一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SWIM2021-18,SC2021-16(2021-08)

コンテキストに基づくサービス連携を支援するイベントルーチングサービスの検討

中田　匠哉†　　中村　匡秀†,††

† 神戸大学 〒 657–8501 神戸市灘区六甲台町 1–1
†† 理化学研究所・革新知能統合研究センター〒 103–0027 東京都中央区日本橋 1–4–1
E-mail: †tnakata@ws.cs.kobe-u.ac.jp, ††masa-n@cs.kobe-u.ac.jp

あらまし　高度なスマートシステムでは，物理世界やサイバー空間における様々な状況（コンテキスト）に応じて，異種分散サービスを動的に連携させることが求められる．現状このようなサービス連携のロジックは，それぞれのアプリケーションで個別に実装されており，連携の複雑さに比例してアプリケーションの複雑度や開発工数が増大する．本研究では，コンテキスト・アウェアなサービス連携を，アプリケーションに中立な形で提供するサービス Uni-messe (Unified Rule-based Message Deivery Service) を提案する．Uni-messe は，ECR (Event-Condition-Routing) ルールに基づき，様々なアプリケーションやサービスで生起されたイベント・メッセージを，定義された条件に応じて，適切なアプリケーションへと配信する．本稿では，まず Publish/Subscribe メッセージ基盤を用いた Uni-messe のアーキテクチャを考察する．次に，6W1H (Who/Whom/When/Where/What/Why/How) イベントモデルに基づく ECR ルールのデータモデルを提案する．最後に，ECR ルールの評価とルーチングの振る舞いを定義する．Uni-messe を使って，スマートホーム内のカーテンの自動制御，および，デイリールーチンの自動化を行うアプリケーションを実装した．Uni-messe によって，ルールに基づくサービス連携をアプリケーションから分離できるため，アプリケーションの開発者や利用者は，サービス連携をより効率的に実装・再利用できるようになる．

キーワード　スマートシステム，マイクロサービス，コンテキストアウェア，ルールベース，Pub/Sub，IoT

Developing Event Routing Service to Support Context-Aware Service Integration

Takuya NAKATA† and Masahide NAKAMURA†,††

† Kobe University, Rokkodai 1–1, Nada, Kobe, Hyogo, 657–8501 Japan
†† Riken AIP, 1–4–1 Nihon-bashi, Chuo-ku, Tokyo 103–0027
E-mail: †tnakata@ws.cs.kobe-u.ac.jp, ††masa-n@cs.kobe-u.ac.jp

Abstract　In advanced smart systems, heterogeneous distributed services are integrated dynamically, based on various contexts in cyber/physical worlds. Currently, the logic of such service integration is implemented specifically in each application. Thus, as the way of integration becomes sophisticated, the complexity and development effort of the application become quite expensive. In this research, we propose a service, called *Uni-messe (Unified Rule-based Message Delivery Service)*, which provides context-aware service integration in an application-neutral manner. Based on ECR (Event-Condition-Routing) rules, Uni-messe routes an event message from an application to a designated application based on a specified condition. We first present the architecture of Uni-messe with publish/subscribe messaging platform. We then propose the data model of the ECR rules based on 6W1H (Who/Whom/When/Where/What/Why/How) event model. Finally, we define the condition evaluation and routing behaviors. Using the implementation of Uni-messe, we demonstrate automatic curtain control and daily routine automation in a smart home. Since Uni-messe de-couples the rule-based service integration from individual applications, it allows developers and users to implement and reuse flexible integration efficiently.

Key words　smart system, micro-services, context awareness, rule-based system, Pub/Sub, IoT

This article is a technical report without peer review, and its polished and/or extended version may be published elsewhere.
Copyright ©2021 by IEICE

1. はじめに

情報通信技術（ICT）により人々の生活は便利になり，昨今では特に**スマートシステム**の発展が著しい．スマートシステムは，物理世界におけるモノ・コトとクラウド上の計算資源をネットワークを介して連携し，付加価値サービス（**スマートサービス**）を実現するシステムである．スマートシステムには様々な適用分野があり，その例としてスマートホーム，スマートシティ，スマートヘルスケアなどが存在する．スマートホーム分野では，IoT を活用して冷蔵庫や空調といった家電を動かしたり，電気料金を考慮したサービス提供などが行われている．スマートシティ分野では，既存の行政サービスに情報技術を組み合わせることで，行政の効率化や市民サービスの向上への取り組みが行われている．スマートヘルスケア分野では，高齢者の見守りや日々の生活のログなどを通じて，健康に生活を送るためのサービスが提供されている．

