東京大学
情報理工学系研究科創造情報学専攻
博士論文

BIM を用いた Software Defined BACS の実現に関する研究

The design and implementation of Software Defined BACS
based on shared BIM repository

粕谷 貴司
Takashi Kasuya

指導教員　江崎 浩　教授

2020 年 6 月

概要

クラウド，IoT (Internet of Things)，人工知能 (AI:Artificial Intelligence)，ビッグデータなどの技術は，デジタル化が遅れていたビルのシステムを急激に変えつつある．高度なエネルギーマネジメントや快適性向上といったスマートビルのソリューションは，既にビジネス領域において提供が始まっているが，その多くは排他的なサイロになっている．サイロ化を助長するような我が国の建設産業の慣習に起因するといえるが，一方で，参入障壁の高さから，スマートビルを実現する技術の研究がほとんど行われてこなかったという背景もある．

本研究では，サードパーティの参入を促し，構築後のアプリケーションの移植性と再利用性を高め，知識ベース等を用いたシステムの自動設定を可能とするスマートビルの構成要素である Software Defined BACS (Building Automation and Control System) の実現を目的とし，そのための実践的なデータ・プラットフォームについて扱う．

データ・プラットフォームの構築のため，スマートビルに必要なセマンティクスについて検討を行い，データモデルの生成手法について提案した．また，セマンティックウェブの技術を用いることで，特定されたセマンティクスが，SDM (Software Defined Media) のアプリケーションにも有効に活用できることを示した．

更に，SDM のための立体音響プラットフォームとそのアプリケーションである「LiVRation」の構築と評価を行い，没入型のインタフェースとオブジェクトオーディオを用いたシステムの新規性と有用性を示した．

上記の検討と，多様なユースケースの分析を基に，スマートビルの汎用データ・プラットフォームである「futaba」を提案した．IoT/AI の適用と，インターネットを介したサードパーティとの連携を前提としたデータ・プラットフォームである．クラウドとオープンシステムの採用を前提としたプラットフォームの設計と，その参考実装について示し，十分な汎用性を備え，かつ現実的なコストで運用可能であることを検証した．また，futaba で収集したデータを用いたスマートビルのためのプロファイリング技術である BAP (Building Activity Profiling) を提案し，その性質について分析するとともに，BAP を用いたアプリケーションについて提案，評価することで，プラットフォームの汎用性を示した．

これらの研究は，建物の中で展開される「デジタルツイン」やその発展である「コモングラウンド」の構築に関わる研究ともいえる．Software Defined BACS の実現は，コモングラウンドの実現を加速し，SDM をはじめとする多様なアプリケーションの社会実装を促す基盤技術になると考える．

本研究によって，閉鎖的といわれる建設業界のデータの利活用が促進されることを期待するとともに，スマートビルの普及と，スマートビルに関する新たなエコシステム，ビジネスモデルが誕生することを期待する．

Abstract

Technologies such as cloud, IoT (Internet of Things), AI (Artificial Intelligence), and big data are rapidly changing the building systems that have been lagging behind in digitalization. Smart building solutions, such as advanced energy management and comfort improvements, have already begun to be offered in the business domain, but many of them have become exclusive silos. On the other hand, due to the high barriers to entry, there has been little research on smart buildings.

We research a practical data platform for smart buildings and Software Defined BACS (Building Automation and Control System), which promotes the entry of third parties, increases the portability and reusability of the application after construction, and enable automatic configuration of the system using knowledge bases.

In order to build a data platform, we discovered the semantics required for a smart building and proposed a method for generating a data model. We also showed that the identified semantics can be effectively applied for SDM (Software Defined Media) applications by using semantic web.

Based on the above discussion and the analysis of various use cases, we proposed "futaba" as a general-purpose data platform for smart buildings. It is a practical data platform that assumes the application of IoT/AI and the collaboration with third parties via the Internet. We described the design of the platform and its reference implementation, and also verified that the platform is sufficiently versatile for smart building applications and can be operated at a realistic cost.

In addition, we proposed BAP (Building Activity Profiling), which is a profiling technique for smart building using the data collected in the platform. We analyzed the profile, and also proposed and evaluated applications using BAP.

These studies can be said to be related to the construction of the "digital twin" and its development, the "common ground", which is developed in buildings. We believe that Software Defined BACS will accelerate the realization of common ground and become a fundamental technology to promote the social implementation of various applications such as SDM.

We hope that this study will promote the utilization of data in the construction industry, and a new ecosystem and business model for smart buildings will be born.

目次

第 1 章　序論 1
1.1　はじめに . . . 1
1.2　論文の構成 . . . 2
1.3　本研究の貢献 . . . 3

第 2 章　研究背景 5
2.1　建設産業のデジタル化 . . . 5
2.2　Software Defined Media . . . 9
2.3　スマートビル . . . 13
2.4　デジタルツイン . . . 25
2.5　研究対象領域 . . . 31

第 3 章　Software Defined BACS 33
3.1　システム構成 . . . 33
3.2　適用時の要件 . . . 35
3.3　構成要素 . . . 38
3.4　まとめ . . . 39

第 4 章　スマートビルのセマンティクス 41
4.1　はじめに . . . 41
4.2　BACS へのセマンティクウェブの技術適用 . . . 42
4.3　スマートビルの要件分析 . . . 45
4.4　提案手法 . . . 48
4.5　SDM アプリケーション . . . 60
4.6　まとめ . . . 63

第 5 章　エンターテインメントシステムの構築 65
5.1　はじめに . . . 65
5.2　関連研究 . . . 65
5.3　本研究の目的 . . . 67
5.4　LiVRation . . . 67
5.5　評価 . . . 75
5.6　まとめと今後の展望 . . . 80

第 6 章　スマートビルのデータ・プラットフォーム 81

6.1 はじめに . . . 81
6.2 関連研究 . . . 81
6.3 本研究の目的 . . . 87
6.4 機能分析と設計要件 . . . 88
6.5 futaba の設計 . . . 92
6.6 実装と評価 . . . 99
6.7 アプリケーション . . . 104
6.8 まとめと今後の展開 . . . 106

第 7 章 スマートビルのデータ分析 107
7.1 はじめに . . . 107
7.2 BAP (Building Activity Profiling) . . . 109
7.3 負荷予測技術 . . . 123
7.4 データ型の推定 . . . 134
7.5 まとめと今後の展開 . . . 138

第 8 章 結言 139
8.1 Software Defined BACS の実現 . . . 139
8.2 コモングラウンドの実現 . . . 140
8.3 まとめと今後の展望・展開 . . . 140

発表論文 143

参考文献 145

用語集 153

付録 A futaba の API 157
A.1 概要 . . . 157
A.2 利用者認証 API . . . 159
A.3 モデル学習データ取得 API . . . 161
A.4 天気情報データ取得 API . . . 167
A.5 建物メタデータ API . . . 168
A.6 WoT API . . . 171
A.7 WoT 拡張 API . . . 179

第 1 章

序論

1.1 はじめに

クラウド，IoT (Internet of Things)，AI (Artificial Intelligence)，ビッグデータなどの技術は，閉鎖的でありデジタル化が遅れていたビルのシステムを急激に変えつつある．高度なエネルギーマネジメントや快適性向上といったスマートビルのソリューションは，既にビジネス領域において提供が始まってきているが，その多くは排他的なサイロになっている．理由の多くは，サイロ化を助長するような我が国の建設産業の慣習に起因するといえるが，その参入障壁の高さからスマートビルを実現する技術の研究がほとんど行われてこなかった背景もある．

建設産業では BIM (Building Information Modeling) や BACS (Building Automation and Control System) などを用いたデジタルデータ活用の高度化が進んでいるが，他業種・融合領域での活用は進んでいない．それらは主として形状（ジオメトリ）と時系列データといえるが，それらは建設産業だけではなく，ゲームをはじめとするエンタテインメント業界において活用が望まれている．

エンタテインメント分野のアプリケーションやコンテンツは，完パケと呼ばれるパッケージ化された形態でサービスされることが多く，プラットフォームを介した他システムとの連携などはされてこなかった．SDM (Software Defined Media) コンソーシアムでは，インターネットを介した立体音響システムの実現に向けて研究を行っており，サービスの再利用や移植性の向上，またはソフトウェアによるサービス定義と制御が可能（Software Defined）なシステム運用を目的として研究を進めている．SDM のアプリケーションはサイバー空間と物理空間を連携させた「デジタルツイン」であり，サイバー空間におけるシミュレーションのため，BIM から取得可能な 3 次元形状や属性情報，IoT により取得した位置情報等の活用が望まれているが，現状ではそれらの連携までは考慮されていない．

一方，スマートビルに適用可能なデータ・プラットフォームやアプリケーションの研究やソリューションは数多くあるが，施設管理者がサービス維持に多くのコストを避けない理由から，特に中小規模のビルに対しては導入が進んでいない．逆に，SDM などの付加価値の高いエンタテインメント分野のアプリケーションは，一般的に省エネや省コストといったとは別の視点で投資がなされる．これらに共通化されたプラットフォーム上で提供するアプリケーションやコンテンツが提供できることで，スマートビル上でより付加価値の高いシステム提供が可能になり，ビルオーナーやテナントからの投資も進むと考える．

本研究では，そうしたスマートビルの高度なサービスを実現する Software Defined BACS の実践的なデータ・プラットフォームについて扱うとともに，融合領域でのエンジニアリングにより実現されるデジタルツインに注目する．具体的には，セマンティクウェブをはじめとしたインターネットのベストプラクティスと，クラウドとオープンシステムの導入を前提とすることで，システムの再利用性を高め，運用コストの低減と可用性の向上を実現する．汎用性の高いアーキテクチャの採用により，スマートビルの多様なユースケースへの対応と，SDM をはじめとするエンターテイメント分野のアプリケーションへの応用も可能にする．

本研究によって，閉鎖的といわれる建設業界のデータの利活用が促進されることを期待するとともに，スマートビル

の普及に加え，スマートビルに関する新たなエコシステム，ビジネスモデルが誕生することを期待する．

1.2　論文の構成

本論文の構成と，各章の概要を以下に示す．

■第 2 章：研究背景　建設産業におけるデジタル化を俯瞰し，研究課題を深耕するとともに，その融合領域であるデジタルツインとその発展であるコモングラウンドについて導入し，研究領域の明確化を行う．

■第 3 章：Software Defined BACS　スマートビルの高度なサービスを実現する Software Defined BACS を導入し，そのシステム構成について深耕するとともに，それらの実現に必要なビルのインフラ要件と，その構成要素について述べる．

■第 4 章：スマートビルのセマンティクス　高度化するスマートビルやデジタルツイン・アプリケーションに必要なセマンティクスについて導入する．建設・維持管理分野で利用されるオントロジーについて述べ，ユースケース分析から必要なセマンティクスを特定した．それらを含むデータモデルの自動生成のため，BIM を用いた生成手法について提案し，実データを用いて検証を行うとともに，BIM より抽出したセマンティクスとジオメトリを用いた SDM アプリケーションを構築し，その汎用性について評価・検証した．

■第 5 章：エンターテインメントシステムの構築　音楽イベントの遠隔配信を対象とし，自由視点映像音声のインタラクティブ再生を行う SDM のためのアプリケーション・プラットフォームについて述べる．収録したライブを自由視点で視聴するとともに，収録された音声を自由にコントロールし，かつ演奏者や楽器のメタデータ，ソーシャルメディアなどで発信されるコンテンツも扱うことが可能な SDM のアプリケーションである「LiVRation」を開発した．それらのシステム要件，設計・実装および評価について述べる．

■第 6 章：スマートビルのデータ・プラットフォーム　スマートビルに向けた汎用データ・プラットフォームである futaba の設計・実装・評価について述べる．関連研究とユースケースを基に要件を設定し，セマンティックウェブやビッグデータ処理などの，インターネットのベストプラクティスとオープンシステムを用いて設計・構築した．参考実装を用いて，実際の建設プロジェクトに適用し，設計の妥当性と有用性等について評価を行った．

■第 7 章：スマートビルのデータ分析　futaba を前提とした，データ分析のアプリケーションについて述べ，futaba の汎用性について考察するとともに，ビルの時系列データの性質について分析する．具体的には，ビルの中で行われる活動やセンサの動きを分析し理解するプロファイリング手法である BAP (Building Activity Profiling) を導入し，デマンドレスポンスなどのユースケースに重要な負荷予測，未知のデータ型の推測技術について述べる．

■第 8 章：結言　本研究について振り返り，Software Defined BACS やコモングラウンドの実現についての考察と，今後の展望を述べる．

1.3 本研究の貢献

本研究における学術的な貢献を以下に述べる．

■スマートビルの実践的なセマンティクスの特定と生成手法 BACS に適用可能なオントロジは多く提案されているが，BACS の専門業者以外のサードパーティとの連携や，エンジニアリングの効率化を意識した実践的な研究は少ない．AI や IoT の専門業者との連携を行ったユースケースから，必要とされるセマンティクスやオントロジを特定し，BIM と BACS のポイントリストを用いたデータモデルの自動生成手法について提案，評価した．

■セマンティックウェブを用いた SDM アプリケーションとの連携手法 SDM コンソーシアムでは，SDM オントロジと呼ばれる語彙が提案しているが，それらは SDM アプリケーションのみ対象としていた．Linked Data などのセマンティックウェブの技術を用いて，BIM から抽出したセマンティクスやジオメトリと連結することで，SDM アプリケーションの開発効率向上が可能であることを示した．

■VR を用いたインタラクティブな立体視聴システム インターネットからの配信される VR コンテンツの視聴が可能になってきたが，配置されている音源は静的なものがほとんどで，かつ高品質なストリーミング再生は困難であった．VR，オブジェクトオーディオ，ロスレス配信技術，SDM オントロジ等を用いて，自由視点でのライブ体験を実現する「LiVRation」を試作・検証し，それらのインタフェース，インタラクション，アーキテクチャの有効性について示した．

■実践的なスマートビル用のデータ・プラットフォーム スマートビルのためのミドルウェアやデータ・プラットフォームは多く研究されているが，多様な IoT デバイスやビッグデータの対応，イニシャル・ランニングコストの低減や可能性の向上を目的とした実践的なデータ・プラットフォームは存在しなかった．クラウドを前提とし，オープンシステムとラムダ・アーキテクチャを用いたデータ・プラットフォームである futaba を提案・実装し，評価を行うことで，その妥当性と有用性を示した．futaba はセマンティクスウェブと IoT を融合させた SWoT (Semantic Web of Things) [44] の 1 事例であるともいえる．

■スマートビルのプロファイリング技術 データ・プラットフォームから取得可能な様々なデータを用いて，ビルのシステムの状態や，その中で行われる活動を分析するプロファイリング技術である BAP を提案，評価した．BAP の分析結果であるモードの性質を明らかにし，負荷予測やデータ型推定に応用可能であることを示した．

第 2 章

研究背景

本章では，スマートビルとその融合領域における研究課題を明らかにし，研究領域の明確化を行う．

2.1 節にて，建設産業のデジタル化について俯瞰し，2.2 節で，融合領域の 1 つである SDM (Software Defined Media) について述べる．2.3 節にて，スマートビルとその中心的な技術である BACS (Building Automation and Control System) 等について述べ，代表的なユースケースについて説明する．2.4 節にて，本研究で対象とするデジタルツイン，そこから派生するミラーワールドやコモングラウンドという概念について深耕し，その構築技術について述べる．2.5 節では，本章を振り返ると共に，本研究の対象領域について述べる．

2.1 建設産業のデジタル化

建設業界はデジタル化が遅れた業界といわれている．2015 年の Kane [46] らの調査において産業別のデジタル成熟度を指標化しているが，全ての産業の中で建設・不動産が最もデジタル成熟度が低いという結果であった．日本の建設業界は特に保守的といわれており，海外に比べてデジタル化が遅れていたが，少子高齢化に伴う労働力不足などの社会背景もあり，デジタル化が急ピッチで進んできた．

図 2.1 にデジタル化の概要を示す．建設プロジェクトのライフサイクルは設計・施工・維持管理の 3 つフェーズに分けており，設計フェーズではコンピュテーショナルデザイン，BIM (Building Information Modeling)，施工フェーズではデジタル・ファブリケーション，i-Construction，そして維持管理フェーズにスマートビル，CAFM (Computer Aided Facility Management) を配置している．それぞれのフェーズでのデジタル成果物は，適切な変換を経て，別なフェーズで再利用されることが理想とされる．なお，設計・施工側のデジタル化の技術は建設テック (ConTech) と呼ばれ，維持管理のフェーズの技術は不動産テック (PropTech) と呼ばれることがある（図 2.2）．

