ウェアラブルコンピューティングのための状況依存アクセス制御機構について

宮前　雅一† 　寺田　努†† 　塚本　昌彦† 　西尾章治郎†

† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 2–1
†† 大阪大学サイバーメディアセンター　〒 565–0871 大阪府吹田市山田丘 2–1
E-mail: †{miyamae,tuka,nishio}@ist.osaka-u.ac.jp, ††tsutomu@cmc.osaka-u.ac.jp

あらまし　ウェアラブルコンピューティング環境では，ユーザの行動・位置・周囲の環境などを考慮してサービスの動作制限を動的に変更するなど，ウェアラブルコンピューティング環境向けの柔軟なセキュリティ機構が要求される．本研究で提案するウェアラブルコンピューティングのためのアクセス制御機構は，ユーザの状況や周りの環境に応じてサービスのアクセス権限を動的に変更することで，ユーザの置かれた状況に応じたサービスの動作制限を行う．また，サービス受信時，サービス実行時，サービス同士の連携時にもアクセス制限を行うことで，さまざまな脅威からウェアラブルシステムを保護する．提案手法を用いることで，ウェアラブル環境における高度なセキュリティを実現できる．
キーワード　アクセス制御，ウェアラブルコンピューティング，セキュリティ，状況依存サービス

A Situation-Based Access Control Mechanism for Wearable Computing

Masakazu MIYAMAE†, Tsutomu TERADA††, Masahiko TSUKAMOTO†, and Shojiro NISHIO†

† Graduate School of Information Science and Technology, Osaka University　2–1 Yamadaoka, Suita, Osaka 565–0871, Japan
†† Cybermedia Center, Osaka University　2–1 Yamadaoka, Suita, Osaka 565–0871, Japan
E-mail: †{miyamae,tuka,nishio}@ist.osaka-u.ac.jp, ††tsutomu@cmc.osaka-u.ac.jp

Abstract　In wearable computing environments, a security mechanism is required to control the access level of data according to environments, users' location, and their behaviors. In this paper, we propose a new security mechanism for wearable computing that realizes the dynamic changes of access control levels. Moreover, this mechanism protects the system from various types of attacks by the access control on timing of the service receiving, the service processing, and the service coordination. Using our mechanism, we can use wearable systems with high level security in wearable computing environments.
Key words　access control, wearable computing, security, context-aware services

1. はじめに

近年，マイクロエレクトロニクス技術の発展による計算機の小型化・軽量化に伴って，ウェアラブルコンピューティングに対する注目が高まっている．ウェアラブルコンピューティングとは，計算機をユーザが常に身に着けて持ち運ぶコンピューティングの一形態であり，図 1 に示すウェアラブルコンピュータのユーザは鞄のように計算機を常に持ち運び，装着型ディスプレイ (HMD: Head Mounted Display) を用いて情報を閲覧している．ウェアラブルコンピューティングは，従来の計算機の利用形態と比較して次の 3 つの特徴をもつ [9].

（1） ハンズフリー：コンピュータを身体に装着しているため，両手を使用せずに情報を参照できる．

（2） 生活密着：常にコンピュータを装着した状態で，日常生活を行う．

（3） 常時オン：コンピュータは常に電源が入っており，使いたいときにすぐに使える．

ウェアラブルコンピューティングで情報を活用するための研究は，これまで多数行われており，例えば各種センサを用いてユーザの位置や状況を取得し，建物の案内やアノテーションを表示するシステム [3, 5] がある．また，ウェアラブル環境のためのサービスプラットフォーム [4, 6] も提案されている．筆者らの研究グループでも，動的にサービスを追加・削除できるウェアラブルコンピューティングのためのイベント駆動型サービスプラットフォーム A-WEAR [5] を提案している．これらのサービスプラットフォームでは，システムの動作中にサービスを追

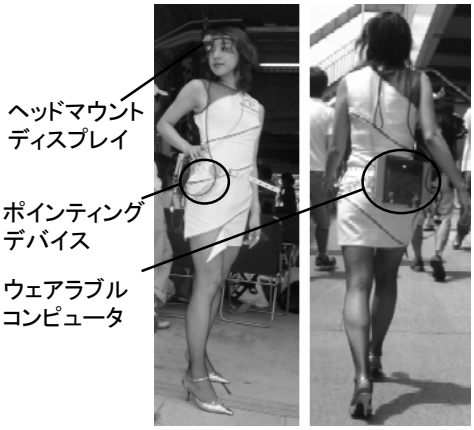

図 1　ウェアラブルコンピューティング

加できる．サービスとは，ウェアラブルシステムが行う複数の処理をまとめてグループ化し，意味をもたせたものである．

このようなシステムを用いることで，遊園地ではアトラクションの案内サービスを提供し，博物館では展示品の案内サービスを提供するなど，多数のサービスの中からユーザの状況に適したものを自動的に取得してシステムに追加し，ウェアラブルコンピュータが効果的にユーザを支援することが可能となりつつある．また，これらの研究で提案されている個人化手法を用い，ユーザの個人情報を基に情報を提供する高度なサービスも提供できる．

一方，ウェアラブルコンピューティングでは各種のセンサを用い，人間の生体情報や個人情報をキーとしたサービスが提供されるため，セキュリティ技術が必要不可欠であるが，既存のウェアラブルシステムにおいてセキュリティが考慮されているものは少ない．そのため，悪意のあるサービスを受信することでシステムやデータの改竄が行われたり，ユーザの嗜好に応じて情報を提供する位置依存サービスがその場を離れても動作し続け，定期的にユーザ情報を漏洩させるといった問題が発生する可能性がある．また，広告用のサービスを受信することで，過剰な広告によってユーザの作業が阻害されたり，音を出したくない状況で音声が再生されるといったことも起こり得る．

