Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

uMediator: 異種サービス間でのハンドオフ支援機構

中川 直樹[†1] 榊原 寛[†1] 中澤 仁[†1]
高汐 一紀[†1] 徳田 英幸[†1]

本稿では，異種サービス間でのハンドオフをサービスハンドオフ（SH）という新たな研究領域として定義し，SH を実現する SH フレームワークを設計・提案する．そして，SH フレームワークの主となる SH 支援機構と連携機構であるサービス情報管理機構をそれぞれ uMediator ミドルウェア，uMediator サーバとして実装する．さらにサービス情報記述言語である USDL の拡張記述を設計する．また評価として異なるインタフェースを備える動画再生サービス間での SH 実証実験を行い uMediator の有効性を実証する．

uMediator: A System for Handoff between Heterogeneous Services

Naoki Nakagawa,[†1] Hiroshi Sakakibara,[†1]
Jin Nakazawa,[†1] Kazuki Takashio[†1]
and Hideyuki Tokuda[†1]

In this paper, we define handoff between different kinds of services as Service Handoff (SH), and design and propose a framework to establish SH. We implement uMediator middleware for information management, and uMediator server for coordinate management. In addition, we designed extension of Unified Service Description Language (USDL). The validity of uMediator server is proven through a handoff experiment, which handovers a streaming service between different environment.

†1 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

1. はじめに

近年，サービス発見プロトコルとセンサから取得した位置情報等を利用し，環境固有のサービス展開が加速している．環境固有のサービスが増加した情報環境では，異なるサービスでも似た内容のサービス（異種サービス）が利用可能である場合が多数存在する．一例としてナビゲーションサービスが挙げられる．あしナビ[1] や上野まちナビ実験[2]，ロボットや RFID[3] タグを利用したナビゲーションサービス[4] など多くの環境固有のサービスが研究，開発され至る所で展開される．環境固有のナビゲーションサービスが存在しない環境では，ユーザ端末内のアプリケーションや Web 上のナビゲーションサービスを利用することが可能である．

そのため，サービスの一貫性を確保しながら各環境ごとで利用可能なサービスに切替え，移動しながらサービスを利用したいという要求が考えられる．しかし，環境が異なると管理主体やサービス開発者が異なり，共通の切替え用インタフェースを備えていないためサービスを切替えられないという問題が存在する．また，各環境内に複数存在するサービスの中から切替え先となり得る似た内容のサービスの判定ができないため，切替え先を指定できないという問題が存在する．

そこで本研究では，異種サービス間での互換性を確保しサービスの切替えを支援する機構の構築を行う．本機構は情報環境を跨ぐ移動の際，サービスの一貫性は保ちつつ各環境に存在する異種サービスへ切り替えを行う．また，切替え先の異種サービスの発見とサービス間の差異吸収は本機構が行う．

本研究ではまず，第 2 節で研究領域をサービスハンドオフと定義した後に問題点を明らかにする．次に第 3 節で機能要件について述べ，サービスハンドオフフレームワークを設計する．さらにサービスハンドオフフレームワークの構成要素である USDL[5] 拡張記述と uMediator の設計について述べ，実装環境について述べる．そして第 4 節でサービスハンドオフの実証実験について述べ，第 5 節で関連研究を挙げ本研究との差異を明確化する．最後に第 6 節にて，本稿のまとめと今後の課題について言及する．

2. サービスハンドオフ

2.1 サービスハンドオフの定義

本節では，本研究の研究領域をサービスハンドオフ（SH）として定義する．

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

サービスハンドオフとは，利用中サービスの一貫性を確保しながら“異種サービスへ”ハンドオフすることである．

この際，利用中サービスのことを SH 元サービス，切替え先サービスのことを SH 先サービスと呼ぶ．また，SH 元サービスと SH 先サービスは，共通のプラットフォームや API を使用せず構築されるものとする．つまり，共通の SH インタフェースを備えないサービス間で，サービスの主となる目的を損なわずにサービスを切替えることがサービスハンドオフである．ナビゲーションサービスを例に挙げると，SH 元のナビゲーションサービスから SH 先のナビゲーションサービスへ切替えても，SH 元で設定した目的地へ SH 先のナビゲーションサービスがナビゲーションしてくれるような切替えをサービスハンドオフという．また，SH を可能にする環境を本稿では SH フレームワークと呼ぶ．