本節では，それぞれについて概要を述べる．

2.1.1 設計フェーズ

コンピュテーショナルデザインとは，ルールやアルゴリズムの記述によりジオメトリ（形状）やデータの生成・操作を行うことであり，具体的な設計に入る前のプランやコンセプト固め，そのためのスタディーやシミュレーションなどをコンピューターを駆使して実施することを指す．実現可能な最善のデザインやプランを見つけるために行う，設計の前処理といえるもので，近年では Rhinoceros のプラグインである Grasshopper[*1]や，Dynamo[*2]といったオーサリングツールが使われることが多い（図 2.3）．これは習熟が容易なビジュアルプログラミング環境が提供されていることに加え，環境シミュレーションなどの有償・無償の多数のプラグインが提供されているためであろう．例えば，図 2.1 に

[*1] https://www.applicraft.com/products/rhinoceros/grasshopper/
[*2] https://primer.dynamobim.org/

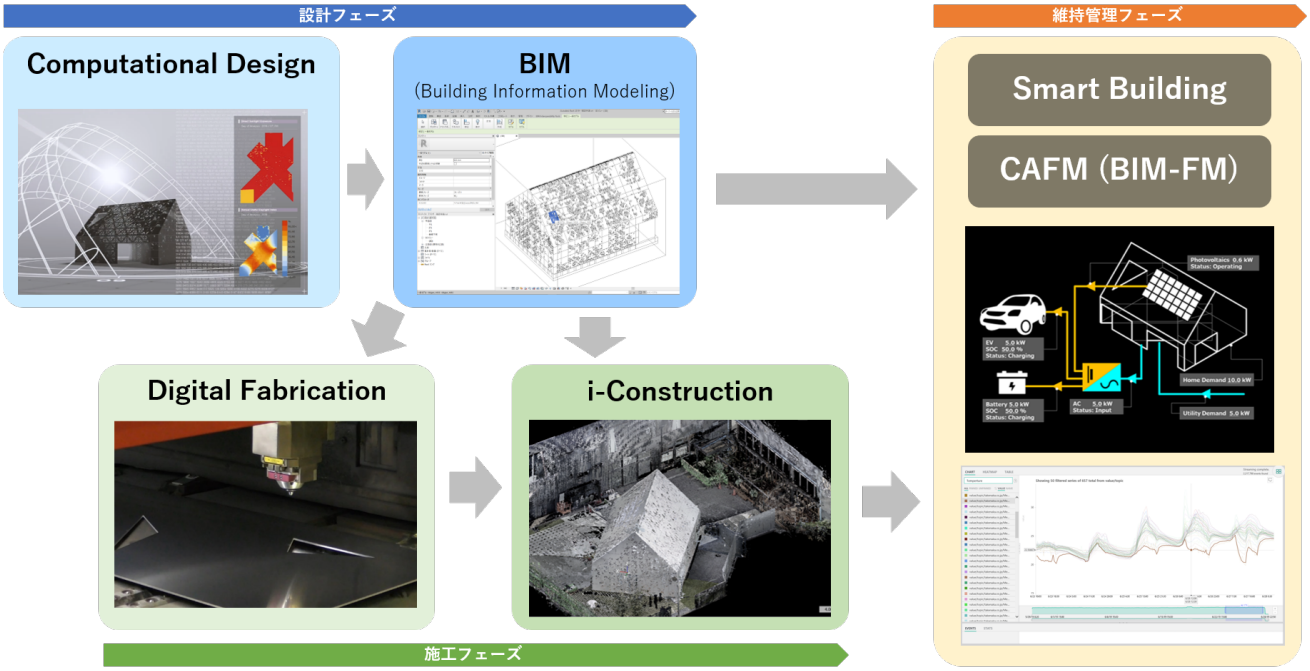


図 2.1: 建設産業におけるデジタル化

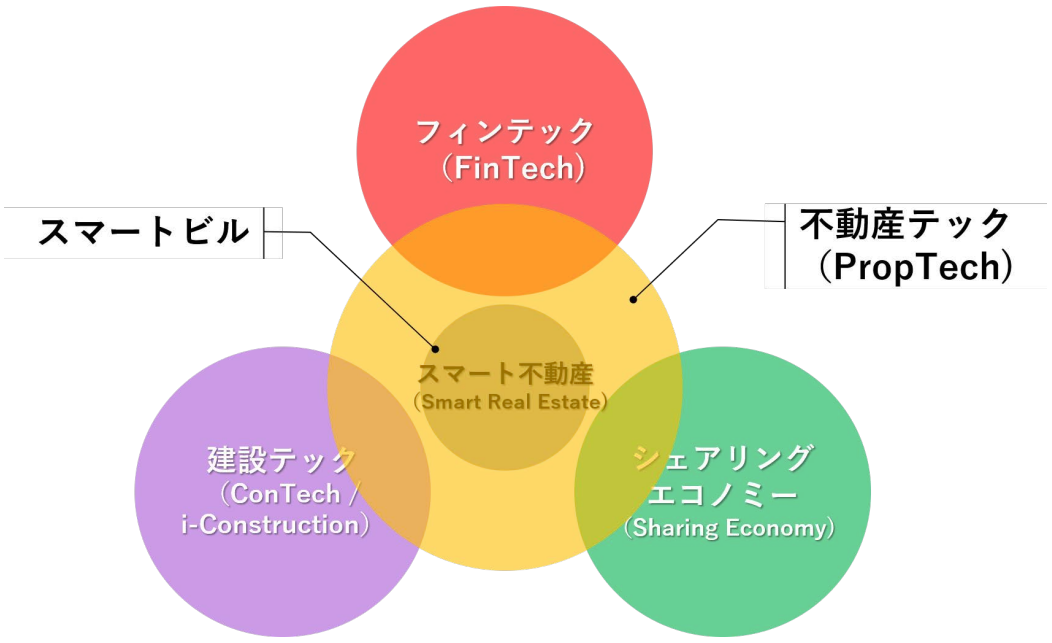


図 2.2: 不動産テックの概念（「不動産テックを考える」 [101] より）

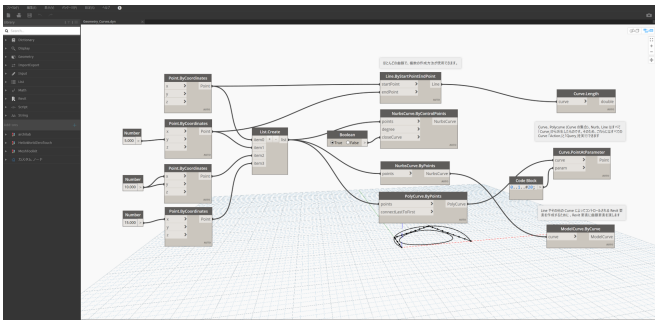


図 2.3: Dynamo によるプログラミング

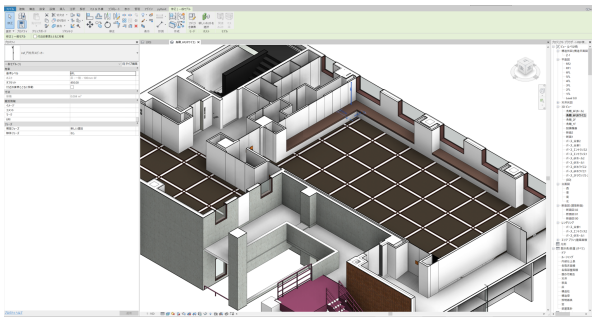


図 2.4: BIM モデル例 (Autodesk Revit)

示されているのは，24 時間 365 日の日射パターンをもとに，コストや熱・光環境などをパラメータとして最適な外皮パネルの形状パターンを計算している画面である [81]．ここで導出された形状データは BIM データ等に変換され，後工程に引き継がれ，設計，施工のためのデータとなる．

BIM は ISO 19650 シリーズ[*3]で規定されており，多様な定義が存在するが，Borrmann らの定義 [7] によると「建築物の物理的・機能的特性をデジタルで表現したものであり，ライフサイクルで発生する何らかの意思決定のために，信頼性の高い情報を提供する共有知識資源である」とされている．狭義には 3 次元の形状情報と室等の名称・面積，材料・部材の仕様・性能，仕上げ等，建築物の属性情報を併せ持った建築情報モデルであり，それら BIM モデル作成のためには，一般的に Autodesk，GRAPHISOFT などが開発した専用のオーサリングツールを利用する（図 2.4）．

日本においては設計事務所や建設会社によって BIM の利活用が牽引されてきたが，2019 年 6 月には国土交通省で建築 BIM 推進会議が設置され，BIM 活用のあるべき姿に向けて国としての目標が設定された．建築 BIM 推進会議では，ツールや専門業者間でのインターオペラビリティ向上や，設計・施工・維持管理といったフェーズ間での連続的なデータ活用を目指して，標準ワークフロー，データの受け渡しルールなどを定めたガイドラインを作成している [85]．なお，2019 年の調査によると，我が国では設計・施工についてはそれぞれ BIM の活用が進んでいるが，維持管理フェーズでの活用はほとんどない．施工については，大手建設会社では相当程度活用されているが，中小建設会社ではほとんど使われていない状況とのことである．維持管理フェーズでの BIM 活用については，日本ファシリティマネジメント協会（JFMA）がガイドライン [102] を出しているが，施設管理者やその委託先の建物維持管理業者に BIM の知識やノウハウがないことが多いため，普及には多くの時間がかかると考えられる．

2.1.2　施工フェーズ

デジタル・ファブリケーションは BIM データをもとにして，部材メーカーや施工者が所有する 3D プリンター，レーザーカッター，デジタル工作機，工業用ロボットなどを用いて，複雑な形状を高品質でありながら安価に実現する手法である．これによってコンピューテーショナルデザインによって作られた複雑な形状が，より実現しやすくなってきている．なお，デジタル・ファブリケーションの推進も i-Construction に位置付けられている．

i-Construction とは「調査・測量から設計，施工，検査，維持管理・更新までのあらゆる建設生産プロセスにおいて抜本的に生産性を向上させる」ことであり，生産性低下と労働力不足の解決のために，建設現場へ IoT（Internet of Things）と ICT（Information and Communication Technology）を導入するとしている [41]．国土交通省は 2017 年 12 月に i-Construction 委員会を設置し，建設現場の生産性向上の取り組みを加速させた．同委員会では，推進のための表彰や人材育成，技術基準類の作成・改定を行っている．

[*3] https://www.iso.org/standard/68078.html



図 2.5: EQ House の概要

2.1.3 維持管理フェーズ

一般的なビル設備の運用監視は，中央監視システムまたは BACS や BEMS (Building Energy Management System) と呼ばれるシステムを用いて施設管理員が行っている．近年，クラウド，IoT，AI などの技術を用いて，これらのシステムを高度化し，省エネや快適性・利便性の向上等を実現するスマートビルが増えてきている．

スマートビルの取り組みで先駆的なものは，坂村健と竹中工務店グループらによって建設された「TRON 電脳住宅」だろう．住宅内の各所に埋め込まれた制御用マイクロコンピュータ間の協調分散によって，居住する住人の希望にあわせた環境調整が行える近未来型の実験住宅で，1000 個に達するマイクロプロセッサやセンサが用いられた[*4]．現在，TRON 電脳住宅があった土地には，電気自動車が普及した未来をコンセプトにした実験住宅である EQ House [115] が建設されている．EQ House は多数の IoT や演出システムを備え，クラウドのデータ・プラットフォームに収集したデータを用いたビル設備の AI 制御を実現している（図 2.5）．

CAFM は施設維持に関わる情報管理のためのシステムで，その機能は個別のビルに対する空間・設備の管理，人員や面積などの執務者のワークプレイスに関わる管理，ビル設備の遠隔監視や施設管理業務の計画・履行管理，複数ビルの財務的な管理など多岐にわたる．BIM や IoT などと連携可能な高度なシステムやサービス[*5][*6]も提供されているが，高価なライセンスや FM に関する専門知識を必要とするなどの問題から，その導入は複数や大型物件を持つビルオーナーやディベロッパーに限られており，中小企業や施設管理会社での活用は進んでいない．

[*4] https://www.youtube.com/watch?v=7jPKEyM44GU
[*5] https://www.vueops.com/
[*6] https://www.starts-cam.co.jp/technology/bim/

2.1.4 融合領域における研究

建設産業のデジタル化について俯瞰したが，これらのデジタル化の取り組みは建設・維持管理 (AEC/FM) ドメインに限定されている．建築家の豊田啓介は，細分化され融合領域の研究が進まない日本の建築業界に警鐘を鳴らしており，そのためデジタル技術をベースとした学問体系である「建築情報学」の必要性を説いている [123]．また，MaaS (Mobility as a Service) の普及によるスマートシティ実現のためには，3D データの提供や測位補助といった環境側のアシスト技術が重要であり，より正確な自動運転の実現のためには，LiDAR や SLAM（Simultaneous Localization and Mapping）などによる計測とデジタル記述（メタデータやセマンティクスの付与を意味する）が必要であると述べられている．室内空間に関しては BIM データの転用が合理的であり，データの互換性を高めることで，建築業界の閉じたアセットを社会基盤に変換できるとしている．BACS についても同様で，それぞれの建物から提供される環境情報や人流情報などは，渋滞やエネルギー問題，災害問題，犯罪といった社会課題を解決するスマートシティの重要なデータになるに違いない．

しかしながら，建設業界で作られたデータの 2 次的な活用は驚くほど進んでいない．理由としては，納品された BIM データや BACS や BEMS から取得される時系列データの所有権の問題や，そのためのデータ抽出・変換のためのエンジニア不足が原因といえる．例えばゲーム業界では，建築物のデータは自ら作ったり，有料のアセットを買ったりすることが多く，BIM データの転用は開発効率や創造性の向上につながるだろう．ライブやコンサートを行うエンタテインメント産業も同様で，ステージセットの検討のために施設から提供される図面は 2 次元であることが多く，音響や演出のシミュレーションなどを行うためには自らモデルを作成する必要がある．そうした背景もあって，近年は BIM とゲームエンジンである Unity[*7], Unreal Engine[*8]といったゲームエンジンとの親和性が向上している．BIM と連携を行うプラグイン[*9][*10]が活用できるようになってきており，それらの動きは今後加速していくといえるが，そのためのプロセスや具体的な連携手法について研究が必要といえる．

2.2 Software Defined Media

本節では，融合領域であるエンタテインメント産業での取り組みとして，SDM コンソーシアムでの活動を取り上げる．

著者らは，2014 年から SDM コンソーシアムを設立し，オブジェクトベースのデジタルメディアと，インターネットを前提とした視聴空間の研究を続けてきた [71, 111][*11]．SDM とは，映像・音響システムの IP ネットワーク化を背景に，これらの設備の機能に対して抽象化・仮想化を行い，サービスとしての映像・音響を提供するための基盤的なアプローチである．

2.2.1 SDM アーキテクチャ

SDM のアプリケーションは，図 2.6 に示す SDM アーキテクチャに従って開発されると定義している．それらを基に多様なアプリケーションの開発を行ってきた [47, 119]．

SDM アーキテクチャでは，視聴空間のデータは抽象化され情報空間（サイバー空間）のサービスに取り込まれ，アプリケーションは API を介して情報にアクセスするとともに，ネットワークを介して別拠点とも連携できる構成となっ

[*7] https://unity.com/
[*8] https://www.unrealengine.com/ja/
[*9] http://aec.unity3d.jp/topics/category/unityreflect/
[*10] https://www.unrealengine.com/ja/datasmith
[*11] https://sdm.wide.ad.jp/

ている．収録環境・デバイスとしては，カメラやマイクだけでなく，IoT のようなセンサも対象としている．空間 3 次元モデルやデジタル地図も使ってサイバー空間を構成しており，デジタルツインの構成であるといえる．

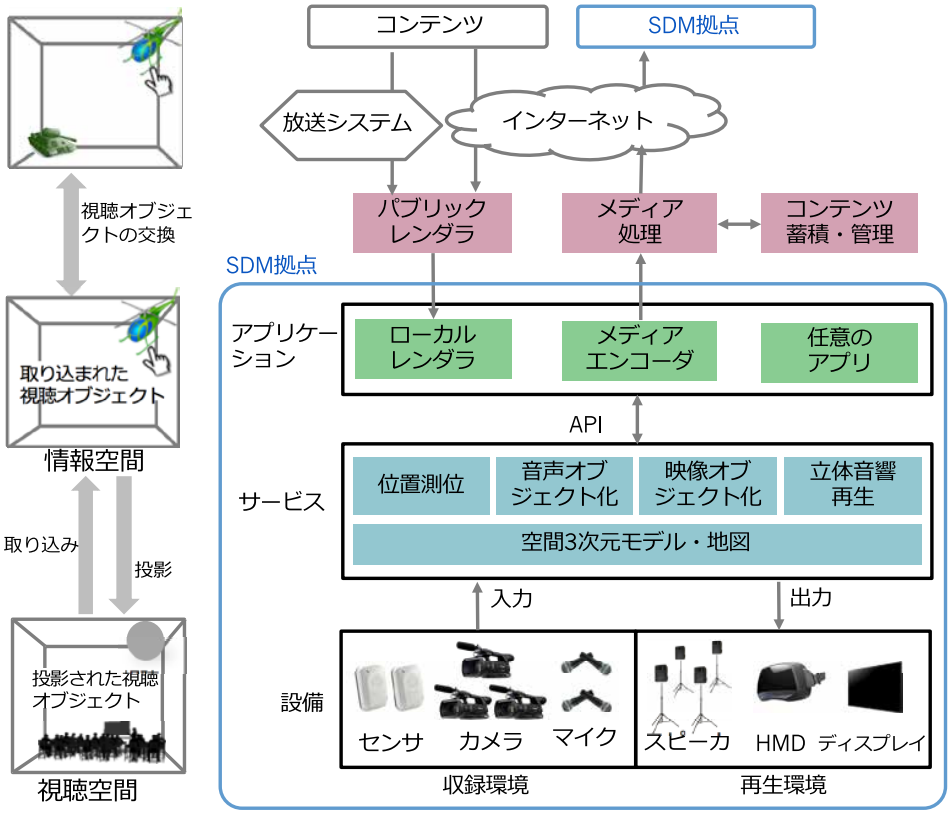


図 2.6: SDM アーキテクチャ

2.2.2 SDM オントロジ

SDM コンソーシアムでは，データを記録・再現するためのシステム構築だけでなく，実際にライブやコンサートのデータを記録し，活用するためのアプリケーション開発も行っている [109]．記録されたデータには対象の映像や音だけでなく，位置情報や楽器の向き，演奏された曲目や会場情報，録音プロセスといった情報も記録される．それらのデータは，様々なアプリケーションから参照，活用されることが望ましく，そのため構造化して相互にも，インターネットからも連携可能とすることが望ましい．これらを実現する技術はセマンティックウェブと呼ばれ，RDF (Resource Description Framework) で記述された情報（リソース）が，ネットワークを介して相互に連携することで LOD (Linked Open Data) を構成する．我々は SDM コンソーシアムが保有・管理しているデータを LOD に変換するため，SDM 用の RDF Vocabulary である SDM オントロジを定義した [8, 88, 48, 79].