現在，悪意のあるプログラムからシステムを保護する手法としては，情報源を信頼度でクラスタ化し，システムに追加するかどうかを判断する手法が WWW ブラウザで Java アプレットをダウンロードする時などに使用されている．しかしウェアラブル環境において，システムのセキュリティを確保するためにはサービスの送信元だけでなく，サービスの処理内容も考慮することが重要である．さらに，サービスをシステムに追加した後も，サービスが実行できる操作内容をあらかじめ制限することでシステムの改竄を防止するだけでなく，ユーザの行動・位置・環境など，さまざまな状況を考慮してサービスの制限を動的に変更する必要がある．また，複数のサービスを同時に利用できるウェアラブルシステムでは，サービス同士が互いに情報を交換できるものがある．このようなシステムでは，ネットワークなどを通して受信したサービスが以前からシステムに存在したサービスに影響を及ぼし，危険な処理を実行させるといったことも考えられる．

そこで，本研究では，ウェアラブルコンピューティングのための状況依存アクセス制御手法 (SBAC: Situation Based Access Control) を提案する．提案手法では，状況に応じてサービスの権限を動的に変更し，サービスの動作を制限する．また，サービス同士が連携して動作する環境において，サービス同士の情報交換を監視することによってシステムのセキュリティを確保する．本手法を用いることにより，動的なサービスの追加が可能なウェアラブルシステムにおいて，高度なセキュリティが確保できる．

以下，2 章で提案手法である SBAC の概要について述べ，3 章で提案手法の詳細について述べる．また，4 章で本手法の実装について説明し，5 章で提案手法の考察を行う．最後に，6 章でまとめを行う．

2. SBAC

本章では，本研究で想定するウェアラブル環境について説明し，提案手法に必要な機能および提案手法の概要について述べる．

2.1 想定環境

ユーザが身に着けているウェアラブルコンピュータにはさまざまなセンサが装着されており，スケジュール情報や日々の行動履歴などを記録している．ウェアラブルコンピュータはそれらの情報を用いて，ユーザが勤務中であるか，通勤中であるか，就寝時間か，屋内にいるか，初めて訪れた場所にいるか，電話中であるか，友人と一緒にいるかなど，ユーザが置かれているさまざまな状況を認識できる．また，ウェアラブルコンピュータは無線 LAN や携帯電話などを用いてネットワークに接続でき，さまざまなサービスを自動的に，またはユーザが指定することでダウンロードできる．サービスの提供元は，ユーザの友人や商店街の店舗などさまざまであり，その信頼性も送信元によって異なる．サービスを提供する外部システムの中には悪意のあるものが存在し，ダウンロードするサービスの中に悪意のあるサービスが含まれている可能性があるため，ユーザはそれぞれのサービスのセキュリティ設定のカスタマイズを行う．さらに，システム内のサービスは連携して動作でき，新たにダウンロードしたサービスも既存のサービスと連携して動作する．ここで，サービスの連携とは，あるサービスが別のサービスの動作に直接的に影響を及ぼすことであると定義する．例えば，ユーザの体温の変化のログを取り続ける体温記録サービスと，体温のログを監視して普段と著しく異なる変化がある場合に警告を表示する体温監視サービスがある場合，体温記録サービスが体温監視サービスの動作に影響を及ぼし，サービスが連携しているとする．一方，あるサービスが CPU やメモリ，ネットワーク帯域などのリソースを占有した結果，他のサービスが正常に動作できなくなるなど，間接的に影響する場合はサービスの連携としない．一般に，サービス間の連携には関数呼び出し・メソッド呼び出し・イベント・トリガなどが用いられる．

2.2 ウェアラブル環境におけるセキュリティ

想定環境では，セキュリティの脅威として以下のものが考え

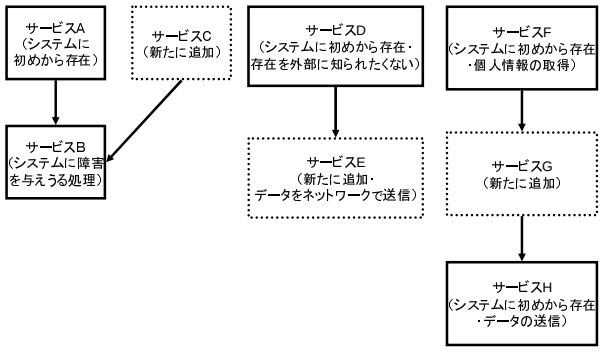

図 2　サービス間の連携における脅威の例

られる．

- **ユーザの状況を考慮しないサービスの実行**

サービスがユーザの望まない状況で実行され，ユーザの行動を妨げたり，計算機の利用性を低下させる．

- **見知らぬ送信者からの悪意のあるサービスの受信**

見知らぬ企業や個人から悪意のあるサービスや過剰広告用サービスを受信することで，ユーザが望まない処理を実行される．

- **信頼している送信者からの悪意のあるサービスの受信**

信頼している企業や人から誤って悪意のあるサービスが送信される．

- **通信経路における盗聴や改竄**

サービスや個人情報などの通信経路において，盗聴や改竄が行われることにより，ユーザの個人情報が漏洩したり，悪意のある操作を実行される．

- **サービス間の連携を利用した攻撃**

システムのセキュリティ機構によって処理が制限されている悪意のあるサービスが，サービス間の連携を用いて攻撃を行う．

サービス間の連携を利用した攻撃の例を図 2 に示す．図 2（a）は，システム内に初めから存在するサービス A と，システムに対して障害を与えられる高い権限をもったサービス B が連携して動作しているときに，新たに追加したサービス C がサービス B と連携して動作することでシステムに障害を与える様子を示している．図 2（b）は，他者に対して存在を知られたくないサービス D が動作しているときに，他のサービスと連携したときに情報を外部へ送信するサービス E を追加することで，他者にサービス D の存在を知られる様子を示している．図 2（c）は，個人情報を取得するサービス F と情報を外部へ送信するサービス H がシステム内に存在するときに，サービス F，H と連携するサービス G を追加されることで，個人情報が外部に送信される様子を示している．