2.2 サービスハンドオフの問題点

本研究では情報環境を跨ぐ移動をした際の SH を想定しているため，まず SH 先となるサービスを発見する必要がある．SH 先となるサービスは，ユーザが利用中のサービスから SH してもサービスの一貫性を保つことが可能なサービスである必要がある．しかし現在，UPnP[6] や Bonjour[7]，Jini[8] 等のサービス発見プロトコルは存在するものの，SH 先になり得る異種サービスを判定できるものは存在しない．そのため**異種サービスの発見問題**が発生する．

また，サービス間で SH を行うには SH 元サービスから SH 先サービスへサービスの設定情報等を含んだ状態情報を渡し，SH 先サービスがそれを解釈し，サービスの一貫性が保てるように設定を行う必要がある．しかし本研究が想定する異種サービスは，共通のインタフェースを備えないため独自定義したデータ形式でデータを作成し，独自定義した SH プロトコルでデータを送受信する．ゆえに，異種サービス間でのハンドオフを実現するには，以下の二点の互換性を確保する必要がある．

- 異種サービス間でデータを送受信しあえる互換性
- 異種サービス間でデータを解釈できる互換性

しかし現状，上記二点の互換性を確保する技術は存在しないため**異種サービス間の互換性確保問題**が発生する．

3. 設計と実装

3.1 機能要件

第 2.2 小節で述べた，“異種サービスの発見問題”と “異種サービス間の互換性確保問題”を解決するため，異種サービス発見機能と SH データ変換機能の二つを機能要件として挙げる．以下で各機能の詳細を述べる．

異種サービス発見機能

SH は，サービスを切替えてもサービスとしての一貫性は保たれる必要があり，同系統のサービス間でなくては実現不可能である．つまり全てのサービス間で実現可能なものではない．ゆえに，複数存在するサービスの中から SH 先サービスとなる異種サービスの発見を行う必要がある．これを実現するのが異種サービス発見機能である．本機能は SH を行う上で不可欠である．

SH データ変換機能

本研究が想定する各サービスは，共通の SH インタフェースや SH プロトコルを備えないため，独自定義した SH データを送受信し，お互いのデータを解釈し合うことはできない．ゆえに，各サービスが備える SH インタフェースや SH プロトコルを理解し，お互いにデータを解釈し合えるよう変換する必要がある．これを実現するのが SH データ変換機能である．本機能を実現することで，異種サービス間での互換性確保が可能となる．

3.2 SH フレームワークの設計

3.1 で述べた機能要件を満たし SH を実現する SH フレームワークについて述べる．異種サービス発見機能は SH 管理機構と SH 支援機構，サービス情報配信機構，サービス情報管理機構により実現される．SH データ変換機能は SH 支援機構とサービス情報管理機構により実現される．図 1 に本フレームワークの全体像を示す．

本フレームワークは複数存在する SH 先候補サービスの中から，異種サービスを判定することで SH 先サービスを発見する．そして SH 元サービスからの SH データを SH 先サービスが解釈できる形に変換し仲介することで SH を実現する．この際，SH 先サービスを決定したり，SH 先が解釈可能なデータ形式を本フレームワークが知るために，サービス情報記述言語が必要となる．以下で各処理機構とサービス記述言語について説明する．

