社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－MBL－40 (11)
2007－UBI－13 (11)
2007／2／22

# 適応的変換機構を用いた異種サービス連携の実現

井上 貴仁 † 服部 篤人 † 田中 宏一 ‡ 河口 信夫 § 西尾 信彦 ‖

**概要**

将来，多数のセンサや情報機器，各種ネットワークによって多種多様なスマート環境が遍在すると考えられる．このようなユビキタス社会では，異種のスマート環境が相互に接続し，サービスの連携が可能となることが望ましい．しかし，異種のスマート環境間を相互接続可能にするにはプロトコルや通信の差異など，多くの問題がある．本論文では，異種のサービスアーキテクチャ間の連携に必要なプロトコル変換・連携手法について述べる．また，異種サービスの変換機構の適応的な連携についてのシステムを実装・評価する．

**キーワード**：スマート環境，サービス連携，サービスアーキテクチャ，ヘテロジニアス

# A Translation Mechanism of Diversed Protocols Utilizing Adaptive Algorithm Selection

**Atsuhito Inoue † Atsuhito Hattori † Koichi Tanaka ‡ Nobuo Kawaguchi § Nobuhiko Nishio ‖**

ABSTRACT

In the future, there will be many kinds of Smart Environments consisting of many kinds of sensors, electronic equipment and networks. In such a ubiquitous society, it is desirable to interconnect Smart Environments and incorporate services, despite the differences between Smart Enviroments. However, there are a number of problems with connecting different Smart Environments to each other and enabling cooperateing services. In this paper, we describe a protcol translation for adapting Heterogeneous service architectures and cooperating Smart Environments to realize the system.

## 1. はじめに

将来，多数のセンサや情報機器，各種ネットワークによって多種多様なスマート環境が遍在すると考えられる．これらの異種のスマート環境を相互に接続し，サービスを連携させるためには，多くの問題がある．例えば，現在利用されるサービスアーキテクチャとして，Bonjour，DLNA，UPnP，Jini，United Spaces [1]，cogma [2] [3]，などが存在するが，これらのサービスアーキテクチャによって実装されたサービスは相互連携することができない．異なるサービスアーキテクチャ同士では，サービスの発見・実行のプロトコルが異なる上，サービス記述形式や利用するデータフォーマットも異なる．これらの問題を解決するために，我々は，異種のスマート環境の差異を吸収し，セキュアな相互接続・サービス連携を可能にする，Secure Semantic Tunneling モデル [4] を提案した．また，Secure Semantic Tunneling モデルを，多種多様な異種のスマート環境に対してスケーラブルにするための設計として，Semantic P2P Network を提案した．

本研究では，Semantic P2P Network を構成する Semantic Peer に実装される，異種のサービスアーキテクチャにおけるサービス連携機構と，これらの異種サービスを連携させる機構を適応的に連携させるための機構を実現させるシステムを実装評価する．

本稿では，まず，2 章で研究背景を述べ，本研究の

† 立命館大学理工学部
Department of Science and Engineering, Ritsumeikan University
‡ 株式会社内田洋行
UCHIDA YOKO COMPANY LTD.
§ 名古屋大学大学院工学研究科/情報連携基盤センター
Graduate School of Engineering, Nagoya University
‖ 立命館大学情報理工学部
Department of Computer Science, Ritsumeikan University

本研究は総務省戦略的情報通信研究開発推進制度 (SCOPE)「異種スマート環境間をセキュアに動的接続・構成する基盤技術」の支援を受けて実施されている．

位置付けを明確にする．3 章では，異種のサービス連携を実現する関連研究との比較を述べる．4 章で異種サービスアーキテクチャ連携させるための変換項目について分析し，異種サービスアーキテクチャの連携モデルを提案する．また，このような異種サービス連携機構を適応的に連携させるための設計について述べる．5 章では，異種サービスアーキテクチャのディレクトリサービスを連携させるモジュールの実装について述べる．また，異種プロトコルを利用する音声通話翻訳モジュールを用いた，適応的異種サービス連携機能の実装について述べる．6 章では，実装したシステムを定量評価する．最後に 7 章で，本論文をまとめ，今後の課題を述べる．

## 2. 研究背景

本章では，Secure Semantic Tunneling の概要について述べる．また，Secure Semantic Tunneling モデルをスケーラブルにするための設計である Semantic P2P Network の設計の概要について述べる．最後に，本研究での位置付けを述べる．