ウェアラブル環境におけるセキュリティ機構は，このような脅威からシステムを保護する機能と，状況に応じてサービスの動作を動的に制限する機能を備えることが要求される．

2.3　提案手法の概要

本研究では，以上の要求事項を満たすアクセス制御機構 SBAC (Situation Based Access Control) を提案する．提案手法のイメージ図を図 3 に示す．図中の網掛け部分が本研究で提案する

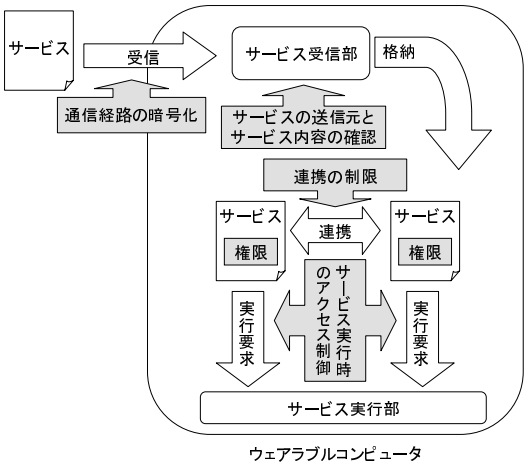

図 3　提案手法のイメージ図

部分である．以下に提案手法の特徴を示す．

- **サービスベースのアクセス制御**

ユーザは，さまざまなサービスを同時に利用するため，サービスごとにアクセス権限を柔軟に割り当てられる必要がある．そこで，提案手法では，RBAC (Role-Based Access Control) を拡張したセキュリティモデルを用いてアクセス制御を行う．このセキュリティモデルを用いることで，信頼している送信者から誤って悪意のあるサービスが送信されてきた場合であっても，システムを保護できる．

- **状況に応じたアクセス制御**

ユーザの置かれた状況と対応するアクセス制限を組にして記述したリストを用い，状況に応じてサービスのアクセス制限を動的に変更する．

- **通信経路の暗号化**

通信の安全を確保するためには，通信経路を暗号化することが有効である．通信の暗号化は，公開鍵暗号手法などが実用化されており，本研究でもそれら既存の手法を用いる．

- **サービスの送信元とサービス内容の確認**

受信したサービスをシステムに追加する際に，悪意のあるサービスや望まないサービスをシステムに追加しないようにする必要がある．そこで，サービスの送信元と実行内容を確認し，サービスをシステムに追加するかの判断や，アクセス権の決定を行う．

- **サービス連携の制限**

連携を行う際にはアクセス制御機構が呼び出し元のアクセス権限と呼び出し先のアクセス権限を確認し，連携処理時のアクセス権限を動的に決定する．動的に決定されるアクセス権限では，呼び出し元と呼び出し先のサービスの両方で許可されたもののみ実行を許可する．

3.　SBAC の詳細

本章では，本研究で提案する SBAC の詳細について述べる．

3.1　RBAC

多くのシステムで使用されているアクセスコントロール手法として，ACL (Access Control List) と RBAC (Role-Based

Access Control) が挙げられる．ACL は，オブジェクトに対して実行可能な操作を複数記述したリストを関連付けて保持することにより，オブジェクトの操作内容を制限する手法であり，簡便で実装が容易であるという特徴がある [1]．しかし，ウェアラブル環境で ACL を用いた場合，勤務時間中にプライベートなサービスの音声出力を禁止するという処理を実現するためにはすべてのプライベートなサービスの音声出力権限を削除する必要があり，状況が次々に変化するウェアラブル環境には適していない．

RBAC は，ユーザに対して役割を割り当て，さらに役割に対してアクセス権限を割り当てるアクセス制御方式であり，ユーザの権限を変更する場合は対応する役割のアクセス権限を変更するだけですむため，セキュリティ管理が容易であるという特徴がある [2, 11]．ウェアラブル環境では，一人のユーザが自分専用の計算機を使用するため，複数のユーザのアクセス制御を行う必要がない一方，サービスごとのセキュリティ権限変更が要求されるため，各サービスを RBAC におけるユーザと考えることで容易にセキュリティ管理が実現できる．

そこで，提案手法では，RBAC を拡張したセキュリティモデルを使用する．既存の RBAC では，役割を用いたアクセス権限の管理しか規定されておらず，状況に応じたアクセス制限やサービス間の連携の制限といったウェアラブル環境において要求される機能を備えていないため，SBAC では RBAC を拡張し，ウェアラブル環境において要求されるセキュリティ機能を提供できるようにする．

以下，SBAC の構成要素と動作の詳細について述べる．

3.2 SBAC の構成要素

SBAC は，アクセス権限・サービス・役割・状況依存アクセス制御リスト・連携オブジェクトの 5 つの要素で構成される．SBAC では，サービスに割り当てられた役割と，役割に割り当てられたアクセス権限を基にサービスのアクセス権限を決定し，そのアクセス権限を用いて図 3 に示すサービス実行時のアクセス制御を実現する．また，状況依存アクセス制御リストを用いて状況に応じたアクセス権限の変更を実現し，さらに連携オブジェクトを用いて図 3 に示す連携の制限を実現する．以下，それぞれの構成要素について述べる．