サービスハンドオフ管理機構

SH 管理機構はユーザモバイル端末上に実装され，SH 自体を管理する．具体的にはユー

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

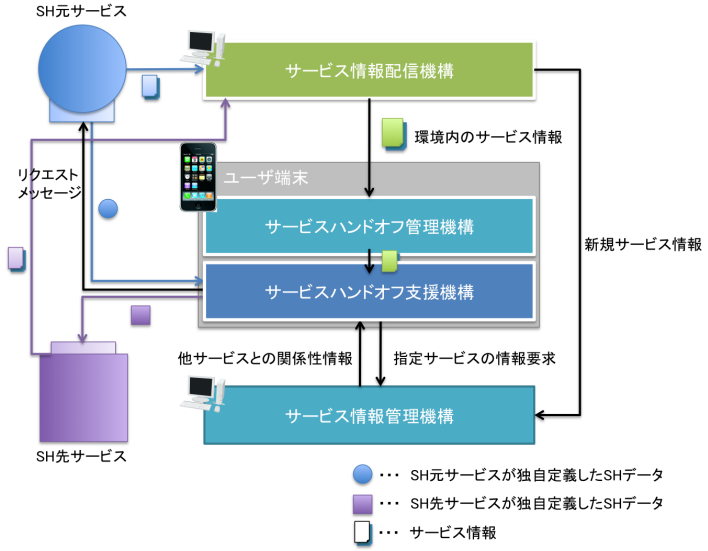

図 1 サービスハンドオフフレームワーク
Fig. 1 Service Handoff Framework

ザへ SH インタフェースを提供し，ユーザの SH 設定を可能にする．ゆえに，ユーザにとってのリッチなサービス等も本機構が定義する．また，ユーザ利用中のサービスや情報環境の移動を監視し，移動を確認した際は SH 先候補となるサービスの情報を収集する．そして，ユーザからの設定に従い SH 先候補のサービス情報ファイルをフィルタリングしたうえで，SH 支援機構に渡すことで SH 自体を管理する．

サービスハンドオフ支援機構

SH 支援機構は SH 管理機構と同じくユーザモバイル端末上に実装され，SH 管理機構から SH 先候補となるサービス情報を受け取り，SH 先を決定する．そして，SH 元サービスから SH データを取得し，SH 先サービスが解釈できるよう変換し SH データを仲介することで SH を支援する．

サービス情報配信機構

サービス情報配信機構は情報環境ごとに必要とされ，環境内のサービス情報を保持する．そして環境内に新たに加わった端末に対してサービス情報を配信することで，SH 支援機構の処理を支援する．また新規にサービスが環境内に構築され，サービス情報が追加された場合はサービス情報管理機構に新規サービス情報を通知する．

サービス情報管理機構

サービス情報管理機構は任意の範囲に一つ必要とされ，サービス情報配信機構に新たなサービス情報が追加されると，その情報を取得，解析し他のサービス間との関係性や継承を整理し，保存する．そして，SH 支援機構からの情報要求に応え，処理を支援する．

サービス情報記述言語

サービス情報記述言語は各サービスが自サービス情報をサービス情報配信機構へ登録する際に利用され，サービス提供サーバの IP アドレスや Port 番号だけでなく，各サービスが独自定義した SH データや SH プロトコル情報についても記述可能である．また，SH データについてはサービス内で独自定義したものなので，他のサービスが解釈できるようにデータ内の情報の一つ一つの意味情報の記述が可能である．

本研究では，SH フレームワークのコアである，SH 支援機構に焦点を絞って研究を進めた．ゆえに，以降では SH 支援機構を中心に述べる．しかし，サービス情報管理機構は SH 支援機構と通信し処理に大きく関わるため，SH 支援機構への応答に必要な処理部を中心に解説する．本研究では SH 支援機構とサービス情報管理機構をそれぞれ uMediator ミドルウェア，uMediator サーバとして実現する．サービス情報記述言語はサービス提供サーバの IP アドレスや Port 番号，サービスの継承情報を標準で記述可能であり，サービス情報を型と実体に分けて記述し，型を継承することで体系的に継承情報を整理可能な言語である USDL（Universal Service Description Language）を採用する．そして拡張記述を新たに設計，定義することで記述要件を満たせるようにする．他の処理機構に関しては SH 支援機構のテスト環境としての簡易実装のみを行ったので，本稿で詳細については述べない．

3.3 USDL 拡張記述の設計

USDL の拡張記述を設計・定義することで，各サービスの SH プロトコル情報をサービス情報の一部として USDL 記述可能にする．それにより異種サービスの判定が可能となるだけでなく，各サービスから uMediator ミドルウェアへの SH プロトコル情報の通知が可能となり，uMediator ミドルウェアでは SH プロトコル情報を解釈することで SH データの変換が可能となる．

拡張箇所は SH プロトコル情報を記述する handoff 要素と SH プロトコル情報の記述で利用される変数定義，変数間の関係性を記述する relationships 要素である．以下で各拡張箇所の詳細について述べる．