これらのスマートシステムに共通する特徴は，複数のサービスを連携・組み合わせることによって提供できるサービスの質や量が改善することである．特に，物理世界やサイバー空間における様々な状況（**コンテキスト**）に応じて，異種分散サービスを動的に連携させることで，その場，その時，その人に最適なサービス（**コンテキスト・アウェア・サービス**）を提供できる．しかしながら，一般的に異種分散サービスの複雑な連携は容易ではなく，コンテキスト・アウェアサービスの効率的な実現には様々な課題がある．

異種分散サービスを連携するための基盤技術として，マイクロサービスアーキテクチャ，Publish/Subscribe メッセージング基盤 (Pub/Sub) が挙げられる．マイクロサービスアーキテクチャは，システムの機能を小さなサービスの単位で公開し，それらを疎結合して新たなサービスを実現するというアーキテクチャ・パラダイムである．Pub/Sub は仲介者（ブローカー）によってサービスの呼び出しを非同期に行うメッセージング基盤である．コンテキスト・アウェアサービスを実現するには，これらの基盤の上に何らかの**サービス連携ロジック**を実装し，状況に応じたサービス連携を実現する必要がある．

コンテキスト・アウェアな連携ロジックを実現するための最も直感的なアプローチは，**ルールベースシステム**である．これは，システムにおいて「ある状況が成立した時には，このサービスを呼び出す」というルールを決め，そのルールに従ってサービス連携を行う方法である．多くの場合，このようなルールベースシステムは，それぞれのスマートシステムやアプリケーションで個別に実装・管理される．ルールのセマンティクスやサービス呼び出しの方法はシステム毎にまちまちであり，その表現能力や複雑度に応じて，アプリケーションの複雑度や開発工数が増大する．

そこで我々は，このようなルールベースシステムを**アプリケーションに中立なサービス**として切り出すことで，コンテキスト・アウェア・サービスを迅速に実現できないかという考えに至った．類似するサービスに，IFTTT [1] や RuCAS [2] が存在するが，連携可能なサービスやアプリケーションが決まっていたり，扱えるコンテキストの汎用性に欠ける等の限界がある．

本研究では，コンテキスト・アウェアなサービス連携を支援する新たなサービス，**Uni-messe（Unified Rule-based MessageDelivery Service）**を提案する．Uni-messe は，イベント (Event), 条件 (Condition), ルーチング (Routing) で定義される **ECR ルール**に基づき，アプリケーションに中立なルールベースサービスを提供する．Pub/Sub 基盤と連動し，様々なアプリケーションやサービスで生起されたイベント・メッセージを，指定された条件に応じて，適切なアプリケーションへと配信する．

提案手法では，まず Pub/Sub メッセージ基盤を用いた Uni-messe のアーキテクチャを考察する．サービス間の連携を全て Pub/Sub で経由で行うことで，サービスの疎結合性を高め，サービスの連携・組み換えを容易にする．また，Uni-messe の責務をサービス実行の指示 (ルーチング) にとどめ，実際のサービス実行は個別のアプリケーションに委譲することで，アプリケーション中立性を維持する．次に，Uni-messe の ECR ルールのデータモデルを提案する．システムで生起するイベントを **6W1H (Who/Whom/When/Where/What/Why/How)** の形式でモデリングし，アプリケーションに中立なコンテキストの定義を可能にする．また，条件部では，6W1H のそれぞれについて汎用的な条件を指定できるようにし，ルーチングには任意の Pub/Sub のトピックを指定する．最後に，ECR ルールの振る舞いを定義する．具体的には，Uni-messe にイベントが到着した際，各ルールの条件部を評価し成立したらルールを発火，ルーチングに指定された Pub/Sub のトピックへイベントを転送する．この時，発火条件をイベントの Why に追記することで，サービス実行の根拠をアプリケーションに伝える．