SDM オントロジは，イベント全体の情報である SDMEvent，収録対象が置かれた状況・環境に関する情報（イベント内のプログラムの名前や内容など）である Context，収録対象の情報（対象の位置情報や種類など）である Target，収録機器情報（機器の位置情報や種類など）である Recorder，生成されるメディア情報である MediaObject の 5 要素（クラス）によって大きく構成される（図 2.7）．SDMEvent の下に Context，Target，Recorder が存在し，それらは相互参照しあう関係になっており，収録行為によって生成された生成物として MediaObject が存在する．なお，デー

タを公開する人が各々の語彙を勝手に定義するとデータの共有が難しくなるため，RDF では可能な限り既存の語彙を使用することが推奨されている．そのため，それぞれのクラス内の語彙には，schema.org[*12]で定義された語彙を取り入れている．

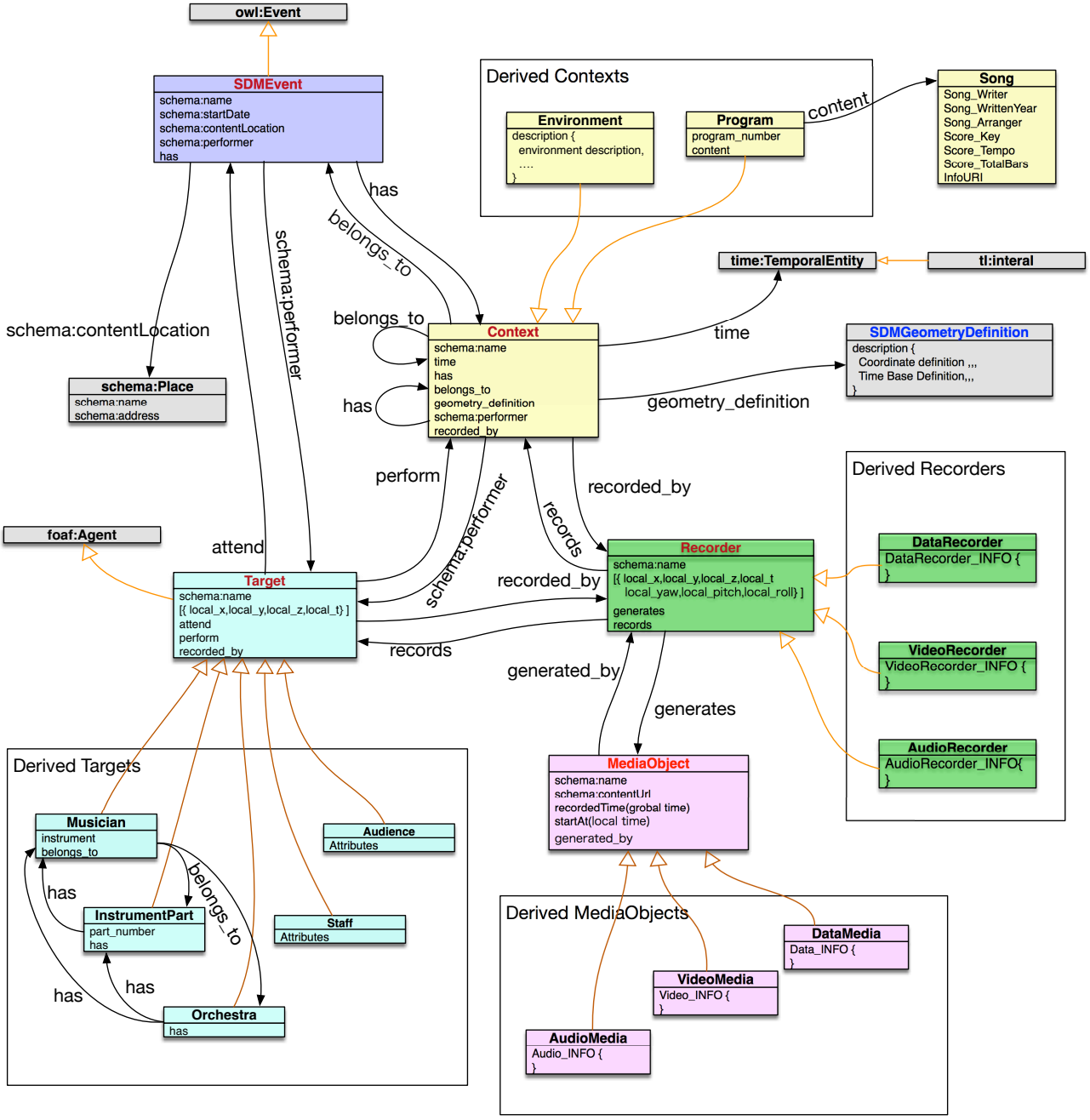


図 2.7: SDM Ontology のデザイン

SDM アプリケーションでは，SDM オントロジで定義された収録対象の RDF が保存された LOD クラウドに，クエリ言語である SPARQL を通じてアクセスし，必要なデータを入手して，加工・再生することを想定している（図 2.8）．

[*12] http://schema.org/

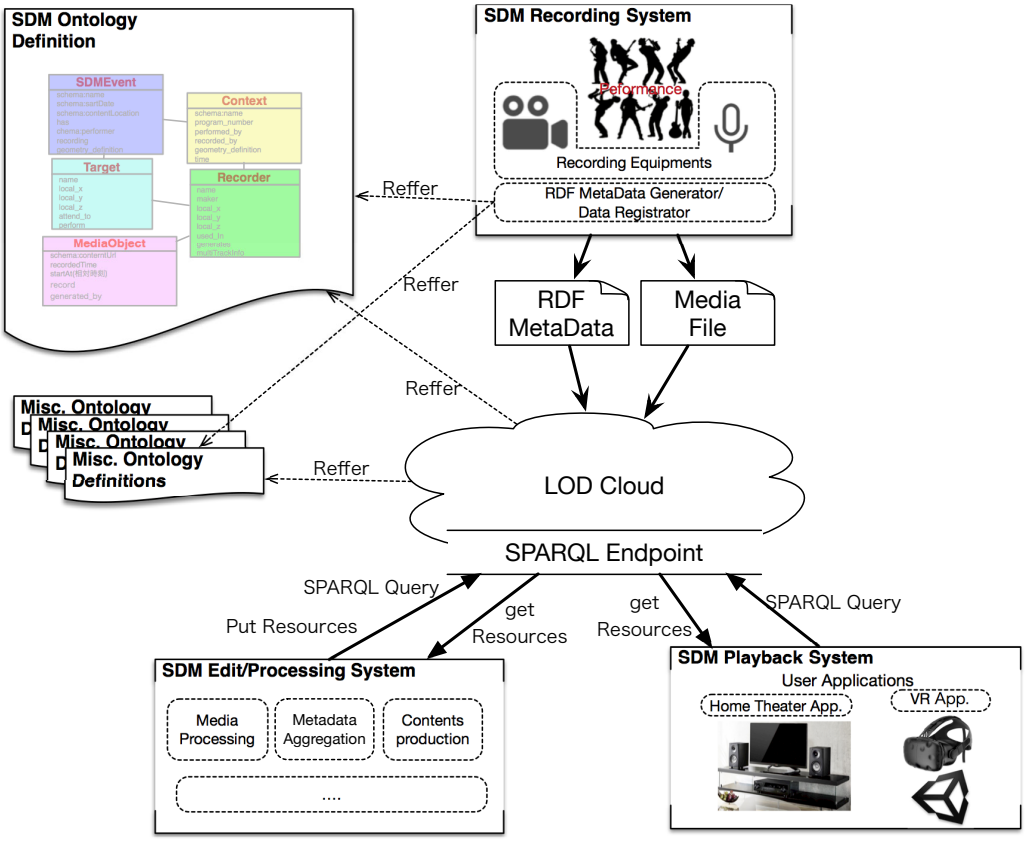


図 2.8: LoD クラウドによる SDM アプリケーションの構成

2.2.3 課題

建物のデジタルデータとの連携

SDM オントロジでは，サイバー空間上の音声オブジェクトの位置を指定したり，サイバー空間を構成する形状データを指定したりすることは想定している．しかしながら，それらはツール等を用いて都度手動で生成していた．BIM にはそれらの形状データや，スピーカー等の設置情報が含まれているため，そこから抽出したデータを用いるのが合理的といえるが，BIM の知識のあるエンジニアや研究者が少ないために，具体的なデータ活用のプロセスや実現技術についての研究が進んでいない．

VR 環境における立体視聴プラットフォーム

近年，Youtube やニコニコ動画などの動画配信サービスや HMD (Head Mounted Display) のような VR (Virtual Reality) デバイスの発展により，4K 映像や 360 度動画といった高臨場感のメディアも容易に再生できるようになってきた．5G のような大容量の通信技術や，MPEG4-ALS [51] などの音声のロスレス配信技術も一般化してきており，今後より高品質でリッチなストリーミングメディアが普及していくと考えられる．一方で，コンサートやライブの需要や人気が増大しているにもかかわらず，それらを記録した Blu-ray などのパッケージメディアは，収録機器の設置位置に制約を受けるため，視聴者の意思による自由な角度，距離による視聴は困難である．一般家庭においても，インターネットからの配信される VR コンテンツの視聴が可能になってきたが，配置されている音源は静的なものがほとんどで，動的で高品質なストリーミング再生は困難であった．

このような VR 環境における立体的視聴のためのプラットフォームに関するシステム・アーキテクチャとインタラクションの研究が必要といえる．

2.3 スマートビル

本節では，スマートビルの現状とその研究課題について論じる．以下では，スマートビル普及の背景について述べた後，スマートビルにおける設備制御の中心である BACS と BACnet について説明する．対象とするスマートビルの機能とユースケースを述べ，その研究課題について述べる．

2.3.1 背景

近年，スマートビルと呼ばれる高度な制御機能を有した建物が増えてきた．スマートビルについては多様な定義が存在しているが，一般的にはクラウド，IoT，AI などの技術を用いて，既存の建物設備制御システムでは実現が難しかった，高度な省エネや快適性・利便性の向上等を実現するビルという理解がされている．スマートビルの増加の理由としては，先に述べた技術革新に加え，BIM の普及，東日本大震災による省エネ・BCP 需要の高まり，人材不足などの社会的要請があるといえる．

省エネの大きな契機としては，2011 年の東日本大震災があり，それからエネルギー消費を極力抑え，災害時でもエネルギー的に自立した建築物として ZEB (ネット・ゼロ・エネルギー・ビル) が注目されるようになった．2012 年には経済産業省による「ネット・ゼロ・エネルギー・ビル実証事業」が開始され，2014 年 4 月閣議決定した「エネルギー基本計画」では，2020 年までに新築公共建築物等で，2030 年までに新築建築物の平均で ZEB を実現することが政策目標とされた．2015 年には ZEB ロードマップ検討委員会が開催され，ZEB の定義，実現・普及に向けたロードマップが策定された．2016 年には「ZEB 設計ガイドライン」などを発行され，経済産業省や SII（環境共創イニシアチブ）が中心となって，ZEB の推進を図っている．なお，ZEB ロードマップ検討委員会の定義によると，ZEB とは「先進的な建築設計によるエネルギー負荷の抑制やパッシブ技術の採用による自然エネルギーの積極的な活用，高効率な設備システムの導入等により，室内環境の質を維持しつつ大幅な省エネルギー化を実現した上で，再生可能エネルギーを導入することにより，エネルギー自立度を極力高め，年間の一次エネルギー消費量の収支をゼロとすることを目指した建築物」であるとされている．

また，2016 年 4 月からの始まった電力自由化によって，デマンドレスポンス (DR) と呼ばれる技術の実証・導入が加速している．経済産業省によると，DR とは「市場価格の高騰時または系統信頼性の低下時において，電気料金価格の設定またはインセンティブの支払に応じて，需要家側が電力の使用を抑制するよう電力消費パターンを変化させること」と定義されており，一般的にはアグリゲータと呼ばれる事業主体が需要家の電力を束ね，電力会社からの節電要請に従って，ネガワットと呼ばれる節電分を生み出すという形態がとられる．この際，需要家側のビルは要請に従って，目標に合わせるための節電制御（デマンドコントロール）が必要となる．照明の照度を下げたり，空調機の設定温度を下げたりする程度であればマニュアルでも可能であるが，発電機や蓄電池，再生可能エネルギーを組み合わせたデマンドコントロールは制御ロジックが複雑化するため，専用システムによる運用が一般化すると考えられる．なお，日本においては，エネルギー・リソース・アグリゲーション・ビジネス（ERAB）検討会によって DR が推進されている．経産省の報告 [98] によると，2017 年度は九州エリアで 2 回，東京エリアで 13 回，2018 年度の夏季も関西エリア 2 回，東京エリア 4 回の DR が発動され，需給調整に活用されるようになってきている．

上記のような ZEB や DR の普及が進む一方で，高度な設備保守・メンテナンスに必要な専門技術者の不足が問題になっている．「ビルメンテナンス情報年間 2019」によると，ビルメンテナンス業務での課題として人材不足と高齢化が深刻化しており，人材の過不足状況では，一番良い関東甲信越でも 7 割，ほかの地域は 8 割程度不足しているという調査結果が出ている．また，高齢者雇用状況においては，60 歳以上の常勤従業員の比率が 36.7 %となっている．慢性

的な人手不足によって，配属された若手従業員が別現場にすぐに配置換えがされることも多く，管理ノウハウの継承も大きな課題となっている．これらの課題解決としては，AI による業務負荷の低減と，遠隔監視や遠隔制御が必要といえる．

更には 2014 年に米国のデロス・リビング社により考案された，well 認証（Well Building Standard）が国内でも認知が広がっている．well 認証とは，心身の健康をサポートしたり，快適性が高く，人間の健康やウェルネスに好影響をもたらす建築物に与えられる認証である．これは，例えばオフィスにおいて，そこで働く従業員が心身共に活発になることでパフォーマンスが向上し，労働生産性が上昇するとの考え方に基づくものであり，認証を取得したビルの不動産価値の向上や，魅力的なオフィス提供による労働人員の確保・定着にもつながるとされている．更には，労働生産性の向上の観点から，労働人口減少のための施策として 2019 年 4 月より施行されている「働き方改革法案」とも相性が良いといえる．評価項目としては，空気，水，食物，光，フィットネス，快適性，こころ，の 7 つがあり，運用によるところも大きいが，建物設備によって制御可能な項目も多い．

上記の背景を受け，省エネだけではなく，DR 対応や AI による省人化，快適性の向上を目指したスマートビルの進化と普及が進んでいる．

2.3.2 BACS (Building Automation and Control System)

BACS は，ISO・TC205[*13] (Building environment design) の WG3 (Building Automation and Control System Design) およびヨーロッパの規格団体である CEN/TC247 (Building Automation, Controls and Building Management) の協働作業により規格化されている ISO16484 シリーズによって定義されている．「自動制御（インターロック含む），監視，最適化，人的操作，建築設備機器の省エネルギー的で経済的・安全な運転操作を達成するための管理を目的とするシステムと全ての製品・エンジニアリングサービスに対する呼称」と定義され，従来の日本における中央監視システム (BAS:Building Automation System) や BEMS を包含する概念である [112].

