統合モビリティサービスの
設計と運用に関する研究

提出先　　大阪大学大学院情報科学研究科

提出年月　2023 年 1 月

平島 陽子

研究業績

A. 学術論文誌論文

1. 沖津潤，平島陽子，朝康博，加藤猛，齊藤達也，“環境配慮型データセンタ向け空調連係 IT 負荷配置最適化方式”，電子情報通信学会論文誌 D, Vol.J95-D, No.4, pp.919-927, 2012.

2. 平島陽子，山崎謙太，森村知弘，薦田憲久，“応答性能保証率向上のためのハイブリッドオートスケール方式”，電気学会 C 部門論文誌, Vol.137, No.3, pp.521-531, 2017.

3. 平島陽子，薦田憲久，藤原融，“統合モビリティサービスの実現に向けたスキーママッチング手法の提案”，電気学会 C 部門論文誌, Vol.139, No.6, pp.686-691, 2019.
   英訳版 : Yoko Hirashima, Norihisa Komoda, and Toru Fujiwara, “A Schema Matching Approach for Integrated Mobility Service”, Electronics and Communications in Japan, Vol.102, Issue 9, pp.12-18, 2019.

B. 国際会議

1. Yoko Hirashima, Kenta Yamasaki, and Masataka Nagura, “Proactive-Reactive Auto-Scaling Mechanism for Unpredictable Load Change”, in Proc. of 4th IIAI International Congress on Advanced Applied Informatics 2016 (IIAI AAI 2016), pp.861-866, 2016.

2. Yoko Hirashima and Norihisa Komoda, “Parameter Optimization for Hybrid Auto-Scaling Mechanism”, in Proc. of 17th IEEE International Symposium on Computational Intelligence and Informatics (CINTI 2016), pp.111-116, 2016.

C. 国内会議

1. 平島陽子，木元賢司，小泉稔，河部展，“QoS 保証型サービスを提供する IP プロビジョニングサーバの検討”, 電子情報通信学会技術研究報告, SSE2000-172, Vol.100, No.451, pp.53-58, 2000.

2. 平島陽子, 片岡健二, 小泉稔, “カスタマーセルフコントロール QoS 制御機能の検討”, 電子情報通信学会技術研究報告, TM2002-45, Vol.102, No.463, pp.11-16, 2002.

3. 永井崇之，名倉正剛，中島淳，平島陽子，森村知弘，“IT システム向け障害対処プラン自動生成システムの検討”，電子情報通信学会技術研究報告，ICM2012-80，Vol.112, No.492, pp.125-130, 2013.

4. 山崎謙太，加藤真理子，平島陽子，森村知弘，“IT システムの障害対応効率化に向けたネットワークトラヒック可視化技術の検討”，電子情報通信学会技術研究報告，ICM2013-24, Vol.113, No.294, pp.7-12, 2013.

5. 山崎謙太，寺村健，平島陽子，“オートスケール時の SLA 保証改善に向けた仮想マシン CPU キャッピング機能の活用に関する検討”，電子情報通信学会技術研究報告，ICM2014-55, Vol.114, No.523, pp.7-12, 2015.

6. 平島陽子，矢野浩仁，橋本博文，牛山純子，花房比佐友，“都市交通シミュレーションのためのオープンデータマッピング手法”，電気学会第 73 回情報システム研究会，IS-18-001, pp.1-5, 2018.

7. 平島陽子，薦田憲久，藤原融，“位置情報を活用した移動需要推定方法”，電気学会第 77 回情報システム研究会，IS-19-009, pp.41-44, 2019.

8. 平島陽子，播金真奈，吉治季恵，廣井和重，”ナッジ応用行動誘導方式の福岡における実地実証評価”，電気学会第 86 回情報システム研究会，IS-21-0038, pp.35-39, 2021.

9. 平島陽子，今村将司，藤原融，薦田 憲久，“ポリシーベース移動需要マネジメント方式の提案”，情報処理学会研究報告 高度交通システムとスマートコミュニティ（ITS），2022-ITS-91(12), pp.1-7, 2022.

D. その他

1. 小泉稔，三宅滋，平島陽子，“ポリシーベースによる QoS 制御”，オーム社, 2001. (2.4 節「LDAP によるポリシーリポジトリの実現」(pp.90-95)，5.1 節「インターネットセキュリティへの展開」(pp.212-225)，5.3 節「QoS 保証型 VPN を実現する MPLS 管理への展開」(pp.240-248) 分担).

2. 青木篤，勝俣修，服部泰明，平島陽子，“ブロードバンドシステムを支える統合運用管理ソフトウェア”，日立評論, 2002 年 10 月増刊号，pp.37-40, 2002.

3. 齋藤達也，平島陽子，山村英穂，吉田伴博，“環境配慮型データセンターを支える省電力化技術”，日立評論，2010 年 5 月号，pp.56-59, 2010.

内容梗概

本論文は，筆者が 2012 年から現在まで（株）日立製作所研究開発グループ，ならびに，2017 年から 2020 年まで，大阪大学大学院情報科学研究科マルチメディア工学専攻在学中に行った，統合モビリティサービスプラットフォームに関する研究成果をまとめたものである．

都市における渋滞・混雑の緩和と，地方における公共交通手段の維持は大きな課題であり，これら課題を解決する方策として Mobility as a Service (MaaS) が注目されている．MaaS は，人々の移動ニーズに合わせて複数の交通サービスを連携させ，統合されたサービスとして提供するものであり，公共交通の利便性を高める手段として期待されている．MaaS のような統合モビリティサービス実現のためには，まず，対象地域で提供される交通サービスの情報を収集，集約し，利用者の移動ニーズに応じて 1 つのサービスとなるよう組み立て，携帯端末のアプリケーション等を通じて案内を提供する必要がある．

このような背景を踏まえ，本論文では以下の 3 点の課題に着目する．第 1 点目は代表的な案内である経路検索機能をカスタマイズする誘導案内ポリシー記述方式である．Covid-19 による外出抑制を契機としてテレワークの普及が進んだが，都心部公共交通のピーク時混雑は解消されていない．その一方で，全体的な乗客数は減少しており，閑散時間帯の公共交通利用促進や駅付帯施設への誘客を含む移動需要マネジメントが交通事業者の課題となっている．そこで，本論文では，混雑回避や店舗への来訪を促すために，利用者の行先や混雑状況等に応じて提示する経路情報を変えるポリシーベースの経路検索機能に着目した．そして，経路検索機能をカスタマイズする誘導案内ポリシーの記述方式として，誘導案内ポリシーの文法と記述支援方式を提案した．さらに，提案する記述支援方式による記述工数の評価実験を行い，従来方式よりも記述工数を削減できることを示した．

第 2 点目は交通サービスの情報の統合である. 複数の交通サービスを連携させ統合モビリティサービスとするには, 交通サービスの運行計画や料金などの情報を事業者から取得し，それらを統一した形式に変換する必要がある．異なる形式を持つデータ同士を対応づける技術として，スキーママッチング手法が研究されてきたが，既存研究が扱うデータ構造は実際の交通サービスのデータよりも単純であり, 交通サービスに適用する場合には精度が課題になる．そこで，本論文では，あるスキーマでは 1 つのオブジェクトに含まれる情報群が, 他のスキーマでは参照関係でつながった複数のオブジェクトに分かれるというデータの特徴を利用して, 事業者固有形式のデータを統一された形式へ対応づける手法を提案した．そして，実オープンデータを用いて提案手法のマッチング精度を評価し, 従来手法である属性名によるマッチングと比較して精度を向上できることを示した.

第 3 点目として，統合モビリティサービスが稼働する IT (Information Technology) 基盤に着目する．統合モビリティサービスは，利用者が移動したい時に，即時にニーズに合った統合モビリティサービスを組み立てて提供する応答性能が求められる．特に，多くの人が一斉に移動する朝夕や, 代替経路を検索する運転見合わせ発生時などは多数のリクエストが集中する．応答性能を保証するには，IT 基盤に潤沢なリソースを割り当てればよいが，その場合はリソースのコストが問題となる．負荷に応じて動的に IT 基盤のリソースを調整する方法としては，仮想サーバを追加するスケールアウトと，仮想サーバに仮想 CPU（Central Processing Unit）やメモリなどのリソースを追加するスケールアップ方法が提案されている．しかし，スケールアウトには仮想サーバの起動に時間がかかるという課題があり，スケールアップについては，追加は迅速だが，仮想サーバの起動前に設定したリソース量しか追加できないという課題がある．そこで，本論文では, 負荷増大の傾向に応じてスケールアウトとスケールアップを使い分けることで応答性能劣化を防ぎ, スケールアップ可能なリソース量を維持するハイブリッドオートスケール方式を提案した．そして，提案方式について複数の負荷パターンに対する評

価を行うことによって, 変動の大きい負荷パターンに対しても応答時間保証率を向上できることを示した.

本論文は全 5 章で構成する. 第 1 章の序論では, 統合モビリティサービスのプラットフォームの概要を述べ, その実現における課題について述べる. さらに, それらの課題に対する従来研究を概観し, 本研究の位置づけを明らかにした上で, 本研究の方針を述べる. 次に, 第 2 章から第 4 章では, 前述した 3 点の課題に対して提案を行う. 最後に, 第 5 章では, 結論として本研究で得られた成果を要約し, 今後に残された課題について述べる.

目次

第1章　序論

1.1　研究の背景

MaaS (Mobility as a Service) と呼ばれる交通サービスの提供形態が広がりつつある[1][2][3][4]．MaaS は，利用者から見れば，タクシー，カーシェア，電車など多様な移動手段をアプリから一元的に検索し，予約，決済できるようにするサービスである．MaaS アプリにより自宅や勤め先などの出発地からの最寄り駅，または，降車駅から目的地までの交通手段が手配できれば，公共交通が利用しやすくなり，利用者増にもつながる．一方，交通事業者から見れば，MaaS アプリは自社の交通サービスの情報を利用者に提供するタッチポイントであり，混雑予測や回避の案内を利用者に届ける手段になる．

本研究では，このような複数の交通サービスをひとまとまりのサービスとしたものを統合モビリティサービスと呼び，その基盤となる情報システムを統合モビリティサービスプラットフォームと呼ぶ．図 1-1 は，統合モビリティサービスプラットフォームの機能の概要を示すものである．統合モビリティサービスプラットフォームの機能は，サービス統合機能とデータ統合機能に分かれる．サービス統合機能は，利用者の要求に応じて旅程を作成し，乗車方法や混雑・遅延情報などを提示する案内機能，複数の移動サービスの予約や決済を一元化する予約統合機能や決済統合機能から成る．データ統合機能は，各交通事業者が公開する運行計画情報など，異なるデータソースから取得した事業者固有形式の交通サービスデータを共通処理可能な形式に統一して，案内機能や予約統合機能，決済統合機能に提供する[5][6]．

そして，これらの機能は，サーバやストレージ装置がネットワークによって接続された IT (Information Technology)システム上に実装され，案内機能や予約統合機能がインターネットを介して利用者に提供される．これらの装置の稼働状況を監視して安定的で安全なサービスを提供するためには，IT システムの運用管理を行う IT マネジメント機能が必要である．IT マネジメント機能は，性能管理，セキュリティ管理，可用性・障害管理機能から構成される[7][8]．

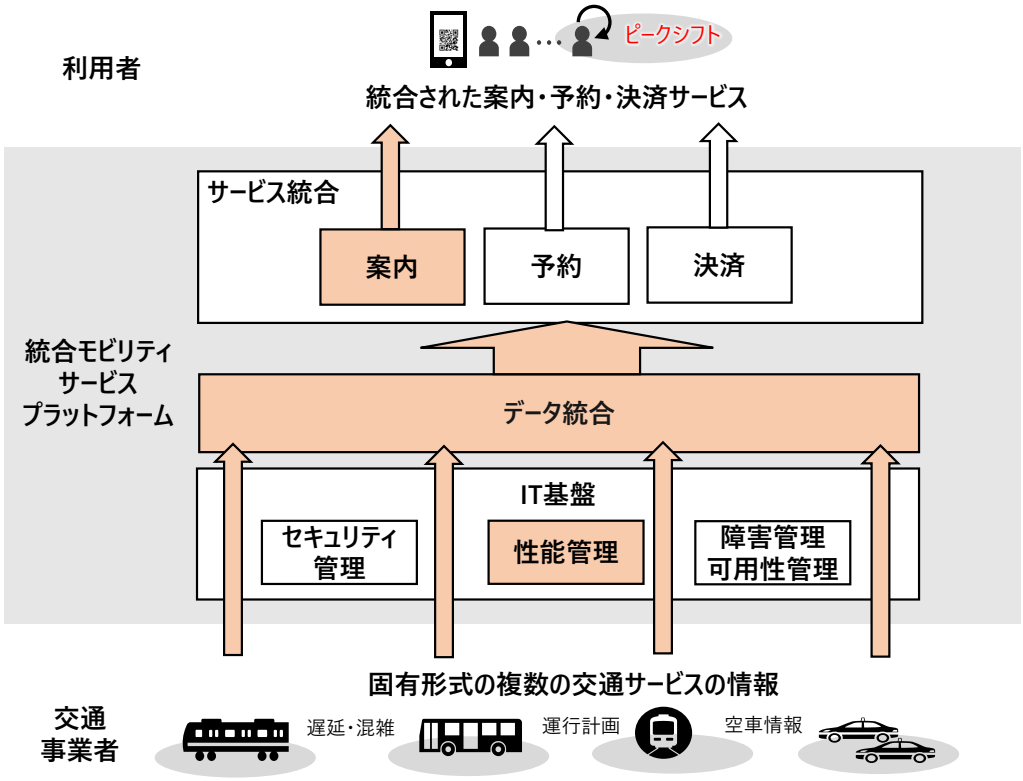

図 1-1　統合モビリティサービスプラットフォーム

統合モビリティサービスプラットフォームの案内機能は，与えられた出発地点から到着地点まで公共交通を利用して移動するための交通便や乗り場，発車時刻などを利用者に届ける．このような案内機能は，既に経路検索サービスとして広く利用されている．経路検索サービスは，一般的には最も早く到着する経路や，最も運賃が安い経路など一般的な観点で良いとされる経路情報を返す．一方で，公共交通サービスにおいては，ピーク時の混雑緩和や閑散時間帯の公共交通利用促進，付帯施設への誘客が課題となっており[9][10][11][12][13]，今後は混雑の回避を促すほか，催事場のある駅に人を集めるなどの移動需要マネジメントに経路検索サービスを活用していくことが期待される[49][58][59]．このため，交通事業者等が混雑やプロモーション等の特別な事情に合わせてカスタマイズできる経路検索機能の実現が課題となる．

また，複数の交通サービスを連携させ統合モビリティサービスとするには，各事業者の交通サービスの運行計画や料金などの情報を集約する必要がある．しかし，日本の公共交通は，各地域の交通行政や交通事業者が個別に発展させてきたものであるためサービスの情報は独自形式で管理されている[14][15]．したがって，独自のデータ形式から

共通のデータ形式への変換が必要になるが，人手で鉄道，バス，カーシェア，駐車場などさまざまなデータを複数のサイトから取得し，共通形式へ変換する負荷は高く，機械的な対応づけが課題となる．

次に，統合モビリティサービスプラットフォームの IT 基盤アーキテクチャに着目する．統合モビリティサービスは，複数の交通事業者の提供するサービス情報を集約し，多数の利用者のリクエストを取扱うスケーラビリティが求められる．さらに，利用者が移動したい時に，即時にニーズに合った統合移動サービスを組み立てて提供する応答性能が必要である[19]．このためには，プラットフォームに潤沢なリソースを割り当てればよいが，その場合はリソースのコストが問題となる．特に，移動需要は時間的な偏りが大きいので，多くの人が一斉に移動する朝夕や，代替経路を検索する運転見合わせ発生時などは多数のリクエストが集中する一方，平常時の昼間などは比較的リスクエストは少ないと予想される．異常時にこそ，統合モビリティサービスプラットフォームは適切な移動手段の情報を提供すべきであるが，不特定多数の利用者がアクセスするプラットフォームにおいて性能保証のため確保すべきリソース量を予測することは難しく，性能保証とコスト低減の両立が課題となる．

以上から，本研究では，以下の課題を解決することを目的とする．

課題(1) 外部定義により提示する情報を変更可能な経路検索機能の実現

課題(2) 固有形式の交通サービスのデータを共通形式に変換する方法

課題(3) 応答性能保証とコスト低減を両立させる IT リソース調整方法

1.2 関連研究

1.2.1 外部定義情報に基づくプログラム動作変更手法の関連研究

経路検索機能をカスタマイズ可能にする方法としては，経路検索機能に外部から設定情報を入力し，設定情報に基づいて経路検索結果を変更できるようにする方法が考えられる．情報システム運用管理の分野においては，外部から与えるポリシーと呼ばれる設定情報によってオペレーション自動化や情報ネットワーク機器における QoS (Quality of Service) 制御やアクセス制御を変更する技術が研究されている[20][21][22][23][24][25][26][27][28]．ポリシー記述言語としては，情報システム管理における標準化団体である DMTF (Distributed Management Task Force) で定義された

CIM-SPL (CIM-Simplified Policy Language) や，IETF (Internet Engineering Task Force) で定義された PCIM (Policy Core Information Model) などがある[29][30][31]. これらのポリシー記述言語では，ポリシーの動作条件であるコンディション，その条件を満たした場合に実行する対処アクションを IF-THEN 形式で記述する．ここで，動作条件は情報システムの応答時間や CPU 利用率のしきい値，ネットワーク装置が受信するパケットに付与されているフラグの値などを用いて記述される．対処アクションは，情報システムの構成や設定を変更するための操作であり，標準化団体によって定義されたコマンドや，管理対象機器の固有のコマンドに渡すパラメータなどで記述される．

ただし，このような既存のポリシー記述言語を経路検索機能のカスタマイズに適用する場合には次のような課題がある．まず，混雑回避やプロモーションなどのさまざまな用途に応じたポリシーを設定できる構造と予約語を持つ記述方式が求められる．一方で，ポリシーは，交通事業者や商業施設の案内担当者が記述，変更できる平易なものであることが望ましい．そこで，記述力のあるポリシー記述方式を定義するという課題と，ポリシーの記述を容易化するという 2 つの相反する課題が生じる．しかし，このような記述能力を持ったポリシーに関する研究はこれまで行われてこなかった．

1.2.2 交通サービスのデータを共通形式に変換する手法の関連研究

異なるデータ形式を持つデータ同士を対応づけ，相互に変換可能にする技術として，スキーママッチング手法が研究されてきた．スキーママッチング手法は，属性名の類似度に着目する手法と，属性型や属性が必須かオプションか，シングルバリューかマルチバリューを許容するかといった制約に着目する手法に分類される[32][33]. これらの内，制約だけでマッチング候補を特定することは難しく，属性名を用いるのが基本的な手法である．さらに，属性名の類似度を求める手法には，単語の意味に着目する手法と，編集距離等を用いて文字列そのものの類似度を求める手法が提案されている[34]．ただし，属性名にどのような語彙を用いるかは設計者に依存しており，同じ属性でも同じ単語や類義語が用いられるとは限らない．特に，交通サービスのデータスキーマにおいては，例えば交通便を示す trip オブジェクトの属性として便の ID である trip_id，その便の路線を示す route_id，便が急行か普通かを示す service_id が含まれるケースがある．このような場合，id や name など複数の属性名に出現する単語が共通であれば文字列の

類似度は高くなるが，同じ属性である確率は低い.

上記の他に，多くの既存研究では，与えられた 2 つのテーブルの属性を対象とした 1 対 1 マッチングを扱っているが，交通サービスのデータ構造はより複雑である．交通サービスに関する情報は，スキーマによってどの情報を 1 つのオブジェクト，または，1 つの属性として定義するかがまちまちである. あるスキーマにおいて 1 つのオブジェクトを構成する情報が他のデータセットでは複数のオブジェクトに分散して記述される場合もあれば，1 つの属性に含まれる情報が他のデータセットでは複数の属性に分かれることもある．例えば，公共交通オープンデータ協議会スキーマ[1]では，時刻表オブジェクトの属性としてバス停情報とカレンダー情報が定義されているが, 標準的なバスフォーマット[2]では，カレンダー情報は時刻表ではなく，便の属性として表現されている．また，1 つのオブジェクトであっても，各属性がフラットに並ぶ構造のものと，他オブジェクトの配列を属性として持つ入れ子構造のものがある．しかし，多くの既存研究では，このような複雑な構造を持つオブジェクトを考慮していない.

1.2.3 応答性能保証とコスト削減を両立させる IT リソース調整方法の関連研究

応答性能の保証とコスト低減の両立を目的として, 負荷に応じ動的に情報システムのリソースを増減させる方法として，オートスケールが研究されてきた．Galante らのオートスケール方式の分類によると，オートスケール方式は，予測型（Predictive）方式と反応型（Reactive）方式とに分類される[35]．予測型方式の研究としては，Mao らは，タスクスケジューリングから必要とするリソース量を算出し，オートスケールを実施する方法を提案しているが, 仮想サーバの起動に時間がかかりジョブのデッドラインを守れない場合があることを述べている[36]．一方，Gong らは，高速フーリエ変換を用いることで，対象システムの繰り返し負荷パターンを導出し，リソース割当の無駄を削減している[38]．また，Hanai らは シミュレーションを実施することで，クラウドへのリソース追加必要量を予測する手法を提案している[40]．これら予測型方式の既存研究では，スケール変更をリソース量の予測に基づき推定する研究が多数見られるが,

1 https://www.odpt.org/
2 https://www.gtfs.jp/

Shen らは，負荷履歴から変動を予測しきれないことを指摘している[41].

一方，反応型方式の既存研究として，Han らは，既存技術であるスケールアップとスケールアウトとを組合せるアルゴリズムを提案している[42]．この提案では，負荷増大検知時にはまずスケールアップを実行し，それでもリソースが不足する場合にスケールアウトを実行することでサービスレベルを改善する．また Deng らは，スケールアップ実現する場合に，リソース利用上限の値を一般的な負荷予測モデルを利用して仮想サーバ起動前に適切に設定する手法を提案している[43].

このように，反応型の既存研究では，突発的な負荷変化の発生時にスケールアップを適用するが，その場合でもリソース利用上限の値を仮想サーバ起動前に決定する必要がある．このため，仮想サーバ運用開始前に設定した利用上限値までスケールアップを実施すると，それ以降の負荷上昇に対応できなくなる．このように既存研究や既存技術では，突発的な負荷変動に対応することは困難である．

1.3　研究の方針

前節までに述べた課題を踏まえ，本研究における各課題の位置づけを図 1-2 に示す.

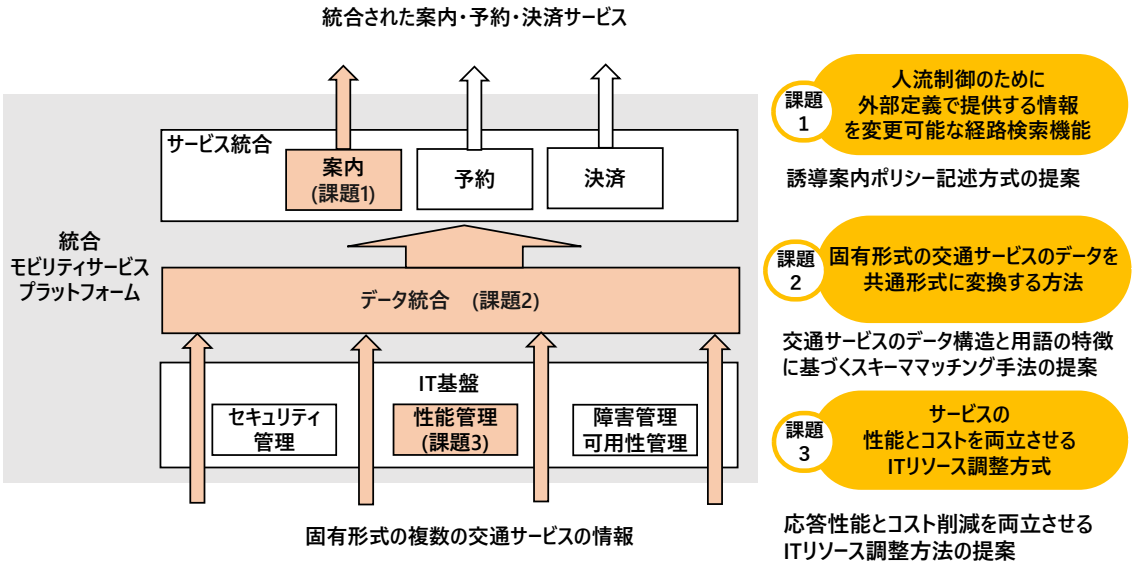

図 1-2　本研究における課題の位置づけ

(1) 誘導案内ポリシー記述方式の提案

外部定義により提示する情報を変更可能な経路検索機能の実現のために，経路検索機能をカスタマイズする誘導案内ポリシーの記述方式を提案する．誘導案内ポリシーを記述するための文法に加えて，IT に詳しいとは限らない交通事業者の案内担当者が誘導案内ポリシーを記述できるように，記述すべき項目数が削減できるテンプレートとテンプレートをあらかじめ登録した記述支援エディタを提案する．そして，記述支援方式について，被験者を用いて誘導案内ポリシーを記述してもらい，従来方式と記述の工数を比較する．

(2) 交通サービスのデータ構造と用語の特徴に基づくスキーママッチング手法

交通サービスのデータ構造では，マッチング対象となる 2 つのスキーマにおいて，あるスキーマでは 1 つのオブジェクトに含まれる情報群が，他のスキーマでは参照関係でつながった複数のオブジェクトに分かれることが多い．例えば，公共交通オープンデータ協議会スキーマ[3]では，バス停とカレンダー情報が時刻表の属性として定義されている．一方で，標準的なバスフォーマット[4]では，バス停は時刻表の属性であるが，カレンダー情報は時刻表と参照関係にある便の属性として表現されている．この特徴を考慮し，マッチング元オブジェクトと参照関係にある他オブジェクトをマッチングの対象に加えることで，マッチングの再現率を向上させるスキーママッチング手法を提案する．例えば，公共交通オープンデータ協議会スキーマの時刻表を，標準的なバスフォーマットの時刻表に変換する場合には，時刻表と参照関係を持つ便もマッチングの対象に加える．さらに，参照元と参照先の属性名を接続した拡張属性名を対象とした属性名の類似度によるマッチングと，属性型別の属性値類似度マッチングを組合せて適用することで適合率を高める．そして，実オープンデータを用いて提案手法と既存手法のマッチング精度を比較評価する．

(3) 応答性能とコスト削減を両立させる IT リソース調整方法

システムを構成する全ての仮想サーバが，その利用上限値までスケールアップを実行した後にも，システム全体へ迅速にリソース追加可能にするハイブリッドオートスケール方法を提案する．提案方法では，既存手法で提案されているようにスケールアウトと

3 https://www.odpt.org/
4 https://www.mlit.go.jp/sogoseisaku/transport/sosei_transport_tk_000067.html

スケールインを使い分けるだけでなく, 既にスケールアップ済であり更なるリソース追加が不可能な仮想サーバを削除し, 代わりにリソース追加が可能な仮想サーバを追加することで, 常にシステムへの迅速なリソース追加ができる仮想サーバが存在するよう制御する. そして, 提案方式に対して複数の負荷パターンに対する評価を行うことによって, 方式の特性や限界, パラメータとその設定方法を明らかにする.

