可能な状況の時にサービスは同期を行うために処理をブロックでき，利用可能な状況になるとブロックが解除されるようにする．

サービス利用の中断をしている間でもサービスだけで，もしくはクライアントだけで処理を進めることができる場合もある．しかし，通常イニシエータ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，イニシエータとの同期をとった処理を行えない．

そのため，サービスとクライアントの同期を取れることが重要となる．これによってイニシエータがサービス利用中断時でもサービスは同期を必要とする直前まで処理を進められる．

ネットワーク切断の予見検知

イニシエータが上記 2 つのことを実現できるよう，イニシエータ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．

サービス利用の中断を行う場合，2 つの理由が考えられる．利用するサービスから物理的に離れる場合と，ネットワーク切断による場合である．ネットワーク切断によるサービス利用の場合，ネットワークが切れた段階で，上記 2 つの処理を行おうとしても無理である．

イニシエータは無線 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．

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．

4. ASDS の動作概要

本章では，本論文で設計・実装を行う ASDS の概要について述べる．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．

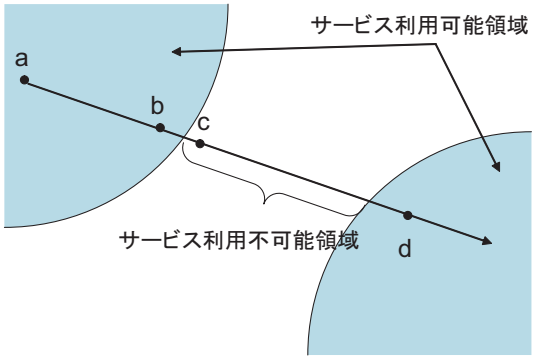

図 2 ASDS が想定するシナリオ
Fig. 2 ASDS's Senario

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イニシエータ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イニシエータが検知するには，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 にサービス利用を中断されてはこまる処理を正常に行えるよう，その処理を行っている間にサービス利用不可能を予見すると，中断のための事前処理を行う．サービスを利用するための処理の中には，その処理を行っている間にサービス利用を中断してもらいたくないものも存在する．そこで，サービスとイニシエータがステートのやりとりをしている間に，サービス利用を中断しないよう ASDS が支援することで，ステートの一貫性が保証できる．

第 3 にイニシエータ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，イニシエータとサービスが同期を取れるようにする．サービス利用を中断している間にイニシエータとの同期が必要な部分までの処理をサービス側が行うことも考えられる．ASDS を利用することで，イニシエータやサービスが同期を取りつつ処理を行える．

5. 設　　計

本章では，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．

5.1 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を利用する．

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 は信号電力をあらわし，N は雑音電力をあらわす．S/N 比の単位は (dB) である．

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

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．逆に閾値を越えたら，ネットワークに接続したと決める．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測定し，その結果を図 3 に示す．

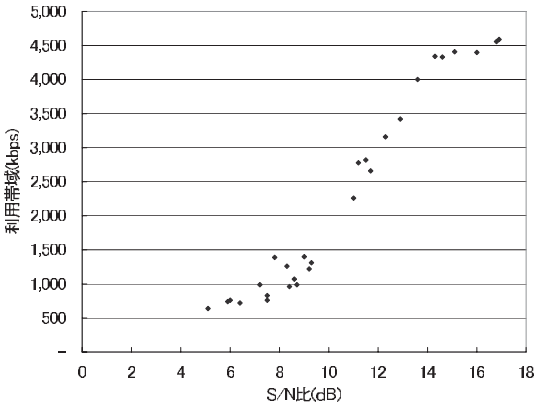

図 3 S/N 比と転送速度の関係
Fig. 3 SNR and Throughput

この結果をもとに，ASDS では切断の閾値を 8，接続の閾値を 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．

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，近づていってるのか，それとも止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，変化してなければ止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を監視し，減少傾向にり 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そうだと判断する．逆に増加傾向にある場合で 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．

5.2 中断・再開のための処理の実行支援

利用中断・再開のための処理を正常に行えるように支援する方法について述べる．ASDS ではネットワーク接続の切断が予見されたり，実際に切断される，また再接続する段階で，以下の 3 つの処理が正常に行えるように支援する．