- **アクセス権限**

実行可能な処理を示す．アクセス権限には，ユーザの個人情報の書き込みや読み取り，データの送信，画面の表示，音声の再生，サービスのアクセス権限の変更，サービスの存在権限などが記述できる．ウェアラブル環境では，計算機にさまざまなデバイスを付けることによってサービスの実行可能な処理が増えることが想定されるため，アクセス権限の種類は動的に追加できる．アクセス権限の変更権をもったサービスは，別のサービスのアクセス権限を変更し，任意の権限を追加・削除できる．

- **サービス**

処理の実行単位を示す．SBAC では，ひとつのサービスに対して複数の役割を割り当てることができ，サービスは役割によって許可されている処理を実行できる．

- **役割**

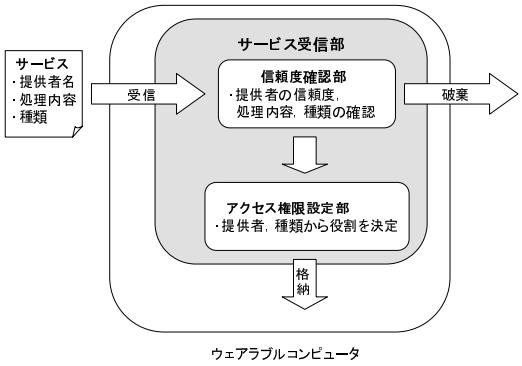

図 4　サービス受信時の処理

役割は複数のアクセス権限の組み合わせで構成される．RBAC では，役割を権利と考えているため,「ネットワーク管理者」と「アカウント管理者」の両方の役割をもつユーザはネットワークの管理とアカウントの管理の両方を行える．これは，ユーザが全く異なる複数の役割を担っていることがあるためである．

SBAC では，サービスに対して役割を与えるが，単一のサービスが全く異なる複数の役割をもつことは少ないと考えられる．そこで，SBAC では役割を制限と考え，サービスが属している全ての役割で許可されている動作しか実行できないようにする．例えば,「位置依存サービス」と「プライベート」の両方の役割をもつサービスは，指定された位置でプライベートな時間にしか動作できない．これにより，プライベートの役割のアクセス権限を変更し，音声出力を禁止することで，プライベートな役割をもつ全てのサービスの音声出力を禁止することが可能になり，状況に応じたアクセス制限の変更を容易に実現できる．また，システムには全てのサービスが属する役割も存在する．この役割のアクセス権限を変更することで，音声出力は全く許可しないといった，システム全体のポリシー変更も実現できる．

- **状況依存アクセス制御リスト**

ユーザの置かれた状況が変化した場合に，変更するアクセス制限を記述したリストを示し，システムが保持する．状況依存アクセス制御リストには，状況名・アクセス権限を変更する役割・アクセス権限の変更内容の組を記述する．このリストはユーザが自由に更新できる．

- **連携オブジェクト**

サービスの連携を担うオブジェクトを示す．例えば，関数呼び出しやメソッド呼び出しを用いてサービス間の連携を行っている場合，呼び出しを実行する処理プロセスが連携オブジェクトになり，イベントやトリガを用いて連携を行っている場合はイベントやトリガ自身が連携オブジェクトとなる．連携オブジェクトはアクセス権限を保持しており，システムは連携オブジェクトのアクセス権限を元にサービスのアクセス制御を行う．

3.3 SBAC の動作

SBAC では，2 章で述べたように，ユーザ状況が変化した時のアクセス権限の変更とサービス実行時のアクセス制限だけでなくサービス連携時やサービス受信時のアクセス制御も行う．以下，それぞれの動作について述べる．

- **ユーザ状況変化時のアクセス権限の変更**

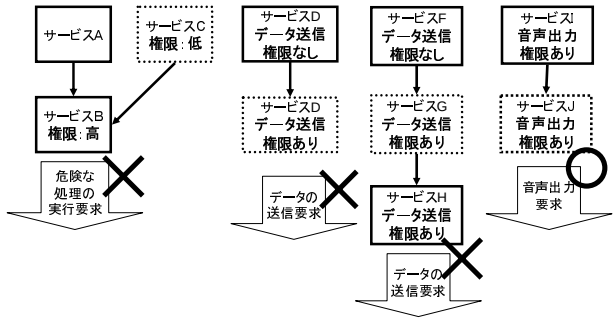

図 5　サービス連携の制限

システムは，ユーザの状況を常に監視し，ユーザ状況の変化を検出すると，状況依存アクセス制御リストを参照して対応する役割のアクセス権限を変更する．

- **サービスの受信**

サービス送信者は，サービスの提供者名，サービスの処理内容，サービスの種類をサービスに添付して送信する．サービス受信時の処理を図 4 に示す．サービス提供者名は，既存の署名アルゴリズム等により詐称されていないことを確認できているものとする．システムはサービスを受信すると，サービス提供者の信頼度，制限されたアクセス権限を越える処理を行うか，システムに追加することを許可されたサービスの種類かといった情報を確認し，サービスをシステムに追加するかどうかを決定する．サービスをシステムに追加する場合，サービス提供者の信頼度から既定の役割を決定する．さらに，サービスの種類に対応する役割が存在する場合，その役割もサービスに割り当てる．以上の処理により，サービスをシステムに追加するかどうかの判定と，サービスに割り当てる役割を決定する．