### 2.1 Secure Semantic Tunneling

Secure Semantic Tunneling は，異種スマート環境を接続するトンネルを構築する (tunneling)．このトンネルは，スマート環境間の通信・サービスの差異 (IPv4/v6，NAT，サービスアーキテクチャなど) を適応的に判断して，容易につなぐことを可能にする (semantic)．また，許可されたユーザ・スマート環境・サービスだけが通ることができる，という安全性を持つ (secure)．このトンネルを通すことで，異種のスマート環境間を越えて，自由かつセキュアにサービスを連携させることができる．

### 2.2 Semantic P2P Network

異種スマート環境を仲介する翻訳を行うレイヤは多岐にわたる，またそれぞれのレイヤにおいてもプロトコルの違いがある．そのため，エンドポイントにある仲介者だけに，上記の翻訳機能を持たせることは困難である．そこで，仲介者の機能を分散化した P2P ネットワークを構成し，P2P ネットワーク全体で仲介機能を実現する．スマート環境を構成するピアを Smart Peer と呼び，単一のサービスアーキテクチャに対応している．また，Semantic P2P Network を構成するピアを Semantic Peer と呼ぶ．Semantic Peer は以下の 4 つの機能を持つ．

1. 異種スマート環境間の通信の差異を吸収する，外部ネットワークとの接続
2. スマート環境内部のユーザやサービスを検索する，スマート環境のオブジェクト検索
3. スマート環境で利用されるサービスアーキテクチャのプロトコルの違いを翻訳する，異種サービスアーキテクチャの連携
4. 異種スマート環境を動的かつ効率的に連携させる，適応的な異種スマート環境の連携

本稿では，Semantic Peer に実装される 3，4 の機構についての設計，及び実装方法について述べる．

## 3. 関連研究

本章では，異種サービスを連携させるためのプロトコル変換を実現したシステムを挙げ，本研究と比較する．

**uMiddle** 異種のサービスアーキテクチャを連携可能にするミドルウェアとして，uMiddle [5] がある．uMiddle は，不特定多数の異種サービスアーキテクチャ間のサービス連携を可能とする，ミドルウェアフレームワークである．uMiddle は，既存のサービスアーキテクチャを利用する機器やサービスに変更を加えず，また任意の数の仲介者をネットワーク内に配置して，サービス連携を柔軟に実現している．

本研究では，遠隔地にある異種のスマート環境で利用されているサービスアーキテクチャを，単純なプロトコル変換で連携させる仲介者を実現する．この仲介者を，Semantic P2P Network で分散化することで，スケーラブル・冗長性といった特徴を持たせたサービス連携が可能になる．このように，本研究は uMiddle とは遠隔のスマート環境連携にフォーカスしているという点で異なる．

**PSGw** 異種プロトコルの相互接続アプリケーションとして RSDev.com [6] の PSGw-Personal Skype to SIP and H.323 gateway (以下 PSGw) が挙げられる．このアプリケーションは，Skype [7] の使用するプロトコルと SIP または H.323 プロトコルを相互に接続するゲートウェイとして機能し，異なるプロトコルで動作する音声通話サービスの連携を可能にする．

本研究では，このようなプロトコル変換機能をピアの機能とすることで，より柔軟な異種サービスの相互接続を実現することを検討する．

## 4. 適応的な異種サービス変換機構の設計

本章では，異種のサービスアーキテクチャを連携させるための連携方式，および変換機構の適応的な連携手法について述べる．ここでは，DLNA と Bonjour といった，同じ目的で異なるサービスアーキテクチャで実装されたサービスについて段階的に検討する．このような異なるサービスアーキテクチャで実装されたサービスを異種サービスと呼ぶ．異種サービスを連携させるために必要となる様々な差異を変換する機構と，適応的な変換機構の連携について設計を行った．