- 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

利用中断に対する事前処理にはサービスとイニシエータとのステートや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っている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っている間にネットワークが切断されてしまった場合，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イニシエータ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ってくる．サービスとの接続を再生成したり，切断中の処理結果をやりとりすることにより，サービスとイニシエータとのステートの一貫性を取れる．

これらの処理はイニシエータとサービスの種類によって内容が異なる．ASDS が統一的な処理を行うのではなく，イニシエータ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．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を ASDS が提供することである．ASDS ではネットワークの切断・接続に連動してこれらの処理が実行できるようにする．

そこで，イニシエータは起動時に，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．イニシエータ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．前節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．

5.3 利用中断中のサービス・イニシエータの処理実行支援

ASDS では，イニシエータ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，イニシエータと同期しなければいけ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，イニシエータがサービス利用を中断した間でもイニシエータとサービスは同期を取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．

サービスがイニシエータと同期を取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，処理結果をイニシエータに送信する時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イニシエータから受けとる場合，イニシエータからのコマンドを待たなければならない．同様にイニシエータ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，イニシエータがサービスと通信を行う際に，イニシエータがサービス利用不可能であると，処理を中断できるようにする．イニシエータとサービスがイニシエータ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，イニシエータの状況を確認する．イニシエータ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，イニシエータが長い間，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．イニシエータ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，サービスはイニシエータとのステー

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．

6. 実　　装

本章では，第 5 章で述べた設計に基づき，ASDS のプロトタイプ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，本プロトタイプは 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．

6.1 システム概要

ASDS は 3 つのモジュールから構成される．図 4 に ASDS のシステム構成図を示す．

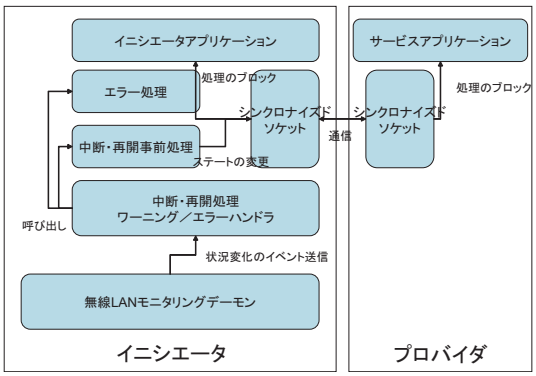

図 4　ASDS: システム構成図
Fig. 4　ASDS: System Architecture

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 比が閾値を越えると，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，各イニシエータの中断・再開処理ワーニング／エラーハンドラを呼ぶ．

中断・再開処理ワーニング／エラーハンドラはイニシエータのサービス利用状況を管理し，必要に応じてイニシエータの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呼ばれた際，イニシエータがサービスを利用していると，イニシエータが用意するワーニングもしくはエラー処理のためのコールバック関数を呼ぶ．イニシエータが中断するための事前処理が必要な場合，イニシエータの状態に拘らずワーニングのためのコールバック関数を呼ぶ．また，ネットワークに再接続された場合も同様である．

シンクロナイズドソケットは，イニシエータの利用状況に応じてイニシエータとサービスのソケットの処理をブロックする．イニシエータの利用状況はイニシエータや利用者が明示的に変更する．

6.2 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 毎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 毎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．

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

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

イニシエータは起動時に切断の何秒前に次節で述べるワーニング／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，ワーニング／エラーハンドラにイベントを送信し，イニシエータは事前処理を行ったり，利用者への注意を促す．

逆に切断と接続の検知は純粋に S/N 比が閾値を越えたことで判断する．切断を検知すると，ワーニング／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ワーニング／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 [15] を拡張して利用する．

6.3 中断・再開処理ワーニング／エラーハンドラ

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・再接続時にイニシエータのコールバック関数を呼ぶ．これにより，イニシエータ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．

イニシエータは setCallBackFunction 関数を用いて事前処理や事後処理のためのコールバック関数を設定する．これにより，イニシエータが必要とする各事前・事後処理をプログラム作成者が自由に用意できる．