本稿では，提案した Uni-messe のプロトタイプを実装し，スマートホーム向けの 2 種類のコンテキスト・アウェアサービスを実現した．1 つは，照度センサと電動カーテンを連携し，外部の照度に応じて自動的にカーテンを開閉するサービスである．もう 1 つは，人感センサを用いて，時刻に応じた宅内のデイリールーチンを自動化するサービスである．Uni-messe によって，ルールに基づくサービス連携をスマートホーム本体から分離できるため，スマートホームの開発者や利用者は，コンテキスト・アウェアサービスをより効率的に実現・運用できるようになる．

2. 準　　備

2.1 スマートサービス

スマートシステムとは，物理世界の情報とサイバー空間の計算資源をネットワークで連携し，付加価値サービスを実現するシステムである．スマートシステム上で動く**スマートサービス**の研究開発が活発であり，我々の研究室でも LINE による高齢者の見守りサービス [3]，物忘れ防止スピーカー ALPS など様々なサービスを扱っている [4]．これらのスマートシステムに共通する特徴として，異種分散サービスを動的に連携させることで，部屋やヒトの状況（コンテキスト）をセンサやサービスで読み取って認識（アウェア）し，状況に合わせた適切なサービス（**コンテキスト・アウェア・サービス**）を提供する点が挙げら

れる．コンテキストアウェア技術を利用したサービスは IFTTT や Nature Remo [5] など多くの商品が実用化されており，一般にも普及している．

異種分散サービスの連携の実現によってスマートサービスを実現するために一般的に使われる技術・概念としては，マイクロサービスアーキテクチャ，Pub/Sub，ルールベースシステムなどが挙げられる．これらの技術を実現するための実装を代表として，各スマートサービスの実装には共通する箇所が多いため，FIWARE [6]，IFTTT，RuCAS，Log4Care [7] といった実装の基盤となるフレームワークを活用することで効率的な実装が可能になる．

2.2 スマートサービス実現のための技術

マイクロサービスアーキテクチャは，システムの機能を小さなサービスの単位で公開し，それらを疎結合して新たなサービスを実現するというアーキテクチャ・パラダイムである．スマートサービスは，様々なセンサやサービスを非常に複雑に連携させる必要があることから，開発コストや実現可能性の問題が存在する．そのため，サービスの開発を効率化するために，サービス同士を疎結合性を保ったまま連携させる必要がある．この疎結合性を高めるための技術としては，Pub/Sub や RESTful API などが活用される．

Pub/Sub（Publish/Subsucribe メッセージング基盤）は，仲介者（ブローカー）によってサービス間の非同期通信を行うミドルウェアである．Pub/Sub におけるメッセージのやり取りは，トピックという私書箱の役割のメッセージの入れ物を大量に用意しておき，このトピックを介しておこなう．各トピックにおいてサービスは，トピックへメッセージを受け渡す Publish と，トピックからメッセージを受け取る Subscribe の 2 種類の操作が可能である．Publish する側と Subscribe する側は互いを意識する必要はなく，Publish と Subscribe のタイミングは非同期である．また，登録した様々なサービスが Publish/Subscribe することができることから，複数のサービスを複雑に連携させることができる．このような Pub/Sub の性質から，Pub/Sub を用いることで**疎結合性**を保ちながらサービスの高度な連携を行うことができる．

コンテキスト・アウェアサービスを実現するためには，マイクロサービスアーキテクチャや Pub/Sub といったサービス連携基盤の上層に，具体的なサービス連携ロジックを実装し，状況に応じたサービス連携を実現する必要がある．サービス連携ロジックの構築における直感的なアプローチとしてルールベースシステムが挙げられる．**ルールベースシステム**は，Event・Conditon・Action の ECA ルールに基づいて処理を行う，推論型の AI システムである．ECA ルールは，生起した Event を Condition で評価し，条件式を満たせば Action が発火する，というプロセスで動作する．Event を評価するための Condition ルールの条件式は，事前にルールベースシステムに登録する必要があり，同時に条件式を満たした場合に発火する Action も事前の紐づけが必要である．ルールベースシステムは，サービスが発生した Event と Action で発火する別のサービスを仲介できることから，コンテキスト・アウェアサービスの実装に直感的に利用できる．