1.4 本論文の構成

本論文では, 第 2 章以降を以下のとおり構成する.

第 2 章では, 文献[47] [48][49][50]に基づき, 経路検索機能をカスタマイズするための誘導案内ポリシー記述方式に関する課題を述べる. そして, 誘導案内ポリシーの文法と, テンプレートにより記述の工数を削減する記述支援方式を提案し, 記述支援方式について工数の削減効果を評価した結果を示す.

第 3 章では, 文献[51][52][53]に基づき, モビリティサービスに関するデータのスキーママッチングにおける課題を述べる. 次に, モビリティサービスに関するデータの特徴を述べ, その特徴を活用することでマッチング精度を高めるスキーママッチング手法を提案する. そして, 提案手法の精度向上効果について, モビリティサービスに関するオープンデータを用い評価を行う.

第 4 章では, 文献[54][55][56][57]に基づき, 既存のオートスケール手法の課題を述べる. そして, 応答性能保証率向上とコスト削減とを両立するハイブリッドオートスケール方式を提案する. さらに, 提案するハイブリッドオートスケール方式について, 平常時負荷パターン, バースト負荷パターンの 2 通りの負荷パターンについて応答性能保証率とリソース利用率の 2 つの観点から評価を行う.

最後に, 第 5 章では, 結論として本研究で得られた成果を要約し, 今後に残された課題について述べる.

第2章　誘導案内ポリシー記述方式

2.1　緒言

本章では，交通事業者等が工事やプロモーション等の特別な事情に合わせて，利用者に推奨経路を提供する経路検索サービスを対象とする．まず，利用者に提供する情報をカスタマイズするため，ポリシーベースの経路検索システムを考え，ポリシー記述とその記述支援について述べる．

経路検索サービスは，目的地，出発地，出発または到着時刻等から成る経路検索条件を満たす経路情報の集合を利用者に提供する．ここで，経路情報は，与えられた出発地点から到着地点まで公共交通を利用して移動するための交通便や乗り場，発車時刻などから成る．以降，経路検索条件を満たす経路情報を「候補経路情報」と呼ぶ．

通常，この候補経路情報は，最も早く到着する経路や，最も運賃が安い経路など一般的な観点で良いとされる経路情報で構成される．これに対して本章で扱う案内では，与えられた経路検索条件に，交通事業者が工事などの事情に合わせて利用者を誘導するために特別な経路検索条件を加えて検索した経路情報を提供する．例えば，工事中で設備の一部が使用できない駅がある場合には，その駅での乗換えを避けるような経路検索条件を加える．また，店舗に誘導したい場合，その店舗がある駅を通ることを経路検索条件に加えて，その経路検索条件で得られた経路情報に立寄って欲しい店舗の施設情報を追加して出す．このため，本章では，経路情報には，経路上の店舗情報なども含まれるものとする．

以降，通常の経路検索で得られる経路情報を「通常経路情報」，特別な条件を加えて検索した経路情報を「誘導経路情報」とし，この 2 種類の経路情報を組にして候補経路情報として提供することを「誘導案内」と呼ぶ．

誘導案内で加味する経路検索条件は，事業者毎に異なるのはもちろん，短期間で変更される．このために，交通事業者等が工事やプロモーション等の特別な事情に合わせて，経路検索サービスをカスタマイズして誘導案内情報を出す仕組みを構築する．カスタマイズとは，利用者が入力する経路検索条件，および，その経路検索条件で得られる通常経路情報と，工事などの事業者の事情に応じて，異なる誘導案内情報を出せるようにす

ることである．

このようなカスタマイズの方法としては，サービスを実現するプログラムに，外部からポリシーと呼ばれるルールを与えて，ポリシーに基づいて動作を変えられるようにすることが一般的である．ポリシーでは，プログラムの動作（アクション）と，その動作を実行する条件（コンディション）とが定義されており，ある条件が満たされた場合に，その条件と対応づけられた動作が実行される．

このカスタマイズ手法を誘導案内に当てはめると，利用者から与えられた経路検索条件と，その検索結果である通常経路情報が特定の条件を満たすかをコンディションとして，どのような特別な経路検索条件を追加するかをアクションとして定義したポリシーを定義すればよい．このポリシーを「誘導案内ポリシー」，特別な経路検索条件を「誘導案内検索条件」と呼ぶことにする．誘導案内システムは，ポリシーに記述された「誘導案内検索条件」を使って，経路検索サービスから経路検索条件に応じた誘導経路情報を取得することができる．

ただし，誘導案内にはさまざまな用途が考えられるため，多様なユースケースに対して誘導案内ポリシーを設定できる構造や予約語を持った記述方式が求められる．一方で，誘導案内ポリシーは，交通事業者や商業施設の案内担当者が記述，変更できる平易なものであることが望ましい．そこで，多様なユースケースに対応できる記述力のある誘導案内ポリシー記述方式を定義するという課題と，誘導案内ポリシーの記述を容易化するという 2 つの課題が生じる．しかし，このような記述能力を持った誘導案内ポリシーに関する研究はこれまで行われてこなかった．

そこで，本章では，経路検索サービスをカスタマイズする誘導案内ポリシーの文法と誘導案内ポリシー記述を支援する方式を提案する．

以下，2.2 節で誘導案内ポリシー記述とその課題について説明し，2.3 節で誘導案内ポリシーの文法を，2.4 節で誘導案内ポリシー記述支援方式を提案する．そして，2.5 節で提案手法に基づいた誘導案内ポリシー記述支援方式の評価と考察を述べ，2.6 節で結果をまとめる．

2.2　誘導案内ポリシー記述の課題

2.2.1　誘導案内システムの構成と誘導案内ポリシーの記述内容

誘導案内システムの構成を図 2-1 に示す. 誘導案内システムは, 利用者が操作するスマートフォン等の端末側の誘導案内アプリケーションと, サーバ側で誘導案内情報を生成する誘導案内プログラムから構成される.

利用者が, 到着地や出発時刻などの経路検索条件を入力すると, 誘導案内アプリケーションは誘導案内プログラムにその経路検索条件を渡し, 誘導案内プログラムは, その経路検索条件を満たす通常経路情報に, 誘導案内経路情報を追加して候補経路情報を作成し誘導案内アプリケーションへ返す.

通常経路情報や誘導案内経路情報は, 商用化されており実績のある経路検索サービスや施設検索サービスを利用して取得する. 経路検索や施設情報検索の実現には, 複数交通事業者の運行計画や, 広範囲な店舗情報を収集する必要があり情報収集と整理に人手と時間を要する. また, 運行計画や店舗情報は変更されるので定期的に更新しなければならない. そこで誘導案内システムは, 正確な情報を出すため, また, 一定のコストと期間で開発するために経路や施設の検索に商用のサービスを利用する. このため, それらのサービスに対して Web API (Application Programming Interface) を発行する通常経路検索部, 誘導経路検索部, 誘導施設検索部を内部に持つ.

誘導経路検索部, 誘導施設検索部が, どのような情報を取得して誘導案内経路情報とするかは,「誘導案内ポリシー」によって制御する.「誘導案内ポリシー」は, 直感的に理解しやすい IF-THEN ルールの集合とする. このルールでは, 誘導案内経路情報を利用者の行先や目的地に応じたものにするため, IF に続くコンディションでは利用者が与えた経路検索条件, その検索結果である通常経路情報に対する条件を定義する. そして, THEN に続くアクションでは, どのような条件を満たす経路情報や施設情報を追加するかを定義する. これが誘導案内検索条件である.

なお, 単に避けて欲しい駅や利用して欲しい経路を指定するのではなく, IF-THEN のルールとする理由は, 移動時間帯や目的地などで誘導案内を出し分けられるようにするためである. これにより, 例えば, 2 路線間の乗換が 2 つの駅 A と駅 B で可能な場合に, 人流が分散するよう利用者の到着地によって乗換えを促す駅を分けることができる.

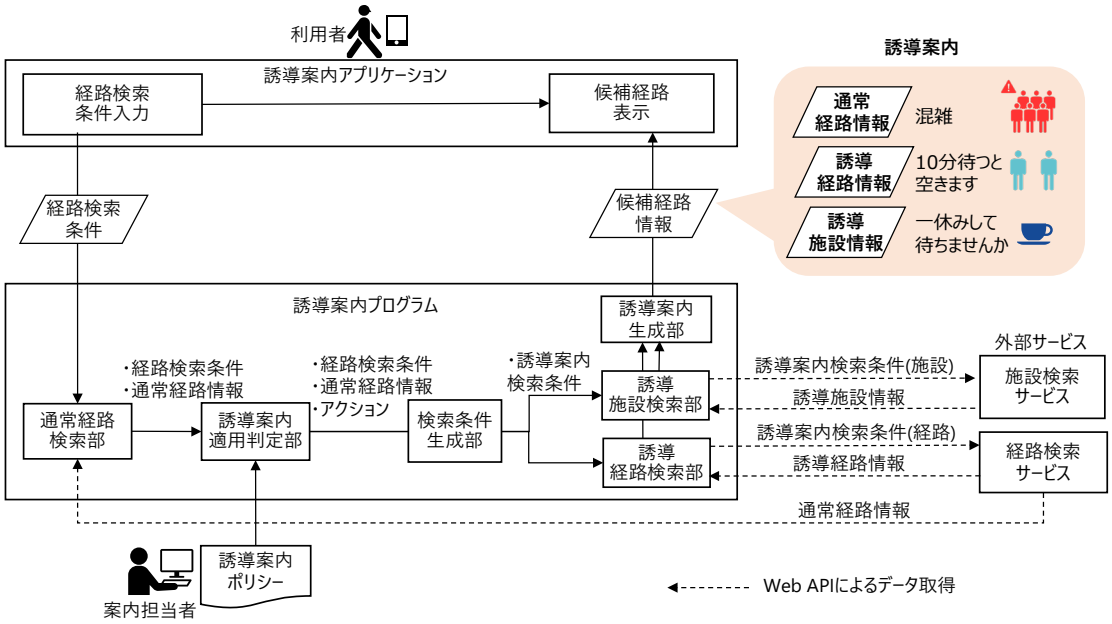

図 2-1 誘導案内システムの構成

誘導案内プログラムの適用判定部は，この誘導案内ポリシーを読み込み，コンディションと，利用者が与えた経路検索条件および通常経路情報とを突合せ，コンディションが満たされる場合には，そのポリシールールのアクションを検索条件生成部に渡す．

検索条件生成部は，アクションに定義された路線や駅の識別子などから誘導案内検索条件を生成し，誘導経路検索部に渡す．誘導経路検索部は経路検索サービスが提供する Web API を用いて誘導経路情報を取得する．同様に，誘導施設検索部は，施設検索サービスから誘導施設情報を取得する．誘導案内検索条件に経路と施設両方の検索条件が定義されている場合の誘導経路情報は，経路情報に立寄り先の施設情報が追加されたものになる．

そして最終的に誘導案内生成部が，通常経路情報と誘導経路情報とを合わせた候補経路情報を誘導案内アプリケーションへ返す．誘導案内アプリケーションは，受け取った通常経路情報と誘導経路情報を比較できるよう，リスト表示やタブ切り替えで表示する．

なお，施設検索サービスも経路検索サービスと同様に Web API を提供しており，検索条件として対象エリアや施設のジャンル，予算などが与えられると，その条件を満たす商業施設や観光スポットなどの施設情報を返す．施設情報は，施設の名称，住所，ジャンル，予算，人気度，位置（緯度経度）などから構成される．一般的に，経路検索サー

ビスの検索の条件として使える項目の数は 20 項目から 30 項目程度，施設検索サービスでは 50 項目以上であり，項目ごとに指定の入力形式や取りうる値の制約がある．

2.2.2 課題

誘導案内にはさまざまな用途が考えられるため，多様なユースケースに対して誘導案内ポリシーを設定できる構造や予約語を持った記述方式が求められる．一方で，誘導案内ポリシーは，交通事業者や商業施設の案内担当者が記述，変更できる平易なものであることも求められる．

そこで，多様なユースケースに対応できる記述力のある誘導案内ポリシーを記述できるようにするという課題と，誘導案内ポリシーの記述を容易化するという相反する 2 つの課題が生じる．以下に，2 つの課題について説明する．

（1）誘導案内ポリシーの記述能力に対する課題

誘導案内には表 2-1 に示すような複数のユースケースが考えられる．これらは，交通分野の専門家へのヒヤリングにより得られたユースケースである．

それぞれのユースケースで，コンディションやアクションの誘導案内検索条件として指定する内容は異なる．例えば No.3 や No.4 では，混雑している経路を検索した利用者に対して誘導案内を出すため，コンディションでしきい値となる混雑度を指定できなければならない．No.6 や No.7 の指定の駅を回避するユースケースでは，コンディションで回避する経由地を指定できなければならない．また，No.2 では，アクションで経路に対する誘導案内検索条件として，グリーン車であることを指定できなければならない．No.1 では，施設に対する誘導案内検索条件として，目的地となる施設の識別子を，No.9 では，立寄ってもらいたい施設がある駅，施設の検索範囲となる駅から施設までの距離，ジャンルを指定できなければならない．

さらに，これらのユースケースから導かれる複数のコンディションとアクションの記述方式は，2.2.1 で述べた誘導案内プログラムの構成の制約を受ける．誘導案内は経路検索サービスの形態で提供し，外部の経路検索サービスを利用して実現する．このため，コンディションの入力情報は利用者が与えた経路検索条件と，その経路検索結果である通常経路情報である．コンディションはそれら入力情報に対する条件として記述できなければならない．そして，アクションの誘導案内検索条件は，外部サービスを使って情

報を取得できるように，外部サービスの検索条件に変換しうる記述とするか，外部サービスの検索結果と突き合わせることができるものでなければならない．

その上で，上記の記述内容は，利用者の個々の経路検索要求に依存しない共通の規則として記述できなければならない．

しかし，このようなアクションとコンディションを記述できるポリシーの記述方式はこれまで研究されてこなかった．

表 2-1 誘導案内のユースケース

No.	分類	ユースケース説明
1	経路	駅から離れた会場で大規模なイベントが開催される場合に，最寄駅から会場までの経路情報を提示
2		グリーン車・指定席に空席がある場合に，グリーン車・指定席を利用する経路情報を提示
3		ラッシュ時に混雑を避ける出発時刻の経路情報を提示
4		ラッシュ時に混雑を避ける経路情報を提示
5		運行見合わせ発生時に，払い戻し額を低減するため，代替交通手段として交通事業者が手配する乗合タクシーの経路情報を提示
6		大規模イベント開催時に，混雑が予想される会場最寄り駅での乗り換えを避ける経路情報と，その乗り換え駅での立寄り施設情報を提示
7		工事により駅設備の一部が利用できない場合に，その駅での乗り換えを避ける経路情報を提示
8		大規模イベント開催時に，混雑が予想される駅の利用を避けてもらうために，その駅を到着地とする利用者に，１駅手前で下車する経路情報を提示
9	施設	駅ナカや駅周辺の商業施設のオープン時に，その駅を利用する人に立寄りを促すため施設情報を提示
10		運行見合わせ発生時に，運行再開を待つ場所として，待合所や商業施設情報を提示

（2）誘導案内ポリシーの記述工数

記述工数として問題となるのは，次の 2 点である．まず，提示する経路情報や施設情報を絞り込むために，アクションにおいて経路検索条件や施設検索条件を複数書かなければならない点である．例えば，混雑回避のために空いている経路を案内する場合には，その経路上の店舗を同時に案内するなど経路案内と施設案内を組み合わせることがある．この場合，アクションには，経路検索条件と施設検索条件の両方を指定する必要があり記述の工数が増える．

もう 1 点は，誘導案内ポリシーの制御文の記述である．制御文とは，誘導案内ポリシーに含まれるポリシールールの区切りや，アクションとコンディションの区別を示す文字列である．

図 2-2 に既存技術による誘導案内ポリシー記述例を示す．この誘導案内ポリシーは，駅 ID が ST100 である駅を経由地とする経路の利用者を対象に，その経由地を回避する誘導案内を出すことを定義している．この記述例では，太字にて示す文字列が制御文である．なお，＃は，それ以降，改行までがコメントであることを示す．

このような制御文や検索条件を含む誘導案内ポリシー記述は，情報システムのコマンドを熟知している情報システムの運用管理担当者には難しいものではないが，案内担当者にとっては負担となり，記述の間違いも起こりやすい．

```
RULE: #1 つ目のポリシールールの開始
  CONDITION: # コンディションの開始
      # 駅 ID が ST100 である駅を経由地とする場合
      via = ST100
  ACTION: #アクションの開始
      # 出発地と到着地と出発時刻は不変で ST100 を経由しない経路検索条件
      URI=https://< URI for route search service >#経路検索サービス URI
      avoid_via= ST100 # 回避する駅 ID
      origin=ST200 # 出発地の駅 ID
      destination=ST300 #到着地の駅 ID
      departure_time=2022-01-14T10:30:00 # 出発時刻
RULE: #２つ目のポリシールールの開始
  CONDITION: # コンディションの開始
      # 混雑度が 80%以上の場合
      crowd_level = 80
  ACTION: #アクションの開始
      # 出発地と到着地と出発時刻は同じでグリーン車を使う経路検索条件
      URI=https://< URI for route search service >#経路検索サービス URI
      origin=ST200 #出発地の駅 ID
      destination=ST300 #到着地の駅 ID
      departure_time=2022-01-14T10:30:00 # 出発時刻
      service_class=green_car # グリーン車
```

図 2-2 既存技術のポリシー記述例

2.2.3 アプローチ

2.2.2 で述べた課題を解決するアプローチを説明する．まず，ユースケースから誘導案内ポリシーで指定すべき条件を洗い出し，誘導案内ポリシー記述の要件を定義する．そして，要件を満たす構造と予約語を決定し，誘導案内ポリシーの文法を定義する．この誘導案内ポリシーの文法はユースケースを網羅する記述力があるが，記述工数は大きい．

そこで，定義した文法を用いて，用途，すなわちユースケースを限定した誘導案内ポリシーのテンプレートを作成する．ユースケースを限定すればコンディションと誘導案内検索条件の一部はデフォルト値を設定できるので，必須入力項目を削減できる．さらに，テンプレートが登録されたポリシー記述支援エディタを提供する．

このようにすることで，誘導案内ポリシーを記述する代わりにテンプレートを選択し，限られたの誘導案内検索条件を設定すれば誘導案内をカスタマイズできるようになる．そして，テンプレートが適用できないユースケースに対しては，文法にしたがって誘導案内ポリシーを記述することで誘導案内を実現できる．

上記アプローチの他に，文法自体をユースケースに限定して簡易化する方法も考えられるが，既存ユースケースに当てはまらない誘導案内を出したい時には，文法に加えて，変更した文法を解釈するように検索条件生成部を変更しなければならない．このため，誘導案内ポリシー自体はユースケースを限定せずに書けるようにし，テンプレートと記述支援エディタで記述を簡単にするというアプローチを提案する．

以下に，誘導案内ポリシー記述の要件と記述支援の要点を述べる．

（1） 誘導案内ポリシー記述の要件

ユースケースとシステム構成から，誘導案内ポリシー記述の要件は次のように定義できる．

（要件１）コンディションでは，利用者が与えた経路検索条件と，その経路検索結果である通常経路情報に対する条件を記述できること．さらに，この条件は，ユースケースで指定すべき条件を網羅しており，かつ，一般的に経路検索サービスで用いられる経路検索条件と通常経路情報に含まれる情報であること．

（要件２）アクションでは，どのような条件を満たす経路情報や施設情報を追加するかを誘導案内検索条件として記述できること．さらに，この誘導案内検索条件は外部の経路検索サービスや施設検索サービスの検索条件に変換し得る条件であるか，または，経路検索サービスから得られる経路情報と突き合わせることで，誘導案内検索条件への適合・不適合が判断できること．

（要件３）誘導案内検索条件は，個々の利用者の経路検索に依存しない共通の規則として記述でき，かつ，固定の条件ではなく利用者が与える目的地や出発時刻に合わせて変更できること．

（要件１）と（要件２）で述べたユースケースで指定すべき条件について説明する．図 2-3 にユースケースから抽出したコンディションを整理する．コンディションは，時間帯，区間・サービス種別などの属性，交通サービスの状態に分類される．例えば，No.6 ユースケースでは，イベントにより駅混雑が予想される時にその駅での乗り換えを避ける経路を提示する．これには，時間帯と利用者の経由駅を指定できなければならない．このように指定すべき項目を洗い出してコンディションの予約語を定義する．

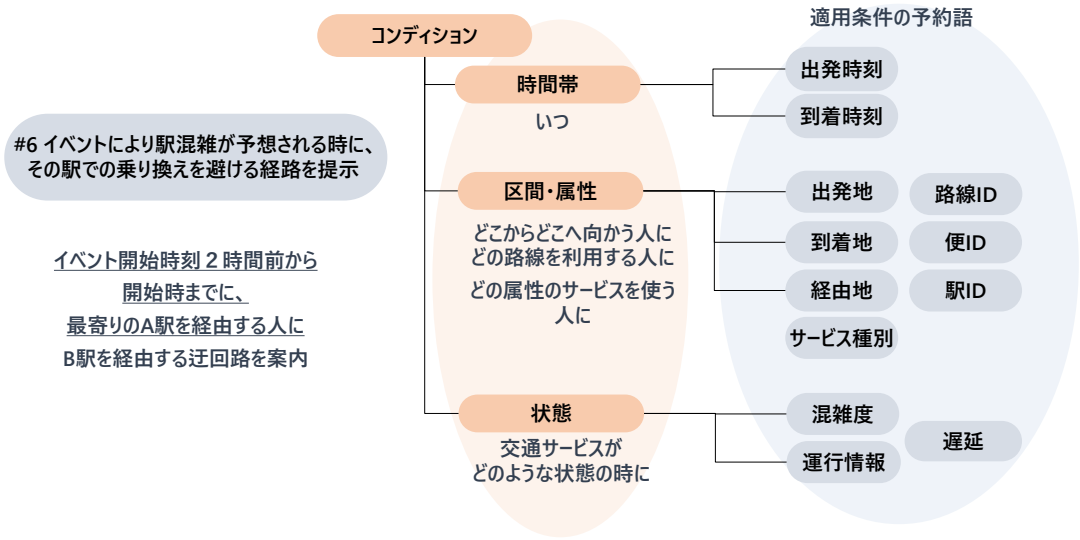

図 2-3 コンディションで指定すべき項目

図 2-4 に，アクションで指定すべき項目を示す．誘導案内は，経路または施設の回避または誘引を行うものなので，アクションは，回避と誘引のどちらを行うかを人流制御指示として，経路と施設のどちらを対象とするかを対象種別として指定し，さらにどのような経路や施設とするかを誘導案内検索条件で指定する．回避と誘引を区別するのは，回避の誘導案内を出す利用者を絞り込むためである．例えば，ある駅での乗り換えを回避して欲しい場合には人流制御指示に回避を指定すると，誘導案内適用判定部はその駅を経由する経路情報を検索した利用者に絞って誘導案内を出すように，コンディションに経由駅の条件を追加する．

さらに，利用者の目的地や出発時刻に合わせて利用を促す経路や施設を選ぶために，利用者の目的地や出発時刻を変数として記述できるようにする．

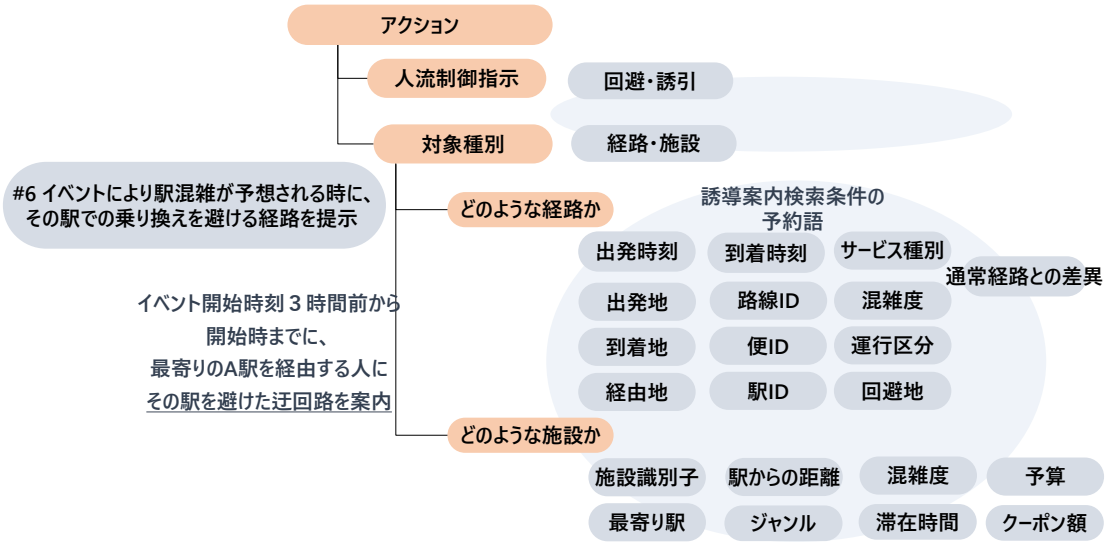

図 2-4 アクションで指定すべき項目

（2）ポリシー記述支援の要点

誘導案内ポリシー記述の工数が増大する原因は，経路や施設を絞り込むために複数の誘導案内検索条件を書かなければならない点，および，構造を定義する制御文を書かなければならない点である．そこで，ユースケースを限定したテンプレートを提案する．テンプレートは，構造を定義する制御文と，ユースケースから決まるコンディションや誘導案内検索条件のデフォルト値を備えた誘導案内ポリシールールのひな型である．こ

のテンプレートを用いて誘導案内ポリシーを記述できるように, テンプレートが組み込まれた専用の記述支援エディタを提供する.