図 2.9 に BACS の構成イメージを示す．概ね 1 万 m^2 以上のビルでは，空調・照明などのサブシステムをモニタリング・制御する中央監視システム等が防災センターに設置され，それらを使って施設管理員が集中監視業務を行っている．一般的にサブシステムはそれぞれ構築ベンダーが異なり，責任範囲を明確にするためスタンドアロンで動作するよう構築される．それらがゲートウェイ機器を通じて BACnet などの専用プロトコルでシステム間の通信を行う構成となっている．BACS は閉域網による構成を前提としていたが，近年インターネットとの連携事例が増えてきており，柔軟な連携の対応が求められている．なお，BACS で扱われるデータは「ポイント」と呼ばれ，デバイスの計測・状態値である計測・計量・警報・状態・設定などを表している．なお，BACnet 以外にも，ISO/IEC14543 (KNX)，ISO22510 (KNXnet/IP)，ISO/IEC14908 (Lontalk)，ISO/IEC/IEEE 18880 などがあるが，本研究では国内における普及の状況を鑑みて，BACnet を取り扱う．

2.3.3 ISO16484 シリーズ

BACS の標準化以前は，中央監視システムは 専門ベンダ 1 社がシステムを閉鎖的に製作し，適用される通信プロトコルも専用であり，他社のシステムと連動させる際は専用のインタフェース装置が必要であった．近年，エンドユーザのメリット及びベンダ自身のメリットの向上の観点から，標準化の動向も後押しして，ベンダー間のシステム構成とインタフェース共通化といった，BACS のオープン化が主流となっていった [121]．実際のプロジェクトでも，サブシステム間の連携を行う場合，多くが BACnet (A Data Communication Protocol for Building Automation and Control Networks) と呼ばれる BACS 専用プロトコルを採用している．しかしながら，BACS の標準化はまだ歴史は浅く，ベンダー専用プロトコルも未だに使われている．

[*13] https://www.iso.org/committee/54740/x/catalogue/

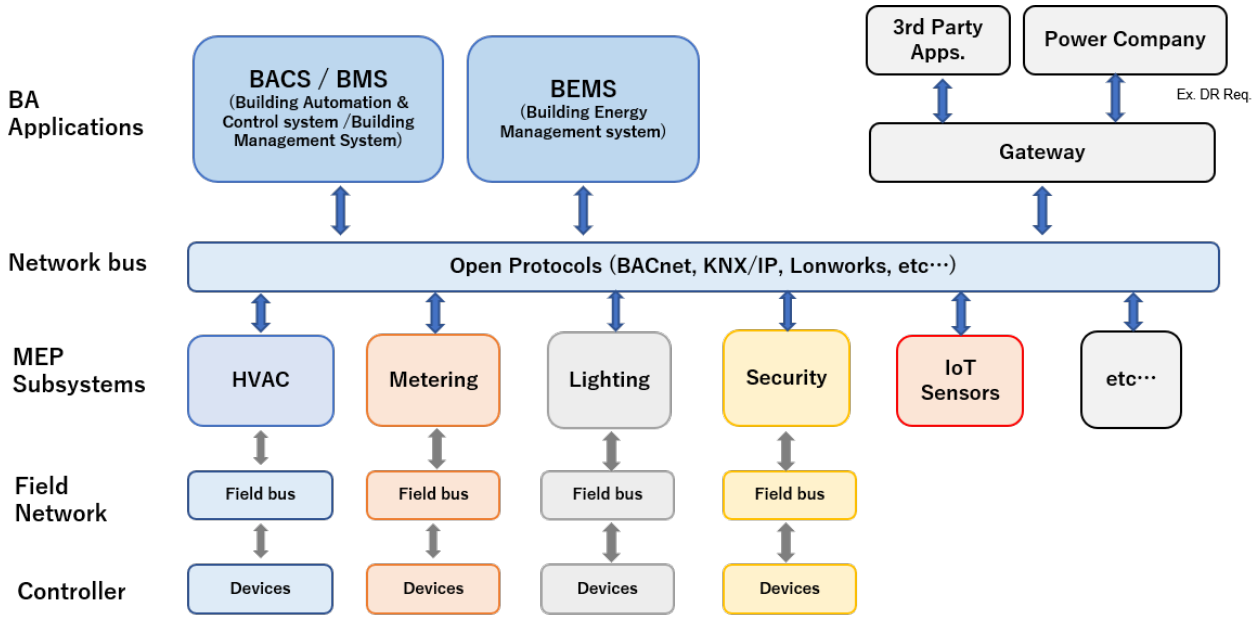


図 2.9: BACS の構成イメージ

表 2.1: ISO16484 シリーズと関連規格

規格コード	名称	公開時期	内容
ISO16484-1	Part 1: Project specification and implementation	2010	BACS プロジェクトの構築と他システムとの統合に関して Design, Engineering, Installation, Completion の各フェーズにおける General Principal を規定する.
ISO16484-2	Part 2: Hardware	2004	設計者・発注者・エンドユーザのための BACS のハードウエアに関する仕様書の作成ガイド. ビルの監視制御システムを BACS (Building automation and control system) と略称しそのハードウエアについて定めている.
ISO16484-3	Part 3: Functions	2005	BACS の基本ソフトウエアに関する仕様書の作成ガイド. BACS に搭載する基本的機能と入出力関係をしめす BACS ポイントリスト等を定める.
ISO16484-4	Part 4: Control applications	-	一般居室オートメーションおよび熱源, FCU, CAV, VAV 等の最適制御のアプリケーションのガイド. 2018 年 9 月オスロ会議で削除決定 (Resolution339)
ISO16484-5	Part 5: Data communication protocol	2004	BACS のデータ通信サービスとプロトコルを定めるもので, ASHARE BACnet の規格を採用している.
ISO16484-6	Part 6: Data communication conformance testing	2014	BACS の ISO16484-5 のプロトコルに対するデータ通信適合試験について定める. ASHRAE BACnet の試験方法を採用している.
ISO16484-7	Part 7: Contribution of BACS to energy performance of building	-	ビルにおける省エネにおける BACS の役割と機能について定める. 2017 年 9 月の東京会議で規格番号を削除し, 新たに NP 52120-1 と 52127-1 への移行が決定. (Resolution316)

BACS における ISO16484 シリーズの関係性を, 表 2.1, 図 2.10 に示す. 現状では, プロジェクトにおける BACS の構築プロセスを定めた ISO16484-1, ハードウェア仕様を定めた ISO16484-2, 基本機能とポイントリストなどを定めた ISO16484-3, 専用プロトコルである BACnet を定めた ISO16484-5, BACnet に対するデータ通信適合試験について定めた ISO-16484-6 が規格化されている.

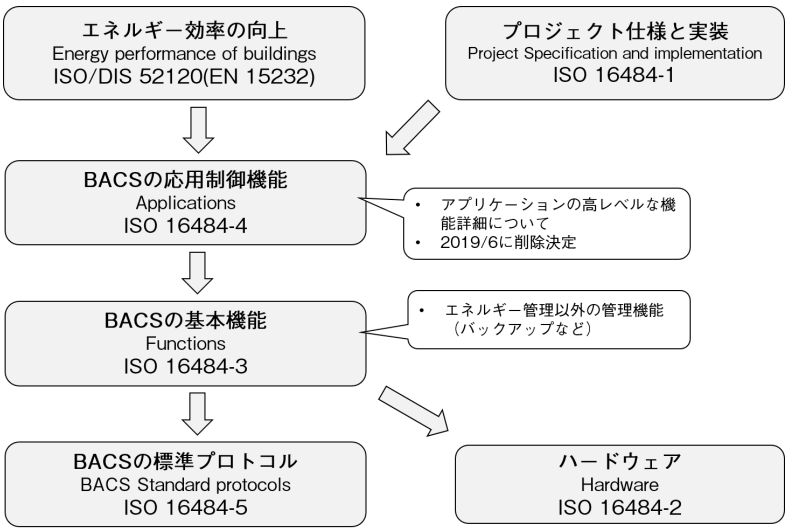


図 2.10: ISO16484 シリーズの関係性

2.3.4　BACS システム構成の変遷

BACS の構成はその時代のニーズや技術の進歩により構成形態を変え，進化してきた．図 2.11 にシステム構成の変化を示す．なお，以下の整理は一般財団法人建築コスト管理システム研究所のレポート [86] を参考にした．

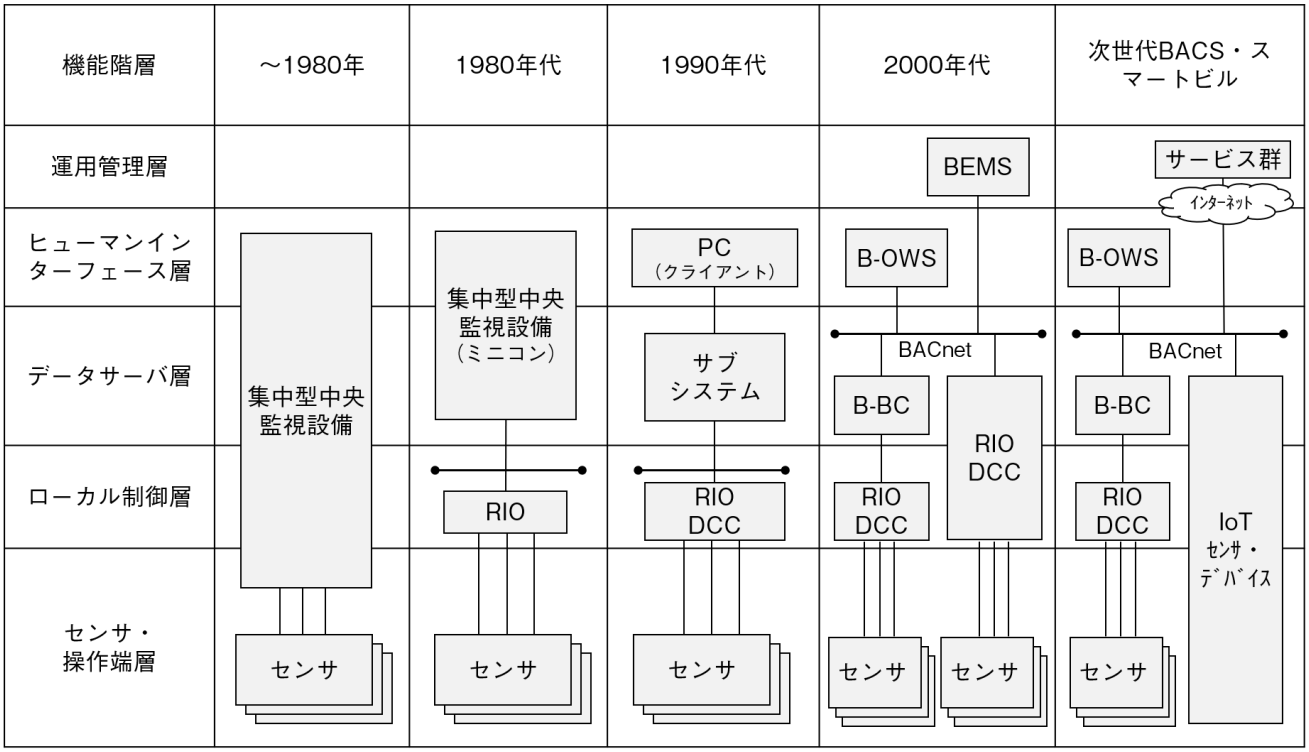


図 2.11: BACS のシステム構成の変遷（建築コスト管理システム研究所・新技術調査検討会資料より）

1980 年代までは，BACS は大型の専用監視卓に全てのセンサ・操作端末が接続される構成であった．これは空調な

どのサブシステムが，マイコンによるデジタル制御可能になってきたためで，それまでのアナログなローカル制御から，スケジュール運転やモニタリングなどの機能を有する，デジタル化された集中制御に切り替わった時期にあたる [96].

1980 年代になると，ミニコンと呼ばれる産業用コンピュータが採用され始め，ネットワーク通信の概念が導入された．入出力装置（RIO：Remote I/O) が分離され，設備機器側に入出力ユニットが置かれることで，直引きの信号配線に比べて大幅な工事費の低減が可能になった．この頃，インテリジェントビルと呼ばれる概念が，日本で普及を始めている．

インテリジェントビルは，「生産性向上のための建築空間・建築設備および共用の情報通信設備を備えて，入居者がそれらのサービスを享受できるオフィスビル」である [103]．1985 年の通信自由化を契機として，様々な設備のネットワーク化が可能となり，地域電話サービスが充実することで産まれたといわれている．ビルの最適制御を行うビルディングオートメーション（BACS)，ビルの共用 LAN などの通信サービスを行うテレコミュニケーション，ビデオテックスやビデオ会議室システムの提供および OA 機器の導入サポートなどを行うシェアードテナントサービスの 3 つを特徴としており，オフィスにコンピュータやプリンタなどの OA 機器導入が進んだ時期であることから，OA フロアの導入といった建築的特徴も有していた．ビル側で共通の通信インフラや設備を持ち，テナントに共有することにより，空調機器のパーソナル化をはじめとした，現在のスマートビルと同様の機能を提供する試みも行われている．ただし，この当時は標準化が進んでいなかったため，排他的な独自ソリューションが多く作られていた．

1990 年代になるとパーソナルコンピュータ（PC）やワークステーションの普及が進み，いわゆるクライアント／サーバモデルによる BACS 構成が一般化してきた．システムの分散制御を可能にする小型コントローラ (DDC:Digital Direct Controller，PLC: Programmable Logic Controller) の普及し，センサ・デバイスの制御ポイントを参照・遠隔制御するサーバとしてのサブシステムと，インターフェースを提供するクライアント PC によって BACS が構成できるようになった．

2000 年代には，BACnet や Lonworks などのオープンプロトコルが普及が進み，BACS の構成要素である B-OWS (BACnet Operator Workstation) や B-BC (BACnet Building Controller) の標準化も進んだ．小型デバイスの処理性能の向上により，より下位のデバイスでデータサーバ層の機能を代替できるようになる．また補助金等の施策により，中小ビルにおいても BEMS の導入が進んだ．

それ以降のスマートビルでは，IoT などの導入やクラウドサービスとの融合が図られ，エネルギーマネジメントシステムの高付加価値機能がインターネットを介して提供されることで，入居者や管理者がより使いやすく，より高度なサービスが享受できるようになってきている．

2.3.5 BACnet (Building Automation and Control Networking protocol)

BACnet は BACS のための標準プロトコルとされており，1995 年に ANSI/ASHRAE Standard 135 に採用され，その後 2003 年に ISO 16484-5 [*14]として採用された．ASHRAE Standing Standard Project Committee 135 によって精力的な改訂作業が行われており，現在（2020 年 5 月）時点の最新のバージョンは ANSI/ASHRAE Standard 135-2016 (ISO16484-5:2017) となっている．日本での普及においては，電気設備学会が日本の独自性を考慮した拡張をを行い，「BACnet システムインターオペラビリティガイドライン」 [91] を発行して普及促進を図っている．

BACnet の特徴は，BACS の構成要素を「オブジェクト」「サービス」によってモデル化しているところにある．オブジェクトとは，BACS の各機能を複数の属性の集合体として抽象化したものであり，サービスとは BACS の動作自体を抽象化したものである．

オブジェクトは例えば，積算値 (Accumulator)，アナログ入出力 (Analog Input/Analog Output)，バイナリ入出力 (Binary Input/Binary Output) などであり，ANSI/ASHRAE 135-2016 においては 60 の標準オブジェクトが定

[*14] https://www.iso.org/standard/71935.html

義されている．また，それら標準オブジェクトごとにプロパティ群とそのデータ型，適合クラスが定義されている．図 2.12 に示すように，デバイス (Device) と呼ばれるユニークなオブジェクトが，複数のオブジェクトオブジェクトを持つ階層構造になっている [78].