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

handoff 要素

handoff 要素は entity 要素の子要素として記述し，SH プロトコル情報である各サービスが独自定義した SH データの構造情報，意味情報，データタイプ（xml や csv などのフェイル形式を指す），リクエストメッセージの記述が可能である．handoff 要素は SH の受信側サービスの情報を記述する handoff_receive_protocol 要素と，SH データの送信側サービスの情報を記述する handoff_send_protocol 要素を子要素として持つ．そして，それらの子要素として実際に送受信される SH データのデータ値部分を意味情報を示す変数に置き換えた情報を記述することで SH データの構造情報を記述可能にする．変数は [変数名] で表現する．図 2 に記述例を示す．

```
<entity>
        ・・・
  <handoff>
    <handoff_send_protocol  data_type="データタイプ"  requestMes="リクエストメッセージ">
        <GoogleMaps>
            <info>
                <address>[addless]</address>          SHデータ構造情報
                <coordinate>[coordinate]</coordinate>  SHデータ意味情報
            </info>
        </GoogleMaps>
    </handoff_send_protocol>                                SH元の場合
    <handoff_receive_protocol  data_type="データタイプ" >
        <csv_data>                                  SHデータ構造情報
            [addless], [coordinate]                 SHデータ意味情報
        </csv_data>
    </handoff_receive_protocol>                             SH先の場合
  </handoff>
</entity>
```

図 2　handoff 要素記述例
Fig. 2　Example of describing handoff element

変数定義法

変数は型として type 要素に記述することで定義する．変数名は通常の型記述における id 要素の値を使用する．通常の型の記述方法と異なる点は，relationships 要素を拡張記述する点である．型の定義はユーザが自由に継承，拡張定義可能であるため，変数も同様にユーザが自由に定義する事が可能である．

relationships 要素

relationships 要素は子要素として relationship 要素を複数持つ事が可能で，relationship 要素の子要素に他の変数との関係性を記述する．関係性の記述には四則演算子を用いる．図 3 に記述例を示す．

```
<type>
    <properties>
        <id>start_sec</id>
            ・・・
    </properties>
            ・・・
    <relationships>
        <relationship> [start_min]*60 </relationship>
    </relationships>
</type>
```

図 3　relationships 要素記述例
Fig. 3　Example of describing relationships element

.

3.4　uMediator ミドルウェアの設計

uMediator ミドルウェアはユーザ端末上で SH 先サービスの発見を行い，SH データを SH 先サービスが解釈可能な形に変換することで仲介を行うミドルウェアである．SH 先サービスの発見は，SH 管理機構が収集した usdl ファイルをもとに，SH 先候補サービス群の中からユーザ利用中サービスの異種サービスを判定することで，SH 先サービスを発見する．また，SH データの変換は SH 元サービスと SH 先サービスの SH プロトコル情報を usdl ファイルから抽出し，SH 元サービスからの SH データを解釈・分解し，SH 先サービスが解釈可能な形に再構築することで SH データの変換を行う．

uMediator ミドルウェアは uMediator 処理管理部と USDL 受信部，異種サービス判定部，USDL 解析部，uMediator サーバ問合せ部，SH 先プロトコル情報取得部，SH データ取得&解析部，データマッピング部，欠如データ算出部，SH データ作成部，SH データ送信部から成る．uMediator 処理管理部は各処理部間のデータ受け渡しを行い，uMediator 全体の処理を管理する．USDL 受信部は SH 元サービスと SH 先候補サービスの usdl ファイルのパスを SH 管理機構より受信し，異種サービス判定部へ受信データを渡す．異種サービス判定部は USDL 解析部を利用し SH 元サービスと SH 先候補サービスの usdl ファイルから異種サービスの判定を行い，SH 先サービスを決定する．そして，SH 先サービス USDL 情報を SH 先プロトコル情報取得部へ，SH 元サービス USDL 情報を SH データ取得&解析部へ渡し処理を渡す．SH 先プロトコル情報取得部は SH 先サービス USDL より，必要とな