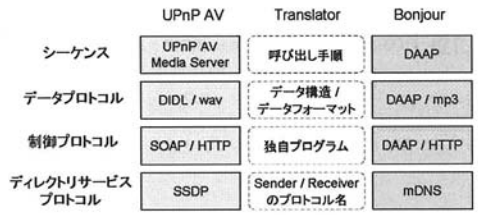


図 1　異種サービスアーキテクチャの連携方式

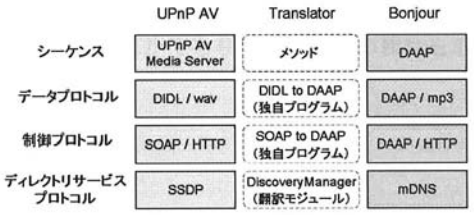


図 2　異種サービスアーキテクチャの連携モデル

### 4.1　異種サービスアーキテクチャの連携

本節では，異種のサービスアーキテクチャを連携させるにあたって必要となるプロトコル変換，最適な連携方式について述べる．異種サービスアーキテクチャにおける異種サービスのプロトコルの調査を行い，翻訳に関する項目を明確化して，異種サービスアーキテクチャ連携の設計モデルを検討する．ここで，プロトコルの違いを翻訳するモジュールを，単に**翻訳モジュール**と呼ぶ．

#### 4.1.1　異種サービスアーキテクチャ連携方式

我々は，異種のサービスアーキテクチャを連携するにあたって，大別して，コントロール用とデータ用のプロトコルを翻訳する必要があると考えている．これらのサービスアーキテクチャにおけるプロトコル群は，定型的であり，機械的な変換を行うことで，連携が可能である．異種のサービスアーキテクチャを連携させる場合，仲介する機能としてトランスレータ方式とプロキシ方式の 2 つが考えられ，機械的な変換ができる場合，機械的な翻訳機能を持つ仲介者をおく，トランスレータ方式の翻訳が適していると考えられる．

また，プロトコルの差異を翻訳する以外に，制御プロトコルの処理手順である，シーケンスの差異を翻訳する必要があると考えている．

これらのプロトコル変換や，シーケンス変換を考慮したトランスレータモデルを図 1 に示す．異種のサービスアーキテクチャを連携させるために，変換が必要なモジュール毎にプロファイルを持たせて，翻訳を行う方式を考える．

#### 4.1.2　異種サービスアーキテクチャの連携モデル

本稿では，異種サービスアーキテクチャ連携させるプロトコル変換の調査の一つとして UPnP-AV の音楽サービスと Bonjour の音楽サービスの例を用いて，異種サービスアーキテクチャ連携の設計モデルを検討した．設計したモデルの概要を図 2 に示す．

異種サービスを連携させるために，前節で述べたトランスレータ方式のプロトコル変換機能を持つ仲介者を考える．この場合，仲介者には，以下の 4 つのプロトコル変換を行う翻訳モジュールが必要である．

1. サービス実行に利用される，制御プロトコルの翻訳
2. サービス管理に利用される，ディレクトリサービスプロトコルの翻訳
3. 制御プロトコルを介して送受信される，データ構造プロトコルの翻訳
4. 制御プロトコルを介して送受信される，データフォーマットプロトコルの翻訳

**■ディレクトリサービスプロトコル翻訳**　ディレクトリサービスのプロトコル翻訳は，あるサービスアーキテクチャのサービスが，別のサービスアーキテクチャのサービスとの連携するために，それぞれのサービスアーキテクチャに互いのサービスの存在を認識させる．例えば，UPnP から Bonjour への変換では，SSDP を利用して発見可能な UPnP の音楽サービスを，mDNS を利用して Bonjour の音楽サービスとして発見可能にする．この時，トランスレータは，SSDP によって UPnP-AV の MediaServer を発見すると，mDNS によって DAAP サービスとして iTunes に通知する．

異種サービスアーキテクチャのディレクトリサービスを連携させるための汎用的なディレクトリサービス翻訳モジュールに，ディレクトリサービスの差異を記述した，プロファイルを与えることで，ディレクトリサービスが連携可能になると考えている．

**■制御プロトコルの翻訳**　制御プロトコルの翻訳は，サービスアーキテクチャ毎に利用している固有の制御プロトコルを利用するサービスの差異を解決する．例えば，UPnP から Bonjour へのでは，SOAP を利用して UPnP の音楽データの問い合わせを行うサービスを，DAAP を利用して Bonjour の音楽データの問い合わせを行うサービスに翻訳する．このような目的が似たサービスを変換する特定のプログラムをプロファイルに記述することで制御プロトコルの差異を解決する．