サービスはそれらのオブジェクトに対する通信方法を定義している．例えば, 図 2.13 では, B-OWS が WriteProperty サービスを使ってプロパティ (Present_Value) に書込みしており，ローカル機器の変更確認を行った後，ConfirmedEventNotification サービスを使って，デバイスの状態変化を別サービスに伝えている．

BACnet では様々な物理層がサポートされているが，現在では BACnet/IP が広く用いられている．ここでは，OSI 参照モデルでのトランスポート層に UDP/IP，データリンク層に IP 層を仮想的にワイヤリングプロトコルとして扱う BVLL (BACnet Virtual Link Layer) が用いられている．なお，BACnet/IP では，IPv4 だけでなく，Addendum 135-2012aj ANNEX U で IPv6 への対応がなされている．

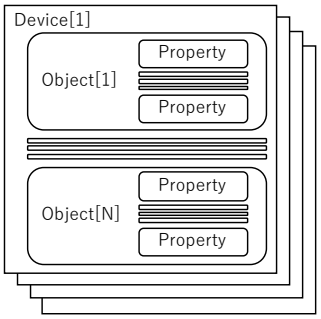


図 2.12: BACnet のオブジェクトモデル

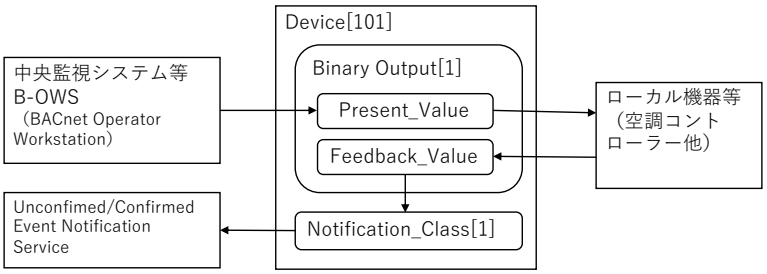


図 2.13: BACnet のサービスモデル

上記のようなポイントを起点としたモデリングは BACS では一般化しているが，オブジェクト指向プログラミング（OOP:Object Oriented Programming）などに比べると，モデリングの粒度が細かく扱いにくい．BACnet ではプロパティの操作をサービスによって規定しているが，値の取得や設定，設定時の通知などといった低レベルの関数にあたるものであり，OOP のようにユーザがメソッドを定義することはできない．また，建物の空間構成や設備間の関係性・種別といったセマンティクスが欠けており，IoT・AI などの専門業者とスマートビルのアプリケーションを開発する際の大きな課題となっている．これらは 4 章で再度考察する．

なお，プロジェクト毎のポイント定義は，ISO16484-3 で規定されている「ポイントリスト」と「機能ブロック線図」等によってなされる．ポイントリストはスプレッドシート形式の文章であり，ハードウェアに関する仕様は含まれないが，BACS で必要とする物理的または通信の入出力，処理装置の性能と記憶容量の決定に関する情報が記述される [93].

2.3.6 スマートビルの機能

Bali らによると，スマートビル（またはインテリジェントビル）の機能には以下があるとされる [12].

1. 安心と安全 (Safety and Security Functions)
2. エネルギー効率化 (Energy-Efficiency Functions)
3. 快適性の確保 (Comfort Functions [Ergonomics of the Building])
4. より高レベルの管理機能 (Higher-Level Management Functions)
5. ディスプレイと制御機能 (Display and Operating Functions [User Interface])

一般的な BACS と同様といえるが，上記の機能（特に快適性など）を提供するスマートビルのアプリケーションは，

個別性が高く汎用性だけを考慮に入れられないこと，それらが入居者に受け入れられるためには，最初にインストールされ最適化した機能だけでなく，様々な状況で個々のユーザの行動パターンに対応できる必要があること，そしてUI (User Interface)/UX (User Experience) が重要な要素であることを指摘している．

Bali らは更に，スマートビルの機能実現のためには，ビルの設計者（建築家）は多くの専門家とコラボレーションを行い，エンジニアリングの結果，システムの稼働に責任を負わなくてはならないと述べている．しかしながら，建築家とエンジニアの間には避けがたいギャップがあり，スマートビルではそのギャップがより顕在化することを指摘している．そのため機能提供に関わる全てのサブシステムを対象として，システム間のインタフェース設計を行う専門家を加えてチームを編成する必要性を述べている．これらは重要な知見であるが，そのような専門家は日本の建設業界においてはほとんどいない．人材育成が大きな課題ではあるが，現状ではそれらの設計を支援するためのフレームワークや，自動化のための施策が求められている．なお，明快には述べられていないが，より高度なエネルギーマネジメントや快適性の確保などのために，IoT や AI の技術適用が想定されていると考えてよいだろう．

2.3.7 スマートビルのユースケース

我々は，スマートビルの機能を実現するため，多数のアプリケーションの実証を行ってきた（表 2.2）．

表 2.2: スマートビルのアプリケーション例

No.	アプリケーション	説明
1	見える化システム	エネルギー，環境情報の見える化．ウェブ，インターホン，デジタルサイネージでの表示，VR，AR などによる 3 次元的な可視化もある．
2	ポータルサイト	入居者がスマートビルの各機能にアクセスするためのポータル機能を提供．
3	コミュニケーションシステム	入居者に光・温熱環境などを申告させ，設備の制御に反映させる．ウェブやスマートフォン，ウェアラブルデバイスからの申告がある．
4	遠隔監視・制御	設備機器の発停，モード変更など．中央監視機能の遠隔からの代替．
5	パーソナル制御	入居者個人が制御可能な照明，空調機器の制御．ウェブでのアクセスが主流．
6	デマンドレスポンス	節電要請を受けた際に，建物側の節電のポテンシャルを計算し，適切な優先度で節電・デマンドコントロールを実施する．
7	ウェルネス制御	IoT などを使って，人に起因する要素（ヒューマンファクター）を抽出し，それらに基づき，より快適な制御を行う．
8	データ出力	指定のフォーマットでデータ・プラットフォームからデータを出力する機能．報告書などに利用する．
9	負荷予測システム	翌日以降のエネルギー負荷（電力，熱）の予測．直近のリアルタイム予測もある．
10	最適計画システム	負荷予測などをもとに，設備機器の最適な運転スケジュールなどを計算する．
11	AI による遠隔制御	事前に合意した評価指標に基づいて，BACS の最適運転を AI エンジンが自動的に行う．

これらの実践の中で，スマートビルのアプリケーションは，パーソナル制御やデマンドレスポンス，一部の見える化システムなどにおけるリアルタイム制御のアプリケーションと，負荷予測やビル設備の最適動作パターンの推定など，概ね 1 日毎のバッチ処理のアプリケーションの 2 つに大別されることが分かった．以下では，それらの中で特徴的なユースケースについて述べる．

多様な電源ソースを組み合わせたデマンドレスポンス

DR (Demand Response) の手法としては，手動によるものと，M2M (Machine to Machine) の通信により自動的に行う ADR (Automated Demand Response）がある．前者は例えば，節電要請の際に電話やメール等で事業者に対して依頼を行う手法であるが，リアルタイム性や実効性に乏しいといえる．従って，ビル設備の遠隔自動制御を伴う

ADR が注目されている．なお，DR には節電だけでなく，上げ DR と呼ばれる，電力消費を増やす要請もある．これは例えば太陽光発電プラントの増加によって，日中に電力が余る際に計画される．

事業者によって詳細は異なるが，DR によるビル制御は例えば次のようなシナリオになる．電力会社からの節電要請 (DR 信号，主として OpenADR[*15]) を受け取った事業者は，要請に記述された削減量，削減時間の目標値に従って，節電可能な設備のデマンドコントロールや蓄電池や発電機などによって，建物のデマンド（消費電力）を事前に定義されたベースラインに対して低減させる．その際，削減量の目標値に対して，指定時間の間± 10 %を達成し続けることでインセンティブを得ることができる．

図 2.14 に実際の DR 発動時の，システムの挙動を示した．なお，この見える化の仕組みは，サーバのアクセスログの可視化等によく用いられる Elasticsearch と Kibana[*16]を用いた．左上のグラフがサマリであり，DR の実行時刻になると，ベースラインに対する目標値（水色のライン）が設定され，蓄電池や発電機，EV（電気自動車）といった多様な電源リソースを，リアルタイム（10 秒間隔）で制御することによって，赤いラインのデマンドを目標値に近づけている．後半，ハンチングしているのは，制御アルゴリズムによるところが大きい．ここでは制御のパラメータとして静的な閾値を設定しており，その周辺でデマンドが上下した際に，頻繁な発電機の発停が起きている．これらの抑止のためには，PV による発電や建物デマンドに対するリアルタイム予測が有効である．詳細について，7 章で再度検討する．



図 2.14: デマンドレスポンスの動作時の挙動

IoT デバイスを用いた快適制御

IoT を使った高度な BACS の事例として多いのが，快適性や利便性の向上のためにワイヤレスのセンサの活用である．例えば，湿度センサにより，入居者により近い環境の温熱環境を捉えて制御に活かすものや，トイレの利用状況の把握のための開閉センサなどである．最近では，入居者の在不在やアクティビティを捉えることがトレンドになっており，ビーコンなどを使って入居者の位置情報を捉えることで，執務室環境の見える化を行うサービス[*17]も現れてきた．

竹中工務店東関東支店において，我々はワイヤレスの環境センサにより細かい温湿度などを捉え，Bluetooth のビーコンによってオフィスの執務者の位置情報を取得するとともに，ウェアラブルデバイス (Apple Watch) によって心拍

[*15] https://www.openadr.org/
[*16] https://www.elastic.co/jp/
[*17] https://mycity.co.jp/

数，加速度などから計算した代謝量をリアルタイムに取得し，執務者の不快感申告も考慮した上で，風量制御が可能なパーソナルファンと天井の放射パネルの温度を制御することで，快適な執務環境の提供を目指したウェルネス制御システムを構築・評価した [83]（図 2.15）．制御対象設備の影響範囲には複数の執務者がいるが，対象グループの代謝量や不快感申告から，PPD（Predicted Percentage of Dissatisfied）と呼ばれる予測不快者率が最小化するような制御値によって，総合的な快適性を保った制御となっている．

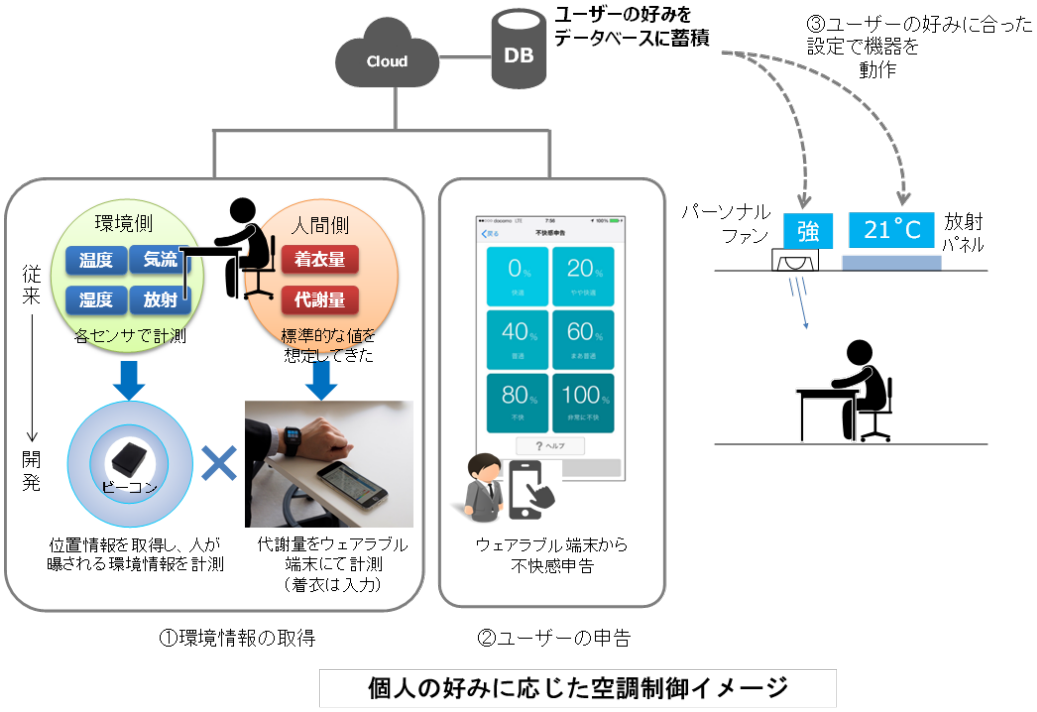


図 2.15: ウェアラブルデバイス，ビーコンを用いた空調最適制御

図 2.16 に，これらのシステムを実現したインフラの構成を示す．MQTT（Message Queue Telemetry Transport）を用いた疎結合なアーキテクチャとなっており，アプリケーション（リアルタイム制御）から発出された制御コマンドが，ビルごとに設置されたゲートウェイによって BACnet 等に変換され，現地システムによって実行される．MQTT Broker によって複数棟をまとめる構成となっており，前に述べた DR アプリケーションも，同じプラットフォーム上で構築されている．これらのアーキテクチャは，BACS と MQTT の親和性について評価した先行研究 [118] の成果による．

強化学習による高度な設備制御

一般的な BACS は，PID (Proportional-Integral-Differential) 制御と呼ばれるフィードバック制御を前提としている．入力値の制御を出力値と目標値との偏差，その積分，および微分の 3 つの要素によって制御を行う方法であり，ビル管理員や入居者の目標値設定や，熱源機器などはスケジュールに従って動作する．一般的に熱源機器の温度設定や運転スケジュールなどは，建物の稼働時に最初に決められた後は，ほとんどチューニングされることはなく，またそれらの変更は 4 半期に一度程度しか行われないのが常態化している．これはビルの維持管理業務の契約には，このようなチューニングや省エネまで含まれていないことも想定されるが，人手不足によって専門性を持った施設管理員が減少していることも影響していると思われる．更に，快適性を両立させようとした場合，人手による制御，または制御ロジックの適用には限界がある．

これらの課題に対応するために，AI を適用する試みが増えてきた．建物制御の研究においては，シミュレーション

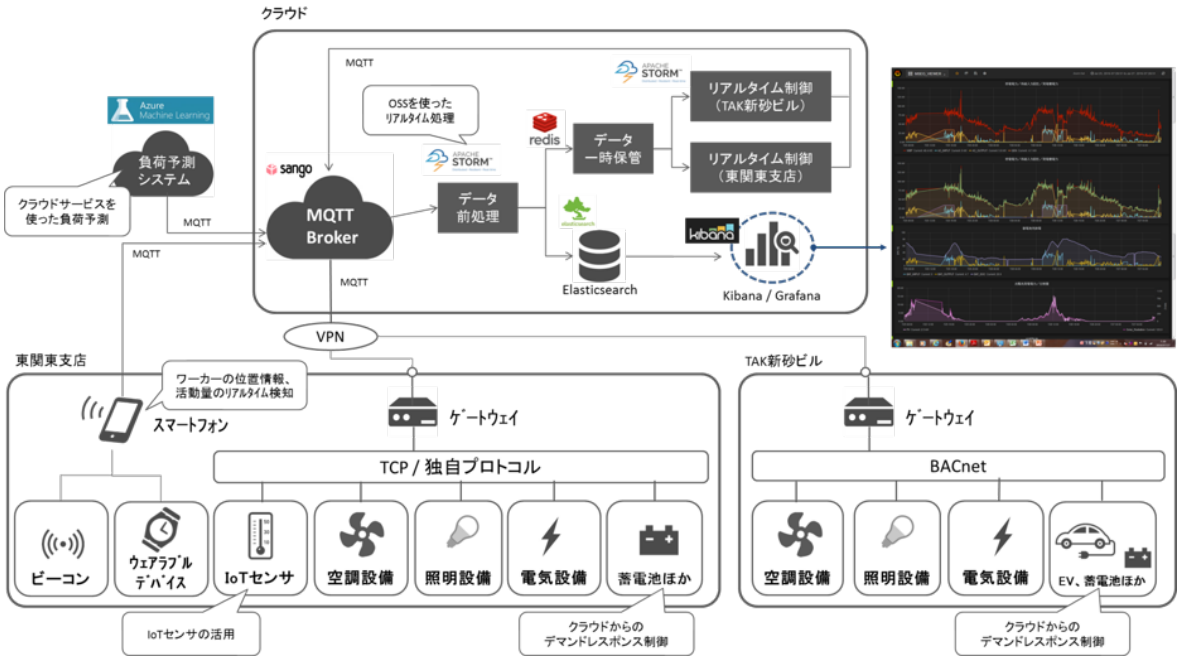


図 2.16: 竹中工務店東関東支店でのシステム構成

をベースとしたモデル予測制御と呼ばれる手法が広く行われてきていたが，我々は強化学習を使って自律的にパラメータの最適化を行いながら制御する研究と実践を行ってきた [115].