テンプレートでカバーするユースケースは, 複数の交通事業者で共通するユースケースを選択した. さらに, ユースケースを整理し, 共通の指定項目で記述できるユースケースは 1 つのテンプレートで記述するようにした. 考えられる全てのユースケースについて個別にテンプレートを作成しない理由は，テンプレート数が増えると誘導案内ポリシー記述者がテンプレートを選択する負担が増えるからである.

2.3 誘導案内ポリシーの文法

2.3.1 誘導案内ポリシーの構造と予約語

(1) 構造

誘導案内ポリシーの構造を図 2-5 に，文法を図 2-6 に示す. 誘導案内ポリシーはポリシールールの集合であり, ポリシールールは 1 つのコンディションとアクションから構成される. コンディションは 1 つ以上のコンディション要素<Condition element 1>から成り，コンディション要素同士は OR 条件で結合される. コンディション要素の中はさらに 1 つ以上のコンディション要素<Condition element 2>に分かれており，それらは AND で結合される. コンディション要素<Condition element 1>と<Condition element 2>は，それぞれ項目名<Key>，演算子 <Operator>，値<Value>の 3 つ組で構成される. このようにすることで，複数の区間に対する条件を OR 条件で記述し，時間と場所を AND 条件で記述することができる. 例えば，複数の駅で工事が行われ，工事の時間帯が 10 時から 15 時であるような場合に，それらの駅で乗り換える人を対象として誘導案内を出すためには，経由駅と時間帯を AND で接続した条件を OR で接続することで，複数の対象駅と時間帯を指定したコンディションを定義する.

アクションは，1 つのポリシールールで，誘導経路情報を生成し，その経路上に施設情報を追加できるように，複数のアクション要素<Action element>を記述できるものとする. アクション要素は，ある経路や施設（対象種別）への，回避または誘引（人流制御指示）を行うものであるため，対象種別<Target>と人流制御指示<Direction>，経路や施設を特定するための誘導案内検索条件<Requirements>の集合から成るものとする. 誘導案内検索条件<Requirements>は経路と施設に関する条件があり，どちらも

項目名<Key>，演算子 <Operator>，値<Value>の 3 つ組で構成される．この誘導案内検索条件には，経路だけでなく施設の検索条件も含まれる．

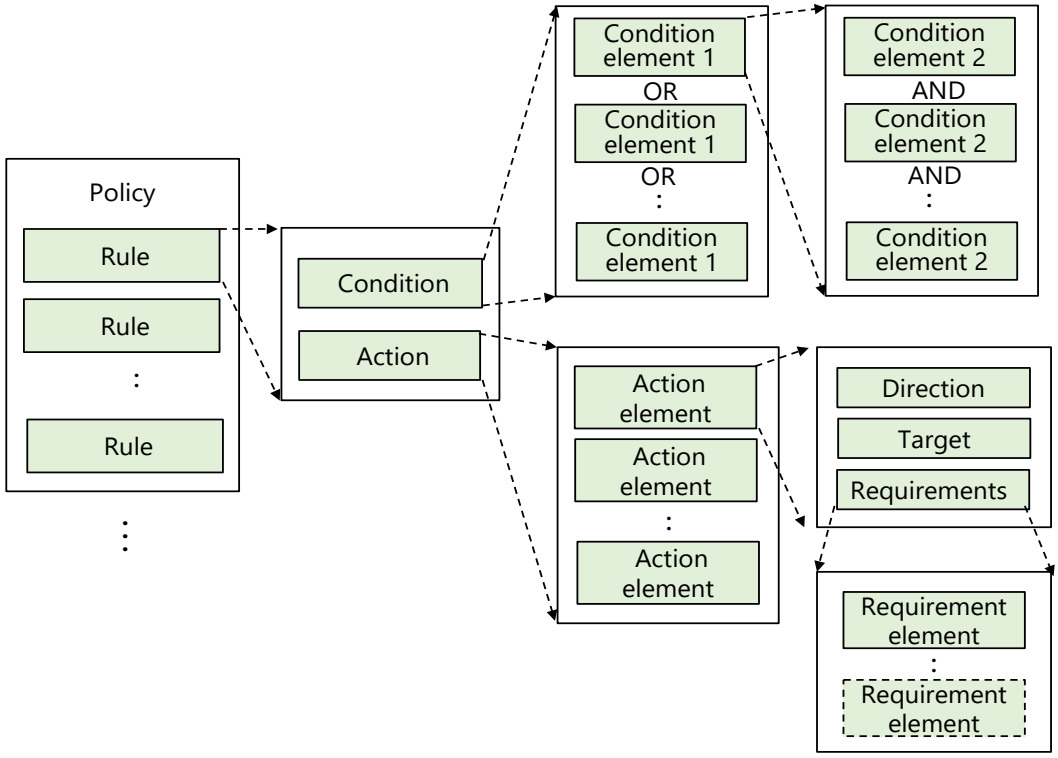

図 2-5 誘導案内ポリシーの構造

```
<Policy> ::= <Rule>*
<Rule> ::= “RULE:” <Condition><Action>
<Condition> ::= “CONDITION :” <Condition_element_1>*
<Condition_element_1> ::= “CONDITION_ELEMENT_1 :” <Condition_element_2>*
<Condition_element_2> ::= “CONDITION_ELEMENT_2 :” < Condition_element>*
< Condition_element> ::= <Key> <Operator> <Value>
<Key> ::= (“origin” | “via” | “destination” | “departure_time_range” |
“arrival_time_range” | “line_id” | “trip_id” | “node_id” | “service_class” |
“crowd_level” | “delay” | “status”)
<Action> ::= “ACTION:” <Action_element>*
<Action_element>:: = “ACTION_ELEMENT:”
        “direction=” <Direction>
        “target=” <Target>
        <Requirements>
<Direction> ::= (“AVOID” | “ATTRACT”)
<Target> ::= (“ROUTE” | “SPOT”)
<Requirements> ::= (<Route_req> | <Spot_req>)
<Route_req> ::= “REQUIREMENT:” <Route_req_element>*
<Route_req_element> ::= <Route_key> <Operator> <Value>
< Route_key> ::= (“origin” | “via” | “destination” | “departure_time” | “arrival_time” |
“line_id” | “trip_id” | “node_id” | “service_class” | “service_type” | “crowd_level” |
“max_time_diff” | “fixed_route”)
<Spot_req> ::= < Spot_req_element>*
<Spot_req_element> ::= <Spot_key> <Operator> <Value>
<Spot_key > ::= (“spot_id” | “center” | “range” | “genre” | “budget” | “stay_time” |
“crowd_level” | “coupon”)
```

図 2-6　誘導案内ポリシーの文法

(2) 予約語

想定するユースケースからコンディションの項目名として指定できる予約語を定義した．予約語とその説明を表 2-2 に示す．

コンディションでは，経路検索要求から得られる出発地，経由地，到着地，出発時刻など利用者の移動予定に応じて誘導案内を出し分けられるように，一般的な経路検索条件に対応した予約語を定義する．

また，経路検索の結果である候補経路情報に応じて誘導案内を出し分けられるように，

候補経路情報を構成する駅や路線，便の識別子の他に，指定席やグリーン車などの交通サービスの属性，および，状態を指定する予約語を定義して，コンディションの項目名として使用する．これら駅や路線，便の識別子，指定席やグリーン車などの交通サービスの属性，および，状態は一般的な経路検索サービスが返す経路情報に含まれる．

表 2-2 コンディションの項目名

No.	カテゴリ	項目名	説明
1	経路検索条件	origin	経路検索条件の出発地
2		via	経路検索条件の経由地
3		destination	経路検索条件の到着地
4		departure_time_range	経路検索条件の発時刻
5		arrival_time_range	経路検索条件の着時刻
6	候補経路	line_id	候補経路を構成する路線番号
7		trip_id	候補経路を構成する便番号
8		node_id	候補経路を構成する駅番号
9		service_class	候補経路を構成する便のサービス種別（指定席，グリーン車）
10	候補経路状態	crowd_level	候補経路を構成する便の最大混雑度（%）
11		delay	候補経路を構成する便の最大遅延時間（分）
12		status	候補経路を構成する便の運行状況（平常・運転見合せ）

アクションで指定する人流制御指示を表 2-2 に示す．人流制御指示は，人を集める誘引（ATTRACT）と，分散させる回避（AVOID）の２種類である．アクションでは，このどちらかを選択する．

アクションで指定する対象種別は，経路（ROUTE）と施設（SPOT）の２種類である．

表 2-3 人流制御指示

Direction	説明	制約
ATTRACT	対象（Target）で指定される経路または施設への誘引を指示する	経路･施設検索条件は必須
AVOID	対象（Target）で指定される経路または施設の回避を指示する	経路･施設検索条件は任意

次に，経路の誘導案内検索条件の項目名として指定できる予約語とその説明を表 2-4 に，施設の誘導案内検索条件の項目名として指定できる予約語とその説明を表 2-5 に示す．

表 2-4 の項目名は，提示する誘導案内経路を決定するための情報であり，出発地，経由地，到着地，出発時刻などの一般的な経路検索条件に加えて，駅や路線など経路の構成要素識別子，属性，状態，通常経路との出発時刻や到着時刻の差を記述できるようにする．ただし，誘導経路の構成要素識別子，属性，状態は一般的な経路検索サービスでは経路検索条件として指定できない．また，通常経路との出発時刻や到着時刻の差も指定できない．しかし，経路情報は構成要素識別子，属性，状態を含むので，誘導経路検索部が，経路情報を誘導案内検索条件の構成要素識別子などに照らし合わせれば条件に適合する経路情報であるかを判断できる．

さらに，経路の誘導案内検索条件では，利用者の目的地や出発時刻に合わせて利用を促す経路を選ぶために，利用者が与えた経路検索条件の値を固有名詞ではなく変数として記述できるようにする．表 2-4 の No.1 から No.5 の項目名の予約語の先頭に@を付与した場合は，利用者が入力した元の経路検索条件で指定された値を代入することを示す．

表 2-4 経路検索条件の項目名

#	カテゴリ	項目名	説明
1	経路検索条件	origin	出発地
2		via	経由地
3		destination	到着地
4		departure_time	発時刻
5		arrival_time	着時刻
6	誘導経路 構成要素識別子	line_id	路線番号
7		trip_id	便番号
8		node_id	駅番号
9	誘導経路の属性	service_type	臨時便等の運行区分
10		service_class	指定席, グリーン車等のサービス種別
11	誘導経路の状態	crowd_level	最大混雑度（%）
12	通常経路 との差異	max_time_diff	通常経路の最も早い到着時間と誘導経路の到着時間との差分（分）
13		fixed_route	通常経路と同一経路であることを条件とする

表 2-5 に示す施設の誘導案内検索条件の検索キーとしては，施設検索の中心位置として使用する駅(center)を指定できるようにする．center の取りうる値は，出発する駅(origin)，経由駅(via)，到着駅(destination)である．例えば，到着駅で立寄りをすすめる場合には destination を選択する．また，移動中の立寄りに適した施設を選ぶために，平均滞在時間や混雑度を指定できるようにする．

表 2-5 施設検索条件の項目名

#	カテゴリ	項目名	説明
1	識別子	spot_id	施設 ID
2	場所	center	中心位置にする駅
3		range	中心位置からの距離（m）
4	属性	genre	提案施設のジャンル
5		budget	平均利用額（円）
6	立寄り	stay_time	平均滞在時間（分）
7		crowd_level	混雑度（%）
8	誘引施策	coupon	クーポン金額（円）

表 2-5 に示す項目の内，項番 6 と項番 7 以外は，一般的な施設検索サービスにおいて検索条件として指定できるものである．項番 6 の平均滞在時間と項番 7 の混雑度については，検索条件としては指定できないが，検索結果である施設情報に含まれるため誘導施設検索部が施設情報を誘導案内検索条件と照らし合わせて条件に適合するか否かを判断できる．

2.3.2 誘導案内ポリシー記述例と要件の確認

混雑回避の誘導案内ポリシーを提案する記述方式を用いて書いた例を図 2-7 に示す．コンディションでは，利用者が検索条件で与えた出発時刻 departure_time_range と，その経路検索結果である通常経路情報の混雑度 crowd_level に対する条件をコンディションとして記述している（要件１）．

アクションでは人流制御指示として回避 (AVOID)，対象種別として経路 (ROUTE) を指定し，誘導案内検索条件として，しきい値となる混雑度 crowd_level が 80%以下であること，および，出発時刻，到着時刻は利用者が与えた元の経路検索条件で指定した時刻より 30 分後にずらすことを記述している．出発時刻，到着時刻は経路検索条件で指定することができ，混雑度は経路検索サービスが返す経路情報に含まれている．また，施設の誘導案内検索条件として，施設検索の中心位置となる地点，範囲，滞在時間を指定している．これらは，施設検索サービスの検索条件に変換することができる．（要件２）．

出発地，到着時刻については，元の経路検索条件として指定された値を使い，出発時刻や到着時刻は，元の経路検索条件で指定した時刻を参照し，その 30 分後にずらす誘導案内経路を出すように指定している．このように，利用者が与えた経路検索条件の値を固有名詞ではなく変数として指定できるので，利用者の与える個々の経路検索要求に依存せずに，出発時刻を前後にずらした経路や，到着地付近の施設を提示する誘導案内ポリシーを記述できる（要件３）．

```
RULE:
  CONDITION:
    CONDITION_ELEMENT_1:
      CONDITION_ELEMENT_2:
        # 出発時刻が 05:00 から 09:00 で混雑度が 80%以上の経路なら
        crowd_level = 80
        departure_time_range = 05:00-09:00
  ACTION:
    ACTION_ELEMENT:
      # 同じ発着地で混雑度が 80％より小さく，出発･到着時刻を 30 分後ろ倒しする
      direction=AVOID
      target=ROUTE
      REQUIREMENT:
        crowd_level=80
        origin=@origin
        destination=@destination
        departure_time=@departure_time+30
        arrival_time=@arrival_time+30
        via=@via
    ACTION_ELEMENT:
      # 出発駅で平均滞在時間が 30 分以下
      direction=ATTRACT
      target=SPOT
      REQUIREMENT:
        center=origin
        range=500
        stay_time=30
```

図 2-7 混雑回避の誘導案内ポリシー記述例

なお，本来は，誘導案内ポリシー記述能力の評価として，提案手法で実際の案内業務をカバーする誘導案内ポリシーが書けるかを検証すべきであるが，本章ではこの評価は行わない．これは，案内業務に対するカバレッジは実際の案内業務において長期間運用しないと分からないためである．実際に，長期間運用した後で，案内担当者にアンケートを取って評価するなどの方法が考えられる．

2.4　誘導案内ポリシーの記述支援

2.4.1 テンプレート

誘導案内ポリシー記述を支援する 7 種類のテンプレートを図 2-8 に示す.

これらのテンプレートは, ユースケースからアクションを類型化することで導き出した. 人流制御指示の誘引と回避, 対象種別の経路と施設との組み合わせは 4 通り考えられる. しかし, 本章では, 施設に対する回避案内は取り扱わない. これは, 想定する誘導案内システムは, 利用者の経路検索操作を契機としておすすめの経路や施設を案内するもので, 利用者が施設を主体的に検索することはなく, ある施設の利用を回避してもらうよう提示することがないからである.

そこで, 誘導案内ポリシーで記述するアクションの基本型は, 特定の経路の回避, 特定の経路への誘引, 特定の施設への誘引の 3 種類になる. これら 3 種類を基本テンプレートとして定義した. そして, これらの基本テンプレートをユースケースに特化して派生させた. 派生とは, デフォルト値として, ユースケースから決まるコンディションや誘導案内検索条件を入れることと, 1 つのポリシールールで複数のアクションを実行できるようにあらかじめ構造を定義することを指す.

基本型のテンプレートは, 経路回避（AvoidRouteRule）, 経路誘引（AtractRouteRule）, 施設誘引（AtractSpotRule）である. 経路誘引, 経路回避テンプレートでは, 出発地と到着地, 経由地, 出発時刻, 到着時刻は元の検索条件で指定された値をデフォルト値として用いるため入力は必須ではない.

さらに, 経路回避には, 出発時刻を変更して回避する方法と, 経由地を変更する経路回避があるため, 通る経路は変えずに出発時刻を変更する時刻変更経路回避（TimeShiftAvoidRouteRule）と, 経由地を変更する経由地変更経路回避（RerouteAvoidRouteRule）の 2 種類を定義した. この時刻変更経路回避, 経由地変更経路回避は, 立寄り先の施設を提示することで, 出発時刻の変更や経由地の変更を促すように, 出発地の立寄り先施設や, 経由地での立寄り先の施設を追加するための第 2 のアクションとして施設誘引を追加した.

施設誘引については, 経路上の立寄り施設への誘引と経路外の施設への誘引があるため, 乗降駅と経由駅で施設情報を提示する立寄り施設誘引（DetourAtractSpotRule）と経路外施設誘引（AddAtractSpotRule）の 2 種類を追加した. 経路外の施設への誘引は,

その施設までの経路も提示できるように，第 2 のアクションとして経路誘引を追加した．追加した経路誘引アクションでは，元の経路の到着地を出発地として，提示する施設を到着地とするの誘導案内検索条件をあらかじめ設定した．

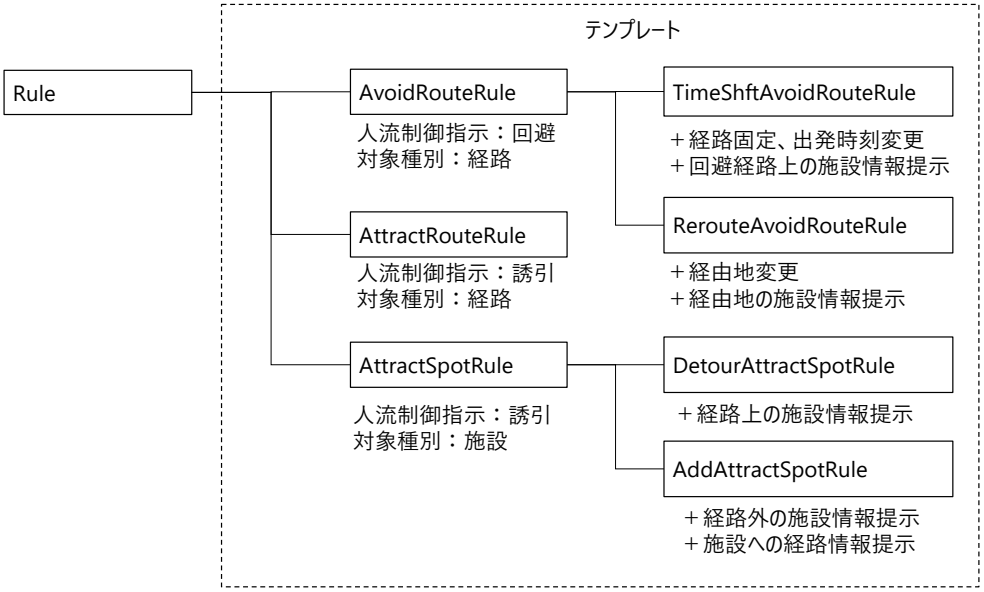

図 2-8 テンプレートの種類

ユースケースに対して適用するテンプレートと，テンプレートを適用しない場合に入力しなければならない検索条件数，適用する場合に入力が必須となる検索条件数を表 2-6 に示す．ユースケース No.2，No.4 は AttractRouteRule で記述する．No.6，No.7 は AttractRouteRule で記述する．No.8 については，テンプレートを作成しない．これは，1 駅手前で下車するという経路は駅間距離が短い限られた路線にしか適用できないためである．

表 2-6 ユースケースとテンプレートの対応

No.	適用するテンプレート	未適用時 検索条件数	適用時 検索条件数
1	AddAtractSpotRouteRule	6 件	1 件
2	AttractRouteRule	6 件	1 件
3	TimeShftAvoidRouteRule	6 件	1 件
4	AvoidRouteRule	6 件	1 件
5	AttractRouteRule	6 件	1 件
6	RerouteAvoidRouteRule	8 件	1 件
7	RerouteAvoidRouteRule	5 件	1 件
8	対応なし	-	-
9	DetourAttractSpotRule	1 件	1 件
10	DetourAttractSpotRule	3 件	1 件

ユースケース No.6 に経由地変更経路回避テンプレートを適用した場合の入力項目を表 2-7 に示す．このテンプレートは，ある経路の回避と，回避経路の経由地に立寄り先の施設を追加して提示するものである．第 1 のアクションの人流制御指示として回避（AVOID），対象種別として経路（ROUTE）が指定されており，第 2 のアクションの人流制御指示として誘引（ATTRACT），対象種別として施設（SPOT）が指定されている．入力が必要な項目は避けて欲しい経由地だけである．提示する施設は，第 1 のアクションによって検索された誘導経路情報の経由地を中心地として，半径 100 メートル以内にある滞在時間が 30 分以下の場所を検索する条件があらかじめ設定されている．

表 2-7 経由地変更経路回避テンプレート

項目			設定済の値
コンディション	経由地		—
第 1 の アクション 経路回避	人流制御指示		AVOID（回避）
	対象種別		ROUTE（経路）
	誘導案内 検索条件	出発地	@origin
		到着地	@destination
		避ける経由地	@via
		出発時刻	@departure_time
		到着時刻	@arrival_time
第 2 の アクション 施設誘引	人流制御指示		ATTRACT（誘引）
	対象種別		SPOT（施設）
	誘導案内 検索条件	中心位置	via (迂回経路の経由地)
		範囲	100
		滞在時間	30

2.4.2 記述支援エディタ

7 種類のテンプレートを組み込んだ記述支援エディタを作成した．記述支援エディタでは，まず利用するテンプレートを選択する．テンプレートを選択すると編集画面が表示される．編集画面では，定義済のコンディションやアクションなどの制御構造と，人流制御指示，対象種別，および，誘導案内検索条件の入力フィールドが表示される．誘導案内検索条件の入力フィールドには，定義済のデフォルト値が入力された状態である．経由地変更経路回避テンプレートの編集画面を図 2-9 に示す．

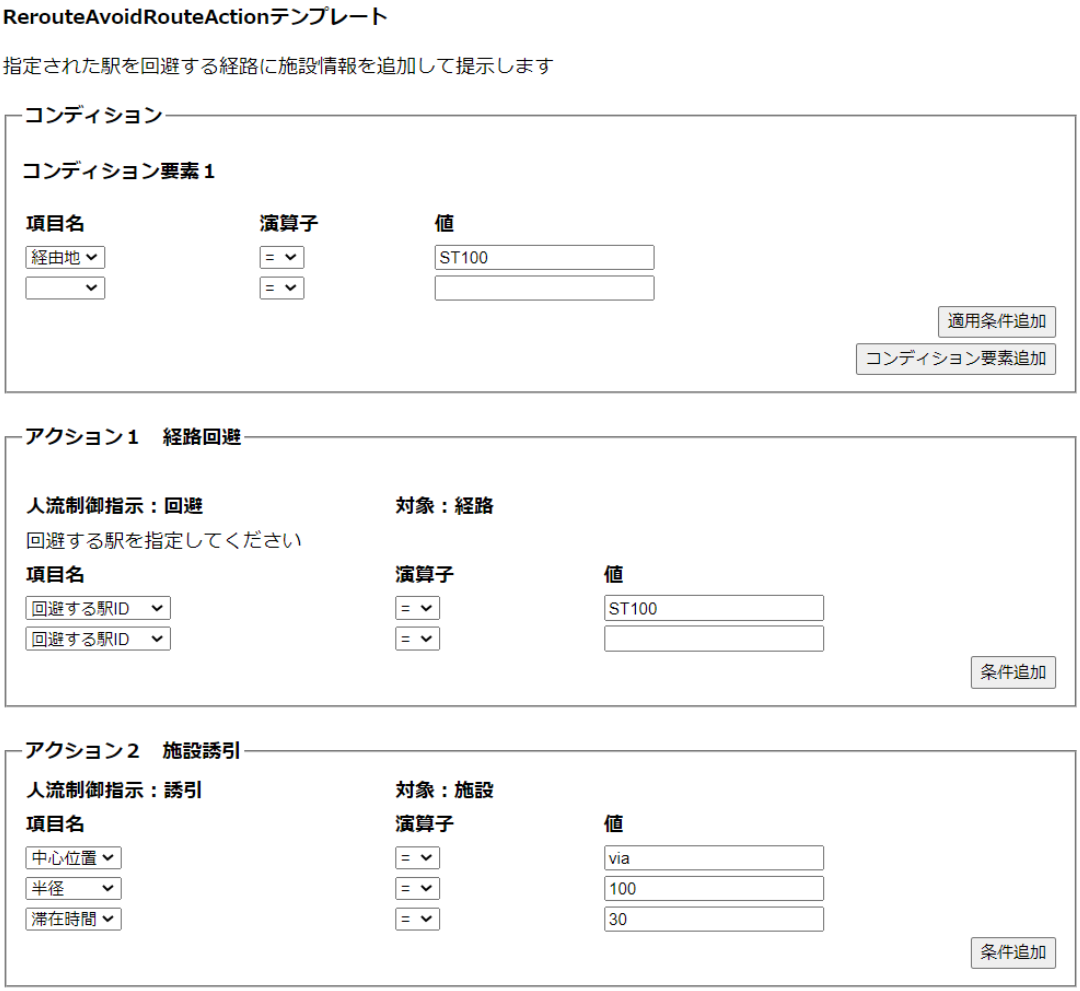

図 2-9 誘導案内ポリシー記述支援エディタ

図 2-9 に示す記述支援エディタで，経由地変更経路回避テンプレートを選択し，経由地を入力すると，記述支援エディタは図 2-10 に示す誘導案内ポリシールールを生成する．

```
RULE:
  CONDITION:
    CONDITION_ELEMENT_1:
      CONDITION_ELEMENT_2:
        via = ST100
  ACTION:
    ACTION_ELEMENT:
      direction=AVOID
      target=ROUTE
      REQUIREMENT:
        avoid_node=@via
        origin=@origin
        destination=@destination
        departure_time=@departure_time
        arrival_time=@arrival_time
    ACTION_ELEMENT:
      direction=ATTRACT
      target=SPOT
      REQUIREMENT:
        center=via
        range=100
        stay_time=30
```