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

る変数名や作成データの構造データ等の SH プロトコル情報と Ip アドレスや Port 番号などの情報を取得し，データマッピング部，SH データ作成部，SH データ送信部に取得データを渡す．SH データ取得&解析部は SH 先サービスへ SH データのリクエストを行いデータを取得し，SH データを解析することで変数名とデータ値のリストを取得しデータマッピング部に渡す．データマッピング部は SH 先プロトコル情報取得部と SH データ取得&解析部から受信したデータをもとに，SH データの再構築に必要となる変数名とデータのマッピングを行う．この際，必要データが必ずしも全て揃うとは限らない．そのため，欠如データ算出部が揃わなかったデータを relationships 要素の情報から可能な限り算出する．これらの行程でデータのマッピングが無事完了した場合は SH 可能であるため，SH データ作成部で SH 先サービスの対応 SH プロトコルに応じた SH データの作成を行う．そして，SH 送信部において完成した SH データを SH 先サービスに送信する．図 4 に uMediator ミドルウェアの構成を示す．

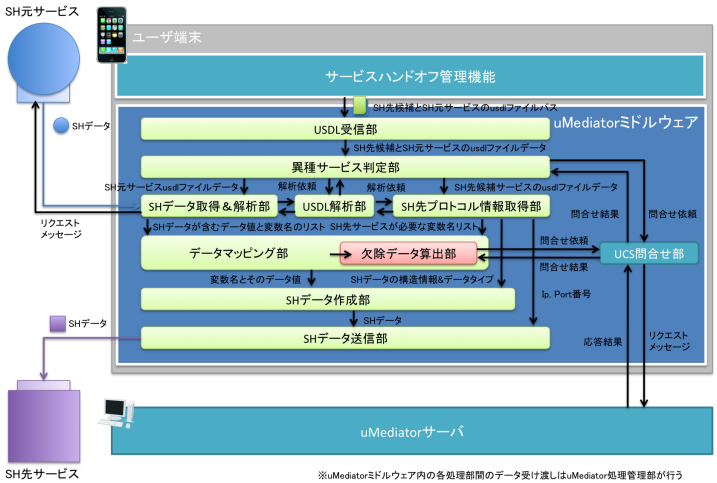

図 4 uMediator ミドルウェアの構成
Fig. 4 Composition of uMediator middleware

3.5 uMediator サーバの設計

任意の範囲で一つ設置され，uMediator ミドルウェアからのサービス情報要求に応答するサーバである．サービス開発者により作成された usdl ファイルはサービス情報配信サーバである Usdl Delivery Server にアップロードされ，Usdl Delivery Server が新規 usdl ファイルを uMediator サーバに送信する．uMediator サーバでは，受信 usdl ファイルから必要情報を抽出しデータベースとして保持することで uMediator ミドルウェアからの要求に備える．

uMediator サーバは uMediator ミドルウェアからの問合せに応答する uMediator 処理系とサービス情報配信機構（UDS）からの新規登録 usdl ファイル情報の処理を行う UDS 処理系，データベース制御部，Usdl Information データベースにより構成される（図 5）．uMediator 処理系は uMediator 接続待機部と uMediator リクエスト応答部，UDS 処理系は UDS 接続待機部と USDL 解析部より構成される．

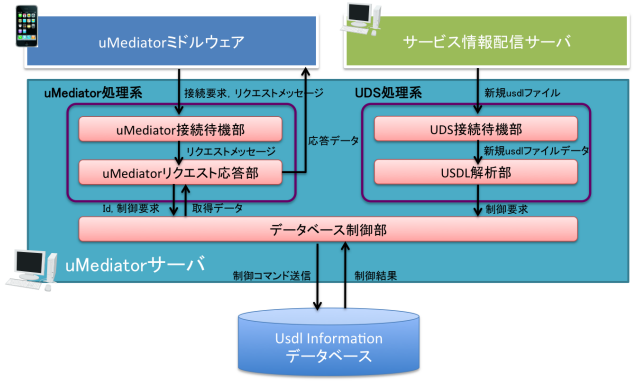

図 5 uMediator サーバの構成
Fig. 5 Composition of uMediator server

3.6 実装環境

本研究では，実装環境として Java[16] と MySQL[17] を用いた．また，データベースのアクセスを行うライブラリとして JDBC（Java DataBase Connectivity）ドライバ[15] を用いた．表 1 に詳細を示す．

