ホームユビキタスコンピューティングのための脱着型サービス間通信ミドルウェアの実現

中澤 仁[†] 徳田英幸[††]

本論文では，ホームネットワーク上でのユビキタスコンピューティング（ホームユビキタスコンピューティング）のための脱着型サービス間通信ミドルウェアを提案する．同ミドルウェアは，ホームネットワークにおけるサービス統合機構である仮想情報家電機器アーキテクチャ（VNA アーキテクチャ）[14] の通信機構である．ホームユビキタスコンピューティングを実現するには，機器間でのストリーミングやイベント通信といった多態な通信を支援するために，複数のミドルウェアを相補的に運用する必要がある．同問題は，既存の通信ミドルウェアにおける通信プロトコルのサポートが限定的であることから，単一のミドルウェア上に多態なサービスを実現できないことに起因する．本論文では，この問題をアスペクト実現妨害問題として提示する．脱着型サービス間通信ミドルウェアでは，通信プロトコルをはじめとする通信アスペクトの実装がモジュールとして提供されており，プログラマはサービス構造定義言語（SML）文書内にモジュール名を指定することによって，サービスのセマンティクスに適した通信を実現できる．これらによって同ミドルウェアは，アスペクト実現妨害問題を解決した．

A Pluggable S2S Communication Middleware for Home Ubiquitous Computing

Jin Nakazawa[†] and Hideyuki Tokuda[††]

This paper proposes a pluggable service-to-service (S2S) communication mechanism in a middleware for home networks, called Virtual Networked Appliance (VNA) architecture. In the architecture, service description method and the pluggable S2S communication mechanism are separated in an orthogonal way. Through the separation, VNA architecture solved a problem of home networks on which users have to operate multiple heterogeneous middleware technologies simultaneously due to complexity of realizing heterogeneous services on one middleware technology: *aspect realization violation* problem. The pluggable S2S communication mechanism provides service programmers with a simple aspect representation method to define a service-specific protocol concern apart from the service's implementation. It also provides off-the-shelf protocol modules for an inter-service communication, and dynamically loads them based on the aspects defined by the programmer. This reduces the complexity of implementing heterogeneous services on the VNA architecture, thereby addressing the problem.

1. 序　　論

ホームネットワークは，計算機やセンサ，白物家電機器を含む多様な情報家電機器で構成される．センサをはじめとする一部の情報家電機器は壁や天井等に埋め込まれ，人間の目につかない場所に計算能力が配置されたスマートスペースを構成する．ユーザは複数の情報家電機器を操作し，それらを協調させながら様々な活動を行う．このような観点から，ホームネットワークはユビキタスコンピューティング環境の一例であるといえる．本研究の目的は，情報家電機器の操作やそれらの協調を，柔軟かつ容易に行えるミドルウェアを提供し，ホームネットワーク上でのユーザの活動を支援することである．同ミドルウェアでは，情報家電機器が持つ機能をサービスとして抽象化し，ホームネットワークを介して制御可能とする．

一方，情報家電機器の統合利用を目的としたソフトウェア技術に関する研究は，ホームネットワークを含めたユビキタスコンピューティング環境を対象として

† 慶應義塾大学政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

いくつか行われている．各技術は，独自のサービス記述方式とサービス間通信ミドルウェアを提供する．通常，前者は API として，後者は通信プロトコルを実装したモジュールとして提供される．しかし，既存の技術では，サービス間通信ミドルウェアが固定的な機能のみを提供しており，サービスプログラマは通信に関する機能要求（本論文では，通信アスペクト，もしくは単にアスペクトと呼ぶ）を指定できない．また，サービス記述方式とサービス間通信ミドルウェアとが相互に依存しているため，どちらかのみを他と入れ替えることはできない．これらは，プログラマとユーザの双方に対して問題となる．本論文ではこれをアスペクト実現妨害問題として提示する．

本研究では，この問題を解決するミドルウェアとして，Virtual Networked Appliance アーキテクチャ[13),14),31),32)]（以降，VNA アーキテクチャと呼ぶ）を開発してきた．本論文では，同アーキテクチャ中のサービス間通信ミドルウェアである脱着型通信ミドルウェアを提案する．VNA アーキテクチャでは，サービス記述方式と脱着型サービス間通信ミドルウェアとが直交的手法により分離されている．プログラマは，通信プロトコルを選択することで，脱着型サービス間通信ミドルウェアに対して，構築するサービスに適した通信アスペクトを指定できる．また，既知の通信プロトコルを選択可能としたことによって，既存のアプリケーションやミドルウェアとの互換性が向上した．

本論文ではまず，2 章において上述した問題について述べ，3 章において VNA アーキテクチャの概要を紹介する．次に 4 章で脱着型通信ミドルウェアを提案するとともに，設計および実装の詳細を述べる．5 章および 6 章ではそれぞれ，実装に基づく実験評価，および関連研究に関する議論を行う．最後に，7 章で本論文をまとめる．

2. ホームユビキタスコンピューティングにおけるサービス間通信

サービス統合ソフトウェアが固定的なサービス間通信ミドルウェアを提供することによって，プログラマは 1 つのミドルウェア上に多様なサービスを実装することが困難となる．結果としてユーザは，ホームユビキタスコンピューティングの実現に複数のミドルウェアの同時運用が必要となる．

2.1 ユビキタスコンピューティング環境とサービス間通信ミドルウェア

本論文では，ユビキタス計算機環境におけるユーザの活動の例として，以下に示すシナリオ「Follow-You-and-Me ビデオ」を扱う．

> 大学の研究員であるとともに主夫であり，また 2 才の娘の父でもある彼は，2010 年の今日，情報処理学会の論文誌投稿締切りであるために，家にはいるが娘の面倒をみてやることができない．そこで彼は，娘におもちゃを渡して，1 人で遊ばせておくことにした．だが彼は心配になってきたので，家中のビデオカメラとテレビ，ディスプレイを使って，娘を遠隔監視することにした．娘の画像は彼女に最も近いビデオカメラで撮影され，ネットワークを介して彼に一番近いテレビやディスプレイに送信される．彼女が別の部屋に移動したときには，移動元の部屋の電灯を消して，移動先の部屋の電灯を自動的につける．また彼が別の部屋に移動したときには，映像を表示するテレビやディスプレイを自動的に切り替える．

このシナリオでは，ビデオストリームがつねに「彼」を追尾（follow）し，また表示される映像はつねに「彼の娘」を追尾する．ホームネットワーク上でこのようなシナリオを実現するためには，情報家電機器を直接制御する機構と，情報家電機器間でデータを交換する機構が必要となる．本論文では，前者を**サービス**，後者を**サービス間通信**と呼ぶ．シナリオ「Follow-You-and-Me ビデオ」では，各機器において**表 1** に示すサービスが必要となる．

表 1 シナリオ中で必要となるサービス
Table 1 Services required in the scenario.