図 2-10 経由地変更経路回避の誘導案内ポリシー

2.5 記述工数の評価

2.5.1 評価方法

提案する誘導案内ポリシー記述方式による記述工数軽減効果を評価するために評価実験を行った．評価実験では，3 名の被験者に表 2-8 および表 2-9 に示す 5 通りの評価用ユースケースを提示し，各ユースケースを実現する誘導案内ポリシーを記述するとい

う課題を出題した．

誘導案内ポリシーの記述は，従来方式と提案方式の 2 通りで作成してもらった．従来方式では，誘導案内ポリシーの仕様書を提供し，一般的なテキストエディタで記述してもらった．提案方式では，テンプレートを登録した記述支援エディタを用いて，必要な入力項目を入れる形で記述してもらった．

評価項目は次の 3 つである．

記述工数：誘導案内ポリシー記述に要する時間

要件達成度：5 件中何件のユースケースについて，人流制御指示，対象種別，経路の誘導案内検索条件，および，施設の誘導案内検索条件を漏れなく指定できたか

間違い件数：文法的記述間違いの件数

3 名の被験者について説明する．被験者 1 は，交通分野の知識，プログラミング経験ともに無く，被験者 3 は，交通分野の知識は有しているがプログラミング経験は無い．被験者 2 は，交通分野の知識は無いがプログラミング経験がある．

被験者 1，被験者 3 は案内担当者のスキルセットを想定している．被験者 2 は情報システムの開発者を想定している．情報システムの知識やプログラミング経験が無いことが，従来方式での誘導案内ポリシー記述では負担となるが，提案方式ではその負担が軽減されることを確認するためにこのようなスキルセットの異なる被験者を選択した．

表 2-8 評価用ユースケース No.1-No.2

No.	大項目	項目	
1	概要	混雑回避のため迂回路を提示	
	コンディション	検索した経路上で混雑度が 80%以上の区間がある	
	アクション	1	混雑度 80%以上の区間を回避する経路
			出発地，到着地は元の経路検索条件と同一
			到着時刻，発時刻指定は元の経路検索条件と同一
2	概要	混雑回避のため移動時刻の変更を提示	
	コンディション	検索した経路上で混雑度が 80%以上の区間がある	
	アクション	1	混雑度 80%以上の区間を回避する経路
			出発地，到着地は元の経路検索条件と同一
			到着時刻，発時刻指定は元の経路検索条件と同一
			元の経路検索条件と同一経路
		2	乗車駅で立寄りをおすすめ
			ジャンルは書店

表 2-9 評価用ユースケース No.3-No.5

No.	大項目	項目	
3	概要	工事で駅設備の一部が利用できない場合や，イベントの開催により駅が混雑する場合にその駅を回避する経路を提示	
	コンディション	検索した経路が指定駅(駅番号 : ST100)で乗換える経路である	
	アクション	1	指定駅(駅番号 : ST100)で乗換えを避ける経路
			出発地，到着地は元の経路検索条件と同一
			到着時刻，発時刻指定は元の経路検索条件と同一
		2	経由地になる施設の立寄りをおすすめ
			経由地からの距離は 0m 以内(駅ナカ)
			平均滞在時間が 30 分以下
4	概要	運転見合わせ時の代替バスへの誘引	
	コンディション	検索した経路上で運転見合わせ発生時	
	アクション	1	運行区分が臨時便である区間を含む経路
			出発地，到着地は元の経路検索条件と同一
			到着時刻，発時刻指定は元の経路検索条件と同一
5	概要	大きなイベントの開催時に会場へのバス便を案内するなど，降車駅を起点として多くの人が向かうことが推測される到着地がある場合に到着地までのスムーズな移動手段を案内	
	コンディション	検索した経路の降車駅が指定の駅（ST200）である	
	アクション	1	施設 ID が指定の ID（SP100）である施設
		2	出発地が元の経路の到着地である経路
			到着地は施設 ID（SP100）
			出発時刻は元の経路の到着時刻の 10 分後以降

2.5.2 評価結果と考察

表 2-10 は各被験者の誘導案内ポリシーの記述工数と，要件達成度，記述間違い件数を示した表である．

記述工数については，提案方式は従来方式と比べて必須入力項目が少ないため，被験者 1，被験者 3 では 4 割以上の所要時間が削減されている．人流制御指示や対象種別を入力する代わりに，テンプレートを選択する必要があるが，被験者 2 からは，人流制御指示や対象種別を入力するよりもテンプレートを選ぶ方が，直感的で分かりやすいという意見が得られた．

要件達成度については，提案方式では全ての要件が達成されたが，従来方式では未達成が 2 件確認された．ユースケース No.1 で，誘引する施設への経路の誘導案内検索条件として，運行区分を臨時便とする条件の指定が不足していた．

記述間違いについては，従来方式で 2 件確認された．選択すべき項目名と類似した別の項目名が選択されたケース，項目名に対応しない値を入力するケースであった．プログラミング経験のある被験者 2 の記述間違いは，誘導案内検索条件の演算子を不等号で記述したものであり，一般的なプログラミング言語の記述方式と間違えたと考えられる．

さらに，工数については，複数のアクションを定義するユースケース No.3，No.4，No.5 の記述に時間を要していた．特に，被験者へのヒヤリングによって，No.5 の 2 番目のアクションである立寄り地までの経路を追加するための誘導案内検索条件の記述が難しかったとの意見が得られた．

No.5 について，既存方式では，被験者は自分が記述した誘導検索条件が，ユースケースとして示された誘導案内経路情報を取得するために十分であるか判断しづらかったと考えられる．提案方式では，テンプレートを誘導案内経路情報の例と対応づけて提供したため，テンプレートによって生成される誘導案内を想像することが容易になり，自分が入力した誘導案内検索条件が，ユースケースに合った誘導案内経路情報を生成するために十分であるかを判断し易くなる．その一方で，提案方式ではユースケースに対して適切なテンプレートを選択する作業が発生するため，テンプレートの説明や例を充実させて選択を支援することが重要になると考えられる．

表 2-10 評価結果

被験者	記述工数(分:秒)		要件達成度		間違い件数(件)	
	提案	従来	提案	従来	提案	従来
被験者 1	19:57	35:01	5/5	4/5	0	1
被験者 2	16:44	21:49	5/5	5/5	0	1
被験者 3	21:02	36:59	5/5	4/5	0	0

2.6 結言

本章では，経路検索機能をカスタマイズするための設定情報である誘導案内ポリシーの記述方式として，誘導案内ポリシーの文法と記述支援方式を提案した．そして，記述支援方式について，従来のポリシー記述方式と比較した記述工数削減効果を被験者を用

いて評価し，記述工数が約 4 割削減できることを示した．

誘導案内ポリシーの文法は，専門家へのヒヤリングによって作成したユースケースを全て記述できるように構造と予約語を定義した．その上で，記述支援方式では，誘導案内ポリシーの記述を容易化するため，複数の交通事業者に共通して適用可能なユースケースに絞って，個別のユースケースに対応づいたテンプレートを作成し，テンプレートがあらかじめ登録された記述支援エディタを提案した．これにより，誘導案内ポリシーを記述する代わりにテンプレートを選択し，限られた誘導案内検索条件を設定すれば誘導案内ポリシーを記述できるようになる．そして，テンプレートが適用できないユースケースに対しては，文法にしたがって誘導案内ポリシーを記述することで誘導案内を実現できる．

このようにして，多様なユースケースに対応できる記述力のある誘導案内ポリシー記述方式を定義するという課題と，誘導案内ポリシーの記述を容易化するという相反する 2 つの課題を解決した．

今後は，誘導案内システムを長期間運用した後で案内担当者にアンケートを取って評価するなどの方法で，提案した誘導案内ポリシーの文法で実際の案内業務をカバーする誘導案内ポリシーが書けるかを検証する必要がある．

第3章
統合モビリティサービスの実現にむけたスキーママッチング手法

3.1　緒言

本章では，複数の交通サービスを連携させ統合モビリティサービスとするため，事業者固有形式の交通サービスの運行計画や料金などの情報を相互に対応づけるスキーママッチング手法を提案する．

複数交通手段の一括予約や一元的な案内の提供には，個別に公開されている電車やバスなどの運行計画情報を収集し，組合せてシームレスなサービスとする必要がある．しかし，日本の公共交通は多数の交通事業者がそれぞれ独自に発展させてきたものであるため，データ公開の共通フォーマットを持っていない．一方，欧州では政府や自治体が公共交通に多額の出資をしており，政府が主導する形でデータ形式の共通化が進んでいる．近年，日本においても公益に資するサービスを民間に開発してもらうことを目的として，政府や自治体は公共交通や水道，電力など社会基盤に関する情報のオープンデータ化を推進しており[14]，公共交通に関するデータを公開するためのスキーマも提案されているが，標準として採択されたものは無い．さらに，提案されているスキーマは，鉄道の駅やバス停，時刻表などの基本的な情報だけを定義したものである[16][17][18]．今日，座席指定の通勤列車や乗り心地を重視した深夜バスが人気を得るなど交通サービスへのニーズは多様化しており，事業者がサービス内容の差別化を競う中で，全てのデータを統一したスキーマで公開することは現実的ではない．基本的な情報を標準スキーマとして定義したとしても，オプション属性で差別化のためのサービス情報を定義する場合には，それらのオプション属性の解釈と対応づけが必要になる．

スキーマが異なるデータを自動的に対応づける仕組みとしては，スキーママッチング手法が研究されている．しかし，既存研究はデータベースのテーブル間マッチングを対象としたものが多く，交通サービスデータのように複雑な構造のデータに適用する場合は精度が課題になる場合がある．

そこで本章では，統合モビリティサービスの実現に向けて，交通サービスに関するデータの特徴を利用して異なるデータソースから取得した交通サービスデータ間のスキーマ変換を実現する手法について提案する．

以降，3.2 節で統合モビリティサービスにおけるデータ統合の課題を述べ，3.3 節で交通サービスのデータ統合を実現するスキーマ変換手法を提案する．3.4 節では，実データを用いた提案手法の評価について述べ，最後に 3.5 節で結果をまとめる．

3.2　統合モビリティサービスにおけるデータ統合の課題

3.2.1　MaaS のためのデータの現状

統合モビリティサービスを実現する機能構成を図 3-1 に示す．この機能構成は，MaaS プロバイダレイヤとデータプロバイダレイヤに分かれる．MaaS プロバイダは，利用者に向けたサービスを統合する役割を持つ．具体的には，MaaS プロバイダは利用者の要求に応じて交通手段を組合せて旅程を作成し，利用者が選択した旅程に含まれる交通サービスが事前予約型であれば，複数事業者にまたがった予約やキャンセルの処理を行い，決済の手段を提供する．さらに，利用者が各交通サービスを円滑に利用できるよう乗車位置や乗換えの案内を行う．

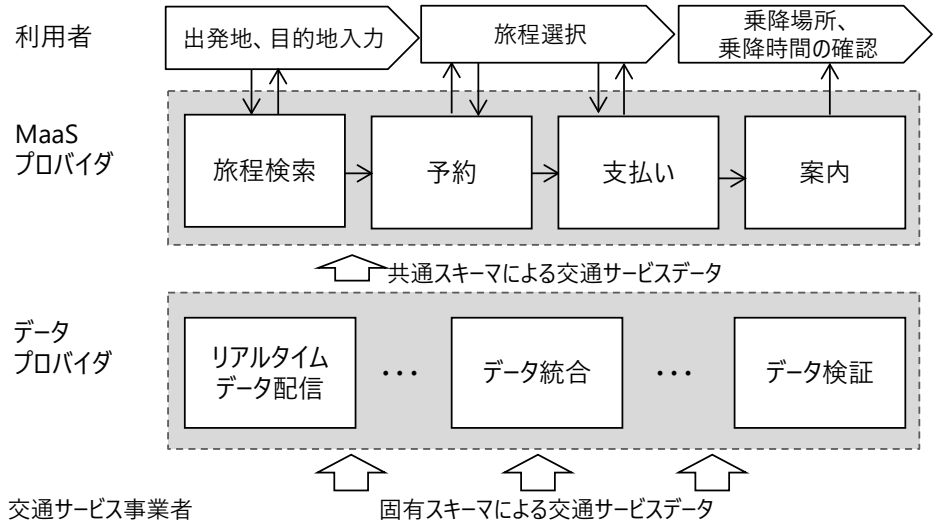

図 3-1　統合モビリティサービスの機能

一方，データプロバイダは，データを統合する役割を持つ．つまり，データプロバイダは MaaS プロバイダと交通事業者との間に位置し，事業者固有の形式で提供される交通サービス情報を集めて統合し，統一された手段で MaaS プロバイダへ提供する．

利用者の多様なリクエストに答えるためには要素となる交通サービスの多様性が必要である．このため，データプロバイダは基本的な駅，路線系統情報，時刻表，運賃に加えて，車椅子乗降の可否や女性専用車両の位置，電源や Wi-Fi 接続サービスの有無，さらには経由する駅の構内図，そして，降車駅の最寄りのシェアバイクステーションに空き自転車があるかなどを取得できることが望ましい．

これらの情報のうち，公共交通サービスの駅，バス停留所，路線系統情報，接続道路情報は，国土交通省の国土数値情報ダウンロードサービス[5]から取得可能である．データ形式は，地図上表示のための GML (Geography Markup Language) 形式，および，GeoJSON (Geospatial JavaScript Object Notation) 形式から選択できる．

この他，公共交通オープンデータ協議会のサイト[6]では，首都圏の鉄道とバス，航空に関して，上記の基本的な情報を JSON-LD (JavaScript Object Notation for Linked Data) 形式で公開しており，API で取得することができる．基本的な情報については，多くの事業者が自社ウェブサイトでも公開しているが，API で情報を提供する事業者は今のところ無い．

一方，国土交通省は，バスに関する情報の公開形式として「標準的なバス情報フォーマット」を発表しており[7]，このフォーマットで情報を公開するバス事業者が現れている．このフォーマットは，公共交通サービス情報のデータ形式として海外で広く利用されている GTFS (General Transit Feed Specification) に日本固有の情報を追加した仕様であり，CSV (Comma Separated Value) 形式のファイルとして取得することができる[15][16][17][18].

上記の基本的なデータに加えて，運休や遅延発生時のリルートや予約の変更を実施するには，運行情報を取り込む必要がある．運行情報は，鉄道の臨時速度制限，不通区間，遅延時間，在線位置情報などから構成される．不通区間，在線位置，遅延時間については，多くの事業者がウェブページで公開しており，在線位置情報についても，一部の事業者はウェブページやスマートフォンのアプリケーションで公開している．上述した公

5 http://nlftp.mlit.go.jp/ksj/
6 http://docs.odpt.org/#api
7 http://www.mlit.go.jp/common/001179007.pdf

共交通オープンデータ協議会のサイトでも，一部の事業者のサービスについては，運行情報や在線位置位置の公開が始まっている．

また, 最近では, 遠回りでも混雑を回避して移動したいというニーズも高まっており，混雑度をアプリケーションで公開する事業者も現れている．

3.2.2 スキーママッチング

データプロバイダは，データ公開サイトや事業者のウェブページから目的のデータファイルやオブジェクトを取得し, 取得したデータの属性を共通データスキーマの属性に統合する．この属性同士の対応付けは人が見れば判断できるものが多いが，鉄道，バス，カーシェア，駐車場などさまざまなデータを複数のサイトから取得し，変換することを想定すると人手での対応付けと変換は負荷が高い．

このような対応付けはスキーママッチングと呼ばれており, 属性名の類似度に着目する手法と，属性型や属性が必須かオプションか，シングルバリューかマルチバリューを許容するかといった制約に着目する手法が知られている．これらの内，制約だけでマッチング候補を特定することは難しく, 属性名を用いるのが基本的な手法である. さらに，属性名の類似度を求める手法には，単語の意味に着目する手法と，編集距離等を用いて文字列そのものの類似度を求める手法が提案されている[33][34]．ただし，属性名にどのような語彙を用いるかは設計者に依存しており, 同じ属性でも同じ単語や類義語が用いられるとは限らない．また，交通サービスのデータスキーマにおいては，例えば交通便を示す trip オブジェクトの属性として便の ID である trip_id，その便の路線を示す route_id，便が急行か普通かを示す service_id が含まれるケースがある[17]．このような場合，id や name など複数の属性名に出現する単語が共通であれば文字列の類似度は高くなるが，同じ属性である確率は低い．

一方, 上記のスキーママッチングとインスタンスが持つ属性値によるマッチングを組合せて精度を高める研究もなされている[60][61]．しかし，属性値に基づく手法では，異なる属性であっても, 類似した値を持っていれば対応づけられてしまうという課題がある．この課題は，交通サービスにおいては発車時刻と到着時刻のように異なる属性がほぼ同じ値を持つ場合もあるため顕著になる．

さらに，多くの既存研究では，与えられた 2 つのテーブルの属性を対象とした 1 対 1

マッチングを扱っているが，交通サービスのデータ構造はより複雑である．交通サービスに関する情報は，スキーマによってどの情報を 1 つのオブジェクト，または，1 つの属性として定義するかがまちまちである．あるスキーマにおいて 1 つのオブジェクトを構成する情報が他のデータセットでは複数のオブジェクトに分散したり，1 つの属性に含まれる情報が他のデータセットでは複数の属性に分かれることもある．例えば，図 3-2 に示す公共交通オープンデータ協議会スキーマでは，時刻表（odpt:BusstopPoleTimetable）の属性としてバス停（odpt:busstopPole）とカレンダー情報（odpt:calendar）が定義されているが，図 3-3 に示す標準的なバスフォーマットでは，カレンダー情報（service_id）は時刻表（stop_time）ではなく，便（trip）の属性として表現されている．また，1 つのオブジェクトであっても，各属性がフラットに並ぶ構造のものと，他オブジェクトの配列を属性として持つ入れ子構造のものがある．しかし，多くの既存研究では，このような複雑な構造を持つオブジェクトを考慮していない．

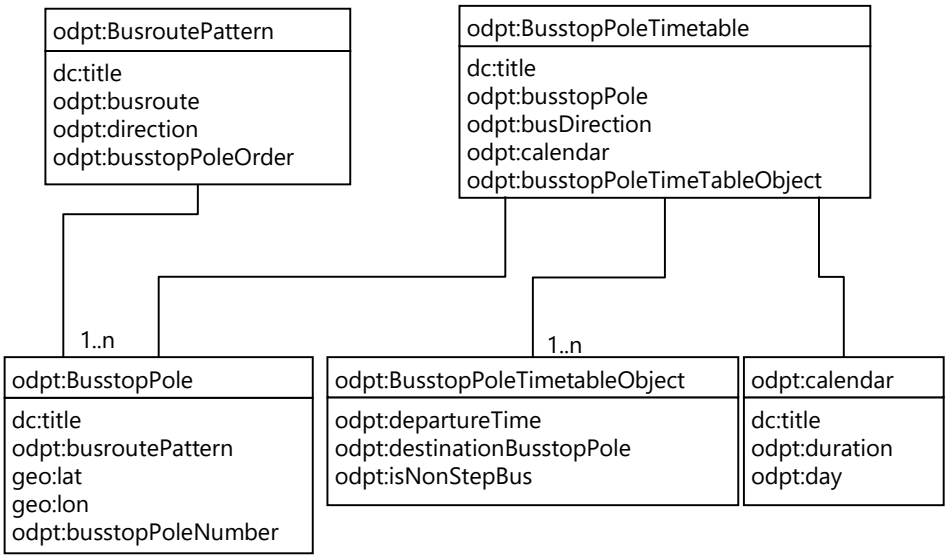

図 3-2　公共交通オープンデータ協議会スキーマ（抜粋）

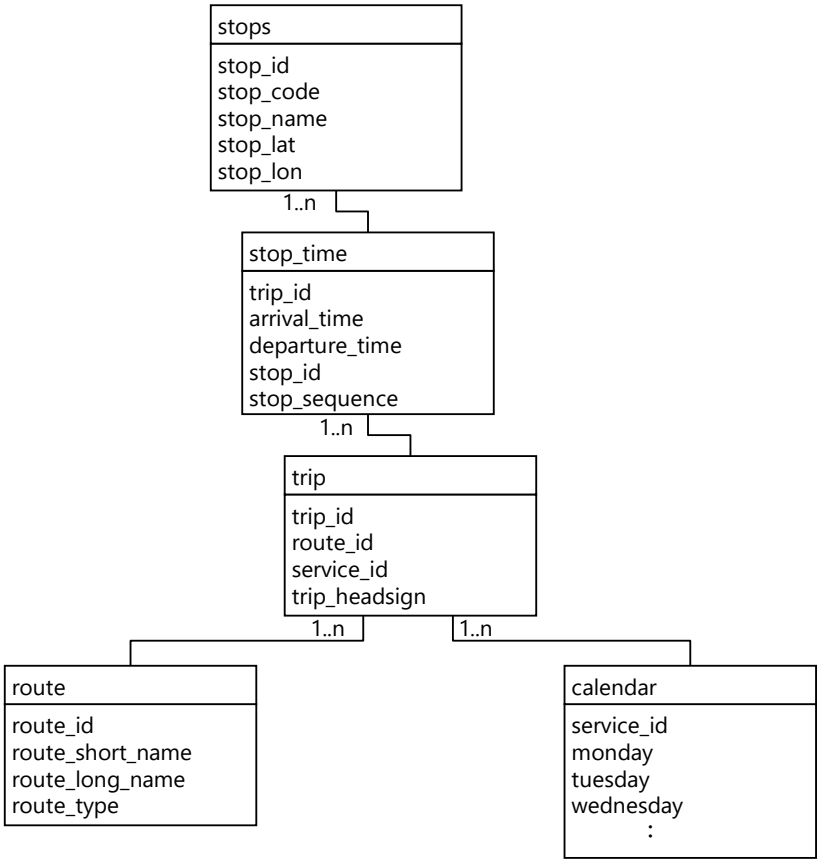

図 3-3　標準的なバスフォーマット（抜粋）

3.3　拡張属性名類似度マッチングと属性型別属性値類似度マッチング

3.3.1　参照関係オブジェクトの取得

交通サービスのデータ構造と用語の特徴を考慮したスキーママッチング手法を提案する．本手法では，3.1 節で述べた複雑なデータ構造に対応するため，マッチング元オブジェクトと参照関係にあるオブジェクトをマッチングの対象に加えてマッチングを行う．オブジェクト間の参照関係は，図 3-4 に示すように(1) 別オブジェクトがマッチング元オブジェクトのキー属性を持つ構造と，(2) マッチング元オブジェクトが別オブジェクトのキーを属性として持つ構造とがある．(1)の場合，マッチング元オブジェクトのキーを持つオブジェクト stop_time は，マッチング元オブジェクト trip の一部と考

え，stop_time の属性を trip の属性と同じように対象に加えてマッチングを実行する．(2) の場合，例えば，マッチング先オブジェクト属性 line_index_code が，キーとなる値を持つマッチング元オブジェクト属性 odpt:busroutePattern と対応づけられた場合には，odpt:busroutePattern が参照しているオブジェクト odpt:BusroutePattern は line_index_code に相当する情報を含む可能性があると考え，line_index_code と odpt:BusroutePattern の全属性との類似度を求める．

なお，マッチング元オブジェクトと参照関係にあるオブジェクトは，階層が深くなるにつれてマッチング先オブジェクトとの関連が弱くなると考え，最初に名称が対応するオブジェクト同士でマッピングを実行し，候補が検出されなかった属性について，参照関係にあるオブジェクトを取得してマッチングを再実行する．

以降の 3.3.2，3.3.3 では，提案手法における属性名の類似度によるマッチング，属性値の類似度によるマッチングについてそれぞれ説明する．

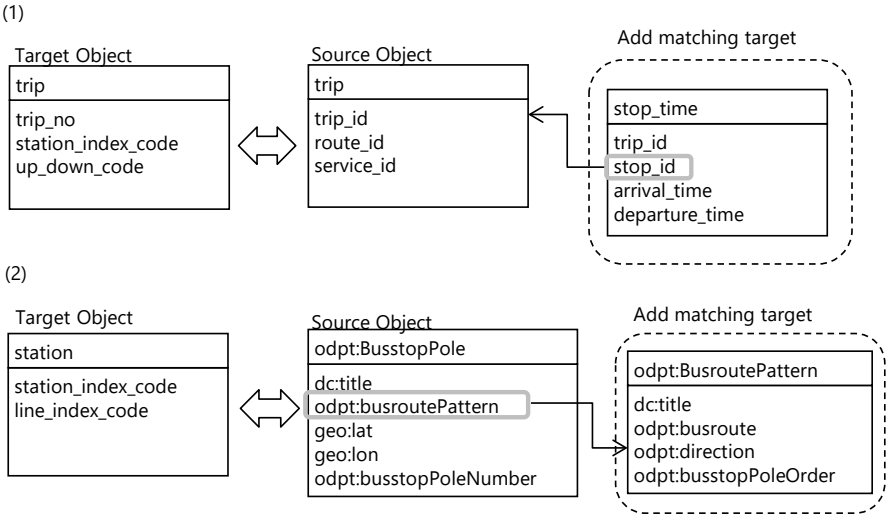

図 3-4　参照関係オブジェクト

3.3.2　拡張属性名類似度マッチング

マッチング元オブジェクトと参照関係にあるオブジェクトを対象に入れる際には，マッチング元オブジェクトの属性名とその属性値によって参照されるオブジェクトの属性名を接続して属性名とみなし，属性名の類似度計算を行う．この接続した属性名を拡張属性名と呼ぶ．拡張属性名の例を図 3-5 に示す．マッチング元オブジェクトの属性

名を入れた拡張属性名を用いることで, バス路線 odpt:BusroutePattern の名前 dc:title は odpt:BusroutePatter+dc:title となり, バス停 odpt:Busstop の名前 dc:title と区別することができる.