サービス送信者は，高いアクセス権が必要なサービスを，処理内容を低く偽ることでサービスをシステムに追加することができるが，サービスに割り当てられる既定の役割で高いアクセス権限が必要な処理を禁止しておけば，このような場合でもシステムのセキュリティを確保できる．

- **サービス実行時のアクセス制限**

システムは，サービスの実行を開始する際にそのサービスが属する役割を元にアクセス権限を調べ，サービスを実行する．実行時に権限を確認することで，役割に与えられたアクセス権限の動的な変更に対応できる．

- **サービス間の連携**

サービス間で連携を行う場合，システムは連携オブジェクトのアクセス権限に連携元と連携先の両方のサービスで許可されている内容を設定する．連携先のサービスの実行は，連携オブジェクトのアクセス権限を元に制限して実行する．

図 2 に示すサービス連携を制限する例を図 5 に示す．図 5（a）では，サービス C がシステムに障害を与えられる高い権限をもっていなければ，サービス B，C の連携時に連携オブジェクトは高い権限をもてないため，システムに障害を与えられない．また，図 5（b）では，サービス D がネットワークを通したデータの送信権限をもっていなければ，サービス D，E の連携時に連携オブジェクトはデータの送信権限をもてない．さらに，図 5（c）でも，サービス G はデータの送信を行わないので連携できるが，ユーザの個人情報を取得するサービスがデータの送信権限をもっていなければサービス H の連携時に連携オブジェクトはデータの送信権限をもてないため，個人情報の漏洩を防止できる．図 5（a）は，音声出力権限をもつサービス I とサービス J が連携して動作できることを示している．

このように，連携オブジェクトを用いることで，サービス連携時におけるセキュリティを確保できる．ただし，さまざまなサービスの動作をシステムログに記録するような場合，ログの記録権限をもたないサービスの動作も記録したいというように，アクセス権限にかかわらず連携したいという要求もありうる．そこで，サービスの強制連携権限を定義する．この権限をもつサービスと連携を行う場合，システムは連携オブジェクトのアクセス権限は利用せず，連携先のサービスのアクセス権限を使用する．サービスの強制連携権限を利用する場合，セキュリティホールを発生させる可能性があるため，ネットワークから受信したサービスに対しては強制連携権限を与えないなどの処置が必要となる．

```
DEFINE Rule-ID
    [IN List-of-belonging-groups]
    [FOR Scope]
    [VAR Variable-name AS Variable-type]*
WHEN Event-type [ (Target-of-event)]
IF       Conditions
THEN DO Actions
```

図 6　A-WEAR の ECA ルール記述構文

```
DEFINE FIND-BUILDING
WHEN GPS_MOVE
THEN
  DO MAP_SET_CENTER (NEW.LATITUDE, NEW.LONGITUDE)
  DO QUERY ('SELECT * FROM GEODATA WHERE  (
    ABS (%NEW.LATITUDE%-X) < 0.005 AND
    ABS (%NEW.LONGITUDE%-Y) < 0.005) ')

DEFINE DISPLAY-WEBPAGE
WHEN SELECT (GEODATA)
IF ?MAP.EXIST (%GPS.X%;%GPS.Y%;%NEW.X%;%NEW.Y%;
  %MOTION_SENSOR.ALPHA_NORTH%;100.0;20.0)
THEN DO SHOW_BROWSER (NEW.URL)
```

図 7　サービス記述例

4. SBAC の実装

筆者らが提案しているウェアラブルコンピューティングのための基盤システム A-WEAR 上に SBAC を実装した．以下，まず A-WEAR の特徴について述べ，次に A-WEAR 上における SBAC の実装について述べる．その後，A-WEAR におけるセキュリティ設定の記述方法について説明し，最後に A-WEAR 以外のシステムにおける SBAC の実装方法について述べる．

4.1 A-WEAR

A-WEAR では，システムの動作をイベント駆動型ルールで記述し，プラグインを用いてシステムの機能を拡張できる．サー

ビス開発者は既存のプラグインを組み合わせ，プラグインがもつ機能を利用するルールを記述することで，容易にサービスが構築できる．これらのサービスはルールを追加，削除することで，システム実行中に自由に追加・削除できる．

A-WEAR はシステムの動作を発生する事象 (イベント)，実行させるための条件 (コンディション)，イベントによって発火する操作 (アクション) の 3 つの組からなる ECA ルールで記述する．A-WEAR で使用する ECA ルールの構文を図 6 に示す．図の *Rule-ID* は ECA ルールを一意に識別する ID を示す．また，イベントを WHEN，コンディションを IF，アクションを THEN で記述する．

ECA ルールを用いた建物案内サービス [6] のルールの例を図 7 に示す．このサービスは，ユーザが移動すると付近の建物を検索する FIND-BUILDING ルールと，検索された建物の Web ページを表示する DISPLAY-WEBPAGE ルールからなる．

このように A-WEAR では，複数のルールをグループ化してサービスを記述し，ユーザはルールを修正・追加・削除することで，容易にサービスをカスタマイズできる．また A-WEAR では，システムにプラグインと呼ぶ拡張モジュールを追加することで，ECA ルールに記述可能なイベント・アクションを動的に追加できる．プラグイン形式の採用により，新たなデバイスへの対応や機能拡張を行う際には対応するプラグインを作成するだけでよく，システム自体の修正を必要としない．