4. SH 実証実験

本節では，SH 実証実験について述べる．本稿では異なる環境に存在し，異なる開発者に

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

表 1 実装環境
Table 1 Mounting environment

実装言語	Java5.1.30
データベース	MySQL（バージョン 5.1.30）
JDBC ドライバ	Connector/J 5.0

より開発された共通のインタフェースを備えないストリーミング動画再生サービス間での SH を想定し実証実験を行った．

4.1 実験環境

本実験では，図 6 に示すように左の環境を環境 α，右の環境を環境 β と想定し，それぞれ端末は別環境に存在すると想定する．環境 α の端末が SH 元サービス提供端末，環境 β の端末が SH 先サービス提供端末である．それぞれ異なる開発者が独自定義した SH プロトコルインタフェースを備えたストリーミング動画再生サービスである．

図 6 実験環境
Fig. 6 Experimental environment

また，実験用計算機として uMediator ミドルウェアが処理を行うユーザ端末，uMediator サーバ，SH 元サービス提供端末，SH 先サービス提供端末を用意した．それぞれ，ユーザ端末，サーバ端末，端末 A，端末 B とする．表 2 に各計算機の詳細を示す．

環境 β では SH 先サービスの usdl ファイル以外に環境内に複数のサービスが存在するよう見せるためにダミーの usdl ファイルを用意した．表 3 に環境 β に用意した 4 つの usdl ファイルの概要を示す．

表 2 実験用端末の性能表
Table 2 Performance table of experimental terminal

	ユーザ端末	サーバ端末	端末 A	端末 B
PC	MacBook Pro	Mac Pro	iMac	iMac
CPU	2.8GHz Core2Duo	2x2.8GHz Quad-Core	2.66GHz Core2Duo	2.66GHz Core2Duo
メモリ	4GB	16GB	4GB	4GB
OS	Mac OS X 10.5.8	Mac OS X 10.5.8	Mac OS X 10.5.8	Mac OS X 10.5.8

表 3 環境 β の usdl ファイル概要
Table 3 Outline of usdl file of environmental β

サービス	SH 元サービスとの関係性
ストリーミング動画再生サービス	ホップ数 1（SH 先サービス）
動画再生サービス	ホップ数 2
音楽再生サービス	ホップ数 3
ナビゲーションサービス	関係性無し

4.2 実験結果

本研究では SH 管理機構を研究対象としていないため，ユーザが環境 α から環境 β へ移動した事を手動で通知し，端末 A で再生中のストリーミング動画が端末 B へ再生途中から切り替わるか実験を行った．図 7 の実験風景より分かるように，ユーザの移動とともにストリーミング動画が端末 B へ再生途中から切り替わる事を確認した．つまり，uMediator ミドルウェアにより環境 β に複数存在するサービスの中から異種サービスの発見が正常に行われ，SH データの変換が正常に行われることで SH が成功したことを確認した．

図 7 実験風景
Fig. 7 Experiment scenery

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

また，実験の際に SH 元サービスが uMediator ミドルウェアに送信した SH データと uMediator ミドルウェアが SH 先サービスに送信した SH データを図 8 に示す．

```
SH元サービス→uMediator（CSV）：
        http://www.youtube.com/watch?v=XXXX,1831

uMediator→SH先サービス（XML）：
  <StreamingMovieServiceB>
      <start_point>18.31</start_point>
      <url>http://www.youtube.com/watch?v=XXXX,1831</url>
  </StreamingMovieServiceB>
```

図 8 本実験で送受信された SH データ
Fig. 8 SH data sent and received by actual experiment

図 8 より分かるように，SH 元サービス→ uMediator ミドルウェアの際には 1831 とミリ秒単位として表現されていたデータが，uMediator ミドルウェア→ SH 先サービスの段階では 1000 分の 1 倍され秒単位に換算されている．これは uMediator ミドルウェアが usdl_info テーブルの relationship 情報を元に正常にデータ変換を行ったことを示している．

本実証実験により，uMediator は複数存在するサービスの中から異種サービスを発見し，各サービスの SH プロトコルに応じた SH データの変換を行える事を実証した．また，異種サービス間での SH に uMediator が有効であることを実証した．