**■データ構造プロトコルの翻訳**　データ構造プロトコルの翻訳，制御プロトコルを介して送受信されるデータのデータ構造を変換する．データ構造は，共通化されていない場合もあり，機械的な変換を行う特定のプログラムが必要である．例えば，UPnP 音楽サービスから Bonjour 音楽サービスへの変換では，SOAP を介

して送受信される音楽情報データを DAAP を介して送受信される音楽情報データのデータ構造に変換する必要がある．この時，SOAP を介して送受信される音楽データのデータ構造である DIDL を，DAAP を介して送受信される音楽データである DAAP に依存したデータ構造に変換する特定のプログラムが必要となる．これらのデータ構造プロトコルを変換する特定のプログラムをプロファイルに記述することでデータ構造プロトコルの差異を解決する．

**■データフォーマットプロトコルの翻訳** データのフォーマットは，異種の音楽サービスの間で，扱う音楽データのデータフォーマットが異なる場合，それぞれの音楽サービスが扱うマルチメディアデータのデータフォーマットを変換する．例えば，UPnP の音楽データが wav で，DAAP の音楽データが mp3 である場合，media-convert [8] のようなプログラムを利用することで，データフォーマットの変換が可能である．音楽データのようなマルチメディアデータは，標準規格を利用することが多く，汎用的なマルチメディアデータ変換プログラムを利用することで，変換が可能となる．これらのデータプロトコルを変換するプログラムの指定をプロファイルに記述することでデータプロトコルの差異を解決する．

### 4.2 適応的翻訳モジュール制御機構

Semantic P2P Network には，Semantic Peer の持つプロトコルの違いを翻訳する機能がモジュール化され配置される．異種サービスを Semantic P2P Network を介して連携するには，Semantic P2P Network 内に分散された翻訳モジュールの中から目的に合った翻訳モジュールを選定し，仲介してもらう必要がある．また，異種サービス間では，Semantic P2P Network 全体で仲介が行われているように見え，どの翻訳モジュールが仲介しているのか意識することなく連携することが望ましい．本節ではそのような連携機構の設計を行い，翻訳モジュールの機能記述，翻訳モジュールの選定機能，翻訳モジュールの制御機能に関して述べる．

#### 4.2.1 翻訳モジュールの機能記述

Semantic Peer は，複数の種類の翻訳モジュールを持っている可能性がある．Semantic P2P Network 上でそれらの翻訳モジュールが機能するためには，翻訳モジュールの情報を，Semantic P2P Network に周知させる必要がある．翻訳モジュールが Semantic P2P Network に対して周知させるべき情報 (プロパティ) として，以下のようなものが挙げられる．

1. 変換できるプロトコルリスト
2. プロトコルに対応したサービス名
3. サービス名に対応したアクセス方法

個々の翻訳モジュールはこれらの項目を記述したプロパティを持つ．

変換できるプロトコルリストに記述する項目としては，Bonjour と UPnP AV の変換が可能な翻訳モジュールならば，Bonjour，UPnP AV という 2 つのプロトコル名を記述する．

プロトコルの中で翻訳できるサービスを明示するために，プロトコルリストのプロトコル名 1 つ 1 つに対して，音楽サービス，TV 会議サービス，などのサービス名を記述する．

また，翻訳モジュールは，接続元サービスと翻訳モジュール間では接続元と同じプロトコルで接続し，接続先サービスと翻訳モジュール間は接続先と同じプロトコルで接続している．そのため，プロトコルに対応したサービス 1 つ 1 つによってアクセス方法が異なる．このことから，アクセス方法をサービス名 1 つ 1 つに対して記述する必要がある．例えば，Skype の音声通話サービスでは，Skype アカウント，H.323 の音声通話サービスでは，IP アドレスといったアクセス方法を記述する．

#### 4.2.2 翻訳モジュールの選定機能

異種サービス間を連携させるためには，必要な翻訳モジュールを検索し，仲介する翻訳モジュールを決定しなければならない．Semantic P2P Network では，各翻訳モジュールのプロパティが共有され，各 Semantic Peer は Semantic P2P Network に配置された翻訳モジュールのプロパティを知ることができる．また，Semantic Peer は翻訳モジュールの共有レベルを設定でき，セキュリティ面を考慮し，特定の信頼できる Semantic Peer の翻訳モジュールのプロパティのみを共有する構成をとることも可能になる．