2.3 既存のスマートサービス実装基盤の課題

スマートサービスの実装は，これらの技術を組み合わせて行われるが，多くのシステムにおいて似通った実装となる．そのため，スマートサービスに必要な技術を組み合わせて提供した枠組みが多く提供されており，これらを用いることで開発の効率化が可能である．既存の枠組みの例としては，FIWARE，IFTTT，RuCAS，Log4Care などが挙げられる．しかしながら，従来の枠組みでは連携可能なサー飛車アプリケーションが決まっていたり，扱えるコンテキストの汎用性に欠ける等の限界があり，ルールベースシステムを中立的に扱う十分な実装基盤は存在しなかった．

コンテキストアウェアアプリにおけるルールベースシステムは，サービスによってルールのセマンティクスや適切なサービス呼び出しの方法が異なり，それらの表現能力や複雑度に応じて，アプリケーションの複雑度や開発工数が増大するという特徴がある．そこで我々は，これらのルールベースシステムを，アプリに中立的なルールの管理（CRUD）・評価処理を行うサービスとして切り出して提供することによって，スマートサービス開発の効率化に繋がると考えた．

3. 提案手法:Uni-messe

3.1 キーアイデア

本研究では，コンテキスト・アウェアなサービス連携を支援する新たなサービス，**Uni-messe（Unified Rule-based Message Delivery Service）**を提案する．

Uni-messe の設計にあたっては，コンテキストアウェアアプリに適したルールベースシステムについてアーキテクチャを考察し，必要な処理を洗い出した．さらに，アプリに中立な処理を取り出してサービス化を行った．

本研究のアプローチは次の通りである．

- **（A1）** Uni-messe のアーキテクチャ
- **（A2）** アプリケーションに中立な ECR データモデル
- **（A3）**Condition ルールによるイベント評価とルーチング

3.2 （A1） Uni-messe のアーキテクチャ

Uni-messe は，Pub/Sub メッセージ基盤の上に，ルール評価及びルーチングを行うサービスとして構築される．各アプリから生起されるイベントとその条件によって，適切なアプリにイベントを転送する．Uni-messe，Pub/Sub，外部アプリの関係を示すアーキテクチャを図 1 に示す．

Uni-messe と周辺アプリにおけるサービスの設計・スコープは次のとおりである．Uni-messe は，ルールの管理・イベント評価（式評価）・処理の指示（ルーチング）の機能を提供する．各ルーチングの規則は，**ECR ルール（Event・Condition・Route）**として定義される．それに対して，Uni-messe と連携する個別のアプリは，個々のルールの定義・意味づけ・解釈・処理の実行（アクション）を行う．各アプリは，Uni-messe のルール管理機能を使用して事前にルールの CRUD を行う．

つまり，Uni-messe と個別アプリでは明確にサービスの責務が異なる．Uni-messe の責務は，イベント評価の結果に応じてどのサービスにどの処理をするか指示すること（ルーチング）

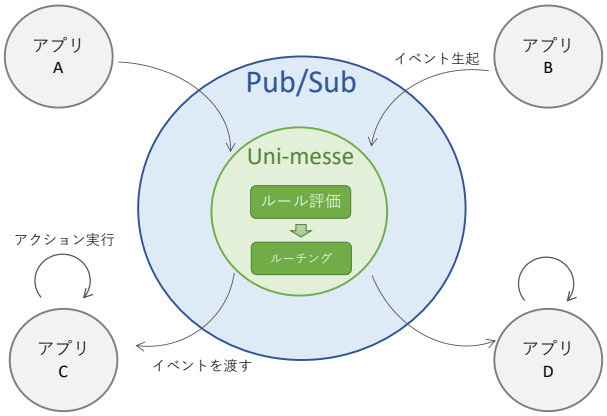

図 1　Uni-messe のアーキテクチャ

であり，処理の実行は責務ではない．処理の実行は，個別のアプリが責務として負うことになる．一般的なルールベースシステムで採用される ECA ルールでは，Route ではなく Action が採用されるが，Uni-messe では新たに Route という概念を導入した．これは，Uni-messe ではサービス処理（Action）を実行する責務はなく，処理を実行するサービスを指示すること（Route）が責務であるためである．