情報家電機器	サービス
ビデオカメラ	電源制御 ビデオストリーム送信
テレビ	電源制御 ビデオストリーム受信 ビデオストリーム再生
ディスプレイ	テレビと同様
照明	電源制御
位置センサ	位置情報管理

ネットワーク上に分散するサービスを統合利用するための主要なソフトウェア技術としては，分散オブジェクトコンピューティング（DOC，Distributed Object Computing）があげられる．CORBA（以降，CCM と呼ぶ）コンポーネントモデル[18)]，Enterprise Java Beans（以降，EJB と呼ぶ）[27)]，および Jini[26)] 等がこの例である．これらのソフトウェア技術は，サービス間通信ミドルウェアとして遠隔メソッド呼び出しを採用している．このためプログラマは，あるサービスの遠隔オブジェクトインタフェースを自由に決定できる．この特性は，クライアントサービスを記述するプログ

表 2 既存のサービス統合アーキテクチャの構成
Table 2 Configuration of existing service integration architectures.

名称	サービス記述方式	サービス間通信ミドルウェア
Jini [26]	Java API	Java RMI
Document-based Framework [9]	XML [30] Document	Remote Method Invocation
UPnP [12),28]	XML Document	HTTP, SOAP [3]
HAVi [23]	Havlet	HAVi Messaging System
ADS [10]	Access Point	Ninja [6),7]

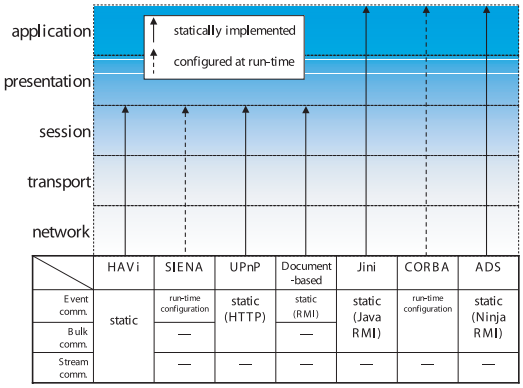

図 1 既存のミドルウェアが提供するサービス間通信形態
Fig. 1 Supported forms of S2S communications of the major existing middleware technologies.

ラマに対して，サーバサービスの遠隔オブジェクトインタフェースをあらかじめ知っておかなくてはならないという制約を課す．すなわち，上述したソフトウェア技術では，クライアントサービスは特定のサーバサービスに依存しており，また未来に開発されるサーバサービスとは通信を行えない．ホームユビキタスコンピューティング環境には多様な情報家電機器が含まれており，また新たに開発された情報家電機器が頻繁に追加されると考えられる．したがって，上述したソフトウェア技術は，ホームユビキタスコンピューティング環境には適用できない．

これに対して，ホームユビキタスコンピューティングの支援を目的として開発されたソフトウェア技術では，サービス間の互換性を向上させる設計がなされている．**表 2** に，そのようなソフトウェア技術のうち代表的なものを示す．これらのソフトウェア技術では，サービスの記述方式，およびサービス間で使用する通信ミドルウェアを共通とし，互換性の向上を図っている．特に，通信ミドルウェアが使用するプロトコルを静的に固定している．

2.2 アスペクト実現妨害問題

サービス間の通信プロトコルを固定することによって，あるミドルウェア上で実現できるサービスが限定されることになる．**図 1** に，既存のサービス間通信ミドルウェアがサポートする通信形態および通信プロトコルを示す．

例として，以下の 2 つのサービスを考察する．

電源制御サービス 他のサービスから受け取ったイベントに基づいて電源を制御する．通信に関するアスペクトとして，高信頼型イベント通信を持つ．

ビデオストリーム送信サービス 他のサービスにビデオストリームを送信する．通信に関するアスペクトとして，実時間ストリーム通信を持つ．

前者のサービスは，イベント通信をサポートする Jini や UPnP，および ADS のすべてで実現できる．これは，上述のサービス間通信ミドルウェアが，サービスが持つ通信に関するアスペクトに適合するためである．これに対して，後者のサービスは，これらのミドルウェア上に実現できない．RMI や HTTP を用いながら，通信に実時間制約を与えることは不可能であるため，上記の各ミドルウェアが，サービスが持つ通信に関するアスペクトに適合しないためである．

このように，サービス間通信プロトコルが固定されているミドルウェア上には，当該プロトコルに適合するサービスのみしか提供できない．すなわち，既存のサービス間通信ミドルウェアは，サービスが持つ様々なアスペクトの実現を妨害している．

2.3 ミドルウェアのフラグメント化

上述のように，サービス間通信ミドルウェアが限定的な通信プロトコルのサポートを提供する場合，プログラマは実装するサービスの特性に合致したミドルウェアを選択しなければならない．結果としてホームユビキタスコンピューティングを実現するために，一様なサービスを提供する多様なミドルウェアが 1 つのネットワーク上で必要となる．これによってプログラマとユーザのそれぞれに関して，以下に示すように負荷が増大する．

まずプログラマは，他のミドルウェア上のサービスをも考慮したプログラミングを行う必要が生じる．ある情報家電機器を開発する際に，イベント通信とストリームデータ通信を実現する必要があるとする．このとき，イベント通信とストリームデータ通信のそれぞ

れに適した異なるミドルウェアを選択すると，片方のミドルウェアを介して受信したイベント情報に連動してストリームデータの送受信を開始するというような，ミドルウェアをまたがった同期を行わなくてはならない．これは，プログラマが実現したいサービス本来の本質的な実装とは異なる，付加的な作業である．このような作業は，たとえばイベント通信に適したミドルウェアにストリームデータ通信機能を追加して，新たなミドルウェアを派生させるといった補完作業を行えば回避できる．しかしこの作業も，そのミドルウェアに元来含まれない通信プロトコルの実装が必要となる点において付加的な作業である．また仮に補完作業が容易に行えたとしても，プログラマごとに補完作業を行えば，派生ミドルウェア間の互換性が失われる．

次にユーザは，複数のミドルウェアを同時に利用および管理する必要が生じる．情報家電機器の制御方法はミドルウェアごとに異なる．上述のようなミドルウェアの選択がプログラマごとに行われると，ユーザはそれぞれ異なる方法で情報家電機器を制御しなくてはならない．また Jini や ADS のように独立したサーバプログラムを必要とするミドルウェアでは，それらのプログラムを動作させるホストの管理が必要である．さらに，情報家電機器の制御に際してエラーが発生した場合，ミドルウェアごとにエラーの通知方法が異なるため，その内容や原因の把握が困難になる．特に前述のようなミドルウェア間の同期が行われている場合には，エラー箇所の特定が困難となる．