筆者らの研究グループでは，これまでに A-WEAR を用いて建物案内サービスの他に農作業支援システム [7]，ウェアラブル環境向けメールシステム [4]，バイクレースサポートシステム，ウェアラブル向け音楽再生システム [8] など，複数のサービスを組み合わせたアプリケーションを多数構築してきた．農作業支援システムは，ユーザが農作業中で手を計算機の操作に利用できない状況でも最新の市況情報や害虫情報をユーザに提示することを目的としたシステムである．ウェアラブル向けメールシステムは，受信したメールの処理をルールで行うことで，受信したメールの自動処理などが行える．また，ECA ルールをメールに添付して送信して受信側で実行できるため，集合場所の案内メールに地図を自動表示する機能を付けるなど，高度な機能をもったメールが実現できる．バイクレースサポートシステムは，レースの状況をリアルタイムでピット内のクルーが装着した HMD に表示してレースをサポートするシステムであり，2003 年に鈴鹿サーキットで行われた 8 時間耐久ロードレースで実際に使用した．

A-WEAR を利用することで，既存のプラグインを組み合わせ，ECA ルールを記述するだけで容易にウェアラブル環境向けアプリケーションを構築できる．必要な機能が既存のプラグインで提供されていない場合，新たなプラグインを開発して不足している機能を補えばよく，A-WEAR を用いずにシステムを開発を行う場合と比較して作業負担を大幅に軽減できる．

しかし，ウェアラブル向けメールシステムは受信した ECA ルールを自動的に実行するため，セキュリティ機構が不可欠である．また，サーキットを訪れたユーザに対してバイクレースサポートシステムを配信したり，建物案内サービスを配信する場合にも，セキュリティ機構が必要である．そこで，A-WEAR に SBAC を実装し，その有効性を確認する．

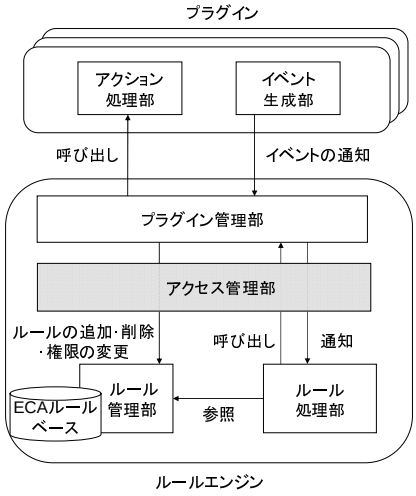

図 8　SBAC に対応した A-WEAR のシステム構成

4.2　A-WEAR における SBAC の実装

本研究では，SBAC の機能を A-WEAR 上に以下のように実現した．

- **サービス受信時の制御**

サービスの受信はネットワーク通信プラグインで行い，サービス受信時の制御はルールで行う．ネットワーク通信プラグインはサービスを受信すると，受信したサービスを一時的に格納したファイル名・サービスの送信者，サービスの処理内容，サービスの種類の情報をもったイベントを発生させる．システムは，あらかじめ記述しておいた ECA ルールでこのイベントを処理し，サービスをシステム内に格納するかどうかや，サービスの既定の役割を決定する．

- **サービス実行時のアクセス制限**

プラグインは，アクションの実行に必要なアクセス権限をあらかじめシステムに登録する．A-WEAR のアクセス制御部は，アクションを実行する際に許可されている実行内容とアクションの実行に必要なアクセス権限を比較し，権限が不足している場合にはアクションの実行を行わないようにする．

- **アクセス権限の動的な変更**

状況に応じたアクセス制限の変更はルールで行う．システムは，ユーザの状況の変化に応じて発生するイベントに対して，状況依存アクセス制御リストを参照してアクセス権限を変更するアクションを記述したルールをあらかじめ保持する．状況依存アクセス制御リストは A-WEAR がもつローカルデータベースに保存する．また，アクセス権限の変更を実行するための権限を用意し，受信したサービスに対してはこの権限を割り当てないことで，受信したサービスが他のサービスの権限を変更できないようにする．

- **サービスの連携の制限**

A-WEAR では，ECA ルールの連鎖によって複数のサービスを連鎖的に実行できるため，ルールの連鎖が SBAC における連携オブジェクトに当たる．そこで，システムはアクション実行

```
// DB の信頼度テーブルに載っているホストからの受信
DEFINE RECEIVE-SERVICE1
WHEN NET_RULE_RECEIVE
IF ?DB.'ConfidenceTable.DefaultRole NAME == %NEW.FROM%'
THEN DO CheckAccessRight(NEW.FILE, NEW.TYPE, DB.'ConfidenceTable
 .DefaultRole NAME == %NEW.FROM%', NEW.RequiredRight)

// DB の信頼度テーブルに載っていないホストからの受信
DEFINE RECEIVE-SERVICE2
WHEN NET_RULE_RECEIVE
IF !DB.'ConfidenceTable.DefaultRole NAME == %NEW.FROM%'
THEN DO CheckAccessRight(NEW.FILE, NEW.TYPE, 'UnknownService',
 NEW.RequiredRight)

// アクセス内容チェックに合格
DEFINE ACCEPT-SERVICE
WHEN CHECK_ACCESS_RIGHT
IF ?NEW.ACCEPT
THEN DO ADD_RULE(NEW.FILE)
     DO SET_ROLE(NEW.FILE, NEW.ROLE, NEW.TYPE)

// アクセス内容チェックに失格
DEFINE REJECT-SERVICE
WHEN CHECK_ACCESS_RIGHT
IF !NEW.ACCEPT
THEN DO DISPLAY_MESSAGE(' 受信したサービスは危険な処理を実行する可能性が
あるので破棄しました')
```