5. 関 連 研 究

関連研究としては，複数の計算機を協調させることによって，ユーザが利用しているサービスを別の計算機へ移送するサービスローミングがある．ここでのサービスは，情報機器に存在するアプリケーションからユーザへ提供される機能を指す．例えば，メールソフトであればメール作成機能やメール送受信機能を指し，電話であれば通話機能を指す．

サービスローミングの実現手法には，アプリケーションが動作しているリモート計算機のデスクトップ画面をローカル計算機へ移送する方法（視覚移送）と，アプリケーションそのものを移送する方法（全移送），アプリケーションの状態のみ移送する方法（状態移送）がある．視覚移送は，リモートのデスクトップ画面をキャプチャして移送するため，既存アプリケーションをそのまま利用でき，VNC[9] や X Window System[10] で用いられている手法である．全移動は，アプリケーション全てを移送するため，アプリケーションを移送元の計算機と同じ状態で移送先の計算機に移送可能である．この手法は主にプロセスマイグレーション[11][12][13] として実現される．状態移送は，移送元の計算機で動作しているアプリケーションの状態を移送し，移送先の計算機で同じ状態を持つアプリケーションを生成する．例として，モバイルオブジェクト[14] が挙げられる．この手法では，送られてきた状態に応じてアプリケーションを再生成するため，移送先の計算機に同じアプリケーションの実行コードが設置されている必要がある．

上記の全ての移送手法において移送先アプリケーションは，移送元アプリケーションの移送手法に対応するよう開発されている．つまり，移送元と移送先のアプリケーションが共通のフレームワークまたは共通の API で開発されている．または専用のソフトウェアがインストールされているのである．これは，移送元と移送先のアプリケーションの管理主体が同一であるか，開発者が同一である場合に可能である．または，標準となるような移送手法が確立されている場合である．しかし現状そのような手法は存在しない．

本研究が想定するような情報環境を跨ぐ移動の場合，SH 元と SH 先のアプリケーションの管理主体や開発者が同一であるような状況は考えにくい．ゆえに，これらの移送手法を適用しサービスローミングを行うことは不可能である．また，ユーザの移動に応じサービスを切替える点は同じであるが，同系統の異なるサービスへの切替えを目的とする本研究と，同一のサービスを移送するサービスローミングとでは目的が異なる．図 9 に SH とサービスローミングの対象領域の差異を示す．

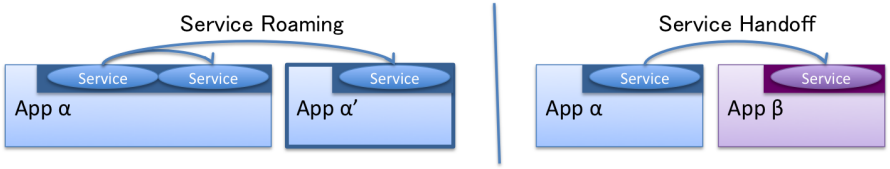

図 9 本研究とサービスローミングの対象領域の差異
Fig. 9 Difference in service handoff and service roaming's target

6. ま と め

本稿では，はじめに現在の情報環境において新たに発生すると考えられる，移動した各環境で利用可能なサービスに切替えサービスを利用したいという要求と，実現の際に発生する

Vol.2010-SLDM-144 No.24
Vol.2010-EMB-16 No.24
Vol.2010-MBL-53 No.24
Vol.2010-UBI-25 No.24
2010/3/26

問題について述べた．具体的には異なる環境に複数存在するサービスの中から切替え先となる異種サービスの発見問題，環境の管理者もサービスの開発者も異なり共通のインタフェースを備えない異種サービス間での互換性確保問題の二つの問題があった．さらに互換性確保問題は各サービスが独自に作成したデータを送受信しあえる互換性と，お互いに独自定義したデータを解釈できる互換性を確保する必要があった．そこで本研究では，これらの問題を解決する研究領域として SH を定義し，SH を実現する環境として SH フレームワークを提案した．