したがって，現在の静的なサービス間通信ミドルウェアを用いたサービス統合ソフトウェア技術の構成手法は現実的ではない．サービスが持つ多様なアスペクトを実現できるサービス間通信ミドルウェアを備えたソフトウェア技術が必要である．

3. 仮想情報家電機器アーキテクチャ

本章では，ホームユビキタスコンピューティングのためのサービス統合ソフトウェアである，VNA アーキテクチャの概要を示す．**図 2** に，同アーキテクチャの概念図を示す．また次章では，同アーキテクチャ中の脱着型サービス間通信機構を提案する．

3.1 VNA アーキテクチャの概要

VNA アーキテクチャは，前述した EJB や CCM と同様に，分散オブジェクトコンピューティング機構である．VNA アーキテクチャがそれらと異なる点は，次章で提案する脱着型サービス間通信ミドルウェアが，通信の形態およびプロトコルを限定しない点である．これによって，VNA アーキテクチャ上に多様なサービスを提供可能となった．**表 3** に，現在サポートされている情報家電機器の例を示す．

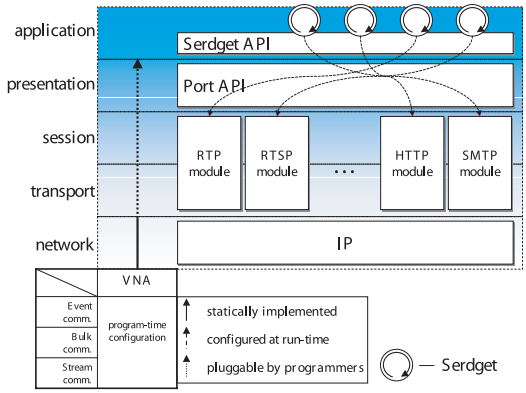

図 2 VNA アーキテクチャにおける通信形態とプロトコル
Fig. 2 S2S communication in VNA architecture.

VNA アーキテクチャの実装は，サービス生成機構，脱着型サービス間通信ミドルウェア，ディレクトリ機構，およびサービス利用機構から構成されている．これらの機構はランタイムソフトウェア（以降，VNA ランタイムと呼ぶ）として統合されており，これの有無によって情報家電機器を 2 つに分類できる．まず，計算機能を持ち VNA ランタイムを直接動作させられる情報家電機器を，マスタ情報家電機器と呼ぶ．これに対して計算機能を持たず，2 点間接続されたマスタ情報家電機器に VNA ランタイムの動作を委譲する情報家電機器を，スレーブ情報家電機器と呼ぶ．本章では，VNA アーキテクチャの実装概要を各機構ごとに述べる．

3.2 サービス生成

VNA アーキテクチャでは，情報家電機器が持つ機能を，Serdget（Service Gadget）と呼ばれるソフトウェアコンポーネントに抽象化する．これによって，情報家電機器に固有の制御プロトコルやデータ取得プロトコルを，VNA アーキテクチャ中の他の部分，あるいは他の Serdget に対して隠蔽する．チューナ，タイマ，テープ録画，およびテープ再生の各機能を持つビデオデッキの場合，チューナとタイマの各機能に対応する Serdget と，テープ録画とテープ再生を同時に制御するビデオテープ入出力 Serdget に抽象化する．機能の制御やデータの交換は，VNA アーキテクチャにおける通信エンドポイントである**ポート**を介して行われる．ポートにはデータ交換用ポートとイベント交換用ポートが存在し，データ交換用ポートはさらに，交換するデータの種類によってストリームデータポートとバルクデータポートとに分類できる．それぞれの

表 3 サポートされている機器の例
Table 3 Summary of supported information appliances.

機器名	ベンダ名	Serdget 名	ポート数	行数	サイズ
Spider（RF タグ読み取り装置）	Ecode	EcodeSpiderSerdget	4	122 行	5.13 KB
FAM3（電源制御装置）	横河電機	FAM3PowerSerdget	2	117 行	4.63 KB
EVI-D30（首振りカメラ）	ソニー	SonyEVID30Controller	4	104 行	4.36 KB
JMF [25]（AV ストリーム受信）	Sun Microsystems	MediaPlayer	1	159 行	6.51 KB
JMF（AV ストリーム送信）	Sun Microsystems	Movie	1	111 行	3.32 KB
実験装置 SSLab [17] 内の電灯	—	RoomLight	3	101 行	3.90 KB
PDP 502M（プラズマディスプレイ）	パイオニア	PDP	6	98 行	3.55 KB
DSP AX-1（アンプ）	ヤマハ	AVAmp	23	147 行	4.47 KB
DA100（温度センサ）	横河電機	DA100Thermometer	4	179 行	7.75 KB
WV-DR9（デジタルビデオデッキ）	ソニー	DVDeck	8	82 行	3.26 KB

ポートごとに入力と出力があるため，Serdget は合計 6 種類のポートを使用して通信を行える．

Serdget の構造は，XML ベースのマークアップ言語である Serdget マークアップ言語（Serdget Markup Language，以降，SML と呼ぶ）を用いて定義される．同文書に定義された情報は，Serdget の属性情報を管理する Profile オブジェクトに保持される．Profile オブジェクトには，Serdget の型，名前，ベンダ，バージョン，位置情報，ポート数等が含まれている．

3.3 ディレクトリ

ディレクトリ機構は，ネットワークに接続された他の全情報家電機器のそれと，互いがホストしている Serdget の Profile オブジェクトを交換して Serdget 管理表を構築する．すなわちすべての VNA ランタイムは，ネットワーク上に存在するすべての Serdget の属性情報をローカルに保持している．したがって，ネットワークを介した検索要求は発生しない．上述した Profile オブジェクトの交換は，新たな情報家電機器がネットワークに追加されたり，ネットワークから情報家電機器が削除された際にのみ発生する．すなわち VNA アーキテクチャでは，管理表の構成を冗長とすることによって検索性能の向上を図った．

3.4 サービス利用

ユーザは，ホームネットワーク上に分散した Serdget を，タスクグラフという概念に基づいて組み合わせて利用する．タスクグラフ中では，ユーザが行いたいタスクに必要となる Serdget がグラフノードとして，また Serdget 間での通信パスがグラフエッジとして定義されている．このタスクグラフを通して複数の Serdget が組み合わせられ，ユーザのタスク要求を満たす 1 台の論理的な情報家電機器を構成することから，本研究ではこれを仮想情報家電機器（Virtual Networked Appliance，VNA）と呼んでいる．