Uni-messe アーキテクチャの動作の流れは次のとおりである．

- **（1. イベントソース）** 各アプリは，部屋やヒトの状況（コンテキスト）を認識し，Pub/Sub を介して Uni-messe にイベントを渡す．
- **（2. イベントルーチング）**Uni-messe は，イベントを ECA ルールで評価する．さらに，評価の結果として Route すなわち Pub/Sub 上のトピックを選び，選んだ全てのトピックにイベントを受け渡す．
- **（3. イベントデスティネーション）** アプリは，Pub/Sub を介して任意のタイミングでイベントを受け取る．さらに，イベントを自由に解釈し，任意の処理を実行する．

3.3 （A2） アプリケーションに中立な ECR データモデル

ECR ルールは，Event・Condition・Route で構築されるルールである．Uni-messe では，ECR それぞれのデータモデルは次のように設計した．

Event は，6W1H（How / Who / Whom / When / Where /What / Why）の 7 つの要素によってコンテキストを表現する構造体である．How は，イベントを検知したサービス・センサの名前・ID を表現する．Who/Whom は，イベントを起こした/起こされた人や機器の名前・ID を表現する．Where は，イベントが発生した場所を表現する．When は，イベントが発生した時刻を ISO8061 準拠で表現する．What は，詳細なイベントの属性を JSON 文字列で表現する．Why は，イベントが発生してきた経緯を辿るリストであり，スタックトレースとしての役割を果たす．

Condition は，Event の値で評価可能な条件式の集合と，それら条件を全て満たしたときに実行するルーチング（Route）の集合の 2 つを保持する構造体である．条件式は Why を除く 6W1H の全てにおいて定義される．How / Who / Whom / Where については，登録したキーワードに対して完全一致・部分一致・完全不一致・集合の要素判定といった基本的な条件評価を行う．条件のデータ構造はキーワードと評価方法のペアで保持される．When については，周期的な時刻を表現する cron 式の一種として保持し，イベントの When 要素が cron 式に含まれる時刻であった場合のみ真を返す．What については，論理式を用いた複雑な条件評価を行う．条件のデータ構造は Java 条件式として評価可能な文字列として保持される．イベントの What 要素の JSON のキーバリューを置換して評価を行い，Java 条件式として真ならば真を返す．Why については，今回は評価式を導入しなかった．

Route は，イベントの宛先を表現する構造体である．データ構造としては，イベントを渡す Pub/Sub 上のトピック名を 1 つだけ保持している．

3.4 （A3） Condition によるイベント評価とルーチング

ECR ルールにおける Condition の役割は，Why を除く 6W1H の評価式と Route の集合を結びつけることである．すなわち，ある一連の評価結果が真となった場合に，Condition を通して結び付けられた Route 全てが発火することになる．Route が発火すると，アプリから受け取ったイベントを Route が保持するトピックに対してそのまま Publish する処理が実行される．このとき，Route の発火原因となった Condition をイベントの Why に追記することで，イベントの因果関係を記録する．これによって，サービスが実行される論理的根拠を明確に示すことが可能になり，Uni-messe において機械学習ではなくルールベースシステムを用いてでイベントを判断することで生まれる利点の 1 つとなる．

4. 実　　装

Uni-messe サービスのバックエンド実装は Kotlin 言語と Spring Boot フレームワークを用いて行った．また，データベースサーバとして MySQL を，Web アプリケーションサーバとして Apache Tomcat を利用した．Uni-messe が利用する Pub/Sub には，別途実装済みの Pub/Sub サービスを外部から利用する形で導入した．Uni-messe の利用手段としては，RESTful API を提供するほか，図 2 に示すフロントエンド UI によってブラウザから Web アプリとして操作できる．フロントエンド UI は JavaScript ライブラリである React で実装した．

5. デモアプリ

5.1 デモアプリ 1

Uni-messe を用いたサービス実装のシンプルな例として，明るさの変化に応じてカーテンを自動で開閉するデモアプリを実装した．イベントを発生するアプリには，照度センサを用いた．アクションを実行するアプリには，別途実装済みのカーテン開閉 API を用いた．