本研究では，SH フレームワークで必要とされる機構のうちコアとなる SH 支援機構と，本機構の処理に大きく関わるサービス情報管理機構に焦点を当て研究を行った．SH 支援機構はユーザ端末上の uMediator ミドルウェアとして，サービス情報管理機構は任意の範囲に一つ設置が求められる uMediator サーバとして設計，実装を行った．また，サービス情報を記述するために USDL の拡張記述を設計・定義した．以上により，問題点として挙げた切替え先サービスの発見と，異種サービス間でのデータ送受信，解釈の互換性確保の問題を解決した．

本稿では uMediator を利用し，異なる環境に存在し異なる開発者が開発した共通のインタフェースを備えないストリーミング動画再生サービス間での SH の実証実験を行い，uMediator の有効性を実証した．

今後は，本研究では焦点を当てていなかった SH 管理機構，サービス情報配信機構へ焦点を当て研究を行う必要がある．特に SH 管理機構は，SH 実用化のためにはどのような設定が必要か，よりユースケースに踏み込み研究を行う必要がある．また SH のトリガであるユーザの情報環境移動は，何を基準にどのように判断すべきなのかなど多くの研究課題が存在する．また，本研究では他サービスの usdl を継承し，その継承を調べることでサービス同士の関係を求め異種サービスを判定している．しかし，他サービスの開発者が記述した usdl を参照する仕組みは存在しない．ゆえに uMediator サーバの情報をサービス開発者が閲覧し，開発サービス情報を継承を利用し記述可能にする必要がある．

謝辞 本研究の一部は，総務省「ユビキタスサービスプラットフォーム技術の研究開発」の成果である

参 考 文 献

1) 山崎 俊作, 伊藤 友隆, 河田 恭兵, 生天目 直哉, 小河原 栄一, 高橋 元, 中澤 仁, 徳田 英幸:**あしナビ：主張するアンビエントナビゲーションシステムの試作**, 情報処理学会シンポジウム論文集. Vol.2006, No.4, pp.95–86 (2008).
2) 上野まちナビ実験:*http://www.kensetsu.metro.tokyo.jp/machinavi/index.html*.
3) UPnP Forum: *http://www.upnp.org/*.
4) Vladimir Kulyukin, Chaitanya Gharpure, John Nicholson, Sachin Pavithran:*RFID in Robot-Assisted Indoor Navigation for the Visually Impaired*, Proceedings of the IEEE International Conference(2004).
5) J. Nakazawa et al:*A Bridging Framework for Universal Interoperability in Pervasive Systems*, Proc. 26th IEEE Int'l Conf. Distributed Computing Systems (ICDCS 2006), IEEE Press, pp.3(2006).
6) UPnP Forum: *http://www.upnp.org/*.
7) Networking Bonjour: *http://developer.apple.com/networking/bonjour/*.
8) JINI NETWORK TECHNOLOGY: *http://jp.sun.com/jini/*.
9) T. Richardson, Q. Stafford-Fraser, K. R. Wood, A. Hopper:*Virtual Network Computing*, IEEE Internet Computing. Vol.1, pp.1–2 (1998).
10) X.Org: *http://www.x.org/*.
11) F. Douglis, J. Ousterhout:*Transparent process migration: Design alternatives and the Sprite implementation*, Software: Practice and Experience. Vol.21, No.8, pp. 757–785(1991).
12) T. Sakamoto, T. Sekiguchi, A. Yonezawa:*Bytecode Transformation for Portable Thread Migration in Java*, In Proceedings of the Joint Sym- posium on Agent Systems and Applications / Mobile Agents (ASA/MA). pp.16–28(2000).
13) H. Liu, H. Jin, X. Liao, L. Hu, C. Yu:*Live migration of virtual machine based on full system trace and replay*, High Performance Distributed Computing Proceedings of the 18th ACM international symposium on High performance distributed computing. pp.101-110(2000).
14) A. Fuggetta, G. P. Picco, G. Vigna:*Understanding Code Mobility*, IEEE Trans. on Software Engineering(1998).
15) MySQL Connector/J Documentation: *http://dev.mysql.com/doc/refman/5.1/en/connector-j.html*.
16) java.sun.com: *http://java.sun.com/*.
17) MySQL 5.1 Reference Manual: *http://dev.mysql.com/doc/refman/5.1/en/index.html*.
18) B. W. Kernighan, D. M. Ritchie:*The C Programming Language*, Prentice-Hall(1988).