VNA は，情報家電機器の開発者をはじめとする，VNA アーキテクチャに関する知識を豊富に持つユーザが定義する[31]．この定義は，XML ベースのマークアップ言語である VNA マークアップ言語（VNA Markup Language，以降，VML と呼ぶ）を用いて行われる．VML 文書中では，`<template>` タグを用いてグラフノード，すなわち当該 VNA が必要とする情報家電機器や Serdget，およびポートの属性情報が指定される．Serdget 間の通信パス，すなわちグラフエッジは，`<message>` タグ内に指定される．同タグの `from` 属性にはメッセージの送信元が，`to` 属性には送信先が指定される．

ユーザがサービス利用機構に対してタスク要求を行うと，同機構はホームネットワーク上に存在する Serdget のうち，当該 VNA のグラフノードの定義を満たすものを検索し，それらを組み合わせる．この，実際のホームネットワーク上で VNA の利用を開始する処理を，マッピングと呼ぶ．マッピングは通常，テレビやディスプレイ，あるいは PC 等のユーザ側情報家電機器で行われる．マッピングを実現したことによって，VNA アーキテクチャや情報家電機器に関する知識を持たないユーザでも，それらの使用や操作が可能となった．ユーザは，小型記憶装置に保存した VNA を携帯し，移動先でマッピング操作を行うことによって，場所を問わずに同一のサービス統合を実現できる．このとき，Serdget の検索処理は VNA をロードした情報家電機器内で閉じている．

4. 脱着型サービス間通信ミドルウェア

本章では，「Follow-You-and-Me ビデオ」の例に基づいて，VNA アーキテクチャにおける脱着型サービス間通信ミドルウェア[15),16)] を提案する．

4.1 設　　計

VNA アーキテクチャでは，脱着型サービス間通信ミドルウェアとサービス記述方式を分離し，Serdget が持つ Port ごとに通信に関するアスペクトの指定を可能とした（**図 3**）．上記の特性は，通信プロトコル

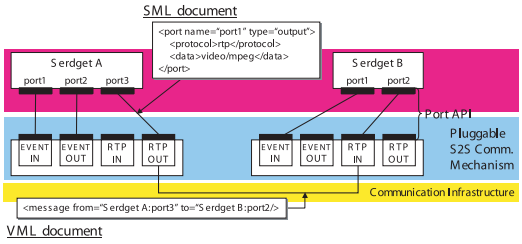

図 3 脱着型サービス間通信機構の概念図
Fig. 3 Pluggable S2S communication mechanism in VNA architecture.

の実装をプロトコルモジュールとして提供し，それらをアスペクト記述に基づいて動的にロードすることによって実現した．本節では脱着の方式，レイヤ，および粒度の各観点から本アプローチの特性を述べる．また 6 章では，同様の観点で関連研究の分析を行う．

4.1.1 脱着方式

通信プロトコルを脱着する方式には，ミドルウェア中の該当部分全体をプログラマの希望するモジュールと入れ替える直接脱着方式と，標準となる実装にそれを追加的に挟み込む間接脱着方式とがある．直接脱着方式では，サービスプログラム中の通信メソッド（もしくは関数）呼び出しがプログラマの希望するモジュールと直接リンクされている．したがって，通信プロトコルを固定している他のミドルウェアと同等の通信性能を得られる．ただしモジュール中にバグが存在した場合，当該モジュールをロードしたミドルウェア全体に影響を与える．間接脱着方式では，プロキシサーバやそれと同様の役割をするオブジェクトがサービス間通信を中継するため，前者と比較して性能が低下する．ただし，あるプロキシサーバが動作を停止した場合でも，ミドルウェア全体の動作停止は発生しない．

本ミドルウェアでは，サービス間通信性能の維持を重視して直接脱着方式を採用した．また，実装に Java 言語を採用したことにより，不正なメモリ操作や型操作に起因するバグの混在を抑制している．

4.1.2 脱着レイヤ

通信プロトコルを脱着するレイヤには，OSI7 層モデルにおけるアプリケーション層プロトコルまでを脱着可能とするものと，プレゼンテーション層以下までのものとに分類できる．前者の場合サービスプログラムは，既定の API を介してモジュールを操作し通信を行う．後者の場合，たとえば CORBA にみられるように，オブジェクトの外部インタフェースをプログラマが自由に決定し，同インタフェースを介した通信に使用するプロトコルが脱着可能となっている．このとき外部インタフェースを知らなければサーバオブジェクトと通信できないという観点から，その外部インタフェースをアプリケーション層プロトコルと見なせる．

本ミドルウェアでは，サービスが持つアスペクトを通信に直接反映できるよう，アプリケーション層プロトコルまでを脱着可能とした．またプロトコルモジュールを操作する共通インタフェースとして *Port* API を規定した．同 API は通信形態ごとの抽象クラスの集合である．情報家電機器やサービスの型に基づくインタフェースと比較して限定的なインタフェースとしたことによって，サービスどうしがより柔軟な組合せで通信可能となったことに加えて，プロトコルモジュールの操作が容易になっている．

4.1.3 脱着の粒度

通信プロトコルを脱着する粒度には，ミドルウェア全体，サービスごと，および通信端点ごとに 1 つのモジュールを脱着可能とする方式がある．ミドルウェア全体を対象とする方式では主に，そのミドルウェアが運用されるネットワークの帯域や遅延等の特性に応じて，管理者がモジュールを選択することが目的となっている．サービスごとに脱着を行う方式は，前述した間接脱着方式を採用するミドルウェアで主に利用されている．通信端点ごとに脱着を行う方式では，たとえば遠隔メソッドごとに使用するプロトコルを変更するといった機構が考えられる．各サービスあるいは各通信端点を対象とする方式では，プログラマがモジュールの選択を行う．またサービス全体を脱着の対象とした場合，多様なサービスのそれぞれに適したモジュールをミドルウェア開発者があらかじめ提供することは困難である．これに対して各通信端点を対象とした場合，ミドルウェア開発者によるモジュールの提供が容易となる．

本ミドルウェアでは，通信端点（ポート）をイベント，バルクデータ，およびストリームデータとに分類し，それぞれのプロトコルモジュールを共通インタフェースである *Port* API に基づいて実装した．またポートごとのアスペクト指定を，前章に示した SML を用いて容易に行えるようにした．これによって本ミドルウェアでは，SML 文書中にアスペクトに適したモジュール名を記述するだけで済み，その変更に際して Serdget プログラムの変更を不要とした．

4.2 実　　装

本節では，脱着型サービス間通信ミドルウェアの Java 言語による実装を概説する．

4.2.1 プロトコルモジュール

本ミドルウェアは，基盤モジュールとプロトコルモジュールとで構成される．基盤モジュールは現在，IP

```
<?xml version="1.0"?>
<!DOCTYPE component>
<component>
  <head>
    <name>Movie</name>
    <comment>...</comment>
    <seealso>...</seealso>
  </head>
  <body>
    <codebase>Movie.jar</codebase>
    <class>Movie</class>
    <port type="output" name="output">
      <content-type>video/mpeg</content-type>
      <protocol>rtp</protocol>
      <comment>...</comment>
    </port>
    <params>
      <param name="addr" type="string">224.0.0.1</param>
      <param name="port" type="integer">23456</param>
      <param name="uri" type="string"></param>
    </params>
  </body>
</component>
```

図 4 Movie Serdget の構造定義
Fig. 4 Structure definition of Movie Serdget.