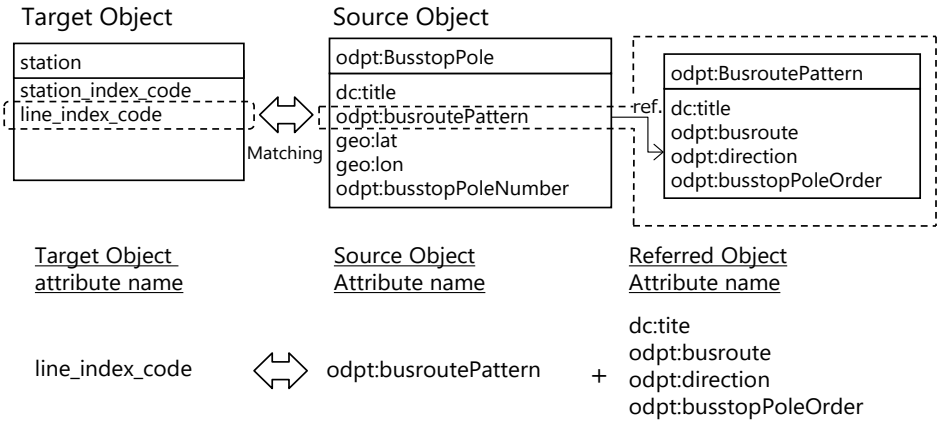

図 3-5　拡張属性名

属性名の類似度は, オブジェクト集合間の類似度を定義するジャッカード(Jaccard)係数を用いて計算する．まず，マッチング元オブジェクトの属性名 Sj，および，マッチング先オブジェクトの属性名 Ti について，各属性名から”_”や”:”で区切られた単語を抽出し，属性名を構成する単語の集合 Ti = {wt1, wt2, …, wtn}，Sj = {ws1, ws2, …, wsm} を作成する．そして，Ti と Sj のジャッカード係数を求める．なお，用語の相違や表記のゆれを吸収するため，類似度を計算する際には，類義語辞書を用いて単語 1 と単語 2 が意味的に同じであれば同一とみなす.

$$\text{類似度 (Ti, Sj)} = \frac{(|\text{Ti} \cap \text{Sj}|)}{(|\text{Ti} \cup \text{Sj}|)}$$

さらに，上記類似度の計算においては，同一オブジェクト中の複数の属性で用いられる単語は，属性を特徴づける単語ではないとみなして，それらの単語の重みが相対的に低くなるようにする．具体的には，統合対象の 2 つのオブジェクト集合を対象として，IDF (Inverse Document Frequency)を用いて計算した各単語の重要度を正規化した値を用いて単語ごとの重みづけを行う．N はあるオブジェクトにおける属性数，df (t) は単語 t を用いている属性数である.

3.3.3 属性型別属性値類似度マッチング

交通サービスの情報では，属性値が文字列である場合は，場所の固有名詞か事業者が定める路線や便のコードであることが多い．そこで，文字列であればマッチング先属性値からパターンを抽出して，そのパターンがマッチング元属性値に合致する割合を類似度とする．例えば，2 桁の数字が":"を挟んで 3 回出現する時刻表現パターン，文字列が "-" や"_"で接続された路線コードや便のコードとして用いられるパターンなどがある．数値であれば双方のデータセットの値域の重複割合を求めて類似度とする．

このために，まず，マッチング元属性の属性型を確認し，マッチング先属性の属性型と一致するか確認する．マッチング元オブジェクトが，スキーマへのリンクを持つ JSON-LD によって定義されたデータセットであれば，スキーマを参照して各属性の型を取得することができる．それ以外の場合には属性値から属性型を推定する．日付や数値は，文字列として記録されている場合もあるため，互換性のある属性型同士であれば型が一致しているとみなす．ある属性型に対して互換性があるとみなす属性型を表 3-1 に示す．特に，時刻の表記は複数の形式があり，午前 0 時からの累計秒数で表現することもあるため，一般的な時刻フォーマットに加えて，整数値も互換性がある属性型とする．上記手順により，属性型が一致する場合には，属性型別の属性値類似度を求め，一致しない場合には類似度は 0 とする．

表 3-1 互換性のある属性型

属性型	互換性のある属性型
文字列	xsd:string
整数/実数	xsd:int, xsd:float, xsd:double, xsd:string
日付	xsd:date, xsd:dateTime, xsd:int, xsd:string
時刻	xsd:time, xsd:int, xsd:string
URI	xsd:anyURI, xsd:string

3.4　評価実験

3.4.1 実験方法

提案するスキーママッチング手法の精度を評価するため, オープンデータを用いて実験を行う．実験では，最も基本的な既存手法である属性名類似度マッチング（NM: Name Matching)，提案手法である拡張属性名類似度マッチング(ENM: Enhanced Name Matching)，ENM と属性型別属性値類似度マッチング（TVM: Type sensed Value Matching)の組合せ（ENM +TVM)の３通りの手法を試行し，再現率，および，適合率を調査する．

マッチング対象範囲は，下記 2 通りとする．ここで，対応する名称とは，オブジェクトの名称同士の類似度が高いことを指している．

(a) 対応する名称を持つオブジェクトのみ対象とする

(b) 対応する名称を持つオブジェクトと参照関係にあるオブジェクトも対象とする

評価に用いたオープンデータを以下に示す. 公共交通オープンデータ協議会サイトから関東地方の鉄道に対するデータ（ODPT）を取得し，統合モビリティサービスの共通スキーマに対応づけた．さらに，標準的なバスフォーマットの形式でデータを公開している事業者サイトから，バスに関するデータ（GTFS）を取得し，共通スキーマに対応づけた．取得したデータのクラス数，属性数，インスタンス数を表 3-2 に示す．統合モビリティサービス共通スキーマの一部を図 3-6 に示す．この共通スキーマは，統合モビリティサービス実現に必要なデータを独自に定義したものであり, バスと鉄道で共通である．

表 3-2　評価用データ

		Class#	Attribute#	Instance#
ODPT	鉄道	16	185	267
GTFS	バス	17	125	889

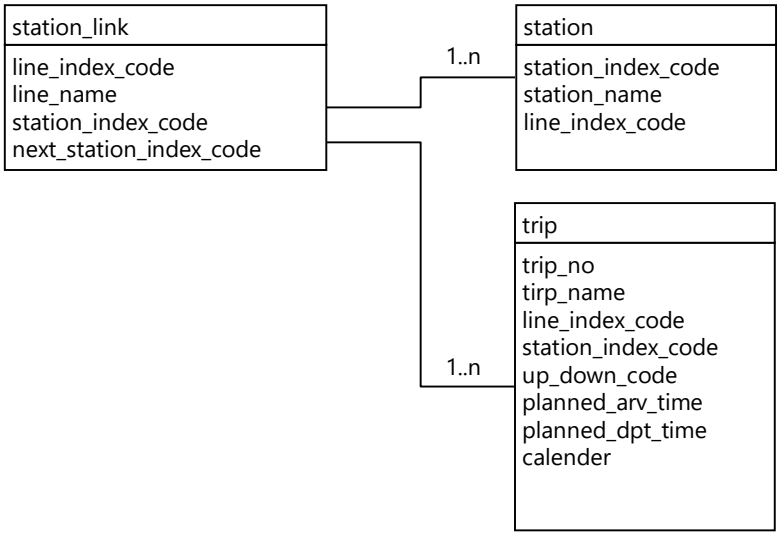

図 3-6　統合モビリティサービス共通スキーマ（抜粋）

なお，本手法では属性名の類似度は類義語辞書に左右される．そこで，日本語 WordNet[8]，および，ドメインの類義語辞書として鉄道用語辞典[9]を用いて，基本的な単語を入力し，得られた検索結果を類義語として用いる．例えば，「路線」と入力した場合，「バス路線」「route」「bus route」「bus line」が得られるため，これらを「路線」の類義語として登録する．

3.4.2 実験結果

3.2 で説明したマッチング手法により，標準的なバスフォーマットのデータ（データ１），公共交通オープンデータ協議会から取得した鉄道に関するデータ（データ２），を統合モビリティサービス共通スキーマへ対応づけた．マッチング手法は，属性名類似度マッチング（NM），提案手法である拡張属性名類似度マッチング（ENM），ENM と属性型別属性値類似度マッチング（TVM）の組合せ（ENM +TVM）の３通りの手法を適用した．マッチングでは最も類似度の高い属性同士を対応づけ，適切な属性が対応づけられた場合は正抽出（Correct），不適当な属性が対応づけられた場合は誤抽出（Wrong）とし，対応するものがあるにも関わらず，検出されなかったマッチング元属性は未検出（Not-Extracted；以下，Not と記す）とした．そして，以下の式で再現率と適合率を求め

8 http://compling.hss.ntu.edu.sg/wnja/
9 http://yougo.rtri.or.jp/dic/

た．なお，正しい対応付けは筆者が手作業で行った．また，参照先オブジェクトの属性を対象に加えない場合と加えた場合の 2 通りについて評価を行った．

$$\text{再現率(recall)} = \frac{\text{Correct}}{\text{Correct} + \text{Not}}$$

$$\text{適合率(precision)} = \frac{\text{Correct}}{\text{Correct} + \text{Wrong}}$$

表 3-3 実験結果（データ 1）

	(a)参照先なし		(b)参照先あり	
	Recall	Precision	Recall	Precision
NM	0.69	0.53	0.92	0.52
ENM	0.71	0.68	0.93	0.67
ENM+TVM	0.71	0.67	0.93	0.74

表 3-4 実験結果（データ 2）

	(a)参照先なし		(b)参照先あり	
	Recall	Precision	Recall	Precision
NM	0.73	0.65	0.75	0.71
ENM	0.76	0.68	0.83	0.75
ENM+TVM	0.76	0.76	0.83	0.83

3.4.3 考察

データ 1 については, 参照関係にあるオブジェクトをマッチングの対象とすることで再現率を向上させることができる．ただし，対象となる属性が増えることで誤検出も増えるために適合率はほとんど向上しない．これに対して，拡張属性名の類似度によるマッチングおよび属性型を考慮した属性値類似度マッチングを行うことで, 候補を適切に絞り込み適合率を 0.74 まで高めることができる．

データ 2 については，参照先をマッチング対象に入れても，再現率はわずかな向上にとどまっているが, 拡張属性名の類似度によるマッチングおよび属性型を考慮した属性値類似度マッチングを適用することで再現率，および，適合率を 0.83 まで高めることができる．参照先をマッチング対象に入れても再現率が向上しないのは，マッチング先

属性が他のオブジェクトに含まれるのではなく, モデル化の考え方自体が異なるためである．以下に例を示す.

(1) 駅並び : 駅並びは, 駅オブジェクトが隣接する駅の識別子を属性として持つ場合, 路線オブジェクトが, 駅の識別子と並び順序を示す番号との組をリスト形式の属性として持つ場合など複数の表現形式がある.

(2) 営業日を示すカレンダー情報 : 平日ダイヤか休日ダイヤかをフラグで示す場合や, 運行する曜日で表現する場合，臨時便などのように特定日付で表現する場合がある．運行する曜日の表現形式にも，カレンダーオブジェクトが曜日を属性として持ち，運行する曜日には 1 を，運行しない曜日には 0 を入れる場合と，曜日を表す識別子を定義し，運行日を示す属性の値として識別子のリストを格納する場合とがある．識別子には，Monday，Tuesday など曜日そのものを用いる場合と，Weekday，または，MondayToFriday のように平日・休日の区別を用いる場合がある.

(3) 時刻表 : 駅における列車の発着時刻で表現する場合と，列車視点で経由する駅の発着時刻で表現する場合がある.

(4) 進行方向 : 上り・下りとして表現する場合と，終着駅や進行方向の地域で表現する場合がある.

上記のような場合に対応するには, データセットが取り得る表現形式を想定してあらかじめ変換ルールを定義し，変換元属性と対応づけておくことで，効率的に対応する情報を取得し，マッチング先オブジェクトの属性値へ変換することができる．例えば，標準的なバスフォーマットのカレンダーオブジェクトで Monday から Friday までの属性値が 1 であれば，統合モビリティサービス共通スキーマの calendar 属性を Weekday とする. このようなドメイン知識に基づく変換ルールを備えておくことで適合率と再現率のさらなる向上が期待できる.

次に，スキーママッチングの再現率，適合率向上の意義について述べる．提案手法を実装したマッチング支援システムでスキーママッチングを行うことで, マッチング元オブジェクトの各属性に対応するマッチング先属性をより高い精度で得ることができる. データ 1 の場合，既存手法では適切なマッチング先属性が対応づけられたのは 79 件, 誤った属性が対応づけられた，または，対応する属性が発見できなかったのは 106 件であった. 提案手法では適切なマッチング先属性が対応づけられたのは 129 件, 誤った属性が対応づけられた，または，対応する属性が発見できなかったのは 56 件であった.

マッチング元スキーマの属性に適切なマッチング先属性が対応づけられている場合には，人が行うべき作業は確認のみである．それ以外の場合，つまり，不適切なマッチング先属性が対応づけられているか，対応づけられたものが無い場合には，人手で対応する属性を探す必要がある．1 件の属性の対応づけを確認する時間を 1 分，人が対応する属性を特定するのに要する時間を 10 分と仮定すると，既存手法では人が行う作業は 1139 分になるが，提案手法では人が行う作業は 689 分になる．データ 1 については，確認と人手による対応づけを含むスキーママッチング作業全体において 450 分の作業時間を短縮することができる．都心部であれば，複数の鉄道事業者，バス事業者のサービスが利用できることが多く，東京では鉄道事業者 9 社，地下鉄・モノレール 6 社，バス事業者は 30 社が運行している．統合対象となるサービスあたり，上記の作業時間短縮となるため，対象サービスが多いほど提案するスキーママッチング手法による作業時間の短縮効果が大きくなる．

また，属性の対応づけを確認する作業においては，類似度が低いほど，適切な属性が対応づけられていない可能性が高い．そこで，対応づけられた属性の組合せを類似度でランキングすることで，慎重に確認すべき属性を絞り込み，確認作業を支援することができると考えられる．特に，提案方式では，拡張属性名類似度と属性型別属性値類似度の 2 種類の類似度を用いるため，それぞれの類似度別にランキングを提示することで，名前の類似度は高いが値の類似度が低い属性の組合せや，その逆の組合せを挙げることができる．このような類似度を用いた人の作業支援方法の検討は今後の課題である．

3.5 結言

異なるスキーマで作成された複数の交通サービスのデータを用いて統合モビリティサービスを実現するため，交通サービスのデータ構造と用語の特徴を考慮したスキーママッチング手法を提案した．提案方式では，マッチング元オブジェクトと参照関係にある他オブジェクトをマッチングの対象に加えることで，マッチングの再現率を向上させ，さらに，参照元と参照先の属性名を接続した拡張属性名を対象とした属性名の類似度によるマッチングと，属性型別の属性値類似度マッチングを組合せて適用することで適合率を高めることが特徴である．

そして，異なるデータソースから取得したオープンデータを，提案手法により独自に定義

した共通スキーマへ対応づけ，既存手法である属性名の類似度によるマッチングと比較して，マッチングの精度を約 20%向上できることを明らかにした．

ただし，提案手法でも対応する属性を発見できない場合が残る．また，適切な属性が対応づけられた場合でも最終的な人による確認は必要である．このため，スキーママッチングの結果によって得られる類似度を活用して，人の行う作業を支援する方法の検討が今後の課題である．

第4章　応答性能保証率向上のための ハイブリッドオートスケール方式

4.1　緒言

本章では, 統合モビリティサービスプラットフォームの応答性能保証率向上とコスト削減とを両立するハイブリッドオートスケール方式を提案する.

インフラ管理コストの削減のため企業システムをクラウド事業者の提供するサーバ上に構築する事例が増えている. 特に近年, 企業基幹系システムをオンプレミスからクラウドサービス上に移行する事例も増加しており, 低コストや柔軟性を特長とするクラウドサービスにおいても, 性能保証や信頼性保証の重要性が増している.

性能や信頼性の保証については, 情報サービスではサービスの提供者と利用者との間で SLA (Service Level Agreement)[63]と呼ばれる契約が結ばれる. SLA では, サービスの応答時間や稼働率に対する目標値が定められ, サービスの提供者は SLA で決められた品質のサービスを提供し, 違反した場合にはペナルティを要求されることもある. サービスの性能を保証するためには, 十分なリソースを確保すれば良いが, それでは必要な時に必要な量だけ使用し, 使用した分だけ支払うというクラウドのメリットを活かすことができない. また, 性能保証のため確保すべきリソース量を, サービス提供開始時に厳密に予測することは難しい. そこで, 性能を保証しつつコストを抑えるように, 双方を意識して割当リソースを調整する仕組みが必要になる.

その代表的な仕組みが, サービスのリソース利用状況に基づいて, 自動的にリソース拡張／削減を実施するオートスケール技術である[35]. ここで, リソースとは, サーバの CPU（Central Processing Unit）コアや周波数, メモリなどの処理能力を決定する要素を指す. オートスケール技術で用いられるリソース拡張／削減方法は, 処理ノードを追加するスケールアップ／ダウンと, ノードの処理能力増強するスケールアウト／インに分類される. スケールアップでは瞬時にリソースの追加が可能であるが, リソースを削減するスケールダウン時にはシステムを停止させなければならないという運用上の難しさがあり, スケールアウトにはリソース追加に分単位の時間がかかり, その間の性能が落ちてしまう可能性がある[36][37]. 既存研究では, スケールアップとスケール

アウトを組合せる研究[45]もなされているが，リソース利用上限の値を運用開始前に決定する必要がある，運用開始前に設定した利用上限値までスケールアップを実施すると，それ以降の負荷上昇に対応できなくなる等の問題がある[46].

本章では，上記課題を解決するため，負荷増大の傾向に応じてスケールアウトとスケールアップを使い分けることで応答性能劣化を防ぎ，かつ，スケールアウト可能なリソース量を維持するハイブリッドオートスケール方式を提案する．提案方式に対して複数の負荷パターンに対する評価を行うことによって，方式の特性やパラメータとその設定方法を明らかにする．

以降，4.2 節でオートスケール方法と課題を，4.3 節で 提案方式を，4.4 節で評価実験とその結果および考察を述べ，4.5 節で結論を述べる．

4.2 オートスケール方式とその課題

4.2.1 従来方式

本章で提案する方式は，パブリッククラウド環境に置かれた EC（Electronic Commerce）サイトなどのように不特定多数のユーザがアクセスサービスを想定している．このようなサービスは性能劣化がサービスの収益に影響するため，性能劣化を未然に防ぐことが望ましいが，パブリッククラウド環境ではリソースを利用した時間に対して料金が発生するため，余分にリソースを割り当てておくことはコスト増大になる．そこで，本章では，サービスの提供者がリソースの利用量を必要，かつ，最低限にすることで，応答性能を維持しつつコストを抑えられる方法を提供し，エンドユーザには応答性能の高いサービスを提供することを目指す．

また，管理対象のサービスは，Web/AP（Application）サーバと DB（Database）サーバから成る Web 三階層システムとする．Web 三階層システムは，Web/AP サーバを永続的なデータを持たないステートレスな構成にすることが容易であるためオートスケールとの相性が良く，多くのアプリケーションがこの Web 三階層を基本として構成されている．このようなシステムに対する従来のオートスケールルールを図 4-1 に示す．オートスケールルールは，オートスケールの実行を決定する条件部と，管理対象システムの処理能力を変更するアクションから成る IF-THEN ルールで記述される．図 4-1 に示すオートスケールルールの条件部では，負荷増大の検知のために用いる 1 つ以上のメトリックやイベントと，

それらに対するしきい値と演算子が定義される．メトリックとは，サービスの性能やサービスを提供するシステムの状態を表す指標であり，具体的にはサービスの応答時間やシステムのリソース使用量などである．そして，アクションには，条件部に対してあらかじめ決められた変更操作と，変更対象リソース，リソース量が定義される．図 4-1 の (a)スケールアップでは，応答時間が 1 秒を超過した場合に，あらかじめ決められた量のリソースを追加することが定義されており，図 4-1 の (b)スケールアウトでは，メトリックが上限しきい値を超過した場合に，仮想サーバ 1 台を追加することが定義されている．オートスケール機構は，このようなルールを読み込んで，条件部で定義されたメトリックを一定間隔で取得して評価し，それらの値が上限しきい値を超過した場合には負荷増大が発生していると判断し，スケールアップまたはスケールアウトを実行する[35].

スケールアップ／ダウンと，スケールアウト／インの概要を図 4-2 に示す．スケールアップ／ダウンは，システムの単一ノードにリソースを追加／削除することによりスケールを変化させる方法である．待ち行列理論からはノードを増やすよりも，1 つのノードにリソースを追加する方がより性能を向上させることができることが明らかにされている[39].

```
IF ＜Metric Name＞ { Comparison Operator } <Threshold>
 THEN <Action> <Resource Amount> <Resource Name>

(a) Scale-up

if ( response time > 1 second ) then
    add 1 vCPU
else if ( response time < 0.5 second ) then
    remove 1 vCPU
end if


(b) Scale-out

if ( response time > 1 second ) then
    add 1 VM
else if ( response time < 0.5 second ) then
    remove 1 VM
end if
```

図 4-1　スケールアップとスケールアウトルール

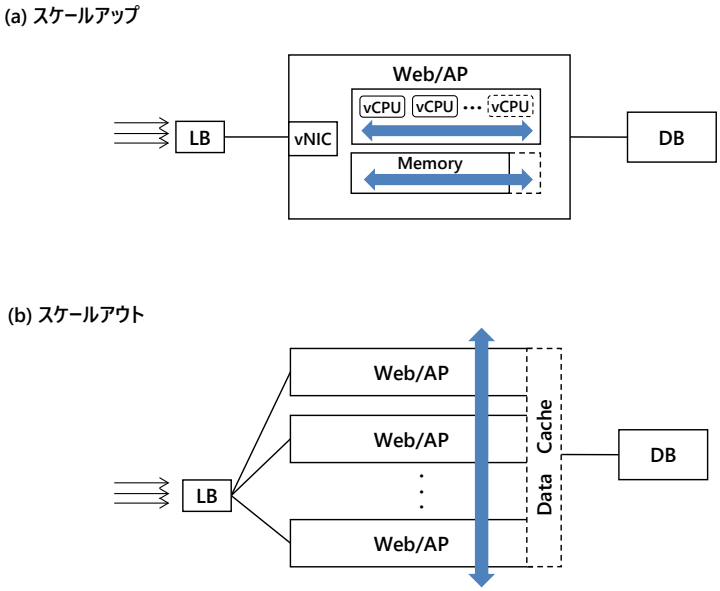

図 4-2　スケールアップとスケールアウト

物理サーバに対して CPU やメモリを増減設する場合は，サーバの停止と起動を伴うためシステム停止時間が発生するが，サーバ仮想化機構を利用したシステムでは，ソフトウェア設定により無停止でリソースを追加することが可能である[62]．しかし，仮想サーバ上で動作するソフトウェアへの割当リソース量に応じて，同時実行スレッド数やコネクション数など性能に影響するパラメータを設定している場合も多く，その場合は，一定以上リソースが増えても使いきれない．このために，一台の仮想サーバに追加して活用できるリソースには限界がある．本章では，これをリソース利用上限値と呼ぶ．さらに，スケールアップは可能でも，システム停止無しでスケールダウン，つまりリソースを削減することができない．例えば CPU リソース量削減のためには仮想サーバ上で動作する OS (Operating System)を含めた全実行中プログラムの CPU 割当て変更が必要であり，システム停止無しでは実現不可能である．このため，スケールアップ後に負荷が低下する場合には不向きである．

一方，スケールアウト／インはシステムにノードを追加／削減することによりスケーラビリティを変化させる方法である．一般的に，大規模な Web システムでは Web/AP サーバを複数台構成とし，サーバ間の共有データキャッシュによりセッション情報を共有し，ロードバランサを利用してタスクを振り分ける．そして負荷変動に応じて Web/AP サーバ数を変化させることでスケールを変更する．ただし，サーバ仮想化機構

を利用したシステムの場合でも，増設する仮想サーバを構築するためには数分単位の時間を必要とするため，負荷の急増に間に合わない場合があることが指摘されている[36].

4.2.2 従来方式の課題

情報システムの SLA を定めたガイドラインでは，サービス性能の達成に対する SLA 要件として，下位レベルで全リクエスト数の 80%，中位レベルで 90%，上位レベルで 100 % に対して目標値を達成することが求められている[63].