サービス全体のアーキテクチャは次のとおりである．まず，Python で定期的にセンサから照度を取得し，照度が短時間で変

図 2　Uni-messe のフロントエンド UI

化した場合にイベントを Uni-messe に渡すプログラムを稼働させた．次に，Uni-messe は，Condition でセンサの ID・場所・照度（明るい/暗い）を判定し，適切な Pub/Sub トピックにイベントを受け渡す．例えば，照度が閾値より高ければ light トピックに，低ければ dark トピックに Publish 処理を行った．最後に，Node.js で Uni-messe が渡したイベントを受け取り（Subscribe），API を実行することでカーテンを開閉した．

このデモアプリにおける Uni-messe の役割は，イベントを発生させたセンサの同定および，照度に応じたイベントのルーチングによって実行する API を切り替えることである．

5.2　デモアプリ 2

次に，より複雑なサービスの連携の例として，複数のデイリールーチンを自動化するデモアプリを実装した．イベントを発生するアプリとして，人感センサと照度センサを用いた．アクションを実行するアプリには，カーテン開閉 API と SwitchBot [8] の家電操作 API を用いた．これらのアプリの組み合わせによって，部屋に人が入ってきたときに，外が明るければカーテンを開け，外が暗ければカーテンを閉め，時刻が朝だったらテレビをつけて湯沸し器のスイッチを押す，という処理を自動化した．

イベントの判定には，次の 3 つの Condition ルールを用いた．

Condition A
- How : センササービス A
- Who : sensorA
- Where : 玄関の棚の上
- What : light >= 200 （明るい）
- Route : RouteA （トピック A）

Condition B
- How : センササービス A
- Who : sensorA
- Where : 玄関の棚の上
- What : light < 200 （暗い）
- Route : RouteB （トピック B）

Condition C
- How : センササービス A
- Who : sensorA
- Where : 玄関の棚の上
- When : * * 6-10 * * *（朝の 6 時から 10 時）
- Route : RouteC （トピック C）

各 Condition ルールは，6W1H の条件をすべて満たしたときに，紐づけられた Route の Pub/Sub トピックにイベントを Publish する．

ここで，各トピックを Subscribe するアプリとして以下を設定した．各アプリは，Subscribe したトピックに Uni-messe からイベントが Publish された場合に処理を実行するシンプルな処理が記述されたプログラムである．

- **（トピック A）** カーテンを開けるサービス
- **（トピック B）** カーテンを閉めるサービス
- **（トピック C）** テレビをつけるサービス，および，湯沸し器のスイッチを押すサービス

また，イベントを生起させるセンササービス A は，人感センサと照度センサを組み合わせたサービスである．すなわち，人感センサが人の存在に反応した場合に，同時に照度センサからも値を取得してイベント中の What に light 値として埋め込み．このイベントを Uni-messe に受け渡すという挙動をするサービスである．

これらの ECR ルールと Subscribe 設定によって，複数のデイリールーチンを自動化する以下のような動作が実現できる．

- **朝に人を検知した場合**：外が明るく時刻が朝であるため，Condition A と C を満たすことにより，カーテンを開ける・テレビをつける・湯沸し器のスイッチを押す，という処理が実行される．
- **昼に人を検知した場合**：外が明るいため，Condition A を満たすことにより，カーテンを開ける，という処理が実行される．
- **夜に人を検知した場合**：外が暗いため，Condition B を満たすことにより，カーテンを閉める，という処理が実行される．

6.　関 連 研 究

既存サービスとして，様々なアプリを手軽に結び付けられるサービスである IFTTT が代表的である．IFTTT を用いたサービス連携の例として，電話の応答に失敗することをトリガー，メールに通知することをアクションとして設定することで，電話の応答に失敗した場合にメールに通知を送ることができる．IFTTT の強みは，電話アプリやメールアプリといったポピュラーな商用既存サービスの機能が既に多数登録されており，プログラムを書く技術を持たない人でも簡単にサービス同士を結び付けられる点である．また，Webhook を用いて独自に開発した新規サービスを連携させることも可能である．