ネットワークの使用を前提として実装されている．同モジュールは，全プロトコルモジュールが共通に使用するモジュールで，通信パスの構築や，各ポートの宛先 IP アドレスおよびポート番号の管理等を行う．基盤モジュールのみを他と入れ替えることによって，IP 以外のネットワークにも対応できる．

プロトコルモジュールは，Serdget が持つ通信端点であるポートの実装である．入力モジュールと出力モジュールの組として提供され，それらはクライアント側サービスとサーバ側サービスにおいて，一対のフラグメントオブジェクト[8] として動作する．これにより，両サービス間の通信路における帯域予約や通信内容の暗号化をはじめとした協調を必要とするアスペクトも実現可能である．

現在，IP プロトコルを使用した基盤モジュール上で，ベストエフォートでの通信を行うモジュールが実装済である．具体的には，EVENT，RAW☆，RTP，RTSP[22]（入力のみ），HTTP，および SMTP 等がサポートされている．これらの既知のプロトコルを採用したことによって，既存のソフトウェアとの互換性が向上した．

4.2.2 脱　　着

プロトコルモジュールの選択は，Serdget の構造定義中で行う．すなわちプログラマは，構造定義中の通信アスペクト記述を変更することによって，プロトコルモジュールを容易に脱着できる．**図 4** に，シナリオで用いる Serdget の 1 つである Movie の構造定義を示す．この構造定義では，`output` ポートの通信アスペクトとして，`<protocol>` タグ中に RTP が指定されている．すなわちこのプログラマは，動画像を実時間ストリーム通信として送信することを，本機構に伝達している．このほかに，動画像をバルクデータ通信により送信したければ，プロトコル属性に HTTP を指定すればよい．

Serdget のインスタンスを生成する際には，まず `<class>` タグに指定された定義クラスをインスタンス化し，同クラスが実装している *init* メソッドと *start* メソッドを呼び出す．次に，通信アスペクトに基づいて，適切なポートをインスタンス化し，同 Serdget に結合する．ポートをインスタンス化する際には，通信アスペクトに基づいて以下のクラスを動的にロードし，インスタンスを作成する．

package.protocol.{*Input* または *Output*}

このうち，*package* はプロトコルモジュールを格納しておくパッケージ名であり，VNA アーキテクチャの設定ファイル中に記述される．また，*protocol* は通信アスペクト中に記述される．

4.2.3 通信パスの構築

ユーザは，VNA の使用を開始する際，それを VNA ランタイムにロードする．このとき VNA ランタイムは，VNA 定義中の `<message>` タグに記述された通信パスを構築する．

図 5 に，「Follow-You-and-Me ビデオ」を実現する VNA を定義した XML 文書を示す．この定義には，5 個の `<template>` タグと 3 個の `<message>` タグが存在する．`<template>` タグのうち `jin` と `daughter` は，それぞれ *Jin* あるいは *daughter* というユーザの位置を特定するための記述である．ただし，両ユーザがそれぞれ，*Jin* と *daughter* と命名された VNA ランタイムが動作するウェアラブル機器を携帯していることを想定している．その他の内容を以下に示す．

screen 「彼」に近接する情報家電機器で，ビデオストリーム受信サービスを提供する Serdget に対応するノード

camera 「娘」に近接する情報家電機器で，ビデオストリーム送信サービスを提供する Serdget に対応するノード

light 「娘」の存在する空間の電灯で，電源制御サービスを提供する Serdget に対応するノード

また，最初の `<message>` タグは `camera` から `screen` に対してビデオストリーム用の通信パスを構築するための記述であり，ほかは，`light` の電源を制御するイ

☆ EVENT プロトコルと RAW プロトコルはそれぞれ，0 バイトのイベントデータおよびサイズ制限のないバイト列を転送する際に使用するプロトコルであり，本研究で実験用に開発したものである．

```
<?xml version="1.0"?>
<!DOCTYPE composite SYSTEM "composite.dtd">
<composite>
  <head>
    <name>follow you and me video</name>
    <comment></comment>
    <seealso>http://www.ht.sfc.keio.ac.jp</seealso>
  </head>
  <body source="SimpleWindow.zip" name="">
    <template name="jin">
      <hint name="type">user</hint>
      <hint name="name">Jin*</hint>
    </template>

    <template name="daughter">
      <hint name="type">user</hint>
      <hint name="name">daughter</hint>
    </template>

    <template name="camera">
      <hint name="type">camera</hint>
      <hint name="location">${daughter->location}</hint>
    </template>

    <template name="screen">
      <hint name="type">MediaPlayer</hint>
      <hint name="location">${jin->location}</hint>
    </template>

    <template name="light">
      <hint name="type">RoomLight</hint>
      <hint name="location">${daughther->location}</hint>
    </template>

    <message from="camera:video out" to="screen:video in"/>
    <message from="light:metamorphosed" to="light:on"/>
    <message from="light:metamorphosing" to="light:off"/>
  </body>
</composite>
```

図 5 「Follow-You-and-Me ビデオ」VNA の定義
Fig. 5 Follow-You-and-Me video VNA.

表 4 各ポートのメッセージ送受信メソッドの一覧
Table 4 Abstract method of each port.

クラス	メソッドリスト
StreamInputPort	*start*, *stop*, *pause*
StreamOutputPort	*start*, *stop*, *pause*
DataInputPort	*receive*, *request*
DataOutputPort	*send*
EventOutputPort	*fire*
EventInputPort	*receive*

ベント用の通信パスを構築するための記述である．

4.2.4 メッセージの送受信

すべてのプロトコルモジュールは**表 4** に示したいずれかのポートに対応する抽象クラスを継承している．表 4 に，各クラスが定義しているメッセージ送受信のための抽象メソッドを示す．表中に示したメソッドのほかに，すべてのインタフェースはメッセージを待機するリスナオブジェクトを登録するために，*setPortListener* メソッドを持つ．同一の親クラスを継承したプロトコルモジュールどうしは，脱着により交換可能である．これによりプログラマは，各プロトコルモジュールの詳細に関知せずに，共通の手法を用いてそれらを使用できる．