図 9　サービス受信時のアクセス制限を行うルール

```
// ユーザ状況変化時に状況依存アクセス制御リストを検索
DEFINE SearchSBACTable
WHEN SITUATION_CHANGE
THEN DO DB_QUERY('SELECT * FROM SBAC_Table
  WHERE SITUATION = %NEW.NAME%')

// 状況依存アクセス制御リストを元にアクセス制限を変更
DEFINE DoSBAC
WHEN DB_SELECT(SBAC_Table)
THEN DO SET_ACCESS_RIGHT(NEW.ROLE, NEW.ACESS_RIGHTS, NEW.ENABLE)

// ユーザが屋外にいることを検出
DEFINE DetectSituation
WHEN GPS_MOVE
THEN DO SITUATION_CHANGE('OutOfDoors')
```

図 10　状況に応じたアクセス制限を行うルール

表 1　状況依存アクセス制御リスト

SITUATION	ROLE	ACCESS_RIGHTS	ENABLE
BeginWork	Working	Enabled	TRUE
BeginWork	Private	Enabled	FALSE
Meeting	All	SoundOut	FALSE

時にアクセス権限をプラグインに伝え，プラグインはアクションに応じてイベントを発生させるときに伝えられたアクセス権限をそのまま渡すようにする．これにより連携オブジェクトによるアクセス権限の伝達が実現できる．

SBAC を適用した A-WEAR のシステム構成を図 8 に示す．図において，ルールエンジンとはルールの管理・実行およびプラグインの管理を行う A-WEAR の中核である．本研究では，アクション実行時のアクセス制御やルールの連鎖の制御を行うために，ルールの管理・実行を行う部分とプラグインの管理を行う部分の間にアクセス管理部を実装した．図からわかるように，サービスとアクセス制限が必要な処理の実行部が切り分けられている場合，SBAC を容易に適用できる．

4.3　セキュリティ設定の記述例

SBAC を提供した A-WEAR では，サービス受信時の制御や状況に応じたアクセス制御に ECA ルールを用いる．本節では，

```
// ウェアラブル環境向け音楽再生システムの一部
// 再生した曲をユーザの嗜好情報として DB に保存
DEFINE SavePlayedSong
WHEN MPLAY_PLAY
THEN DO DB_QUERY('INSERT INTO PLAYED_SONG (NAME, TIME)
  VALUES (%NEW.NAME%, %NOW.TIME%)')

// 受信したルール．DB の更新時にデータを送信
DEFINE SendDBData
WHEN DB_INSERT
THEN DO NET_SEND('192.168.0.1', 'PlayedSong', 'Name', NEW.NAME)
```

図 11　連鎖制限が必要なルール

以上のアクセス制御を実現する ECA ルールについて述べる．

図 9 にサービス受信時の制御を実現するルールを示す．**RECEIVE-SERVICE1** は，受信したサービスの送信元がユーザの信頼度テーブルに載っている場合に，対応する既定の役割のアクセス権限と受信したサービスのアクセス内容を比較するアクションを実行する．**RECEIVE-SERVICE2** は，受信したサービスの送信元がユーザの信頼度テーブルに載っていない場合に，詳細が不明なサービスに割り当てられる役割のアクセス権限と受信したサービスのアクセス内容を比較するアクションを実行する．**ACCEPT-SERVICE** は，アクセス権限のチェックの結果，サービスの実行が許可された場合に，受信したサービスをシステムに追加して既定のアクセス権限を割り当てる．**REJECT-SERVICE** は，アクセス権限のチェックの結果，サービスの実行が許可されない場合に，サービスの実行が拒否されたというメッセージを表示する．

また，図 10 に状況に応じたアクセス制御を実現するルールを示す．**SearchSBACTable** と **DoSBAC** はシステムが初めからもっているルールであり，ユーザ状況の変化時に状況依存アクセス制御リストを参照して役割のアクセス権限を変更するルールである．ユーザが勤務中であるという状況に対応した状況依存アクセス制御リストの例を表 1 に示す．この表は，ユーザの勤務時間中には，仕事用の役割をもったサービスに対して実行可能権限を割り当て，プライベート向けサービスの実行可能権限を削除し，ミーティング中には全てのサービスの音声出力権限を削除することを示している．**DetectSituation** は，GPS を用いてユーザが屋外にいることを検出した場合，状況の変化をシステムに通知するルールである．このように，アクセス制御をルールを用いて行うことで，ルールの修正による柔軟なアクセス制御のカスタマイズが可能になる．

さらに，サービスの連携を制限しなければユーザの情報が漏洩するルール例を図 11 に示す．この例は，ウェアラブル環境向け音楽再生システムが再生した曲名をデータベースに保存し，ユーザの嗜好情報として利用している場合に，データベースにデータを追加したときに発生する DB_INSERT イベントに応じてデータを送信するルールを受信した場合を示している．このような場合でも，ユーザの嗜好情報蓄積サービスに対してデータの送信を禁止しておくことで，データの漏洩を防止できる．

4.4　その他のシステムにおける SBAC の実装方法

MEX [3] や NETMAN [2] では，サービスプラットフォーム

上で動作するモジュールとしてサービスを記述し，サービスが他のサービスやプラットフォームを呼び出すことで複雑な処理を実現している．これらのシステムでも，A-WEAR と同様に，サービス受信部やプラットフォームを修正し，サービス受信時の制御やサービス間の連携をプラットフォームで管理できるようにすることで SBAC を適用できる．

ただし，サービスが OS の機能を直接利用し，安全でない処理を実行できるシステムでは，サービス実行時のアクセス制御が行えないため，OS の機能の利用をプラットフォームが動的に制限できなければ SBAC を適用できない．このようなシステムでは，サービスの OS 呼び出しをプラットフォームが中継するように拡張することで SBAC を適用できる．