Smart Peer は，Semantic P2P Netowork には属しておらず，仲介する翻訳モジュールの検索を行うことができない．そこで，Smart Peer は，ローカルネットワーク内に存在する Semantic Peer を探し出し，連携させたい異種サービス間の仲介を行う翻訳モジュールを検索依頼要求を出す．ここでは，Smart Peer から要求を受け取った Semantic Peer を他の Semantic Peer と区別するため，**End-Point-Peer** と呼ぶ．

End-Point-Peer は Semantic P2P Network 内で共有された翻訳モジュールのプロパティを元に要求を満たす翻訳モジュールを検索する．そして，発見した翻訳モジュールの中から，仲介させる翻訳モジュールを決定する．

#### 4.2.3 翻訳モジュールの制御機能

翻訳モジュールが仲介者として機能し，異種サービス間を連携させるためには，異種サービス間が翻訳モジュールを経由する設定に変更する必要がある．翻訳モジュール制御機構を図 3 に示す．End-Point-Peer は

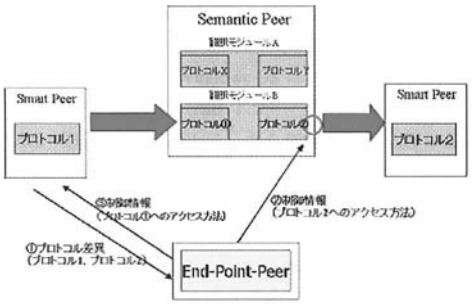


図 3　翻訳モジュール制御機構

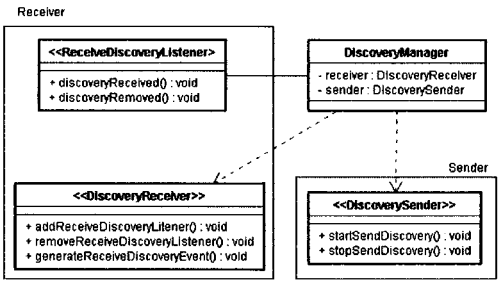


図 4　DiscoveryManager

図 3 の 1 で受け取った要求に基づいて，翻訳モジュールを選定する．選定した翻訳モジュールを経由するために End-Point-Peer は翻訳モジュールに制御情報を送り，Smart Peer には，選定した翻訳モジュールに接続するように制御情報を送る．つまり，図 3 では，異種サービスの接続元 (プロトコル 1) は，翻訳モジュールに接続先を設定し，翻訳モジュールはその接続を受けて異種サービスの接続先 (プロトコル 2) に接続するように設定を変更する制御情報を送信する．翻訳モジュールは，受信した制御情報を元に，接続先を変更することで，特定の異種サービス間を仲介する仲介者として機能する．

Semantic Peer はこのような制御を，選定した翻訳モジュールすべてに対して行い，異種サービス間が選定した翻訳モジュールを経由することで、異種サービス間のプロトコル変換を行い，連携を可能にする．

## 5.　実装

本章では，異種サービスアーキテクチャ連携モデルの検証の 1 つとして，ディレクトリサービス連携モジュールを実装した．また，異種サービス連携の適応的連携機構を実装した．本稿では，適応的連携機構が制御する異種サービス連携機構として，PSGw を使用した．

### 5.1　ディレクトリサービス連携モジュールの実装

本節では，異種のサービスアーキテクチャのディレクトリサービスを連携させる，汎用的なディレクトリサービス翻訳モジュールについて設計・実装を行った．本設計では，以下の 3 つの機構で構成する．

1. 翻訳されるサービスアーキテクチャの受動的なサービス通知する機能を持つ，Receiver
2. 翻訳されたサービスアーキテクチャの能動的なサービス通知する機能を持つ，Sender
3. Sender と Receiver を管理して連携させる機能を持つ，DiscoveryManager

ディレクトリサービス翻訳モジュールの設計を図 4 に示す．

Receiver では，翻訳されるサービスアーキテクチャのサービス通知を受け取りを ReceiveDiscoveryListener によって実現する．Sender では，翻訳されたサービスアーキテクチャの能動的なサービス通知を，startSendDiscovery によって行う．DiscoveryManager は，サービス通知を受け付ける Receiver の機能と，それを翻訳し，別のサービスアーキテクチャのサービス通知として送信する，Sender の機能を連携させる．この連携をさせるためには，サービスアーキテクチャの翻訳項目をプロファイルに記述する．Sender(DiscoverySender) と Receiver(DiscoveryReceiver) を実装したクラスをそれぞれ指定する．また，Receiver の discveryReceived, discveryRemoved に用いられる値と，Sender の startSendDiscovery, startSendDiscovery に用いられる値を記述する．記述した項目の差異を Receiver と Sender の実装クラスによって翻訳し，ディレクトリサービスの連携が可能になる．