IFTTT の欠点として，トリガーとアクションの結びつきが強く，柔軟性に欠けるという点が挙げられる．そのため，Uni-messe のように定期的に発生しているイベントを柔軟に別サービスの

実行トリガーとして用いる，といった運用が困難である．また，Uni-messe はイベントのルーチングを目的としたサービスであるのに対し，IFTTT はサービスを結合して新たなサービスを生成することを目的としたサービスであり，サービスの目的が全く異なる．

コンテキストアウェアと Pub/Sub を連携した先行研究には，IoV（Internet of Vehicles）フレームワークを提案した Iqbal ら [9] の研究や，強化学習を用いた Kim ら [10] の研究がある．これらの研究は，Uni-messe とは異なりコンテキストの推論に機械学習を用いることを提案している点が対照的である．

7. ま と め

本研究では，サービス連携の効率化によりコンテキストアウェアアプリの開発コストを低下させることを目的として，ECR ルールの管理・評価を行うサービスである Uni-messe を提案した．Uni-messe は，Condition ルールに基づくイベントのルーチングをアプリケーションに中立な形で代行することにより，各アプリで行われていた該当処理の冗長な実装を不要とすることができる．また，Pub/Sub を介してアプリとイベントをやり取りすることによって，サービス間の疎結合性を保つことができ，アプリ間の複雑な結合を可能にした．本研究では，Uni-messe を用いたスマートサービスの実装例として 2 種類のデモアプリの実装を示した．

今後の課題としては，Uni-messe を機能拡張することによって，さらに複雑なサービス連携を可能にする必要がある．具体的には，複数イベントを統合・連携して評価することによって，例えば別サービスとして提供される 2 つ以上のセンサの値をもとに他のサービスを動かす，といった処理を実現可能にしたいと考えている．また，今回の研究ではイベントにおける Why の評価を行わなかったが，Why を活用して時相論理（テンポラルロジック）の評価を行うことができると考えている．これにより，過去に発生したイベントの内容によって処理の分岐を変化させるようなイベント時系列に応じたサービス連携が可能になる．

文　献

[1] “My applets - ifttt,” https://ifttt.com/home. (Accessed on 08/01/2021).

[2] H. Takatsuka, S. Saiki, S. Matsumoto, and M. Nakamura, “RuCAS: Rule-based framework for managing context-aware services with distributed web services,” International Journal of Software Innovation, vol.3, no.3, pp.57–68, July 2015.

[3] C. Miura, H. Maeda, S. Saiki, M. Nakamura, and K. Yasuda, “Implementing mind monitoring service for elderly people at home using line chatbot,” ISG’s 12th World Conference of Gerontechnology, vol.19(suppl), pp.••–••, Oct. 2020. Online.

[4] 明石拓弥，佐伯幸郎，中村匡秀，安田清，“在宅高齢者を支援する個人適応型スピーカーサー ビスの提案，” 日本認知症予防学会学術集会プログラム・予稿 集認知症予防学会，pp.••–••June 2021.

[5] “Nature remo（ネイチャーリモ），” https://nature.global/nature-remo/. (Accessed on 08/01/2021).

[6] “The open source platform for our smart digital future - fiware,” https://www.fiware.org/. (Accessed on 08/01/2021).

[7] H. Maeda, S. Saiki, M. Nakamura, and K. Yasuda, “Log4Care: Unified event logging service for personalized care,” Digital Human Modeling 2018 (DHM 2018), Held as Part of HCI International 2018, vol.LNCS 10917, pp.466–477, Springer, July 2018. Las Vegas, USA.

[8] “Switchbot（スイッチボット）| スマートホームにらくらくスイッチ，” https://www.switchbot.jp/. (Accessed on 08/01/2021).

[9] R. Iqbal, T.A. Butt, M.O. Shafiq, M.W.A. Talib, and T. Umar, “Context-aware data-driven intelligent framework for fog infrastructures in internet of vehicles,” IEEE Access, vol.6, pp.58182–58194, 2018.

[10] J. Kim, S. Hong, S. Hong, and J. Kim, “Context-aware pub/sub control method using reinforcement learning,” Concurrency and Computation: Practice and Experience, vol.33, no.15, p.e5727, 2021. https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5727