5. 考　　察

5.1 セキュリティの完全性

アクセス権限を動的に変更すると，アクセス権限が変化する直前にサービス間の連携を開始した場合に，連携元のサービスのアクセス権限で禁止された処理が連携先のサービスで実行される恐れがある．このように，SBAC はセキュリティの完全性が不足している部分がある．そこで，本節では SBAC におけるセキュリティの完全性について述べる．

ウェアラブル環境におけるセキュリティの脅威は，ユーザの個人情報の漏洩やシステム改竄など，常に完全に防止すべき内容と，広告の表示や音声の再生など，状況に応じて防ぐべき内容に分けられる．状況に応じて防ぐべき内容は，アクセス内容が状況によって許可されていることから，仮にアクセスが禁止された直後に禁止された処理を実行されたとしても，システムに致命的なダメージを与えることはなく，ユーザのウェアラブルコンピュータの利用性を一時的に低下させるにとどまると考えられる．そのため，セキュリティの常に防止すべき処理に対して完全性が満たされていれば，最低限の機能を備えているといえる．

状況に応じて動的にアクセス権限を変更しない場合，SBAC は RBAC と同等の完全性を備えているため，提案手法はウェアラブル環境における最低限の完全性を満たしている．

5.2 サービスのアクセス権限不足時の処理

SBAC では，サービスの実行時に必要なアクセス権限を満たしているか確認しているため，サービスの実行中にアクセス権限不足になることが考えられる．アクセス権限不足時の処理は，サービスの実行を中止したり，禁止されている処理を無視して続きの処理を実行する，禁止されている処理が再び許可されるまで実行を中止するといったことが考えられる．本研究で実装したプロトタイプシステムでは，状況に応じて禁止する内容を音声の再生やメッセージの表示であると想定し，禁止されている処理を無視するようになっているが，サービスの種類や処理内容によってアクセス権限不足時の対応方法が動的に変更できることが好ましいと考えられる．A-WEAR における SBAC では，ECA ルールによって処理を変更することを考えているが，SBAC におけるアクセス権限不足時の処理は今後の課題である．

6. ま と め

本研究では，ウェアラブルコンピューティングのための状況依存アクセス制御機構 SBAC を提案し，プロトタイプシステムを筆者らの研究グループで提案しているウェアラブルコンピューティングのための基盤システム A-WEAR 上に実装した．提案手法は，サービスベースのアクセス制御・通信経路の暗号化・サービスの送信元とサービス内容の確認・サービス連携の制限という特徴により，ウェアラブルコンピューティングにおけるさまざまな脅威からシステムを保護する．

今後は，アクセス権限不足時の処理を動的に変更できる機構について考案する予定である．

謝　　辞

本研究は，文部科学省振興調整費「情報フィルタリングの数学的基盤の確立」,「モバイル環境向 P2P 型情報共有基盤の確立」，および文部科学省 21 世紀 COE プログラム「ネットワーク共生環境を築く情報技術の創出」，科学研究費補助金 (基盤研究 (B)(2))「大規模な仮想空間システムを構築する放送型サイバースペースに関する研究」(プロジェクト番号:15300033) の研究助成によるものである．ここに記して謝意を表す．

文　　献

[1] Barkley, J.: Comparing Simple Role Based Access Control Models and Access Control Lists, *Second ACM Workshop on Role-Based Access Control*, pp. 127–132 (1997).

[2] Kortuem, G., Bauer, M. and Segall, Z.: NETMAN: The design of a collaborative wearable computer system, *Mobile Networks and Applications*, Vol. 4, No. 1, pp. 49–58 (1999).

[3] Lehikoinen, J., Holopainen, J., Salimaa, M. and Aldro-Vandi, A.: MEX: A Distributed Software Architecture for Wearable Computers, *ISWC '99 (Third Int. Symp. on Wearable Computers)*, pp. 52–57 (1999).

[4] 三浦直樹, 宮前雅一, 寺田努, 塚本昌彦, 西尾章治郎: Aware-Mail: ウェアラブルコンピューティング環境のためのイベント駆動型メールシステム, 第 65 回情報処理学会全国大会講演論文集 (5), pp. 207–210 (Mar. 2003).

[5] 宮前雅一, 中村聡史, 寺田努, 塚本昌彦, 西尾章治郎: ウェアラブルコンピューティングのための拡張可能なルール処理システム, 情報処理学会研究報告 (情報家電コンピューティング研究グループ 2002-IAC-3), pp. 41–46 (June 2002).

[6] 中村聡史, 宮前雅一, 寺田努, 塚本昌彦, 柳瀬康宏, 釣裕美, 堀雅和, 西尾章治郎: ウェアラブルコンピューティングのためのルール処理システムを用いたサービス, 第 1 回情報科学技術フォーラム (FIT2002) 論文集第 4 分冊, pp. 217–218 (Sep. 2002).

[7] 中尾太郎, 寺田努, 塚本昌彦, 宮前雅一, 庄司武, 岸野泰恵, 義久智樹, 西尾章治郎: ウェアラブル型ルールベースシステムを用いた農作業支援システム, 第 65 回情報処理学会全国大会講演論文集 (5), pp. 211–214 (Mar. 2003).

[8] 寺田努, 塚本昌彦, 宮前雅一, 西尾章治郎: ウェアラブル環境のためのルールベース BGM プレーヤについて, 日本ソフトウェア科学会第 11 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2003) 論文集, pp. 25–30 (Dec. 2003).

[9] 塚本昌彦: モバイルコンピューティング, 岩波書店 (2000).