このディレクトリサービスの連携モジュールを用いて，UPnP-AV と Bonjour のディレクトリサービスの連携を実現させた．図 5 にシステムの構成を示す．Receiver の機能を実現する ReceiveUPnPAVDiscoveryImpl を実装した．また，Sender の機能を実現する SendBonjourDiscoveryImpl を実装した．ReceiveUPnPAVDiscoveryImpl は，UPnP-AV MediaServer の発見の通知を，generateReceiveDiscovery を用いてイベントとして DiscoveryManager に通知する．DiscoveryManager は，discoveryReceived を用いてイベントを受け取ると SendBonjourDiscoveryImpl の startDiscovery を用いて Bonjour の DAAP サービスとして通知する．これらの連携により，iTunes が UPnP-AV Media Server を発見することが実現できた．

このようなディレクトリサービス連携を，DiscoveryManager にプロファイルを与えることで，様々なサービスアーキテクチャに対して行う．段階的な検証の一つとして，本稿で実装した UPnP-AV と Bonjour の

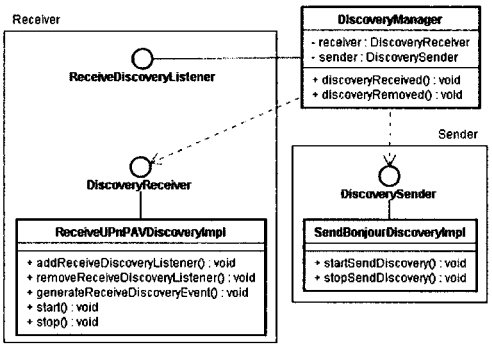


図 5　システム構成図

```
<?xml version="1.0" encoding="utf-8"?>
<properties>
  <common>
    <input>DiXiM Media Server</input>
  </common>
  <receiver>
    <implClass>
      jp.ac.ritsumei.ubi.translator.discovery.ReceiveUPnPAVDiscoveryImpl
    </implClass>
    <input>object.item.audioItem.musicTrack</input>
  </receiver>
  <sender>
    <implClass>
      jp.ac.ritsumei.ubi.translator.discovery.SendBonjourDiscoveryImpl
    </implClass>
    <input>_daap._tcp</input>
  </sender>
</properties>
```

図 6　プロファイル

ディレクトリサービスの連携では，図 6 のようなプロファイルを与えることで，連携を実現した．receiver の implClass に Receiver を実装したクラス名を指定し，input には generateReceiveDiscoveryEvent の引数の値を記述する．同様に，sender の implClass に Sender を実装したクラス名を指定し，input には startSendDiscovery の引数の値を記述する．また，これらに共通の引数の値を common の input に記述する．

今回の実装では，音楽サービスを連携させるために，receiver の input には，UPnP-AV の ContentObject で音楽ファイルを示す object.item.audioItem.musicTrack を指定する．sender の input には Bonjour の音楽サービスのサービスタイプである _daap._tcp を指定した．このように，サービスを通知する引数の差異をプロファイルに記述することで，ディレクトリサービスの連携を実現した．

今後は，様々な異種サービスアーキテクチャのディレクトリサービスを実装し，プロファイルの記述に必要な項目を明確にすることで，汎用的な連携に最適なプロファイルの記述を検討していく．

### 5.2　適応的翻訳モジュール連携機構

本節では，異種サービス連携として異種プロトコルの音声通話サービスを接続する機構を実装する．音声通話サービスのプロトコルとして，Skype, H.323, SIP を用い，PSGw を制御するプログラムを組み込み音声通話サービス連携モジュールとして用いる．