`from` 属性に指定されたポートと，`to` 属性に指定されたポートで，指定された通信アスペクトが合致しない場合には，`from` 属性側の VNA ランタイムから，`to` 属性側の VNA ランタイムに対して，適切なプロトコルモジュールを移送する．`to` 属性側の VNA ランタイムでは，移送されたプロトコルモジュールが本来のものと交換可能であれば脱着を行う．この処理を動的脱着と呼ぶ．ただし，`to` 属性に指定されたポートが既存の通信パスの一部となっている場合は動的脱着を行わず，通信パスの構築は失敗する．

5. 評　　価

情報家電機器は，ユーザの操作に対する迅速な応答が求められる．VNA アーキテクチャでは，Serdget の実装を軽量化するとともに，必要なプロトコルモジュールだけを動的にロードすることによって，メモリ使用量を低減している．しかし，Java 実行環境自体の CPU 負荷およびメモリ負荷が大きく，情報家電機器に対しても現在の PC と同等の性能を期待できない．本研究では，Java 実行環境が動作し，かつ性能上の制約を持つ機器として，**表 5** に示す 2 種類の機器において，VNA アーキテクチャのオーバヘッドを測定した．ただし実験は，本アーキテクチャのサービス間通信に関連する部分，特にポートおよび通信パスの生成に重点を置き，本章ではユーザの視点から測定結果をまとめ，考察する．

5.1 情報家電機器の起動

VNA アーキテクチャでは，情報家電機器にインストールされている Serdget のロードのためのオーバヘッドが生じる．

Serdget のロードには，Serdget 自体のインスタンス化とそれに付随するポートのインスタンス化が必要となる．したがって本実験では，4 章に示したものを含めた全実装済 Serdget を測定対象とした．**図 6** に測定結果をまとめた．測定対象となった Serdget のうち，バイトコードの大きさに関しては，最小が 3.26 KB，最大が 10.2 KB であったが，ロード時間との関連は認められなかった．これに対してポート数に関しては，最小が 0 個，最大が 23 個であり，多数のポートを持つ Serdget ほどロード時間を要した．ロードに要したオーバヘッドは，ThinkPad X22 では最大でも 1 秒弱であり，その他の場合では 350 ミリ秒未満であった．また，iPAQ でもポート数 8 個で 6 秒強，23 個と多数の場合でも 15 秒弱であった．

表 5 実験に使用した iPAQ H3660 と IBM ThinkPad X22
Table 5 Specification of iPAQ H3660 and IBM ThinkPad X22.

	Compaq iPAQ H3660	IBM ThinkPad X22
CPU	Intel StrongARM (206 MHz)	Intel Pentium III 800 MHz
RAM	64 MB	256 MB
Display	3.8" LCD	12.1" LCD
OS	Familiar Linux v0.3	FreeBSD 4.5R
JDK	Blackdown JDK 1.3.1	Sun Microsystems JDK 1.3.1 + FreeBSD Patch

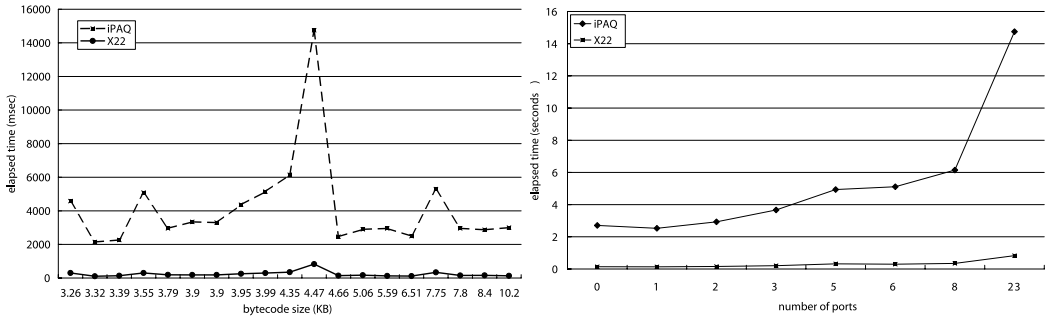

図 6 iPAQ（上）と ThinkPad X22（下）Serdget のロードに関する性能
Fig. 6 Performance of Serdget loading on iPAQ (the upper line) and ThinkPad X22 (the lower line).

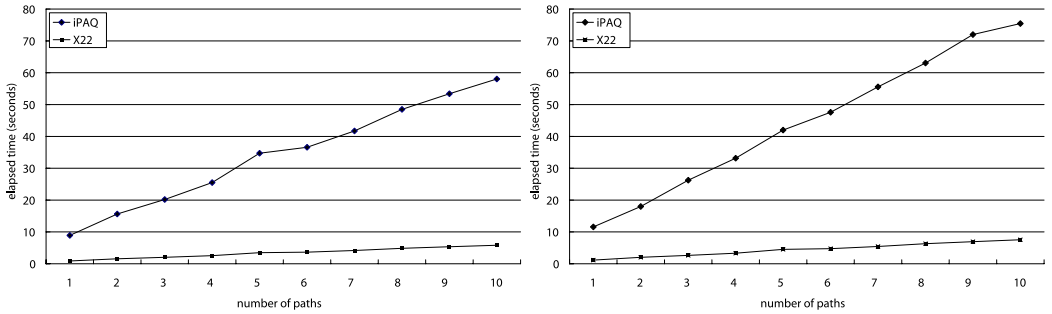

図 7 動的脱着を行わない場合（左）と行う場合（右）のマッピング処理の性能
Fig. 7 Time costs for mapping operation.

Serdget のロードは情報家電機器に通電された際にのみ発生する．したがって上述の結果は，iPAQ のような性能上の制約が強い情報家電機器でも現実的な時間内で起動できることを意味する．1 つの情報家電機器に 8 個のポートを持つ Serdget が 10 個動作するとしても，そのすべてを 1 分程度でロードできる．これは近年の計算機の起動時間と同等である．

5.2 マッピング

ユーザは，VNA のロードを VNA ランタイムに指示し，ロードが完了するまで待機する．すなわちマッピングに関するオーバヘッドは，ユーザが実際に機器を使用する際の待ち時間である．したがってこれを抑制し，ユーザに対する直接的な負荷を軽減することが望ましい．本実験では，2 個から 20 個の `<template>` タグと，それらを接続する 1 個から 10 個の `<message>` タグを含む VNA 定義を用意し，それぞれマッピング処理に要した時間を測定した．**図 7** に，通信パス数の観点から結果をまとめた．このうち，動的脱着に要した時間は全体のおよそ 30%であった．

実験から ThinkPad X22 では，最も複雑な VNA の場合でもオーバヘッドが 6 秒未満に抑制されている．「Follow-You-and-Me ビデオ」VNA をこの結果に当てはめると，オーバヘッドは 2.02 秒となる．この値は，シナリオに示したすべての機器を人間が手動で操作した場合と比較して，高速である．この結果から，現在ほぼ各家庭にある標準的な PC を用いれば，マッピング処理を十分高速に行えることが示された．これに対して現在の iPAQ が持つ性能は，VNA アーキテ