図 2.17 は，照明制御を対象とした際の強化学習による設備制御のイメージである．データ・プラットフォームから取得したデータを使って，制御のための目標値を予測する AI と，決められた目標値から報酬を定め，それに従って行動（制御）を決める強化学習 AI の 2 つがあり，それぞれが独立して学習を行う．なお，強化学習は最適化のために指標を定める必要がある．図 2.17 の事例だと，電力量の最小化，照明であれば輝度，空調であれば予測平均申告 PMV (Predicted Mean Vote) [28] と呼ばれる温熱環境評価指数を指定範囲に収めることで報酬を得るモデル設計としている．

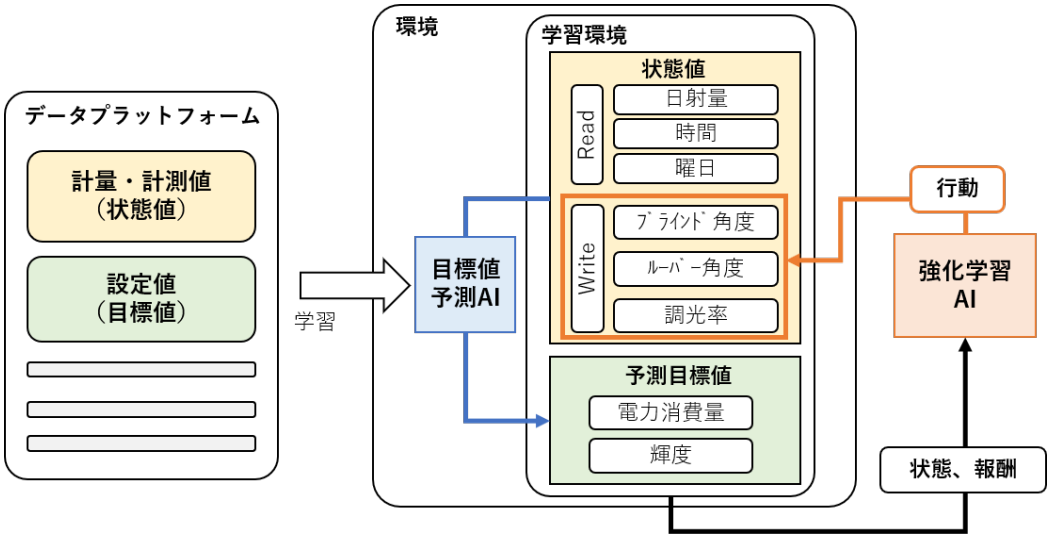


図 2.17: 強化学習による設備制御イメージ

なお，強化学習によって最適な設備の制御値を求める研究は Han らの調査 [35] に詳しい．この調査の時点では，非常に限定的な環境や制御対象での評価が多いといえるが，AI 技術の進展から，本節で述べた多くの特徴量と制御対象による制御も実用段階に入ってきたといえる．

2.3.8 課題

以下では，実際の建設プロジェクトにおいてスマートビルを実現する際に考慮すべき要件と課題を考察する．

スマートビル構築の要件

IoT，AI の適用を前提とした新築時のスマートビル建設プロジェクトの流れは図 2.18 のように表現できる．計画，調達，設計（基本設計，実施設計），施工，維持・管理，更新（改修）というライフサイクルが存在し，竣工時に建物は引き渡しとなるが，スマートビルのアプリケーションは，試運転や性能検証が施工後の対応になることが多い．これは，実際に建物の利用や入居が始まって，データが集まらないとチューニングが難しいためである．

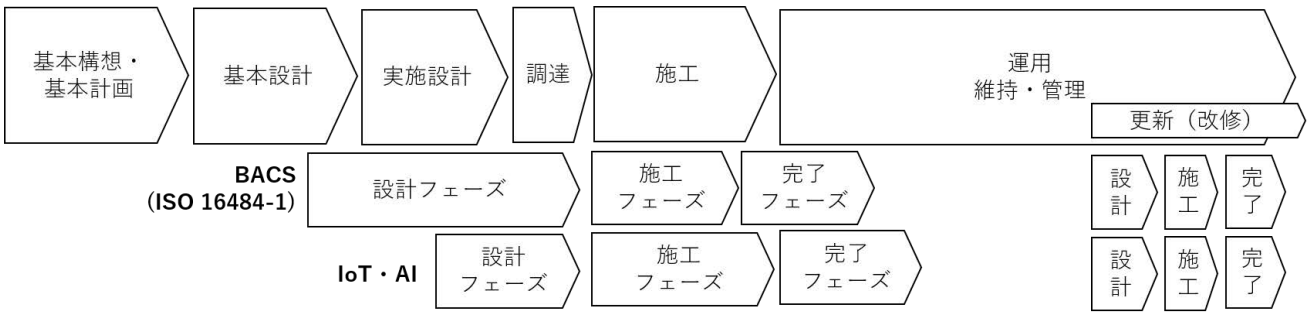


図 2.18: 建設プロジェクトのライフサイクル

また，国内の建設プロジェクトでは，一般的に総合請負業であるゼネコンが元請となってプロジェクトを契約し，元請業者（ゼネコン）から工事の一部を請け負う下請業者であるサブコンが存在し，その下請けに専門業者がいるような重層下請構造になっている．具体的には，設計会社が発行した設計図書をもとに，ゼネコンの設備設計が詳細設計を行い，ゼネコンの設備担当が電気工事，衛生工事，空調工事などを担当するサブコンをまとめて施工管理する．

BACS や IoT・AI などの専門業者は，商流としてはサブコンの下請となるが，設計意図を実現するため，実施設計中に設計者と綿密な打ち合わせを行い，機能詳細の合意，デバイスの選定，構築などを行う．IoT や AI などを用いた高度な BACS のアプリケーションを構築しようとする場合，ネットワークやクラウドなど従来的な設備設計者とは異なる専門性が求められる．

また，ソフトウェアハウスのような専門業者は，工事業者とはが大きく慣習が異なるため，その振る舞いがプロジェクトの進行上問題になることが多い．建築に関するドメイン知識がないことが多く，BACS にどのようなデータが存在し，どう連携させるべきか，期中に細かいやり取りや調整が必要になってくる．そのための調整役として専門家が必要であることは，2.3.6 節で述べた通りである．

上記も考慮すると，スマートビルを実現するための要件・課題は，表 2.3 のようにまとめられる．これらは，先端技術を有した専門業者の新規参入を拒む障壁となっている．スマートビルは，上記を考慮しつつ，専門業者間の調整のための具体的な業務フロー，データ受け渡しのルールを考慮した上で設計，構築される必要があると考える．そのため，具体的には以下の研究が必要と考える．

表 2.3: スマートビルを実現のための要件・課題

No.	項目	説明
1	複数ベンダー，システムが混在する	サブシステムごとに構築ベンダーや商流が異なり，品質保証の責任範囲が異なる．全体をまとめる調整役（SIer) が不在であることが多い．調整には高度なドメイン知識が要求されるため，業界全体で人材がそもそも不足している．
2	拡張性を考慮しない	竣工してからはシステム構成が変化することや，設計で意図された動作以外の拡張は考慮されない．機能拡張の場合は，改修工事の際に都度エンジニアリングを行う慣習となっている．
3	参入障壁が高い	AI や IoT のエンジニアは建設のドメイン知識がないことがほとんどであるために，参入障壁が高い．データにセマンティクスが付与されていることが稀であるため，サードパーティーベンダーによるデータ理解が難しい．
4	期間中での性能検証が困難	BACS や IoT は建設プロジェクトにおいては最も後期に構築が始まるため，データが取れるのも最後の方である．入居者の利用を前提とするシステムもあるため，プロジェクトの期中で評価が行えないことが多い．なお，BACS の専門業者はそのまま保守契約を受託することが多く，そのまま囲い込みが発生しやすい構造になっている．
5	特有の商習慣	ソフトウェアハウスと建設プロジェクトでは商習慣が大きく異なる（工事契約を前提とする）．そのため新たな製品，サービスなどは建設請負の商流から離れて，建築主が直接発注するのが好ましい場合がある．また，IoT は一般的に大量のポイントとなるため BACnet に取り込むことは，コスト増大を招く（計量ポイントに比例してコストが増える）．BACnet での連携するシステム数によっても初期コストが変わってくるため，連携先は可能な限り集約する必要がある．

スマートビルのセマンティクス

上で述べた建設特有の商習慣が，参入障壁を高くしている一因となっているといえるが，スマートビルの構築においては，AI や IoT といった専門業者とのコラボレーションが必須である．それらの参入を促し，構築後のアプリケーションの移植性と再利用性を高め，知識ベース等を用いたソフトウェアによるシステムの自動設定を可能とする Software Defined BACS の実現のためには，スマートビルのための実践的なセマンティクスとオントロジの特定と，それらを用いたデータモデルの生成方法の研究が必要である．

実践的なデータ・プラットフォーム

IoT などのシステムは工事請負会社の商流ではない場合があり，責任範囲を明確化にするためにも，データ・プラットフォームを介した疎結合なシステム構成が望まれる．また，初期コストを抑える意味でも IoT による大量のポイントはデータ・プラットフォームを介して，BACS と連携することが好ましい．スケーラビリティの高いデータ・プラットフォームによって，1 棟あたりの運用コストやサービス料金の低減も図れると考える．建設特有の要件と，多様なユースケースを考慮した実践的なデータ・プラットフォームのアーキテクチャや構築に関する研究が必要である．

ビルシステムのプロファイリング技術

2.3.7 節で述べたように，DR のための制御においては多様な制御対象を考慮する必要となり，かつ導入設備によっては太陽光パネルによる発電量も考慮しながらターゲットのデマンドをアルタイムに合わせていく必要もある．複合的な設備システムの挙動は複雑であるため，設備単体ではなく全体のシステムを対象としたプロファイリング技術の導入が必要である．また，正確なプロファイリングが可能になることで，AI 適用が容易になり，DR に必要な負荷予測の精度向上も期待できる．

2.4 デジタルツイン

BIM 等から抽出した形状データや実空間から収集した時系列データを用いて，サイバー空間でシミュレーションやデータの可視化などを行うアプリケーションは「デジタルツイン」と呼ばれる．本章で述べたコンピュテーショナルデザインや BIM，i-Construction といった技術や，SDM の一部の事例も広義のデジタルツインであるといえる．

デジタルツインは元々製造業で先行してきた概念であり，シミュレーションによって現実と同様のプロダクトをサイバー空間上に作り，想定されるトラブルの検証などに利用されていた．2015 年に発表された第 5 期科学技術基本計画[*18]では，我が国が目指すべき未来社会の姿として提唱された Society5.0 において，実空間の状況が直接サイバー空間に再現され，サイバー空間での情報処理の結果が実世界の動きを制御する CPS (Cyber Physical System) として再定義され，各産業において取り組みが進んでいる．

本節では，特に建設産業におけるデジタルツインについて俯瞰し，その発展であるコモングラウンドの概念について触れる．

2.4.1 建設産業における取組み

IoT や XR (VR/AR:Augmented Reality/MR:Mixed Reality) 技術の興隆により，空間情報を IoT で取得（キャプチャ）し，BIM モデルにそれらの情報を紐づけることが容易になってきた．そのため AEC/FM 分野においても，よりリアルで臨場感の高いプレゼンテーションやステークホルダー間の情報共有やシミュレーション等を目的としたデジタルツインが注目を集めている．

Bentley Systems の定義[*19] では，「デジタルツインは，周囲の環境と関連付けた物理的な資産とシステムのデジタル表現で，パフォーマンスを理解しモデル化するため，エンジニアリング情報と統合されている．それらが示す現実世界の資産と同様，デジタルツインは常に変化している．〜中略〜 デジタルコンテキストとデジタルコンポーネントをデジタル年表と組み合わせることで，デジタルツインは実質的に 4D を通じて BIM と GIS を進化させている」とされており，IoT のみならず時間の概念も含めた高度な情報プラットフォームであると述べられているが，日本の建設業界においては，図 2.19 に示すように BIM データと IoT を用いた，センシングとシミュレーションを統合したようなモデルで語られることが多い [106]．例えば，IoT によってキャプチャしたデータと，数値解析やシミュレーションによる予測結果との差異分析をすることで，予測手法の進化を促したり，それらの分析結果をもとに高度な制御を行ったりするスマートビルのアプリケーションもその１つといえる．

AEC/FM 分野におけるデジタルツインの実装例としては，Bentley Systems が OpenCities Planner[*20]を用いてヘルシンキ等で行っているスマートシティの事例が先進的な事例といえるだろう．Dassault Systèmes の 3D エクスペリエンス・プラットフォーム[*21]も同様の機能を提供している．また，建設生産プロセスにおけるデータ共有環境である CDE (Common Data Environment) の周辺機能として，BIM から抽出した情報を見ながら，多様なステークホルダーが協調できるプロダクト・サービスもある[*22][*23]．

FM 分野でも，Archibus[*24]など多くの CAFM ツールが，BIM との連携機能を有するようになってきた．BIM などのデータ連携・変換のためのプラットフォームである Autodesk Forge[*25]を使って，BIM と IoT を連携させるような

[*18] https://www8.cao.go.jp/cstp/kihonkeikaku/index5.html
[*19] https://kyodonewsprwire.jp/release/201910242565
[*20] https://www.bentley.com/en/products/product-line/reality-modeling-software/opencities-planner
[*21] https://www.3ds.com/about-3ds/3dexperience-platform/
[*22] https://streambim.com/
[*23] https://www.autodesk.co.jp/products/bim-360-design/overview
[*24] https://www.archibus-jp.com/
[*25] https://forge.autodesk.com/

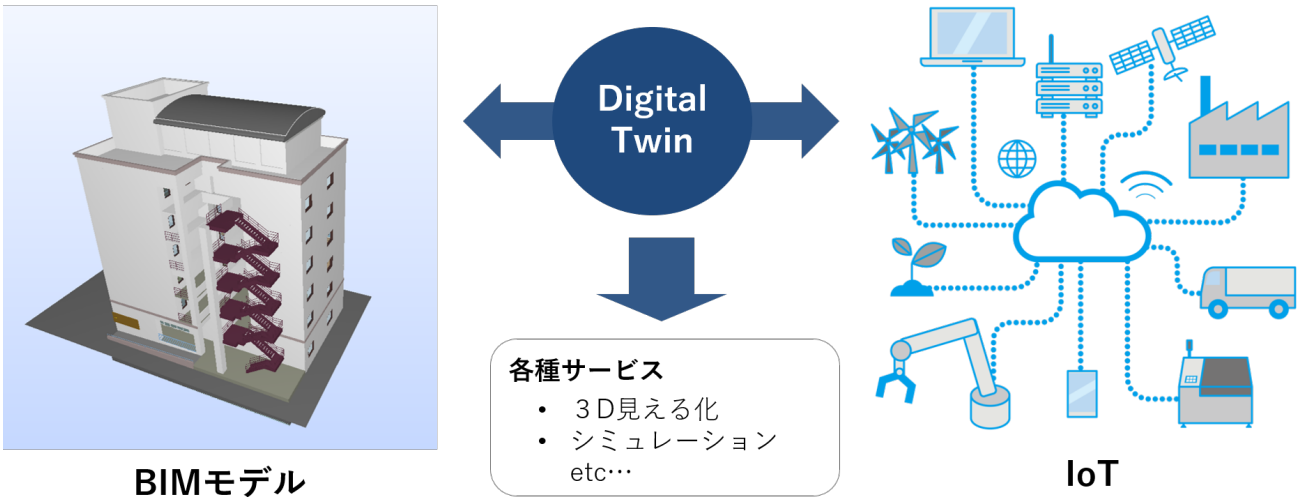


図 2.19: 建設業におけるデジタルツインのイメージ

事例も増えてきている．Microsoft は Azure Digital Twins (ADT) と呼ばれる PaaS (Platform as a Service) を提供している．ADT はスマートビルのプロジェクトでの対応経験から抽出されたデザインパターンを抽出してサービス化したものであるが，対応する形状の管理は直接行わない．現在（2020 年 5 月現在）プレビュー版ではあるが，今後の多くの機能拡張を検討しており，今後デジタルツインの汎用プラットフォームになるポテンシャルを秘めている．

2.4.2　デジタルツインの構築技術

本節では，特に BIM を用いてデジタルツインを実現しようとする研究や普及のための取り組み，構築手法の調査について述べる．

先行研究

BIM と IoT を用いたデジタルツインの構築技術については，Tang [69] らが詳細に調査している．アプリケーションを 4 つのカテゴリ (Constriction Operation and Monitoring, Health and Safety Management, Construction Logistic and Management, Facility Management) に分類し，特性について考察するとともに，それらのアプリケーション構築を実現するプロセスと手法についてまとめている．