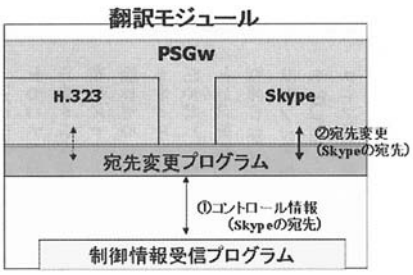


図 7　翻訳モジュールの実装

音声通話サービスを実行できる PC を 2 台用意し，1 台には Skype，もう 1 台には H.323 プロトコルを利用できる Polycom PVX を導入する．音声通話サービスの連携を可能にする PSGw に機能を追加し、翻訳モジュールとして動作するようにした．Skype と H.323 の翻訳モジュールの内部構成図を図 7 に示す．翻訳モジュールは，制御情報を受け取り，接続先を動的に変更することができなければ仲介者として機能しない．PSGw にはそのような機能がないため，制御受信プログラムと宛先変更プログラムを追加した．Skype と H.323,Skype と SIP の音声通話サービス翻訳モジュールを各 1 台ずつ用意する．翻訳モジュールの選定を行い，翻訳モジュールに対して制御を行う機能を持つ End-Point-Peer を 1 台用意する．また，今回の実装では，翻訳モジュールの数が少なく，制御機能に関する検証を中心に行うため，Semantic P2P Network を構築するということは行わない．それに代わり，翻訳モジュールのプロパティを管理するデータベースを用意し，データベースの情報を翻訳モジュールの検索に用いる．End-Point-Peer は，データベースの中を検索し，翻訳モジュールを選定する．システム構成図を図 8 に示す．

## 6.　評価

本章では，基本的な変換性能を評価する目的で，ディレクトリサービス連携モジュールの変換時間を測定した．また，PSGw を翻訳モジュールに用いた時の音声の遅延と，そのときの制御に要する時間の測定を行った．

### 6.1　ディレクトリサービス連携モジュール

異種のサービスアーキテクチャにおける，ディレクトリサービス連携モジュールの変換時間を測定した．本項では，UPnP の SSDP と Bonjour の mDNS を例として，ディレクトリサービス連携モジュールの変

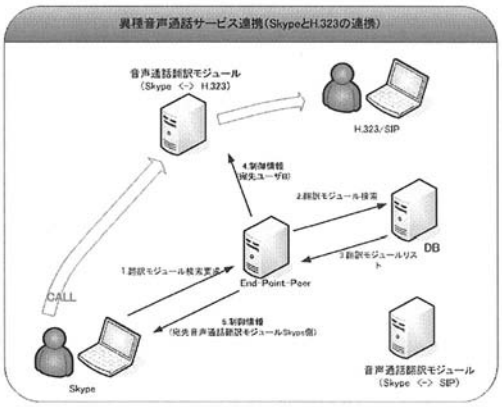


図 8　システム構成図

表 1　ディレクトリサービスの変換時間

| 変換プロトコル | 平均変換時間 (ms) |
|---|---|
| SSDP/mDNS | 10 |
| SSDP・SOAP/mDNS | 10343 |

換時間を測定した．測定結果を，表 1 に示す．

SSDP/mDNS では，UPnP の SSDP によって，ディレクトリサービス連携モジュールが単純に翻訳し，Bonjour の mDNS によってサービス通知を送信するまでの時刻を測定した．

SSDP・SOAP/mDNS では，SSDP/mDNS で述べた変換時間に加え，翻訳する際に UPnP の SOAP サービスで UPnP-AV の MediaSever にある全ての音楽データを問い合わせて取得する時刻が追加される．

この時，UPnP-AV MediaServer に含まれていた音楽データは 29 個であった．

今後，音楽データの取得する時間に関して，改善の必要がある．

### 6.2　翻訳モジュール選定機能を用いた音声サービスの連携

音声通話サービスの適応的な連携に関して評価を行う．まず，通常の音声通話サービスのに関して測定を行った．ここでは，Skype を用い，ローカルネットワーク内で，Skype 同士を接続する場合の，音声の遅延時間を測定した．ローカルネットワーク内では，Skype 同士による音声通話の遅延はほとんどなく，0.1 秒以下の遅延に過ぎなかった．次に，本稿で実装した異種音声サービスに関して測定を行った．通常の音声サービスとの比較を行うために翻訳モジュール，Skype，Polycom PVX(H.323/SIP) はすべてローカルネットワーク内に配置し，以下のパターンに関して音声通話の遅延を測定した．

表 2　Skype と H.323 の音声遅延時間

| 接続タイプ | 仲介数 | 平均遅延時間 (ms) |
|---|---|---|
| Skype/H.323 | 1 | 610 |
| Skype/SIP | 1 | 670 |
| H.323/SIP | 2 | 1030 |