これに対して，従来技術のスケールアウトでは，リソースの追加に分単位の時間を要するため，スケールアウトが完了するまでの間にサービス応答性能が目標値を超過してしまうことがある．一方，負荷変動に対してスケールアップで対処することで，スケールアウトと比較して目標値の保証率を高めることができるが，スケールアップ後に負荷が低下する場合には，リソースを余らせることになってしまう．このような課題に対して，スケールアップとスケールアウトを組合せる研究もなされているが，既存研究ではリソース利用上限の値を運用開始前に決定し，その上限値にもとづいてスケールアップ設定を行うため，予測が外れた場合には負荷上昇に対応できなくなる．

4.3 ハイブリッドオートスケール方式

4.3.1 ハイブリッドオートスケール方式の概要

本章では，上記課題を解決するためにスケールアップとスケールアウトとを組み合わせたハイブリッドオートスケール方法を提案する．まず，ハイブリッドオートスケール方式の概要を，図 4-3 を用いて説明する．提案方式では，応答性能やリソース使用量の上限しきい値による負荷増大検出時(t)に，スケールアウトによる仮想サーバ追加完了時点(t+Δt)のリソース使用率予測値を求め，予測値がリソース使用率の上限値に達する場合にはスケールアウトでは間に合わないと判断し，スケールアップを実行し性能劣化を防ぐ（図 4-3 (1)-(a)）．リソース使用率の上限値を超えない場合には，スケールアウトを実行する（図 4-3 (1)-(b)）．なお，リソース使用率の上限値とは，中位レベルの SLA である応答時間保証率 90%が達成できなくなる時点の使用率を指している．

そして，負荷減少検出時には，スケールアウト／インにより仮想サーバサイズと仮想

サーバ数を調整することでリソースを削減する．具体的には，管理対象システム全体に対してスケールアップで追加し得るリソース量であるスケールアップ能力が一定量残っているか確認し，不足する場合には，既にスケールアップ済であり更なるリソース追加が不可能な仮想サーバを削除して，リソース追加が可能な仮想サーバを追加する．このようにして，サービスを提供するシステム全体としてのスケールアップ能力を確保する（図 4-3 (2)-(a)）．それ以外の場合には，スケールインを実行する（図 4-3 (2)-(b)）．

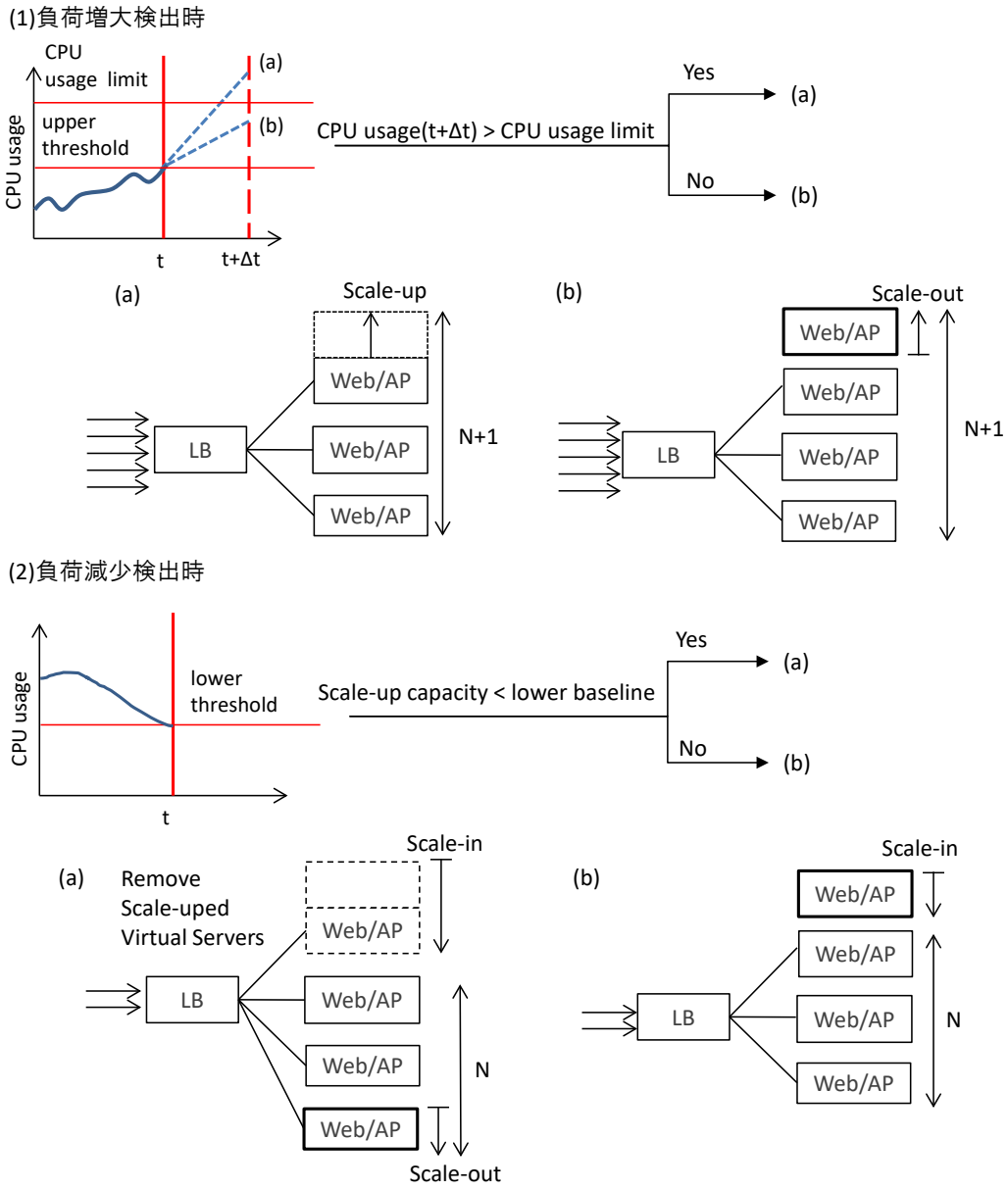

図 4-3　ハイブリッドオートスケール方式の概要

4.3.2 オートスケールルール

提案するハイブリッド方式のオートスケールルールを図 4-4 に示す．図 4-4 に示すオートスケールルールの条件部では，負荷増大をサービス応答時間と CPU 使用率によって判断する．我々の過去の調査 によると，典型的な企業の業務アプリの一つである SugarCRM (Sugar Customer Relationship Management))では，汎用スクリプト言語 PHP(Hypertext Preprocessor)で書かれたプログラムによる動的ページ生成のため CPU リソース量の不足がアプリケーション全体のボトルネックになる[54]．そこで提案方式では，このようなアプリケーションを対象として，応答時間と CPU リソースに着目し，原因となる事象である CPU リソース不足とその結果である応答性能劣化が同時に発生した場合に，CPU リソースの不足によって応答時間の増大が引き起こされたと判断する．なお，提案方式は条件部で用いるメトリックに依存するものではない．アプリケーションによってボトルネックとなるリソースは異なるため，メモリがボトルネックとなるアプリケーションを対象とする場合には，メモリの不足とサービス応答時間の増大を条件としてオートスケールを実行することが可能である．

まず，負荷増大検出時のアクションについて説明する．負荷増大検出時は，まず，スケールアウトを実行した場合に仮想サーバ追加にかかる所要時間Δt 経過後の CPU 使用率を予測する．そして，その値が CPU リソース使用率の上限しきい値を超える場合に，急激な負荷増大が発生したとみなし，CPU またはメモリをあらかじめ決められたスケール変更単位 s 追加するようスケールアップを実行する．一方，下回る場合には，スケールアウトで間に合うと考え仮想サーバを追加する．スケールアップ実行後には，現状のスケールアップ能力を確認し，この能力があらかじめ定義した下限基準値を下回る場合には，リソース追加可能な最小構成の仮想サーバを 1 台追加する．

続いて，負荷減少検出時のアクションについて説明する．負荷減少検出時は，まずスケールアップ能力を確認し，スケールアップ能力が下限基準値を下回る場合には，リソース量 1 の仮想サーバを n-s 台数追加後にリソース量 n である仮想サーバを削除する．削減する仮想サーバとしては，リソース追加済のものを優先的に選択する．これにより，システムからリソース量 s を削減し，かつ，スケールアップ能力のある仮想サーバに入れ替えることでそれ以降の負荷増大に備える．

一方，負荷減少検知時に，スケールアップ能力が下限基準値以上である場合には，従

来方式のスケールインと同様に仮想サーバ 1 台を削除する.

```
if (response time > response time upper threshold
       & cpu usage > cpu usage upper threshold) then
       if (predicted cpu usage ＞ cpu usage limit) then
              add resource unit(s)
       else if
              add VM(s)
       end if
       if (scale-up capacity ＜ lower baseline) then
              add VM(1)
       end if
else if (response time < response time lower threshold
       & cpu usage < cpu usage lower threshold) then
       if (scale-up capacity ＜ lower baseline) then
              add n-s VM(1)
              remove VM(n)
        else
              remove VM(s)
       end if
end if
```

図 4-4　ハイブリッドオートスケールルール

4.3.3 システム構成

提案方式を実現するためのシステム構成を説明する．システム構成は，図 4-5 に示すようにオートスケール機構と管理対象システムに分かれる．オートスケール機構は，監視モジュール，負荷予測モジュール，スケーリングプラン生成モジュール，構成管理モジュール，構成管理 DB から構成される．オートスケールルール評価モジュールは，監視モジュールが収集したサービス応答時間と CPU 使用率を取得し，および，負荷予測モジュールによって求めた Δt 経過後の CPU 使用率，構成管理 DB から取得した現在のスケールアップ能力を参照し，4.3.2 で述べたハイブリッドオートスケールルールに基づいてオートスケールアクションを決定する．次に，スケーリングプラン生成モジュールは, 構成管理 DB に格納された各仮想サーバの起動時の仮想サーバタイプと追加済リソース量を参照して，あらかじめ決められたリソース利用量上限値と比較し，リソース追加や削除アクションの対象となる仮想サーバを決定することでアクションを具体化する. 最後に, 構成変更モジュールが管理対象システムに対して仮想サーバ追加，削除，リソース追加指示を出し，構成管理 DB に格納されている仮想サーバ構成とス

ケールアップ能力を更新する．

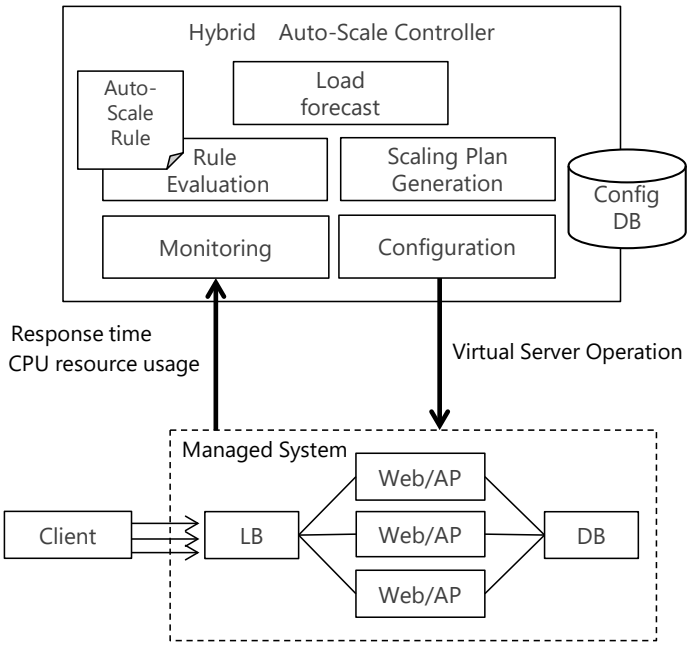

図 4-5　システム構成

4.4　評価実験

提案方式の有効性を検証するために，提案方式と既存方式であるスケールアウト／インの２つの方式について，評価実験をおこなった．負荷パターンは，実サービスに対するリクエスト数推移から作成したパターン 2 種類を適用した．また，オートスケール制御は，オートスケール条件のしきい値と，一度に追加・削除を行うリソース量によって効果とコストが変わるため，オートスケール条件を 5 通り，一度に追加・削除を行うリソース量を 2 通りに変化させ，計 50 通りの評価をおこなった．なお，スケール変更単位とは仮想 CPU の個数である．

表 4-1　仮想サーバインスタンスタイプ

Server		仮想 CPU[個]	メモリ[GB]
Load Balancer		1	1
Web	Small	1	2
	Medium	2	4
Database		2	8

4.4.1 実験の条件

(1) 実験システム構成

実験システム構成について説明する．管理対象システムは，物理サーバ 5 台から成る仮想化環境上の仮想サーバで構成される．各物理サーバ は CPU が Intel Xeon E5430 2.66GHz（8 コア），メモリが 12 GB の PC サーバであり，VMware vSphere 5.1 により，物理サーバ 5 台を 1 つの仮想化環境として利用する．各仮想サーバ（図 4-5 の破線部分）は 5 台の物理サーバのいずれかで稼働し，リソースに空きのある物理サーバを選択して起動させている．なお，これらの仮想サーバのイメージは，Fiber Channel を介して接続された外部ストレージに格納されている．

管理対象システムは，ロードバランササーバ（LVS）で，重み付き最小コネクション方式による負荷分散を行う．ロードバランサの配下には Web サーバ（Apache）を最小構成で 2 台，最大構成で 10 台配置する．各 Web サーバは 1 つのデータベースサーバ（MySQL）と接続している．仮想サーバのサイズを表 4-1 に示す．これら仮想サーバタイプは AWS (Amazon Web Services)が Web アプリケーション向けに提供しているインスタンスとほぼ同等のリソース量を備えるよう定義した. Web/AP サーバには 2 種類 (small/medium) のサイズを用意し, 初期構成は small サイズのサーバ 2 台とした. そして，従来方式の評価では，スケール変更単位が 1 の場合は small サーバ，2 の場合は medium サイズのサーバを追加・削除する．提案方式の評価では，各仮想サーバタイプは, 仮想 CPU 数を 4vCPU まで増加させることができるものとする. 前述の物理サーバは，これらの仮想サーバを同時稼働させるための必要なリソースとして，CPU40 コア，メモリは 60GB を提供できるようにした．

この管理対象システム上で動作させるアプリケーションとしては, オープンソースの

CRM（Customer Relationship Management）アプリケーションである SugarCRM を採用する．SugarCRM は企業内で用いられる業務アプリケーションであるが，Web/AP サーバから成る 3 階層システムである点，Web サーバとして Apache を用い，アプリケーションの実装に PHP を用いている点，DB サーバに MySQL を用いている点では，多くの EC サイトと同様である．さらに，PHP の処理内容は，DB サーバに対する検索，読み出し，および，それらの情報を用いた動的なページの作成である点も同じである．SugarCRM と EC サイトとの相違は，写真や動画などの有無であるが，実際のサービスでは，それらのコンテンツは高速化のために CDN（Content Delivery Network）やキャッシュサーバから提供されることが多く，Web/AP サーバの負荷にはならないことから負荷特性に大きな相違はないと考え，データの準備が容易である SugarCRM を採用した．

次に，オートスケール機構の監視モジュールとしてはオープンソースの監視サーバである Zabbix を用いた．そして，オートスケールルール評価モジュール，負荷予測モジュール，スケーリングプラン生成モジュール，構成管理モジュール，構成管理 DB は J2EE にて実装した．オートスケールルール機構は，SQL (Structured English Query Language) によって Zabbix の MySQL データベースからサービス応答時間，および，各仮想サーバの CPU 使用率を取得し，負荷増大や減少を検知した場合には，仮想化環境管理機構である VMware vCenter に対して vCenter API を発行して仮想サーバ追加，削除，リソース追加指示を出す．また，ロードバランサに対して，追加した仮想サーバタイプや追加したリソース量に応じて，操作対象サーバへのリクエスト転送の重み付け変更要求を発行する．

(2) 評価項目

SugarCRM はリクエスト数増加に伴い主に CPU 負荷が増大する特徴を持つため，本実験では，提案するハイブリッドオートスケール方式と，既存手法であるスケールアウト方法を用いて CPU リソースに対するオートスケールを行い，サービス応答時間保証率，CPU とメモリの消費量を評価する．サービス応答時間保証率は，評価対象期間中の全リクエストに対して，目標値以内に応答が返ってきたリクエストの割合である．CPU 消費量は使用した仮想 CPU 数と時間（秒）との積で計算し，メモリはメモリ量と時間（秒）との積で計算する．さらに，仮想 CPU 数と CPU 使用率の推移も計測し，

CPU リソース使用量の変化を確認する．

本来，Web アプリケーションの性能は CPU，メモリ，ネットワーク I/O，ディスク I/O によって決まるため，これらについて評価が必要である．ただし，メモリとディスク I/O，ネットワーク I/O は仮想サーバ追加・削除時に一緒に増減するため，メモリ消費量から他 2 つを類推できると考え，CPU とメモリ消費量を評価することにした．なお，本実験では CPU のみの連続追加（スケールアップ）は最大 3vCPU であり，最小構成の 2vCPU から一気に 3vCPU 追加された場合でも，その時点での CPU リソース 5vCPU を使い切る負荷に対して，他のリソースには余裕がありボトルネックにはならないことを確認している．

(3) オートスケール条件

オートスケール条件のしきい値について説明する．しきい値については，低く設定すれば負荷増大に対して早めにリソースを追加でき，性能保証しやすくなるが，リソースを浪費する可能性も高くなる．一方，高く設定すると負荷増大に間に合わず性能保証しづらくなる．そこで，本論文では，しきい値を表 4-2 に示すように 5 通りに変化させる．

このしきい値を決定するため，CPU 使用率と応答性能の関係を測定する予備実験を行った．対象システムの Web/AP サーバの合計仮想 CPU 数が 2 から 8 の場合について，リクエスト数を段階的に増加させ評価を行った．各仮想 CPU 数の場合について，スケールアウトやスケールアップを実行することを考慮して，取りうるインスタンスの組み合わせを複数通り作成した．例えば，仮想 CPU 数 4 の場合は，small サーバ 2 台の最小構成に small サーバ 2 台を追加した構成と，medium サーバ 1 台を追加した構成を作成した．その結果，各パターンに共通して，応答時間の最低値が 0.3 秒前後であること，応答性能増大傾向と CPU 使用率は正の相関を持ち，応答時間が 0.35 秒を超え始めるのは，55%を超過した場合であり，0.5 秒を超え始めるのは CPU 使用率が 70%を超過した場合であり，CPU 使用率が 80%を超過すると急激に応答性能が劣化するという関係が得られた．この関係から，劣化が現れ始める点から劣化傾向が加速する点までの間を 5 点取り上げてしきい値とした．また，リソース使用率の上限値は 80%とした．さらに，応答時間の下限しきい値を 0.4 秒，CPU 使用率の下限しきい値を 40%とし，それらの条件を満たす場合は，オートスケール機構はリソースの削減を実施する．

なお，負荷増大検知に用いる CPU 使用率やサービス応答性能は，瞬間的な値の増減に左右されないよう 1 秒間隔の計測値 5 点の移動平均を取る. 以上のオートスケールルールを，本実験では 10 秒間隔で評価する.

表 4-2 ハイブリッドオートスケールのしきい値

	上限しきい値		下限しきい値	
	応答時間	CPU 使用率	応答時間	CPU 使用率
Rule0	0.35 sec	55%	0.4 sec	40%
Rule1	0.4 sec	60%	0.4 sec	40%
Rule2	0.5 sec	70%	0.4 sec	40%
Rule3	0.6 sec	80%	0.4 sec	40%
Rule4	0.65 sec	85%	0.4 sec	40%

(4) オートスケールアクションと選択ロジック

オートスケール実行時に追加／削除するリソース量は，1vCPU (small サーバ) 単位，2 vCPU (medium サーバ) 単位の 2 通りとする．なお，本実験環境では新規仮想サーバの追加に要する時間は 120 秒であった．これにより，スケールアウトとスケールアップの選択ロジックは，CPU リソースの使用量の増加率を元に，線形外挿によってスケールアウトに要する時間である 120 秒後の CPU リソース使用率を求めて, その値が CPU 使用率の上限しきい値を超過する場合にスケールアップを選択する．また，スケールアップ能力の下限基準値は 3vCPU とする.

(5) 負荷パターン

EC サイトやブログサイトなどでは，生活時間帯に合わせて緩やかに負荷が推移するが，広告などのキャンペーン実施後や，ソーシャルメディアでの注目度が高まった場合には急激に負荷が増大することがある．特に近年は，携帯端末の普及により通勤時間帯や昼食時間帯にもアクセスが急増するケースが現れている．そこで本実験では，上記のような特徴を持つ負荷パターンである World Cup'98 の Web サイトに対するリクエスト数推移として公開された 7 日分のデータ[10]を基に作成した 2 通りの負荷パターンを用いる．World Cup'98 の Web サイトアクセス数推移は，開催期間中で試合があった日

10 http://ita.ee.lbl.gov/html/contrib/WorldCup.html

と試合が無かった日とで負荷変動パターンが異なる．試合が無かった日は，夕方から夜にかけて負荷が増大し，深夜にかけて緩やかに減少する．一方，試合があった日は，試合の前後に急激な負荷増大が発生している．この World Cup'98 Web サイトに対するアクセスは read が主であり，本研究の対象アプリケーションである EC サイトや経路検索サイトに対するアクセスは write が含まれるという違いがある．しかし本研究では，平常時の負荷パターンと急激な負荷増大が発生するバースト負荷パターンに対して提案方式の効果を検証し比較するために，双方のパターンを含むことが重要であると考えて World Cup'98 Web サイトへのアクセス数推移を採用した．作成した 2 通りの負荷パターンである平常時負荷パターン，バースト負荷パターンを図 4-6 に示す．平常時負荷パターンは，緩やかに上昇し減少するパターンである．バースト負荷パターンは，急激に上昇し，急激に減少した後でゆるやかに減少する．どちらのパターンも経過時間 240 秒で負荷が大きく上昇するが，平常時負荷パターンの増加割合に比べて，バースト負荷パターンの増加割合は 2 倍である．この 2 通りの負荷パターンに応じたリクエストを，負荷生成プログラム（JMeter）を用いて発生させる．発生させるリクエストは，SugarCRM に対する一般的な操作であるログイン，ドキュメントの検索，ドキュメントの読み出し，ドキュメントに対する書き込み，ログアウト処理を連続して行うものである．

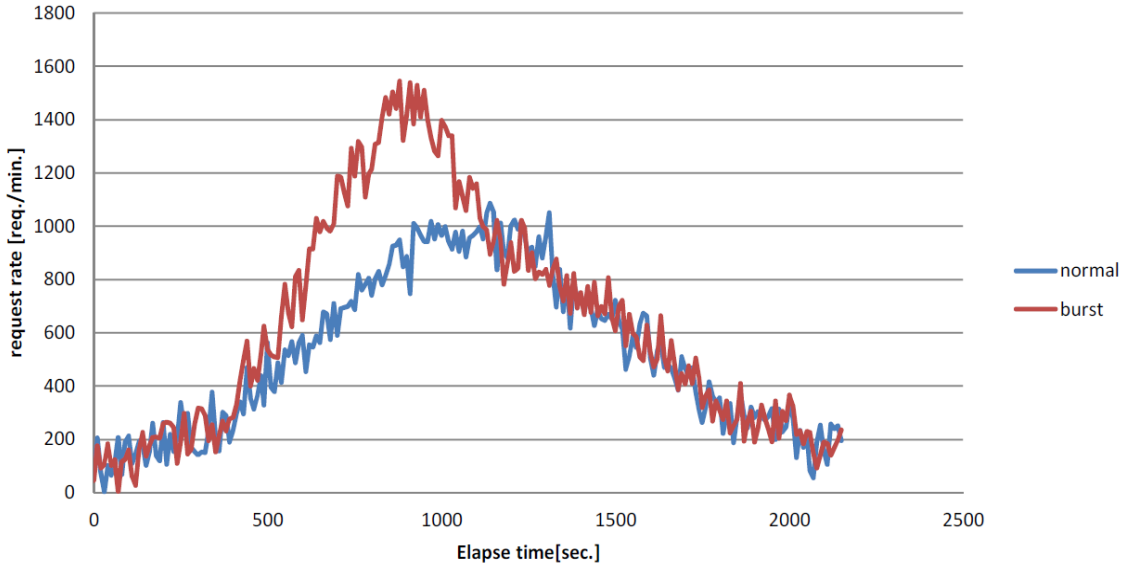

図 4-6 リクエストパターン

4.4.2 平常時負荷パターンを用いた実験結果と考察

本節では，平常時負荷パターンに対する実験結果と考察について述べる．Web サイトの応答時間についてユーザビリティの観点で考察した記事[11]によると，ユーザの思考を途切れさせないための応答時間は最長 1 秒であると述べられている．ただし，ユーザが体感する応答時間は，システムの応答時間にネットワーク遅延が加算されるため，この遅延を考慮してサービス応答時間の目標値は 0.8 秒以内とする．

まず，従来方式で最も高いサービス応答性能保証率を達成したルール 0 と，スケール変更単 1vCPU の場合，提案方式で最も高いサービス応答時間保証率を達成したルール 0，スケール変更単 1vCPU の場合の仮想 CPU 数，CPU 使用率の推移を図 4-7，図 4-8 に示す．この CPU 使用率は，全 Web/AP サーバの平均値であり，各仮想サーバの CPU 使用率としては 1 秒間隔で計測した CPU 使用率の過去 5 回分の平均値を用いている．また，仮想サーバの CPU 使用率は，物理サーバと同様な方法で取得すると同一物理サーバに同居する他仮想サーバの影響を受けるため，仮想化環境管理サーバである vCenter から取得する．

平常時負荷パターンの場合，従来方式では，負荷が増大し始める 200 秒前後から 7 回のスケールアウトが実行される．一方，提案方式では，まず 3 回のスケールアップが実行され，負荷の伸びが収まるとスケールアウトが実行されることで CPU 使用率を抑え，負荷減少時は small サイズ仮想サーバの追加とスケールアップ済仮想サーバの削除により，スケールアップ能力を維持しつつ段階的にスケールを縮小している．

11 Nielsen Norman Group: Website Response Times (online),
available from <http://www.nngroup.com/ articles/website-response-times/>

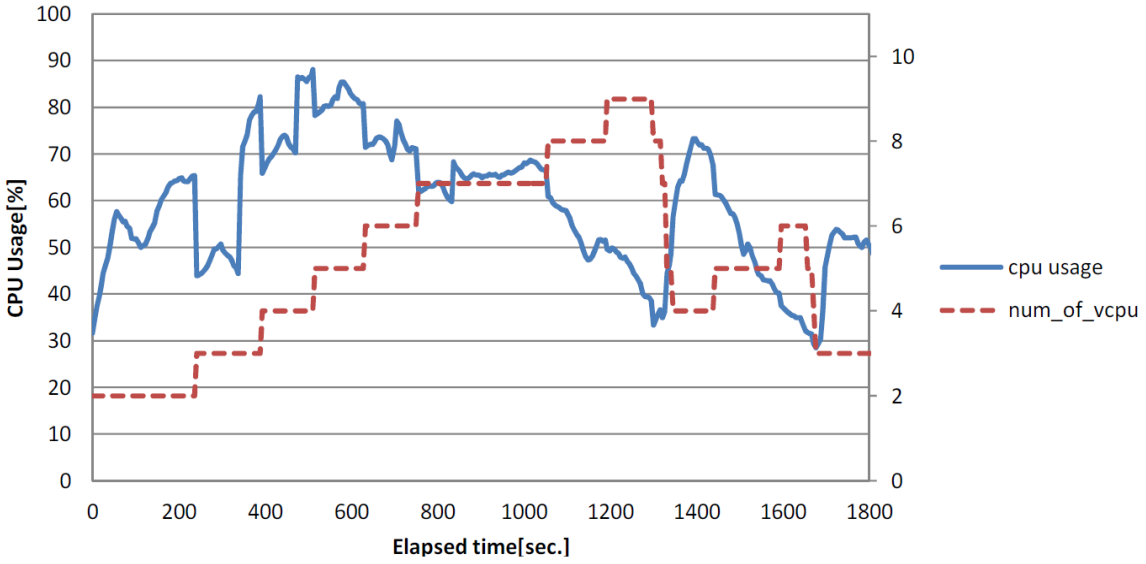

図 4-7　CPU 使用率と仮想 CPU 数
（従来方式，ルール 0，スケール変更単位 1）

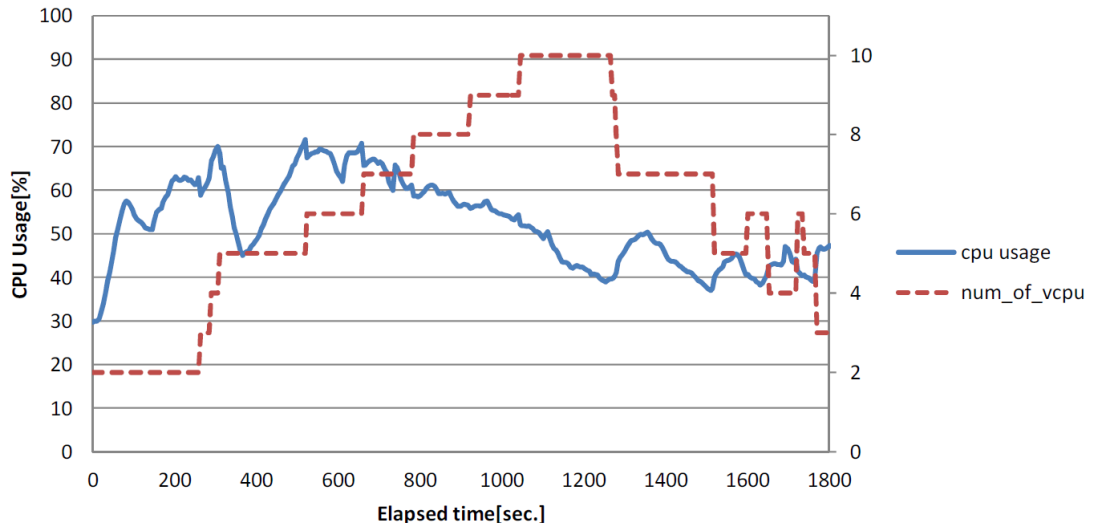

図 4-8　CPU 使用率と仮想 CPU 数
（提案方式，ルール 0，スケール変更単位 1）

次に，従来方式と提案方式の検索リクエスト応答時間推移を図 4-9，図 4-10 に示す．

ここでは，SugarCRM に対して実行した操作の内で応答時間が最も長い検索リクエストの応答時間を取り上げ説明する．従来方式では，負荷が上昇し始める 300 秒前後か