例として，BIM から抽出した形状，属性をユニーク ID (GUID:Global Unique ID) を用いて IoT のポイントと結びつけるのに，関係データベースを用いたアーキテクチャが紹介されており，汎用性を高めるための技術として，独自のクエリ言語などによる BIM の形状・属性の抽出例なども紹介されている．将来的には，SOA (Service Oriented Architecture) と呼ばれるウェブサービスと親和性の強いアーキテクチャに代わっていくことが示唆されており，クラウドの利用も一般化してくると述べられている．しかしながら，AEC/FM 分野においては，先に述べたようにクラウドを前提としたシステム実践は，ビジネス領域で先行しており，インターネット由来のビッグデータ処理技術などの導入も進んでいる．また，ここでは AI や SDM のような音響メディアの連携に関わる検討もない．

他には，ゲームエンジンを中心に据えて，歴史的建造物のビジュアライズを行った研究 [16] や，BIM オーサリングツールの情報をリアルタイムに VR と連携するシステムの提案 [25]，ゲームエンジンを用いたプレゼンテーションシステムに応用する研究 [87]，ゲームエンジンを用いた没入環境による建物維持管理への適用 [26] などもあるが，特定のアプリケーションに限定されており汎用性は高いとは言えない．

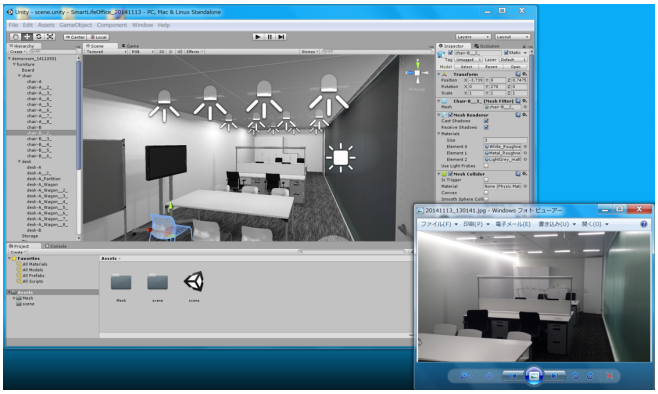


図 2.20: 竹中工務店技術研究所のスマートライフオフィスの Unity アセット

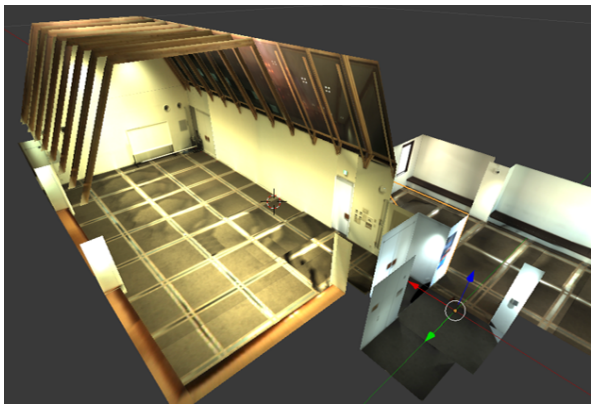


図 2.21: 東京大学 I-REF 棟の Unity アセット

スマートライフ・ハッカソン

我々は，ビルに関するデジタルツイン・アプリケーションの普及を目指してスマートライフ・ハッカソンと呼ばれるイベントを 2014 年 12 月から 4 回に渡って実施してきた．BIM や点群キャプチャデータから作成した 3D データと，各種設備制御のためのインタフェースを Unity のアセットとして提供し，参加者は会場でチームを組んで，自由な発想でアプリケーションを開発するイベントである．

第 1 回は竹中工務店技術研究所にて実施し，以降は東京大学の I-REF 棟で実施した．図 2.20，図 2.21 に提供した Unity の 3D データのアセットを示す．第 1 回は，パーソナルファンと呼ばれる個別空調や，照明制御，持ち込んだスマート家電のインタフェース[*26]のみを提供した．2 回目以降は FIAP Storage [95] に集約されたエネルギー使用量，リアルタイム位置情報[*27]などのセンシングデータの提供に加えて，空調・照明制御，立体音響制御が可能な SDM のシステムを制御対象として提供した．これらは MQTT を用いた疎結合のアーキテクチャになっており，筆者らのビル制御用プラットフォームの先行研究 [118] を拡張している．

参加チームは，実空間だけではなく，ゲーミフィケーションを交えた仮想空間とのインタラクションによる設備・機器制御を行うなど，実際の建設プロジェクトにおいては実現が難しい多様なコンテンツを開発した．また，これらの検証によって，SDM をはじめとする多様なシステム連携を実現する，MQTT を用いた疎結合なアーキテクチャの有効性が確認されたといえる．

海外事例

上記のスマートライフハッカソンと同様の取り組みとして，MIT メディアラボが開催している Reality Virtually Hackathon[*28]がある．VR や MR の技術を用いた総合的なハッカソンといえるが，2019 年の受賞作の一つである SoundSpace [57][*29]は，配布されたと思われる MIT メディアラボの BIM データと Unity，BIM のデータ変換のために Autodesk Forge を用いて，インタラクティブな音響シミュレーションシステムを開発している．これらを開発したのは設計事務所のエンジニアであり，海外ではゲームエンジンによる VR，MR と BIM の境界が低い状況にあることが伺える．

レンセラー工科大学の建築スクールでは，建築（Architecture）と XR などを連携させた教育のために，CRAIVE (Col-

[*26] https://github.com/KAIT-HEMS/node-picogw
[*27] https://www.ubisense.jp/
[*28] https://realityvirtuallyhack.com/
[*29] https://devpost.com/software/sound-space

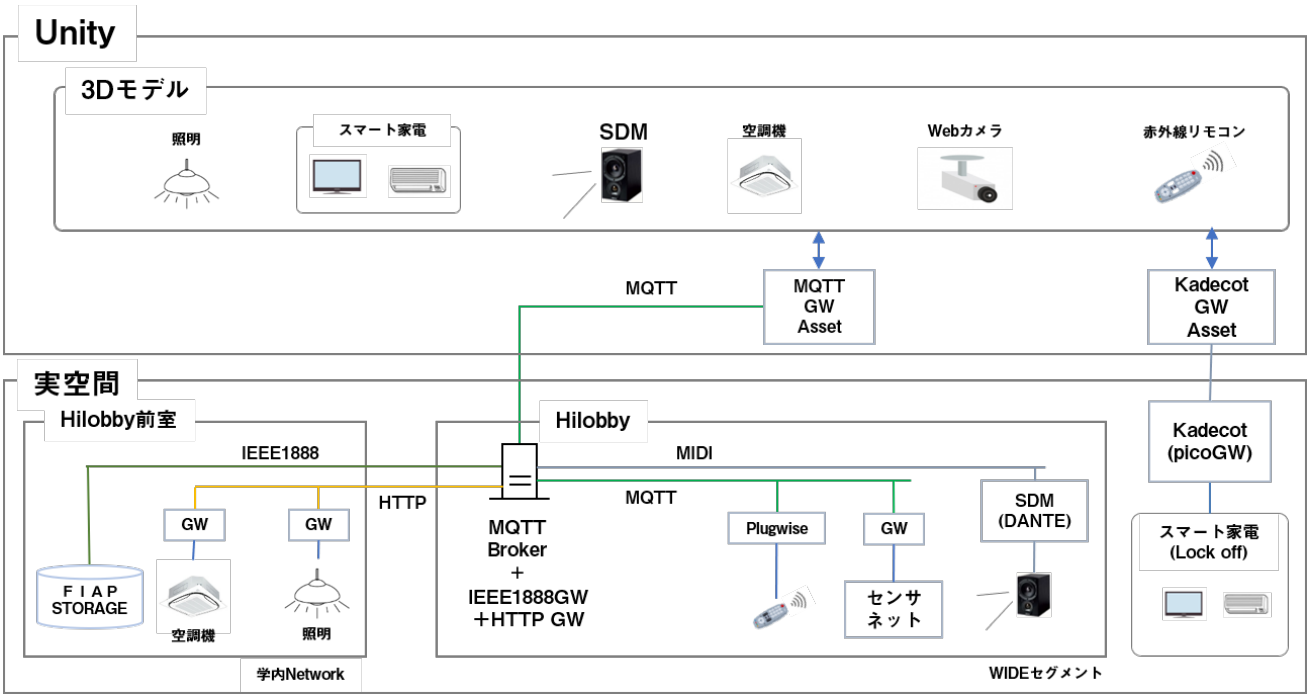


図 2.22: スマートライフハッカソンのシステム構成（第 2 回以降）

laborative Research Augmented Immersive Virtual Environment) Lab[*30]を構築し，それらのコンテンツ・クリエータのためにプラットフォームを解放している．$10m \times 12m$ の空間に 128 台のスピーカと 8 台のプロジェクタを配置し，インタラクティブで没入感のあるコンテンツ開発のためのアセットを提供している．このようなコンテンツ・クリエーションと建築空間は切り離すことができなくなってきており，それらの実現・効率化のための研究が必要といえる．

実態調査

ビジネス領域におけるデジタルツイン構築の実態調査のために 6 社（国内 3 社，国外 3 社）にヒアリング調査を行った [116]．ヒアリング項目としては，クラウド活用の実態やそのベストプラクティス，BIM と連携など多岐にわたる．それらのヒアリングの内容と，先行研究 [118] での取り組み・ノウハウも併せて，BIM から形状，属性データ（メタデータ）を抽出し，IoT と組み合わせてデジタルツインのアプリケーションを構築するための標準的なデータ処理のパイプライン／アーキテクチャを整理した (図 2.23)．ソフトウェア開発で一般的には MVC (Model, View, Controller) に従って整理をしている．

1. BIM からデータ交換フォーマットである IFC (Industry Foundation Classes) に変換
2. Parser（解析器）を介して，IFC を形状，空間構成グラフ，属性に分離・変換
3. 形状についてはアプリケーションに応じて，最適なファイルフォーマットに変換
4. 空間グラフ，属性をそれぞれ最適なデータベースに格納
5. デジタルツイン・コントローラが空間グラフや属性データ，空間グラフに紐づいた IoT/BAS のポイント情報を結びつけて，アプリケーションに最適なデータモデルに変換して保持．必要な制御ロジックも内包
6. アプリケーション／ビューワで可視化

重要なのは，BIM から抽出した汎用的なデータモデル・知識ベースの情報を，それぞれのアプリケーションが最適

[*30] https://www.clatcraive.net/

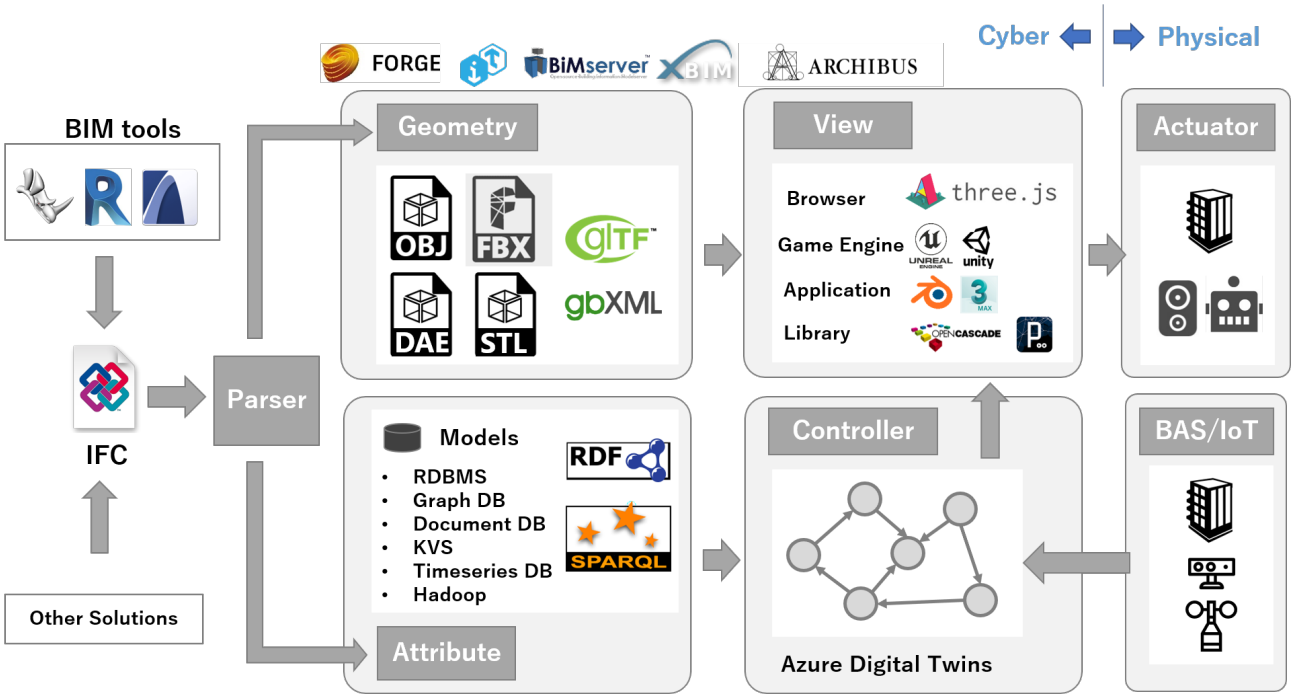


図 2.23: デジタルツインのパイプライン

なデータモデルで再解釈し，利活用することである．上記の具体化したプロセスについては，第 4 章で再度検討する．

2.4.3　コモングラウンド

AEC/FM 分野以外にも目を向けると，NTT グループが実世界を反映した高精度デジタル情報の掛け合わせによる革新的サービスとして「デジタルツインコンピューティング（DTC）構想」を発表している[*31]．DTC とは，デジタルツインを大きく発展させ，実世界を表す多くのデジタルツインに対して交換・融合・複製・合成等の演算（デジタルツイン演算）を行うことにより，モノ・ヒトのインタラクションをサイバー空間上で自由自在に再現・試行可能とする新たな計算パラダイムと説明されている．デジタル化される対象をヒトにも拡張していることが特徴であり，それらを実現するためデータ・プラットフォームを NTT グループで開発中である．

また，Wired の特集[*32]では，デジタルツインの先にある世界としてミラーワールドという概念を紹介している．見た目だけではなく，場所やモノのコンテキストまでを把握し提示するもので，「現実世界（リアルワールド）にあるすべての場所やモノ——すべての道路，街灯，建物，部屋——の実物大のデジタルツインがミラーワールドに存在するようになる．いまはまだ，その片鱗を AR ヘッドセットを通して見ているに過ぎない．ひとつまたひとつと，ヴァーチャルな断片が縫い合わさり，ついには現実世界のパラレルワールド版として，開かれた永続的な場所が形づくられるのだ」とある．些か誇張したような表現で語られているが，BIM や 3D センサ，カメラによる物体認識技術などが一般化している昨今，技術的な素地はすでに出来上がっているといえる．スイスのスタートアップである Nomoko[*33]は，実空間情報のキャプチャと，キャプチャされたオブジェクトに対するデジタル記述によってミラーワールド構築を始めている．

上記の DTC やミラーワールドで必要性が示唆されているのは，実空間やシステムの数だけデジタルツインは存在し，それらを解釈し包み込むプラットフォームの存在である．豊田は，これらを人間社会と AI がともに依拠できるサ

[*31] https://www.ntt.co.jp/news2019/1906/190610a.html
[*32] https://wired.jp/special/2019/mirrorworld-next-big-platform
[*33] https://nomoko.world/

イバー空間であり，知識ベースである「共通基盤（コモングラウンド）」という概念 [100] を引用しつつ，スマートシティのプラットフォームとして提案している [123].