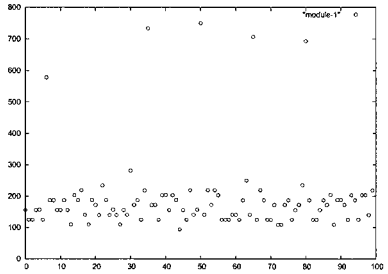


図 9　翻訳モジュール 1 つに対する制御時間
縦軸 [ms] / 横軸 [回数]

1. Skype と H.323 の接続（翻訳モジュール 1 つ経由）
2. Skype と SIP の接続（翻訳モジュール 1 つ経由）
3. H.323 と SIP の接続（翻訳モジュール 2 つ経由）

表 2 に結果を示す。

音声の遅延はネットワーク環境，翻訳モジュールの性能によって左右される．異種音声通話サービスを実現するためには，音声の遅延以外に，接続するまでに要する時間に関しても評価する必要がある．

本稿の翻訳モジュール制御機構の実装では，Semantic P2P Network を用いず，データベースを用いて実装しているので，翻訳モジュールの検索時間は考慮しないことにする．今回は，選定された翻訳モジュールを制御するのにどれほどの時間がかかっているのか測定し評価する．翻訳モジュールに対して制御情報を送信するところから，制御完了メッセージを受信するまでを制御にかかる時間とする．翻訳モジュール 1 つに対して制御を行った結果を図 9 に示すし，翻訳モジュール 2 つに対して制御を行った結果を図 10 に示す．

制御時間に多少はばらつきがあるが，ほとんどの場合，翻訳モジュール 1 つを制御する際は 200ms，2 つ制御する際は 350ms かかる．この程度の制御時間ならば，特に気になることなく利用できる．

## 7.　まとめと今後

本稿では，異種スマート環境の連携を行うために必要な Semantic Peer の機能を述べ，その中でも，異種サービスアーキテクチャの連携させる変換機構と，適応的な変換機構の連携について詳しく述べた．

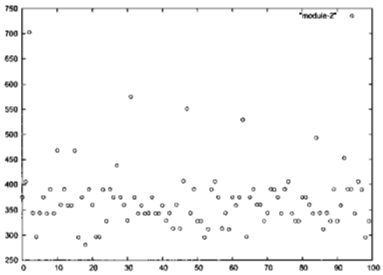


図 10　翻訳モジュール 2 つに対する制御時間
縦軸 [ms] / 横軸 [回数]

我々は，本稿で提案した異種サービス変換機構と適応的連携機構を用いて，異種サービスを適応的に連携できると考えている．

今後，異種サービスアーキテクチャを連携させる変換機構については，様々なサービスアーキテクチャを通して，段階的な調査をさらに進めて翻訳に必要な問題を明確化していく．また，適応的な連携機構については，P2P アルゴリズムの適応，変換機構の多段構成のサポート，複数の変換機構が存在する場合の適応的な選定アルゴリズム，などを検討していく．

## 参考文献

[1] Yu enokibori, Nobuhiko Nishio, “A Secure Extension for Reading Multi-Institutional Ubiquitous Service System”, UCS2004(2004)

[2] 河口信夫，春原雅志，“WebCodget : Web サーバに移動して動作する組み込み機器向け移動ソフトウェア”，情報処理学会ユビキタスコンピューティング研究会/映像情報メディア学会技術報告，2005-UBI-2(8)(2005)

[3] Nobuo Kawaguchi, “VPcogma : A Light-Weight Cooperative Middleware for Ubiquitous Embedded Devices”, International Workshop on Software Architectures for Self-Oranization with pervasive2005(2005)

[4] 田中宏一，河口信夫，西尾信彦，“Secure Semantic Tunneling による異種スマート環境接続の設計”，情報処理学会ユビキタスコンピューティングシステム研究会報告，2006-UBI-12，pp.13-20，(2006)

[5] Jin Nakazawa, Hideyuki Tokuda, W. Keith Edwards, Umakishore Ramachandran “A Bridging Framework for Universal Interoperability in Pervasive Systems” ICDS'06(2006)

[6] http://www.rsdevs.com/

[7] http://skype.com/

[8] http://media-convert.com/ja/