ら仮想サーバ追加が完了する 500 秒過ぎまでの間，応答性能が著しく劣化する．これに対して提案方式では，この負荷上昇時の応答性能劣化が短時間で抑えられている．

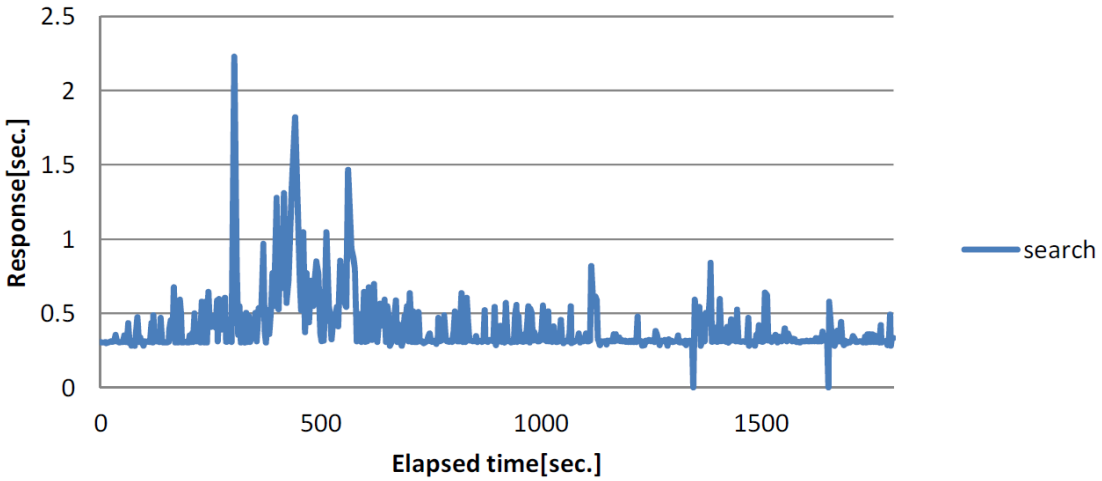

図 4-9　応答時間（従来方式，ルール 1，スケール変更単位 1）

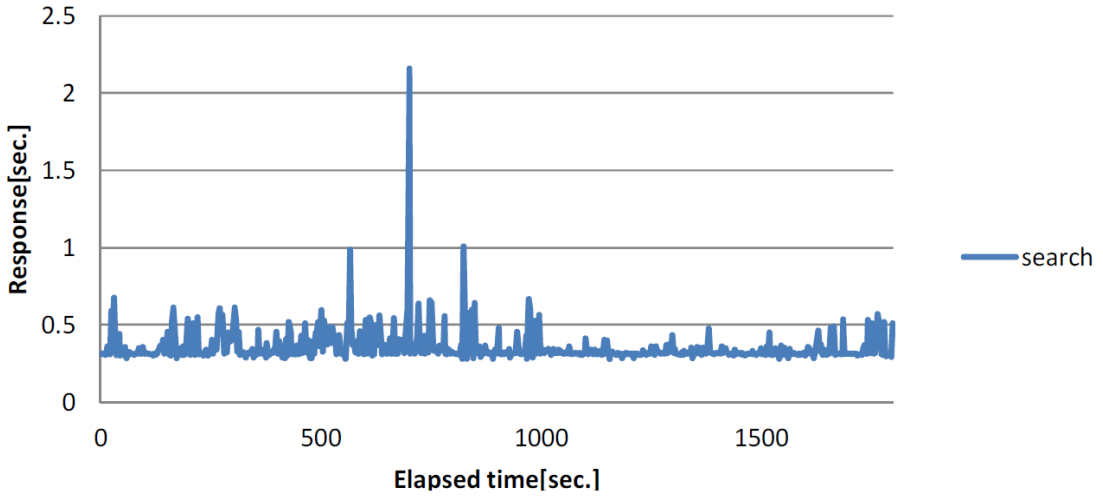

図 4-10　応答時間（提案方式，ルール 0，スケール変更単位 1）

従来方式，提案方式の各条件のサービス応答時間保証率と，リソース消費量を図 4-11，図 4-12 に示す（図中，O は従来方式のスケールアウト，H は提案方式のハイブリッ

ドを示し, 数字はスケール変更単位を示す). これらの値は 5 回の計測の平均値である.

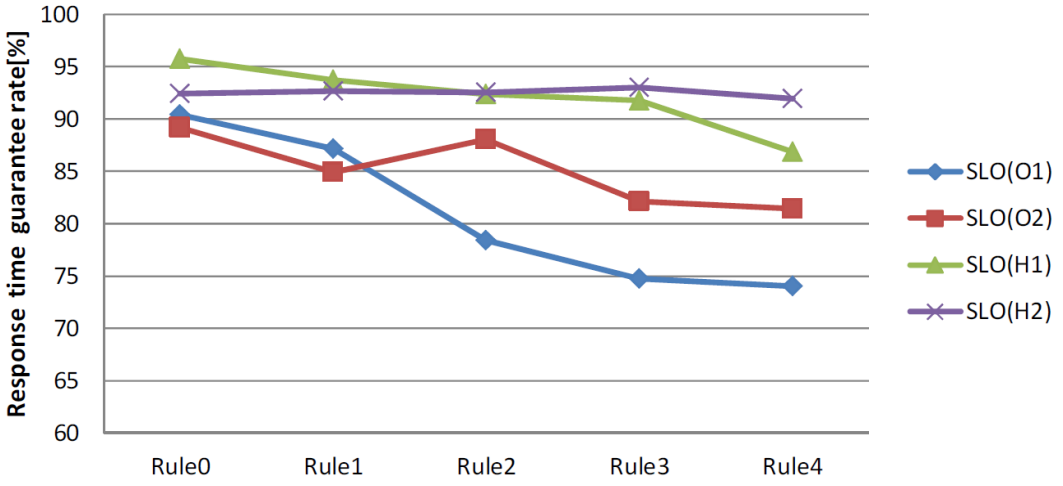

図 4-11 サービス応答時間保証率

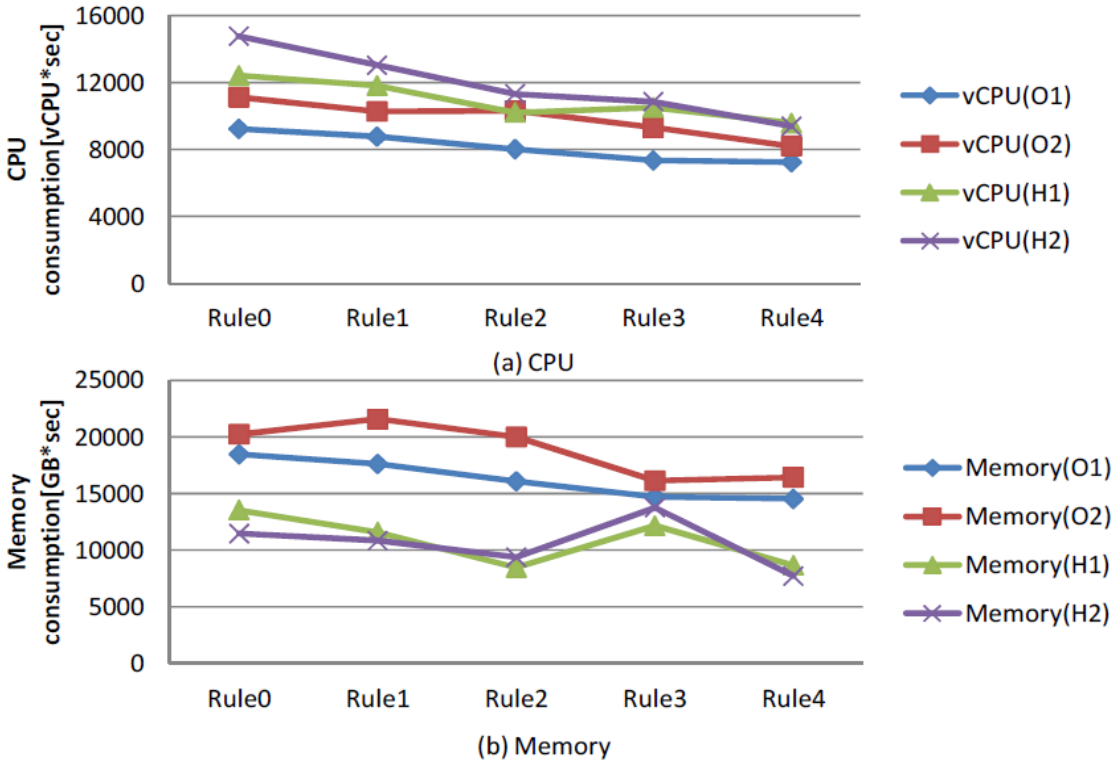

図 4-12 リソース消費量

従来方式においてスケール変更単位１，すなわち 1vCPU の場合，および，提案方式においてスケール変更単位２，すなわち 2vCPU の場合については，しきい値を低くしたルール 0 が最もサービス応答時間保証率が良い．一方で，従来方式においてスケール変更単位２の場合は，ルール１のサービス応答時間保証率がルール２よりも劣っている．これは，CPU 使用率 55%の中程度の負荷の段階で 2vCPU の仮想サーバを追加したため，リソース使用量が一旦低下してスケールダウンの条件である 40%以下になってしまい，スケールアウト実行直後にスケールダウンが実行されるフラッピングが発生したためである．スケールアウトでもしきい値を低くしたり追加リソース単位を大きくすることで性能向上可能であるが，しきい値を低くしすぎたり，追加するリソース単位を大きくとりすぎることでこのようなフラッピングを引き起す場合がある．

提案手法の場合でも，しきい値が高いルールになるほどサービス応答時間保証率が低下していくが，スケール変更単位が 2vCPU の場合よりも，1vCPU の場合が低下の傾向が著しい．この点について考察する．提案方式のスケールアップ能力は，スケールアップを行うと減少しスケールアウトを実行すると増加する．オートスケールの上限しきい値が低い場合は，しきい値超過からリソース使用率の上限値に到達するまでに余裕があるためスケールアウトも実行されるが，上限しきい値が高いとスケールアップを連続して実行することになり，スケールアップ能力が限界に達してしまうことがある．ただし，1vCPU 単位のスケールアウトでは 1vCPU，2vCPU 単位でのスケールアウトの場合は 2vCPU 単位で拡張されるため，2vCPU 単位で追加を行う場合の方が一回のスケールアウトで追加されるスケールアップ能力が大きく限界に達しづらい．このような理由により，1vCPU 単位よりも 2vCPU 単位で追加を行う方が，サービス応答性能保証率の劣化が抑えられる．

次に，各方式の最良値を比較すると，サービス応答時間保証率は従来方式で 90.4%，提案方式で 95.7%であり，5.3%改善される．この時のリソース消費量は，従来方式で 9223[vCPU 秒]，提案方式で 12416[vCPU 秒]であり，約 34%増加している．ただし，従来方式のルール２，スケール変更単位 2vCPU の場合と，提案方式のルール 2，スケール変更単位 1vCPU の場合とを比較すると，保証率と CPU リソース使用率はほぼ同等である．提案方式では負荷減少時のスケールインとスケールアウトの過程で従来方式よりリソースを消費するが，この条件では負荷減少時の消費量は従来方式でしきい値を低くとることによって増加するリソース消費量より若干少ないことが分かる．一方，メモ

リ使用量は従来方式で 18466[GB 秒]，提案方式で 13558[GB 秒]であり，約 26%削減される．これは，従来方式では仮想サーバ追加により CPU リソースとメモリリソースが常に同時に追加されるが，提案方式では不足する CPU リソースのみが追加される場合があり，メモリ使用量が削減されるためである．同様にディスク I/O とネットワーク I/O についても，提案方式の方が少ない消費量になると考えられる．

以上より，提案方式では，従来方式と同等のサービス応答時間保証率を達成するために必要なリソース量はほぼ同じであり，従来方式では達成できないサービス応答時間保証率を実現することができる．

4.4.3 バースト負荷パターンを用いた実験結果と考察

本節では，バースト負荷パターンに対する実験結果と考察について述べる．まず，従来方式で最も高いサービス応答時間保証率を達成したルール 1，スケール変更単位 2vCPU の場合と，提案方式のルール 0，スケール変更単位 1vCPU の場合の仮想 CPU 数，CPU 使用率の推移を図 4-13，図 4-14 に示す．

従来方式では，段階的にスケールアウトを実行しているが，負荷上昇にスケールアウトが間に合わず，CPU 使用率が 100%に達している．さらに，スケール変更単位が大きいために，一旦負荷が低下した際にスケールインを実行し直後に負荷が再度上昇している．これに対して提案方式では，最初の負荷増大検知時にスケールアウトを実行し，負荷が高くなった段階でスケールアップを実行し，CPU 使用率をほぼ 70%以下に抑えている．

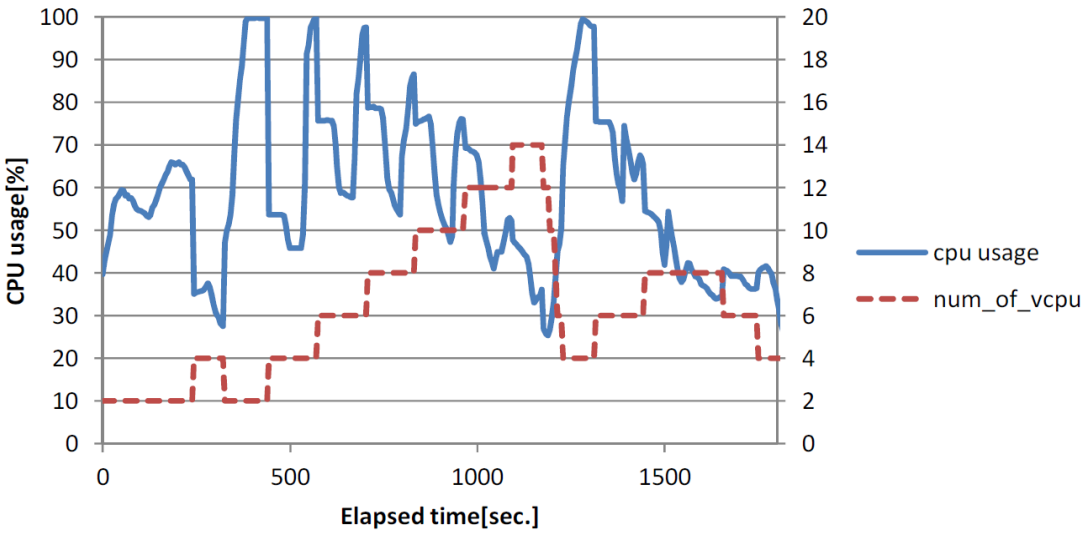

図 4-13　CPU 使用率と仮想 CPU 数（従来方式，ルール 1，スケール変更単位 2）

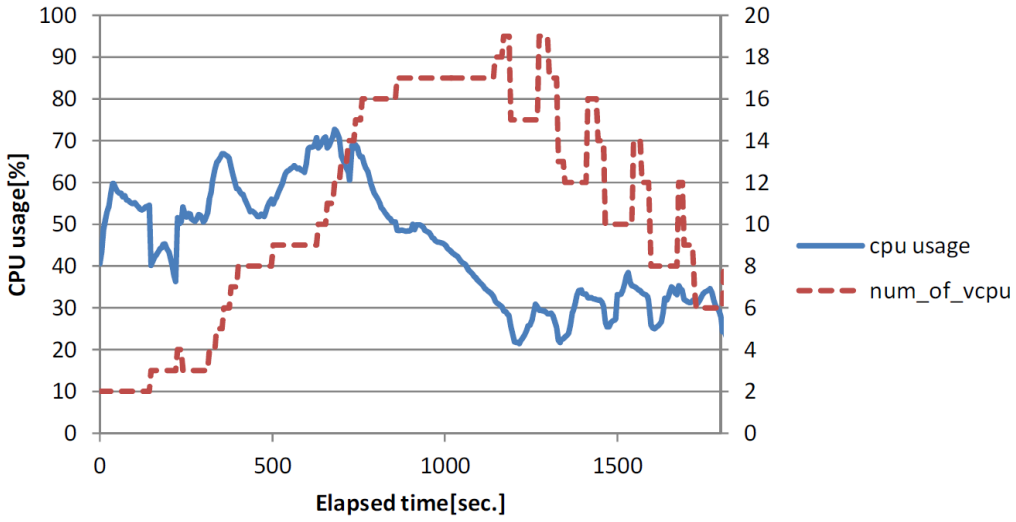

図 4-14　CPU 使用率と仮想 CPU 数
（提案方式，ルール 0，スケール変更単位 1）

次に，従来方式で最も高いサービス応答時間保証率を達成したルール 1，変更単位 2vCPU の場合と，提案方式において最も高いサービス応答時間保証率を達成したルール 0，スケール変更単位 1vCPU の場合の検索リクエスト応答時間推移を図 4-15，図 4-16 に示す．

バースト負荷パターンの場合は平常時負荷パターンに比べて負荷上昇の傾きが倍に

なるため，従来方式では 300 秒前から応答性能が急激に劣化する．また，スケール変更単位が大きいためにスケールイン後に瞬時的な性能劣化が発生している．一方，提案方式では，最初にスケールアップを実行することで性能劣化を抑えている．

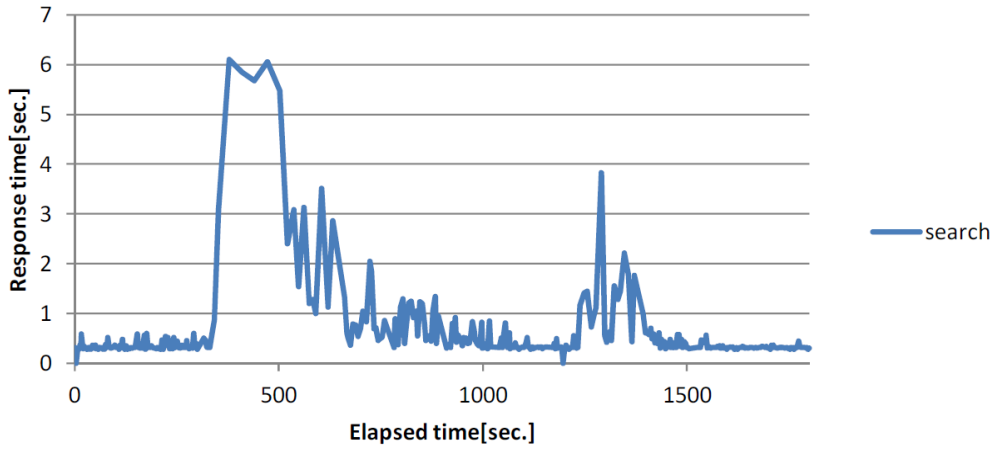

図 4-15　応答時間
（従来方式，ルール１，スケール変更単位２）

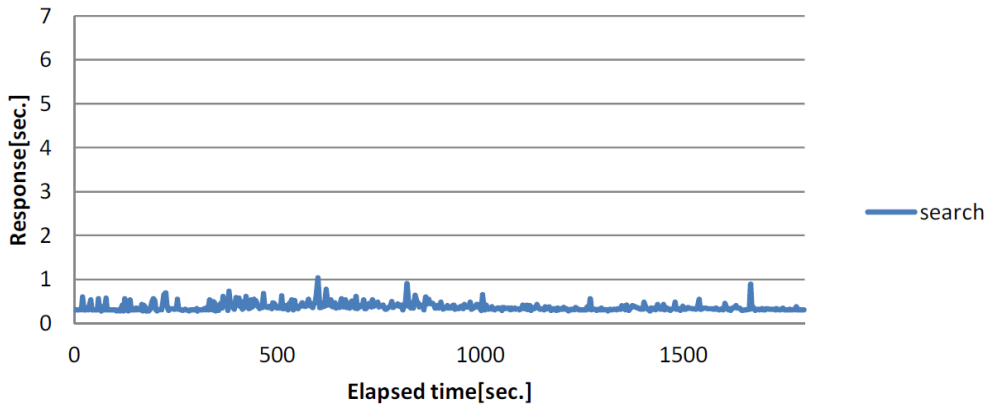

図 4-16　応答時間
（提案方式，ルール１，スケール変更単位１）

従来方式，提案方式の各ルールのサービス応答時間保証率を図 4 17 に，リソース消費量を図 4 18 に示す．バースト負荷パターンの場合，従来方式はルール 1，提案方式はルール 0 の場合に最も高いサービス応答時間保証率を達成している．各方式の最良値を比較すると，サービス応答時間保証率は従来方式で 74.8%，提案方式で 92.5%であり

17.7%改善される．このように，提案方式では，バースト負荷パターンの場合でもスケールアップ能力を維持しつつサービス応答時間保証率を 90%以上に維持することが可能になる．なお，CPU リソース使用量は，従来方式で 12249[vCPU 秒]であり，約 44%増加する．ただし，従来方式のルール 0，スケール変更単位 2vCPU の場合と，提案方式のルール 4，スケール変更単位 1vCPU の場合とを比較すると，提案方式の方が少ないリソース量で，より良いサービス応答時間保証率を達成している．メモリ使用量は従来方式で 12680[GB 秒]であり，ほぼ半減している．これは，従来方式ではメモリ量が大きい medium サイズの仮想サーバが追加されるのに対し，提案方式では不足する CPU リソースのみが追加される場合があるためである．この差は仮想サーバのメモリ量によって変わるため，従来方式では対象アプリケーションのリソース消費バランスにあった仮想サーバサイズを選択することが重要である．

以上より，バースト負荷の場合も，提案方式は従来方式と同等のサービス応答時間保証率を達成する場合に必要なリソース量は同等であり，かつ，従来方式では達成できないサービス応答時間保証率を実現できる．

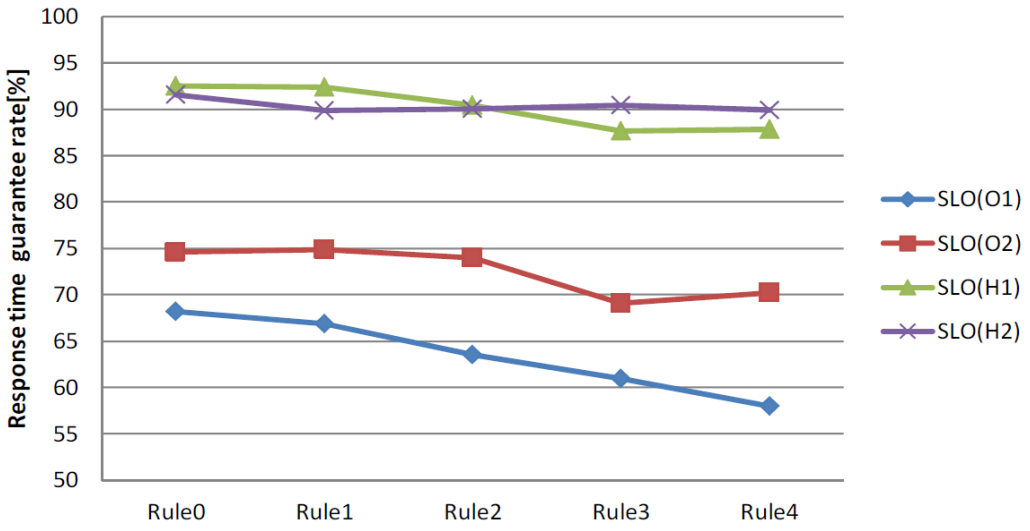

図 4-17　サービス応答時間保証率

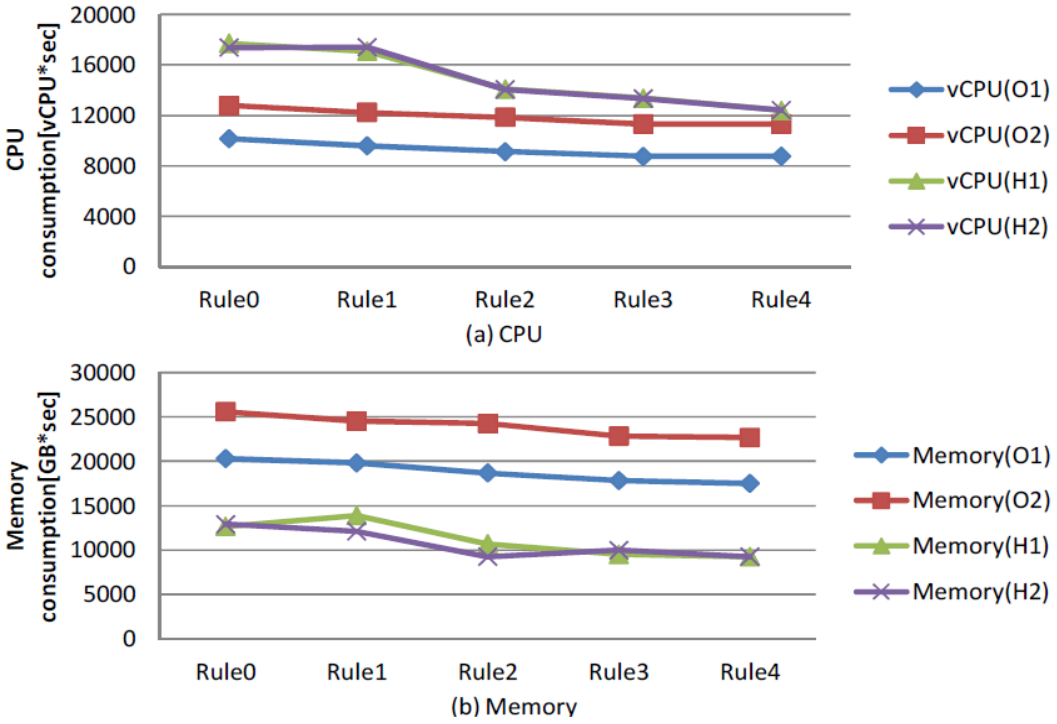

図 4-18　リソース利用量

4.4.4　実験結果の全体的考察

従来方式と比較した提案方式のサービス応答時間保証率の改善度合いは，平常時負荷パターンで 5.3%，バースト負荷パターンの場合で 17.7%であった．提案方式は，バースト負荷パターンの方で効果が顕著であり，負荷変動傾向が変化する負荷パターンに対して有効である．