クチャにおけるマッピング処理には不足しているといえる．VNA 定義の解釈には，DOM を用いた XML 文書の操作をともない，メモリ使用量の観点から性能上の制約が強い機器ではオーバヘッドが大きくなる．また Serdget の検索時には，全 Serdget のプロファイルと `<template>` タグ情報との，正規表現を用いたマッチングが必要となり，同様にオーバヘッドが大きくなる．ただし，3.2 節で述べたように，すべての VNA ランタイムはネットワーク上に存在する全 Serdget の Profile オブジェクトを保持している．したがってマッピング処理はどの情報家電機器上で行っても同様の結果が得られる．すなわち本実験の結果は，ネットワーク上に性能上の制約を持つ情報家電機器が接続されていたとしても，PC 程度の性能を持つ機器上でマッピング処理を行えば，それらを現実的な性能で統合利用できることを意味する．

6. 関 連 研 究

TAO [21] では，CORBA サービス間で時間制約をともなった通信を実現するために，Pluggable Protocols Framework [19] を提案している．同フレームワークでは，下位層ネットワークの特性に応じたトランスポートプロトコルモジュールの直接脱着による，ORB の動的構成を行う．動的構成は ORB の開始時に，設定ファイルに基づいて行われる．TAO におけるプロトコルモジュールの選択が ORB 全体に対して行われるのに対して，脱着型サービス間通信ミドルウェアは，サービスが持つ通信端点ごとに行われる．すなわち，あるサービスが複数のポートを持つとき，各ポートごとに異なるプロトコルを使用できる．これによって，サービスごとに粒度の細かい通信アスペクト指定が可能である．

QuO アーキテクチャ[20),29)] は，一対の CORBA オブジェクト間にプロキシオブジェクトを挿入して通信の QoS 制御を行う，間接脱着方式を採用している．同アーキテクチャでは，通常の IDL に加えて QoS Description Language（QDL）を導入し，QoS パラメータを指定する．プロキシオブジェクトは *delegate* と呼ばれ，QDL と IDL の定義から自動生成される．QDL における通信アスペクトの指定はサービスごとに行われ，本論文が提案する通信端点ごとのプロトコルモジュール選択の方が粒度が細かい．これに加えて，本研究で実現した通信アスペクト指定は，上述した QoS パラメータ指定と同程度に単純である．また脱着型サービス間通信ミドルウェアにおいても入力モジュールと出力モジュールが協調的に動作して，End-to-End の QoS 管理を行える．

AspectIX [8] は，CORBA を拡張してサービス単位の QoS 管理を実現している．同機構では，Qoslet と呼ばれるフラグメントオブジェクトを，クライアント側オブジェクトとサーバ側オブジェクトの間に間接脱着し，両オブジェクト本来の機能に，QoS 管理機能を付加できる．Qoslet はサーバ側オブジェクトからクライアント側オブジェクトへ移送されるため，両オブジェクト間でフラグメントの不一致が発生することはない．Qoslet はサービス単位で脱着が行われ，プレゼンテーション層以下のプロトコルを代替する機能を持つ．AspectIX では Qoslet をプログラマが実装する必要があるのに対して，脱着型サービス間通信ミドルウェアではプログラマが所与のモジュールを選択する方式をとっているため，サービスの本質的な実装に集中できる．また本ミドルウェアではモジュール選択を通信端点ごとに行えるため，より柔軟な指定を行える．

Smart Proxies [11] も，AspectIX と同様に，CORBA を拡張して QoS 管理機能を実現している．同機構では，CORBA オブジェクト間に QoS 管理機能を付加するためのプロキシオブジェクトがサービスごとに間接脱着される．同プロキシオブジェクトは，IDL によって定義された CORBA オブジェクトのインタフェースと同一のインタフェースを持つため，クライアントからは透過的に扱える．これらの機構では，QoS に関するアスペクトをプログラマ自身が記述することを前提としている．脱着型サービス間通信機構でも，サービスプログラマ自身がプロトコルモジュールを実装することは可能である．しかし上記の機構では，サービスの機能的な側面に加えて非機能的な側面までをも，プログラマが必ず実装する必要がある．したがって，脱着型サービス間通信機構における通信端点ごとのモジュール選択によるアスペクト指定は，プログラマに対してより柔軟かつ単純な方式を用意しているという点で，ホームユビキタスコンピューティングを目的としたサービス開発をより促進すると考えられる．

通信アスペクトの指定および新たなプロトコルモジュールの実装に関しては，プロトコル記述言語の採用を検討している．Gryphone プロジェクト[1),24)] は Distributed Interaction Language（DIL）と呼ばれるプロトコル記述言語を提案している．サービス記述とは別にプロトコル動作を DIL を用いて定義し，プロトコルスタックの動的生成を実現している．また Ensemble プロジェクト[2)] では，マイクロプロトコルと呼ばれるオブジェクトを組み合わせてプロトコルスタックを構築する機構を提案している．本論文で提案

した脱着型サービス間通信ミドルウェアは，これらの機構において動的に生成されるプロトコルスタックを，所与のモジュールとして提供している．プログラマが既存のプロトコルモジュールを複合的に使用したり，まったく新しいプロトコルモジュールを実装する際には，上述したプロトコル記述言語等の提供によりそれらの作業を円滑化できると考えられる．

7. 結　　論

本論文では，既存のミドルウェアが持つアスペクト実現妨害問題を提示し，それらを解決するミドルウェアとして，VNA アーキテクチャ中の脱着型サービス間通信ミドルウェアを提案した．

既存のミドルウェアでは，サービス間通信に使用するプロトコルをプログラマが選択することはできない．したがってプログラマは，実装するサービスに適したプロトコルを採用したミドルウェアを選択する必要がある．結果として，ホームネットワーク上には一様なサービスを提供する多様なミドルウェアが存在し，ユーザはそれらすべてを扱う必要がある．

VNA アーキテクチャでは，サービス記述方式とサービス間通信ミドルウェアとを分離し，後者で用いるプロトコルや付随する通信アスペクトをプログラマがサービスの通信端点ごとに指定可能とした．イベント通信，バルクデータ通信，およびストリームデータ通信を行える複数のプロトコルを，VNA ランタイムに直接脱着できるモジュールとして提供した．したがって VNA アーキテクチャ上には，多様なサービスを少ないプログラミングコストで実現できる．またユーザは，VNA アーキテクチャを介して，多様な情報家電機器を統合利用できる．

今後の課題として，より多様な通信アスペクトのモジュール化が必要である．現在，通信路の暗号化，通信帯域予約，およびソフトリアルタイム通信のモジュール化を進めている．また，複数のモジュールを同時に指定して，異なるアスペクトを協調的に動作させるための，モジュールの階層化に関する研究も同時に進めている．