図 5 にイニシエータのサービス利用状況の状態遷移の様子を示す．

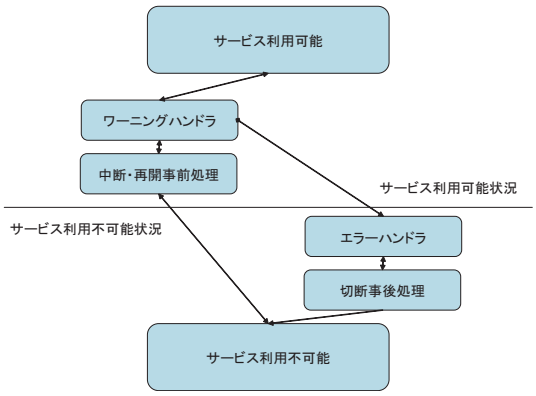

図 5 サービス利用状況の状態遷移図
Fig. 5 Initiator's service availability state transition diagram

現在，イニシエータ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ワーニング／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．

イニシエータ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．

6.4 シンクロナイズドソケット

イニシエータからのサービス利用中断・再開の処理によって，イニシエータ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シンクロナイズドソケットについて述べる．

イニシエータは 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．シンクロナイズドソケット内ではサービス利用状況を表すフラグが管理され，イニシエータからの setServiceAvailabilityState 関数によって更新される．フラグがサービス利用不可能な状況を示している時に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．

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．これにより，サービスの無駄な占有や計算資源の浪費を防げる．

7. 評　　価

本章では，ASDS のプロトタイプ実装の評価として，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，実証実験する．今回の実証実験では，ノート型 PC 上にイニシエータのサンプルプログラムを実行させ，予見するための時間を 5 秒に設定した．利用者はイニシエータが動作するノート PC を持ちながら実際に移動し，その変化に適応する．図 6 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．

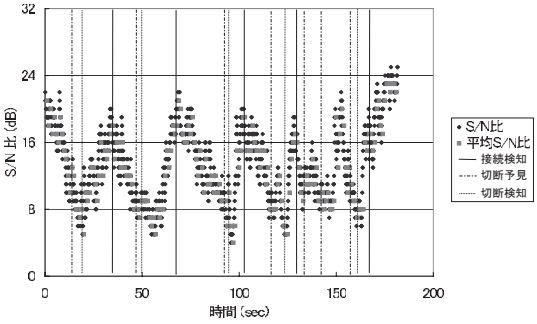

図 6 S/N 比の変化と各種処理
Fig. 6 SNR and each performances

まず，S/N 比が激しく変動しているのがわかる．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い，イニシエータがその都度サービス利用状況の変化に適応し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．

8. 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共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，イニシエータ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・プロトタイプの実装を行い，評価を行った．

サービスの利用状況の変化に適応する際に必要な中断前の事前処理や中断処理失敗時のエラー処理，また再開時に必要な処理を行えるよう，ASDS が状況を監視し，必要に応じてイニシエータの各処理を実行させる．ASDS では無線 LAN の S/N 比をその特徴を考慮しつつ利用することで，適切に状況変化を予見，検知できる．

今後の課題として，ASDS を利用した応用アプリケーションを実装し，実証実験を行いつつ評価を行うことが挙げられる．その際，様々なイニシエータの種類を分類し，ASDS の提供する機能が十分であるか検討する．

参 考 文 献

1) 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).
2) 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).
3) 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).
4) 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).
5) 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 Princeples (SOSP)*, pp. 156–171 (1995).
6) 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).
7) Mann, S.: Wearable Computing: A First Step Toward Personal Imaging, *IEEE Computer*, Vol. 30, No. 2, pp. 25–32 (1997).
8) 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).
9) Perkins, C.: IP mobility support, *Internet Request For Comments RFC 2002* (1996).
10) 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).
11) Sun Microsystems, Inc.: Jini Architecture Specification (1998). http://www.javasoft.com/products/jini/specs/jini-spec.pdf.
12) Universal Plug and Play Forum: Universal Plug and Play (UPnP) (1999). http://www.upnp.org/.
13) 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).
14) Weiser, M.: Computer of the 21st Century, *Scientific American*, No. 3, pp. 94–104 (1991).
15) 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).
16) 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/.