また，従来方式でバースト負荷に対応する場合，スケール変更単位を 1vCPU から 2vCPU にすることで最大 12%の性能改善がみられ，サービス応答時間保証率を 74%まで上げることが可能である．ただし，増減の単位を大きくすると負荷上昇率が落ちた場合にはリソースを余分に追加することになる．また，負荷減少時には削除しすぎてしまいフラッピングを招く場合がある．これに対して提案方式では，スケール変更単位を 1vCPU としたままで中位レベルである 90%以上のサービス応答時間保証率を達成している．

今回の実験では，World Cup’98 の負荷パターンを用いたが，現在は 1998 年当時と比べインターネット人口の増加と携帯端末の普及によりサービスへのアクセス数も増大している．例えば，2010 年の JAIST の FTP（File Transfer Protocol）サーバへのアクセス数はピーク時で 50000req./min.であり，本実験の 30 倍となっている[64]．提案方式を用いてこのような負荷に対応するためには，スケールアップ可能なリソースが大量に必要になるが，大量リクエストを処理するサービスでは，平常時の負荷も大きいためリソースの在庫を持っていると想定され，提案手法を用いたピーク負荷への追随が可能であると考えられる．ただし，しきい値やスケール変更単位などのパラメータは対象システムに応じて調整する必要がある．

4.5 結言

本章では，サービス性能の維持とリソース有効利用を目的として，スケールアップとスケールアウトとを組み合わせたハイブリッドオートスケール方式を提案した．そして，平常時負荷パターンとバースト負荷パターンの 2 種類の負荷に対する評価によって，提案方式は，従来のスケールアウトに比べて応答性能を最大 1 割向上させることができることを明らかにした．また，負荷増大の傾向が一定である場合よりも，増大の傾向が変化する場合に，従来方式と比較した改善効果が大きくなることが分かった．

提案方式は，突発的な負荷増大検出時にはスケールアップを実行して迅速にリソースを追加し，負荷減少検出時には，上限値までリソースを追加したスケールアップ済の仮想サーバをリソース追加可能な別の仮想サーバへ入れ替えることで，その後の負荷増大に備えスケールアップ能力を維持する点が特徴である．

ただし，提案方式を適用するためには，対象サービスごとにオートスケールのしきい値やスケール変更単位などのパラメータを調整する必要がある．そこで，多様，かつ，大規模なサービスに提案方式を適用していくため，今後は運用中に適切なパラメータを学習しオートスケールを行う方法を検討することが課題である．

第5章　結論

5.1　本研究のまとめ

本論文では, 複数の交通サービスを連携させた統合モビリティサービスの実現に関する研究成果を，以下の 4 章に分けて述べた.

第 1 章では，複数の交通サービスをひとまとまりのサービスとした統合モビリティサービスについて述べ,「(1) 外部定義により提示する情報を変更可能な経路検索機能の実現」「(2) 固有形式の交通サービスのデータを共通形式に変換する方法」「(3) 応答性能保証とコスト低減を両立させる IT リソース調整方法」の 3 つの課題が重要であることを示した. 更に，各課題に対して,「(1) 経路検索機能をカスタマイズする誘導案内ポリシーの記述方式の提案」「(2) 交通サービスのデータ構造と用語の特徴に基づくスキーママッチング手法の提案」「(3) 迅速なリソース追加が可能なスケールアップとリソース削除が可能なスケールアウトを組み合わせる IT リソース調整方式の提案」という研究方針を定めた.

第 2 章では経路検索機能をカスタマイズするための設定情報である誘導案内ポリシーの体系的記述方式として，誘導案内ポリシーの文法と記述支援方式を提案した. 誘導案内ポリシーの文法は, 想定されるユースケースを全て記述できるように構造と予約語を定義した. その上で，記述支援方式では，誘導案内ポリシーの記述を容易化するため，ユースケースに対応づいたテンプレートと記述支援エディタを提案し，必須入力項目を削減できるようにした. そして, 記述支援方式について, 提案方式と従来のポリシー記述方式での記述工数削減効果を評価し, 記述工数が約 40%削減できることを示した.

第 3 章では, 異なるスキーマで作成された複数の交通サービスのデータを統合するため, 交通サービスのデータ構造と用語の特徴を考慮した拡張属性名類似度マッチングと属性型別属性値類似度マッチング手法を提案した. そして，提案手法により異なるデータソースから取得したデータを独自に定義した共通スキーマへ対応づけ, 既存手法である属性名の類似度によるマッチングと比較して，マッチングの精度を約 20%向上できることを明らかにした. これにより，例えば，鉄道の運行情報の 185 属性の内，既存手法では対応づけることができなかった 50 属性について自動的な対応づけが可能になり,

1 件あたりの対応づけに要する時間を 9 分と仮定すると，確認と人手による対応づけを含むスキーママッチング作業全体において 450 分の作業時間を短縮することができる．

第 4 章では, 応答性能と統合モビリティサービスプラットフォームのリソースコスト低減を実現する IT リソース調整方式として，ハイブリッドオートスケール方式を提案した．提案方式は負荷増大の傾向に応じて，リソース追加に時間を要するが追加量に制限の無いスケールアウトと, 迅速なリソース追加が可能であるが追加量に上限のあるスケールアップを使い分けることで応答性能劣化を防ぐ．さらに，上限値までリソースを追加したスケールアップ済の仮想サーバをリソース追加可能な別の仮想サーバへ入れ替えることで，システム全体としてのリソース増強能力を維持する．本論文では，この提案方式に対して複数の負荷パターンに対する評価を行い, 従来のオートスケールと比較して，使用するリソースを同等に抑えながら応答性能を最大 1 割向上できること，および，方式の特性や限界，パラメータとその設定方法を明らかにした．

本論文では統合モビリティサービスの設計と運用に関する 3 つの課題を議論した．これらの課題を解決する 3 つの技術を適用することで, 統合モビリティサービスプラットフォームのデータ統合機能において, 複数の交通事業者から収集した異なる形式のデータを共通形式へ変換する際の工数を低減できる．そして，案内機能である経路検索サービスは, 混雑や工事を避ける経路へ利用者を誘導するようカスタマイズすることができる．また，IT 基盤の性能管理機能は，経路検索サービスに対する検索要求が増大した場合でも，IT リソースのコストを抑えながらサービスの応答性能を維持するように IT リソースを調整することができる．

5.2 今後の課題

最後に，本研究における今後の課題について述べる．本論文では,「(1) 外部定義により提示する情報を変更可能な経路検索機能の実現」「(2) 固有形式の交通サービスのデータを共通形式に変換する方法」「(3) 応答性能保証とコスト低減を両立させる IT リソース調整方法」，という 3 つの課題に対して，第 2 章〜第 4 章で解決方法と，ここの解決方法における今後の課題を示した．今後は，これらの解決方法を組み合わせて，実環境に適用してその効果や影響を評価する必要がある．また，その他の課題としては以下のようなものがある．

(1) 誘導案内ポリシーの矛盾の検出と解消

本論文では，統合モビリティサービスの案内機能として，通常の経路検索結果に迂回経路や立寄り地などの誘導案内情報を追加できる経路検索サービスの実現に向けて，誘導案内の設定情報を記述する誘導案内ポリシーの文法と記述支援方式を確立した．

これにより，経路検索機能をカスタマイズして，事業者の事情に応じた誘導案内を出すことが可能になるが，誘導案内ポリシーに複数のルールが記述された場合には，それらのルール同士で矛盾が生じる可能性がある．このため，複数のルール間の矛盾の検出と，矛盾の解消方法について検討する必要がある．

(2) 新たなモビリティサービスのデータへの対応

本論文では，統合モビリティサービスのデータ統合機能において，異なるスキーマで作成された複数の交通サービスのデータを用いて統合モビリティサービスを実現するため，交通サービスのデータ構造と用語の特徴を考慮したスキーママッチング手法を確立した．しかし，統合モビリティサービスでは，電動マイクロモビリティや充電サービス付き駐車場など新たな統合対象となる交通サービスが現われている．これらのサービスのデータについても，提案したスキーママッチング手法を適用することでマッチングの工数を削減できるか検討する必要がある．

(3) パラメータ学習機能

本論文では，統合モビリティサービスの IT 基盤性能管理機能において，負荷増大の傾向に応じて，リソース追加に時間を要するが追加量に制限の無いスケールアウトと，迅速なリソース追加が可能であるが追加量に制限のあるスケールアップを使い分けることで応答性能劣化を防ぎ，スケールアップ可能なリソース量を維持するハイブリッドオートスケール方式を確立した．ただし，提案方式を適用するためには，対象サービスごとにパラメータの調整が必要である．そこで，多様，かつ，大規模なサービスに提案方式を適用していくため，今後の課題として運用中に適切なパラメータを学習しオートスケールを行う方法を検討する必要がある．

謝　辞

本研究の全過程を通じて，終始懇切丁寧なるご指導とご鞭撻，格別のご配慮を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 藤原融教授，薦田憲久招へい教授（大阪大学名誉教授）に深く感謝申し上げます．

本研究をまとめるにあたり，貴重なお時間を割いて頂き，丁寧なるご教示を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 下條真司教授，前川卓也准教授に深く感謝申し上げます．

本研究をまとめるにあたり，親切なるご助言を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 原隆浩教授，鬼塚真教授，松下康之教授に深く感謝申し上げます．

本研究の機会と大学院博士後期課程に進学する機会を与えて頂くとともに，研究を進めるにあたりご配慮を賜りました（株）日立製作所 研究開発グループ 技師長 山本彰博士，社会イノベーション協創統括本部 前統括本部長 森正勝博士（現イノベーション成長戦略本部副本部長），サービスシステムイノベーションセンタ センタ長 加藤博光博士，グリーンインフラシステム研究部 部長 崎川修一郎氏，UL 主任研究員 足立進吾博士に心より御礼申し上げます．

第 2 章および第 3 章の研究に関して，ご討論やご助言を頂くとともに多大なるご支援を頂きました（株）日立製作所 矢野浩仁博士，元（株）日立製作所 故廣井和重博士に心から御礼申し上げます．

第 4 章の研究に関して，共同研究者として有益なご討論やご助言を頂きました元（株）日立製作所 名倉正剛博士（現南山大学），（株）日立製作所 山崎謙太氏に心から御礼申し上げます．

ならびに格別なるご指導とご配慮を賜りました（株）日立製作所 研究開発グループ サービスシステムイノベーションセンタの各位に心から御礼申し上げます．

最後に，学位取得に向け応援してくれた両親と義母に心から感謝いたします．また，本論文の執筆にあたり，理解し支えてくれた夫 賢太と子供たち 啓太郎，郁，新に感謝いたします．

参考文献

[1] J. Sochor, H. Arby, M. Karlsson, and S. Sarasini, “A Topological Approach to Mobility as a Service: A proposed Tool for Understanding Requirements and Effects, and for Aiding the Integration of Societal Goals”, in Proc. 1st International Conference on Mobility as a Service (ICoMaaS), 2017.

[2] 日高洋祐，牧村和彦，井上岳一，“MaaS モビリティ革命の先にある全産業のゲームチェンジ”，日経 BP 社, 2018.

[3] 日高洋祐，牧村和彦，井上岳一, “Beyond MaaS 日本から始まる新モビリティ革命 ― 移動と都市の未来”，日経 BP 社, 2020.

[4] 深尾三四郎，“MOBILITY 2.0”，日本経済新聞社, 2018.

[5] CATAPULT, “Mobility as a Service - Exploring the Opportunity for Mobility as a Service in the UK”, https://cp.catapult.org.uk/wp-content/uploads/2021/07/Exploring_the_Opportunity_for_Mobility.pdf, 2016, (accessed 2022-12-18).

[6] F. Callegati, “I Want to Ride My Bicycle: a Microservice-based Use Case for a MaaS Architecture”, in Proc. 22nd IEEE Symposium on Computers and Communication Workshops DENVECT, pp.18-22, 2017.

[7] 永井崇之，名倉正剛，中島淳，平島陽子，森村知弘，“IT システム向け障害対処プラン自動生成システムの検討”，電子情報通信学会技術研究報告，ICM2012-80, Vol.112, No.492, pp.125-130, 2013.

[8] 山崎謙太，加藤真理子，平島陽子，森村知弘，“IT システムの障害対応効率化に向けたネットワークトラヒック可視化技術の検討”，電子情報通信学会技術研究報告，ICM2013-24, Vol.113, No.294, pp.7-12, 2013.

[9] 東京都市圏交通計画協議会，“新たなライフスタイルを実現する人中心のモビリティネットワークと生活圏”，第 6 回東京都市圏 PT 調査，https://www.tokyo-pt.jp/static/hp/file/publicity/toshikoutsu_1.pdf, 2021, (2022-12-18 閲覧).

[10] 東京都産業労働局, “令和 2 年度 働き方改革に関する実態調査”, https://www.sangyo-rodo.metro.tokyo.lg.jp/toukei/koyou/jouken/r2/index.html, 2020, (2022-12-18 閲覧).

[11] 大久保敏弘, “第 4 回テレワークに関する就業者実態調査報告書”，NIRA 総合研究開発機構, 2021.

[12] 奥田竜雄, 平澤茂樹, 松隈信彦, 福本恭, 志村明俊, “スマートシティを実現するスマートモビリティ”, 日立評論 2011 年 12 月号, Vol.93, No.12, pp.816-821, 2011.

[13] 藤井聡, 谷口綾子, “モビリティ・マネジメント入門”, 学芸出版社, 2008.

[14] 財務省財務総合政策研究所, “インフラとしてのオープンデータ ―政府・自治体保有データのオープン化が日本経済に及ぼす影響―”, フィナンシャル・レビュー, 平成 27 年第 4 号(通巻第 124 号), pp. 29-47, 2015.

[15] 国土交通省, “MaaS 関連データの連携に関するガイドライン ver2.0”, https://www.mlit.go.jp/sogoseisaku/transport/content/001399363.pdf, 2021, (2022-12-18 閲覧).

[16] 国土交通省, “標準的なバスフォーマット”, https://www.mlit.go.jp/sogoseisaku/transport/sosei_transport_tk_000067.html, 2017, (2022-12-18 閲覧).

[17] 国土交通省, “静的バス情報フォーマット（GTFS-JP）仕様書 （第 2 版）”, http://www.mlit.go.jp/common/001283244.pdf, 2019, (2022-12-18 閲覧).

[18] 国土交通省, “動的バス情報フォーマット（GTFS リアルタイム）ガイドライン”, http://www.mlit.go.jp/common/001283242.pdf, 2019, (2022-12-18 閲覧).

[19] J. Allspaw and J. Robbins, 角征典訳, “ウェブオペレーション サイト運用管理のテクニック”, オライリー・ジャパン, 2011.

[20] R. Boutaba and I. Aib, “Policy-Based Management: A Historical Perspective”, Journal of Network and Systems Management, Vol.15, No.4, pp.447-480, 2007.

[21] H. Chan and T. Kwok, “A Policy-Based Management System with Automatic Policy Selection and Creation Capabilities by Using a Singular Value Decomposition Technique”, in Proc. of the 7th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'06), pp.96-99, 2006.

[22] 小泉稔, 三宅滋, 平島陽子, “ポリシーベースによる QoS 制御”, オーム社, 2001.

[23] 平島陽子, 木元賢司, 小泉稔, 河部展, “QoS 保証型サービスを提供する IP プロビジョニングサーバの検討”, 電子情報通信学会技術研究報告, SSE2000-172, 交換システム, Vol.100, No.451, pp.53-58, 2000.

[24] 平島陽子, 片岡健二, 小泉稔, “カスタマーセルフコントロール QoS 制御機能の検討”, 電子情報通信学会技術研究報告, TM2002-45, Vol.102, No.463, pp.11-16, 2002.

[25] 青木篤，勝俣修，服部泰明，平島陽子，“ブロードバンドシステムを支える統合運用管理ソフトウェア”，日立評論 2002 年 10 月増刊号, pp.37-40, 2002.

[26] 工藤裕，森村知弘，菅内公徳，薦田憲久，“障害原因解析のためのルール記述方法とその実行方式”，電気学会 情報システム研究会, IS-09-71, pp.1-6, 2009.

[27] 工藤裕，森村知弘，増岡義政，薦田憲久，“情報システムの運用自動化に向けたポリシー記述形式とポリシー実行スケジューリング方式”，電気学会 C 部門論文誌, Vol.131, No.10, pp.1803-1830, 2011.

[28] 工藤裕，森村知弘，増岡義政，薦田憲久，“業務システム動的構成変更のためのポリシー実行スケジューリング方式”，電気学会 情報システム研究会, IS-10-74, pp.5-10, 2010.

[29] DMTF, “CIM Simplified Policy Language (CIM-SPL) 1.0.0”, http://www.dmtf.org/sites/default/files/standards/documents/DSP0231_1.0.0.pdf, 2009, (accessed 2022-12-18).

[30] D. Agrawal, S. Calo, K. W. Lee, and J. Lobo, “Issues in Designing a Policy Language for Distributed Management of IT Infrastructures”, in Proc. of the 9th IFIP/IEEE International Symposium on Integrated Network Management (IM 2007), pp.30-39, 2007.

[31] B. Moore, E. Ellesson, J. Strassner, and A. Westerinen, “Policy Core Information Model 106 (PCIM) Version 1 Specification”, Request for Comment 3060, Network Working Group, 2001.

[32] E. Rahm and P. A. Bernstein, “A Survey of Approaches to Automatic Schema Matching”, Very Large Data Base Journal, Vol.10, No.4, pp.334–350, 2001.

[33] P. Shvaiko and J. Euzenat, “A Survey of Schema-Based Matching Approaches”, Journal on Data Semantics IV. (Lecture Notes in Computer Science book series (LNCS), Vol.3730), pp.146–171, 2005.

[34] F. Giunchiglia, M. Yatskevich, and P. Shvaiko, “Semantic Matching: Algorithms and Implementation”, Journal on Data Semantics IX. (Lecture Notes in Computer Science book series (LNCS), Vol.4601), pp.1-38, 2007.

[35] G. Galante, L. Carlos, and E. de Bona, “A Survey on Cloud Computing Elasticity”, in Proc. of IEEE Fifth International Conference on Utility and Cloud Computing (UCC), pp.236-270, 2012.

[36] M. Mao, J. Li, and M. Humphrey, “Cloud Auto-Scaling with Deadline and Budget Constrains”, in Proc. of Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on IEEE, 2010.

[37] H. Fernandez, G. Pierre, and T. Kielmann, “Autoscaling Web Applications in Heterogeneous Clod Infrastructures”, in Proc. of 2014 IEEE Int’l Conf. on Cloud Engineering (IC2E), pp.195-204, 2014

[38] Z. Gong, X. Gu, and J. Wilkes, “Press: Predictive Elastic Resource Scaling for Cloud Systems”, in Proc. of 2010 IEEE International Conference on Network and Service Management (CNSM), pp.9-16, 2010.

[39] W. Dawoud, I. Takouna, and C. Meinel, “Elastic VM for Rapid and Optimum Virtualized Resource’s Allocation”, in Proc. of 2011 5th International DMTF Academic Alliance Workshop on Systems and Virtualized Management (SVM), 2011.

[40] M. Hanai, T. Suzumura, A. Ventresque, and K. Shudo, “An Adaptive VM Provisioning Method for Large-Scale Agent-Based Traffic Simulations on the Cloud”, in Proc. of 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2014), pp.130-137, 2014.

[41] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, “Cloud- Scale: Elastic Resource Scaling for Multi-Tenant Cloud Systems”, in Proc. of the 2nd ACM Symposium on Cloud Computing, pp.5:1-5:14, 2011.

[42] R. Han, L. Guo, M. Ghanem, and Y. Guo, “Lightweight Resource Scaling for Cloud Applications”, in Proc. of 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.644-651, 2012.

[43] D. Deng, Z. Lu, W. Fang, and J. Wu, “CloudStream-Media: A Cloud Assistant Global Video on Demand Leasing Scheme”, in Proc. of 2013 IEEE International Conference on Service Computing (SCC 2013), pp.486-493, 2013.

[44] D. Huang, B. He, and C. Miao, “A Survey of Resource Management in Multi-Tier Web Applications”, IEEE Communications Surveys & Tutorials, Vol.16, No.3, pp.1574-1590, 2014.

[45] K. Hwang, Y. Shi, and X. Bai, “Scale-Out vs. Scale-Up Techniques for Cloud Performance and Productivity”, in Proc. of 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2014), pp.763-768, 2014.

[46] M. Mao and M. Humphreyi, “Auto-Scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows”, in Proc. of 2011 IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2011), pp.1-12, 2011.

[47] 平島陽子，薦田憲久，藤原融，“位置情報を活用した移動需要推定方法”，電気学会第 77 回情報システム研究会，IS-19-009, pp.41-44, 2019.

[48] 平島陽子，播金真奈，吉治季恵，廣井和重，“ナッジ応用行動誘導方式の福岡における実地実証評価”，電気学会第 86 回情報システム研究会，IS-21-0038, pp.35-39, 2021.

[49] 宮崎邦彦，廣井和重，小林悠一，Giulio Mofa, Eduardo Fernandez-Moral, Wisinee Wisetjindawat，“デジタル技術が切り拓く新たなモビリティサービス”，日立評論 2021 年特別号, Vol.103, No.5, pp.92-96, 2021.

[50] 平島陽子，今村将司，藤原融，薦田憲久，“ポリシーベース移動需要マネジメント方式の提案”，情報処理学会研究報告 高度交通システムとスマートコミュニティ（ITS），2022-ITS-91(12), pp.1-7, 2022.

[51] 平島陽子，矢野浩仁，橋本博文，牛山純子，花房比佐友，“都市交通シミュレーションのためのオープンデータマッピング手法”，電気学会第 73 回情報システム研究会，IS-18-001, pp.1-5, 2018.

[52] 平島陽子，薦田憲久，藤原融，“統合モビリティサービスの実現に向けたスキーママッチング手法の提案”，電気学会 C 部門論文誌, Vol.139, No.6, pp.686-691, 2019.

[53] Y. Hirashima, N. Komoda, and T. Fujiwara, “A Schema Matching Approach for Integrated Mobility Service”, Electronics and Communications in Japan, Vol.102, Issue 9, pp.12-18, 2019.

[54] 山崎謙太，寺村健，平島陽子，森村知弘，“オートスケール時の SLA 保証改善に向けた仮想マシン CPU キャッピング機能の活用に関する検討”，電子情報通信学会技術研究報告, ICM2014-55, Vol.114, No.523, pp.7-12, 2015.

[55] 平島陽子，山崎謙太，森村知弘，薦田憲久，“応答性能保証率向上のためのハイブリッドオートスケール方式”，電気学会 C 部門論文誌, Vol.137, No.3, pp.521-531, 2017.

[56] Y. Hirashima, K. Yamasaki, and M. Nagura, “Proactive-Reactive Auto-Scaling Mechanism for Unpredictable Load Change”, in Proc. of 4th IIAI International Congress on Advanced Applied Informatics 2016 (IIAI AAI 2016), pp.861-866, 2016.

[57] Y. Hirashima and N. Komoda, “Parameter Optimization for Hybrid Auto-scaling Mechanism”, in Proc. of 17th IEEE International Symposium on Computational Intelligence and Informatics (CINTI 2016), pp.111-116, 2016.

[58] 楊中平, 小関隆章, 曽根悟, “鉄道ネットワークにおける異常時の乗客個別誘導案内法の基礎検討”, 電気学会 交通・電気鉄道 ITS 合同研究会, TER-02-11 ITS-02-1, 2002.

[59] 上野爽, 大西亘, 小関隆章, “利用者から信頼される旅客行動属性を考慮した都市鉄道の個別案内システムの提案”, 電気学会 交通・電気鉄道研究会, TER-21-055-057.059, 2021.

[60] D. Engmann and S. Massmann, “Instance Matching with COMA+”, in Proc. of BTW Workshops-Model Management and Metadaten-Verwaltung, pp.28-37, 2007.

[61] B. Gu, Z. Li, X. Zhang, A. Liu, G. Liu, K. Zheng, L. Zhao, and X. Zhou, “The Interaction Between Schema Matching and Record Matching in Data Integration”, IEEE Transaction on Knowledge and Data Engineering, Vol.29, No.1, pp.186-199, 2017.

[62] VMware, Inc. “VMware White Paper: Virtualizing Business-Critical Applications on vSphere”, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/whitepaper/solutions/vmware-virtualizing-business-critical-apps-on-vmware_en-white-paper.pdf, (accessed 2022-12-18).

[63] 社団法人電子情報技術産業協会, “民間向け IT システムの SLA ガイドライン第三版”, 日経 BP 社, 2006.

[64] 二宮恵, 宇夫陽次郎, 長建二朗, “集約 Web トラフィックにおけるリソース配分モデルの提案”, インターネットコンファレンス 2010 (IC2010), pp.83-91, 2010.