参 考 文 献

1) Astley, M., Sturlman, D.C. and Agha, G.A.: Customizable Middleware for Modular Distributed Software, *Comm. ACM*, Vol.44, No.5 (2001).
2) Birman, K., Constable, B., Hayden, M., Hickey, J., Kreitz, C., van Renesse, R., Rodeh, O. and Vogels, W.: The Horus and Ensemble Projects, *Proc.DARPA Information Survivability Conference and Exposition* (*DISCEX00*) (2000).
3) Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S. and Winer, D.: Simple Object Access Protocol (SOAP) 1.1, World Wide Web Consortium Technical Report (2000).
4) Carzaniga, A., Rosenblum, D.S. and Wolf, A.L.: Challenges for Distributed Event Services: Scalability vs. Expressiveness, *Proc. International Workshop on Engineering Distributed Objects '99* (*EDO'99*) (1999).
5) Carzaniga, A., Rosenblum, D.S. and Wolf, A.L.: Design and Evaluation of a Wide-Area Event Notification Service, *ACM Trans. Comput. Syst.*, Vol.19, No.3, pp.332–383 (2001).
6) Gribble, S.D., Brewer, E.A., Hellerstein, J.M. and Culler, D.: Scalable, Distributed Data Structures for Internet Service Construction, *Proc. Fourth Symposium on Operating Systems Design and Implementation* (*OSDI2000*) (2000).
7) Gribble, S.D., Welsh, M., Brewer, E.A. and Culler, D.: The MultiSpace: An Evolutionary Platform for Infrastructural Services, *Proc. 1999 Usenix Annual Technical Conference* (1999).
8) Hauck, F.J., Becker, U., Geier, M., Meier, E., Rastofer, U. and Steckermeier, M.: AspectIX: A quality-aware, object-based middleware architecture, *Proc. 3rd IFIP International Conference on Distributed Applications and Interoperable Systems* (*DAIS*) (2001).
9) Hodes, T.D. and Kats, R.H.: A Document-based Framework for Internet Application Control, *Proc. Second USENIX Symposium on Internet Technologies and Systems* (*USITS '99*), pp.59–70 (1999).
10) Huang, A.C., Ling, B.C., Barton, J. and Fox, A.: Making Computers Disapper: Appliance Data Services, *Proc. International Conference on Mobile Computing and Networking* (*Mobicom*), pp.108–121 (2001).
11) Koster, R. and Kramp, T.: Structuring QoS-supporting services with smart proxies, *Proc. IFIP/ACM Middleware Conference* (*Middleware 2000*) (2000).
12) Microsoft Corp.: Universal Plug and Play Device Architecture Reference Specification (1999).
13) 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).
14) Nakazawa, J., Tobe, Y. and Tokuda, H.: On Dynamic Service Integration in VNA Architecture, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol.7, No.E84-A, pp.1610–1623 (2001).
15) Nakazawa, J., Tobe, Y. and Tokuda, H.: A Pluggable Service-to-Service Communication Mechanism for VNA Architecture, *Proc. 22nd International Conference on Distributed Computing Systems* (*IEEE ICDCS2002*) (2002).
16) Nakazawa, J. and Tokuda, H.: A Pluggable Service-to-Service Communication Mechanism for Home Multimedia Networks, *Proc. ACM Multimedia 2002*, pp.621–630 (2002).
17) Okoshi, T.: Smart Space Laboratoty Project: Toward the Next Generation Computing Environment, *IWNA2001* (2001).
18) OMG: CORBA Component Model, OMG Technical Document no orbos/99-07-01.
19) O'Ryan, C., Kuhns, F., Schmidt, D.C., Othman, O. and Parsons, J.: The Design and Performance of a Pluggable Protocols Framework for Real-time Distributed Object Computing Middleware, *Proc. IFIP/ACM International Conference on Distributed Systems Platforms* (2000).
20) Schantz, R., Loyall, J., Atighetchi, M. and Pal, P.: Packaging Quality of Service Control Behaviors for Reuse, *Proc. 5th IEEE International Symposium on Object Oriented Real-time Districuted Computing* (*ISORC 2002*) (2000).
21) Schmidt, D.C., Levine, D.L. and Mugnee, S.: The Desing of the TAO Real-Time Object Request Broker, *Computer Communications*, Vol.21, No.4 (1998).
22) Schulzrinne, H., Rao, A. and Lanphier, R.: Real Time Streaming Protocol (RTSP), RFC2326 (1998).
23) Sony, Matsushita, Philips, Thomson, Hitachi, Toshiba, Sharp and Grundig: Specification of the Home Audio/Video Interoperability (HAVi) Architecture (1998). http://www.havi.org/home.html
24) Sturman, D., Banavar, G. and Strom, R.: Reflection in the Gryphon Message Broking System, *Proc. Reflection Workshop at Object-Oriented Programming Languages and Applications* (1998).
25) Sun Microsystemds: Java Media Framework Specification, Version 1.0 (1997). http://java.sun.com/products/java-media/jmf/1.0/apidocs/spec-license.html
26) Sun Microsystems, Inc.: Jini Architecture Specification (1998). http://www.javasoft.com/products/jini/specs/jini-spec.pdf
27) Sun Microsystems Inc.: Enterprise Java Beans (TM) Specification, Version 2.0 (2001)
28) Universal Plug and Play Forum: Universal Plug and Play (UPnP) (1999). http://www.upnp.org
29) Vanegas, R., Zinky, J., Loyall, J.P., Karr, D., Schantz, R.E. and Bakken, D.E.: QuO's Runtime Support for Quality of Service in Distributed Objects, *Proc. IFIP/ACM Middleware Conference* (*Middleware 98*) (1998).
30) World Wide Web Consortium: Extensible Markup Language (XML) 1.0 (1999).
31) 大越　匡，中澤　仁，田村陽介，望月祐洋，戸辺義人，西尾信彦，徳田英幸：VNA：仮想情報家電の実現へ向けて，第 59 回情報処理学会全国大会 (1999).
32) 中澤　仁, 大越　匡, 望月祐洋, 徳田英幸：VNA 構築用ライブラリの設計と実装，情報処理学会全国大会 (1999).

(平成 14 年 12 月 21 日受付)
(平成 15 年 4 月 16 日採録)

中澤　仁

2003 年慶應義塾大学大学院政策・メディア研究科博士課程修了．同年 4 月より日本学術振興会特別研究員．ホームネットワークミドルウェア，ユビキタスコンピューティング，分散オブジェクト指向システム，オブジェクト移送ミドルウェア等の研究に従事．博士（政策・メディア）．

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，慶應義塾大学環境情報学部教授，同大学大学院政策・メディア研究科委員長．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．