コモングラウンドを言い換えると，ヒト・モノ・システム・AI が共通理解可能なプラットフォームといえる．筆者の整理では，図 2.24 に示すように，BACS データや IoT を含む時系列データを扱うプラットフォーム，BIM や点群等から取得した形状を扱うプラットフォーム，音声や動画を含む SDM のためのプラットフォーム，それらのデジタル記述である知識ベース (Knowledge-base)，それらを解釈しデジタルツインを構成するデジタルツイン層，更には AI システムをはじめとする多様なアプリケーションを提供するサードパーティに対する API から構成される．

ここで述べた知識ベースとは，これらは Society5.0 を支えるとされている，産業別に分散しているビッグデータ連携のための分野間データ連携基盤 (日本版 NIEM：National Information Exchange Model) または，その中核機能とされているデータ連携共通語彙基盤 （IMI：Infrastructure for Multi-layer Interoperability) の具体的なインスタンスと呼べるだろう．分野間データ連携基盤では，全ての産業を網羅する概念図が描かれているが，まずは特定のドメインにおいて研究を進めていくことが必要といえる．

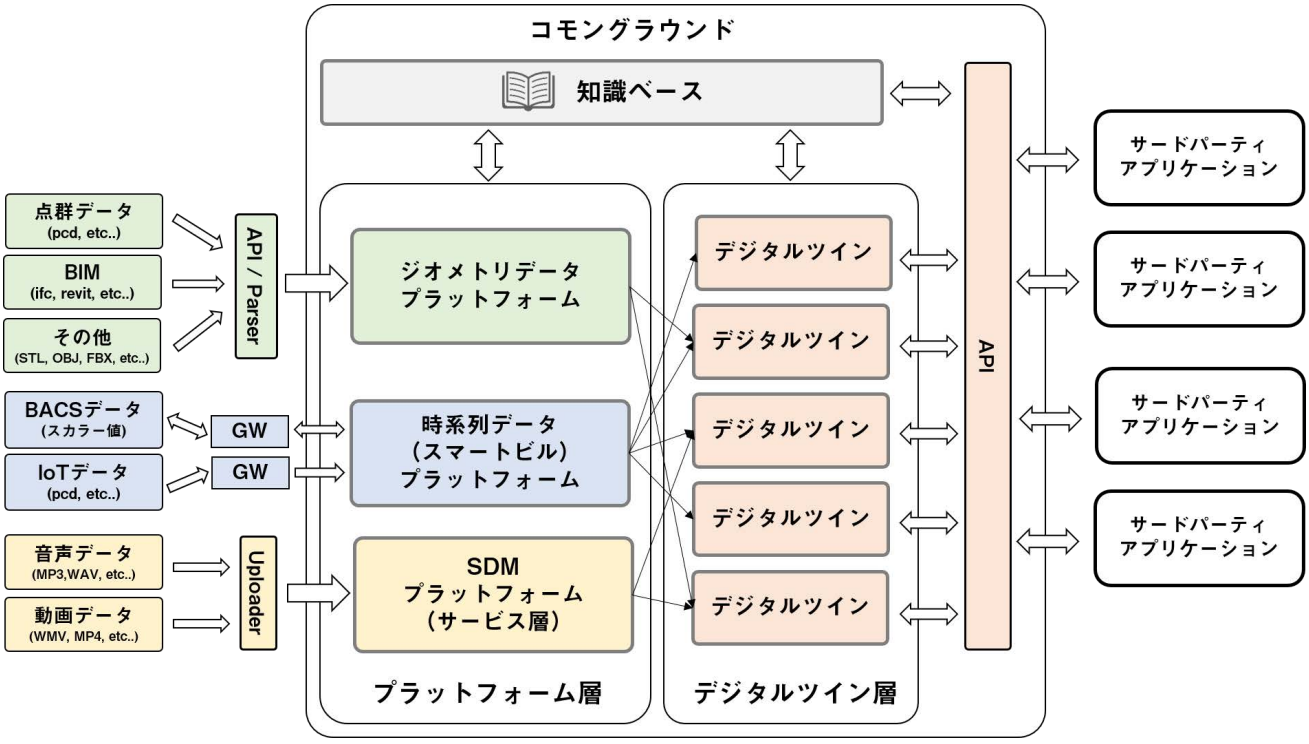


図 2.24: コモングラウンドの構成イメージ

2.5 研究対象領域

本章では建設産業によるデジタル化について述べ，BIM や BACS などのデジタルデータの活用が他領域で進んでいないこと，ゲームやエンタテインメント分野での活用が望まれていることを述べた．エンタテインメント分野における取り組みとして SDM を取り上げ，融合領域における BIM の活用といった研究課題について触れた．また，スマートビルにおいて，AI・IoT などを専門業者とのコラボレーションのために，実践的なセマンティクスやオントロジ，データ・プラットフォームの必要性について述べた．更に，デジタルツインやコモングラウンドについて説明し，BIM を用いたデジタルツイン・アプリケーションの構築手法について調査結果をもとにまとめた．

それぞれの節において，個別の研究課題についても触れたが，全体を俯瞰すると前節で述べたコモングラウンドの全体像を引用し，研究領域としては図 2.25 のように表現できる．本研究で扱うそれぞれのプラットフォームがコモングラウンドを構成する 1 要素となっている．

本研究では，スマートビルに注目し，Software Defined BACS を実現する要素技術とデータ・プラットフォーム，インタラクティブな操作を実現する立体視聴を実現する SDM のアプリケーションを中心に述べるが，これらが発展していくことで，デジタルツインやその先のコモングラウンドが実現すると考える．

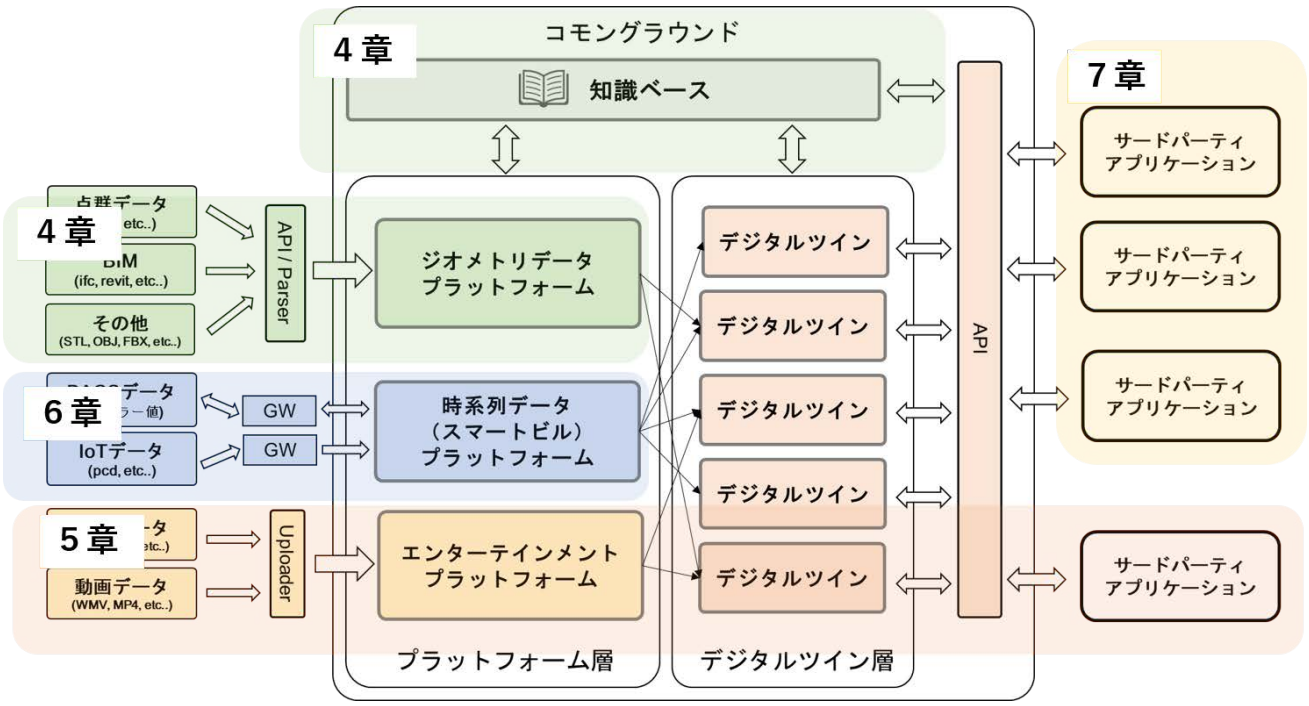


図 2.25: 研究対象領域

第 3 章

Software Defined BACS

本章では Software Defined BACS について深耕する．

Software Defined とは，サービスの再利用や移植性の向上すること，またソフトウェアによる定義と制御が可能ということであり，近年では，例えばソフトウェアによるネットワーク機器の集中制御によって，ネットワーク構成や設定などを柔軟かつ動的に変更する Software Defined Network や，物理的なストレージデバイスを抽象化し，それらの構成・制御・管理をソフトウェアによって行うことで，高い効率性と高度な自動化を実現する Software Defined Storage などに使われている．

Software Defined BACS は，可用性の高いローカルシステムである BACS の機能を維持しつつも，その機能的限界を超えるため，ハードウェアの抽象化とデータの管理機能を持ったデータ・プラットフォームを有し，API 連携によって再利用や移植性の高いサービス構築が可能であり，それらのサービス定義（構造や手順）や制御パラメータ等の更新がソフトウェアによって可能なシステムと定義する．Software Defined BACS は，以下に述べるプロジェクト適用上の課題を解決するため，BACS の構築ベンダーと協調可能な形で，アプリケーションの移植性と再利用性を高める実践的なアーキテクチャを志向する．即ち，責任範囲が明確で可用性が高い既往の BACS を保持し拡張する実践的なシステムといえる．

3.1 システム構成

Software Defined BACS と同様の概念として SDB [23] がある．SDB は本研究と同様の課題意識を持ち，サイロ化を打破し，アプリケーションの移植性と再利用性の向上を目的としたスマートビルのための情報インフラを目指している．Mazzara らは，スマートビルと SDB の関係性を図 3.1 のように説明している [55].

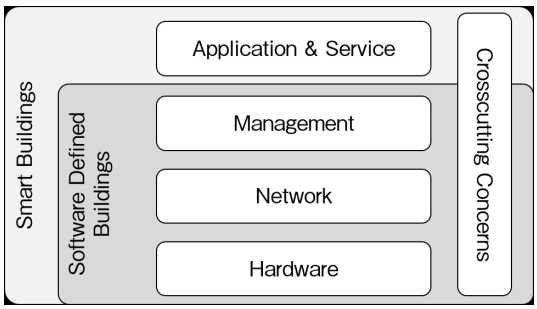


図 3.1: Software Defined Building とスマートビルの関係

SDB の構成要素をハードウェア，ネットワーク，マネジメントの 3 つとし，スマートビルは適用されるアプリケーションとサービスを含むものとしている．SDB では，ハードウェアの抽象化を行い，多様な通信プロトコルに対応し，

各種のリソースやデータを管理するとしており，スマートビルでは省エネ，快適性向上，安全・安心などのアプリケーションを提供する．また，Abid らは SDB におけるリソースの仮想化を更に推し進めた Virtualized Software Defined Buildings (VSDB) を提案している [3]．SDB はビル単体の制御を対象にしていたが，VSDB ではリソースを仮想化することで，スマートシティなどを対象とした面的なアプリケーションの実現を容易する．ユースケースとしては，例えば 2.3.7 節で述べた DR (Demand Response) を複数棟で行う VPP (Virtual Power Plant) が該当する．

SDB におけるビル内のデバイスやリソースの抽象化・仮想化はコンピュータサイエンスにとっては一般的であり，理想的ともいえるが，実際のプロジェクト適用においては実現が難しい．理由としては，主として設備設計・施工において重要な責任範囲がある．建築プロジェクトにおいては空調，照明などのサブシステムはそれぞれ別の施工会社によって構成され，それぞれの責任範囲の中で完結させる必要がある．計装ベンダーなどが，それぞれのサブシステムと B-BC (BACnet Building Controller) と呼ばれるコントローラを介して，B-OWS (BACnet Operator Workstation) と呼ばれる中央監視機器と通信・連携させることで BACS を構築する．大型プロジェクトにおいては，BACnet によるインテグレーションが一般化しており，IoT などの機器も B-OWS との通信のために，共通のデータバスである BACnet に変換する構成が合理的といえる．ただし，ポイント数が多い場合は，2.3.8 節で述べたように，イニシャルコスト低減のために一度クラウド上のデータ・プラットフォームに収集してから，必要なポイントのみを BACnet で連携するといった構成がとられる．いずれにせよ，責任範囲と製品保証の問題から計装ベンダーは，設計図書にて合意されたポイントリストや機能以外の提供は行うことができないため，遠隔からプログラマブルに機能更新を行うような構成は困難といえる．

Software Defined BACS は，こうしたプロジェクト適用上の課題を解決するため，BACS の構築ベンダーと協調が可能な形で，アプリケーションの移植性と再利用性を高める実践的なアーキテクチャを志向する．図 3.2 に既往の BACS，SDB，Software Defined BACS のシステム構成の比較を示す．なお，SDB の詳細は 6.2.1 節で詳細を述べる．

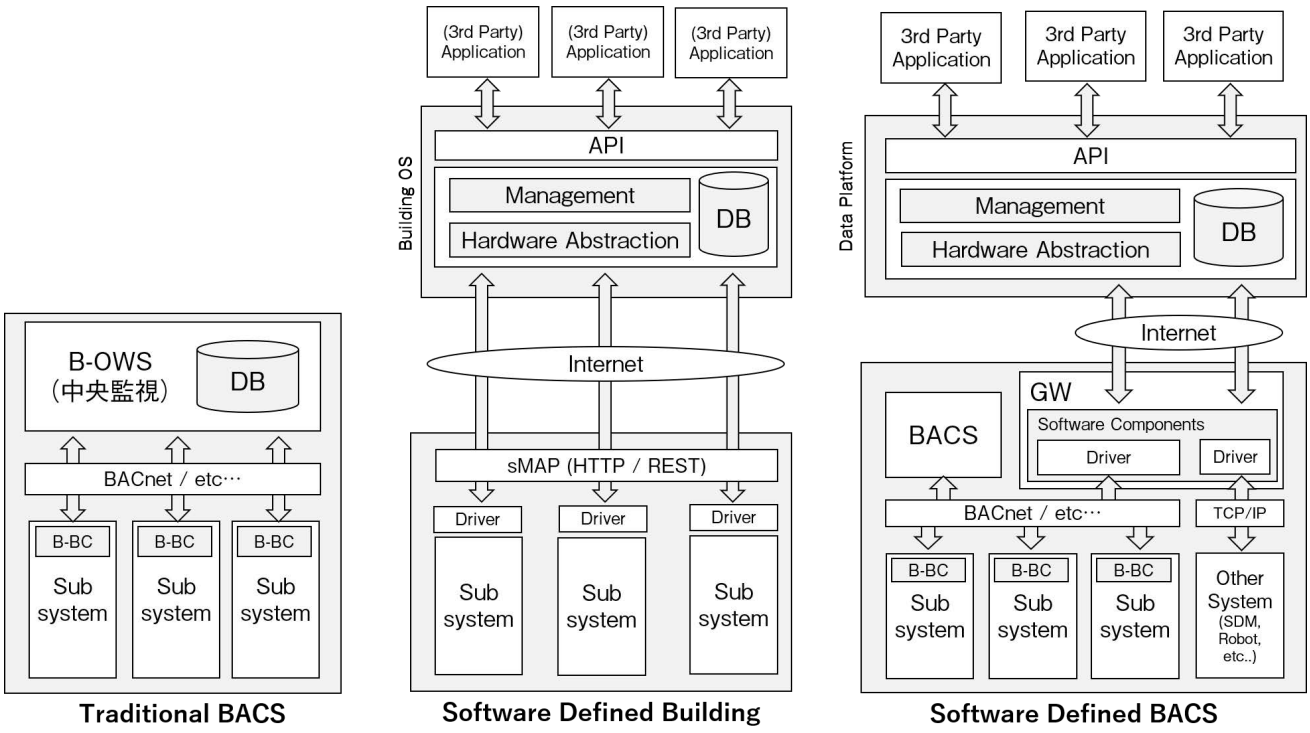


図 3.2: Software Defined BACS のシステム構成比較

SDB は既往の B-OWS の代替として，ビル用の OS (Building OS) の構築を目指しているといえる．Building OS

と各種のサブシステムを結ぶデータバスは sMAP[21] で統一される．一方，Software Defined BACS はローカルシステムである BACS の構成を保持しつつ，それらと BACnet で連携を行う GW (ゲートウェイ) を介してクラウド上のデータ・プラットフォームと連携する．データ・プラットフォームは，BACnet 等で発出・収集された現地設備，デバイスのデータを保存し，ハードウェア抽象化とメタデータ定義を行うとともに，サードパーティに対してデータを仲介する API を提供する機能を持つ．Building OS との違いは，提供する管理機能である．例えば，SDB は制御プロセスや建物の進化管理，更には拡張性の高い UX を提供するとしている．

Software Defined BACS では，BACS としての基本機能はローカル設備に委譲し，GW 以降（ネットワーク，クラウド）をクラウドベンダーの責任範囲とすることで，クラウドからの遠隔機能更新を可能にする．なお，図 3.2 では，ローカル側にゲートウェイを配置しているが，VPN で接続することで，B-OWS となる GW をクラウドに配置する構成も考えられる．

また，Software Defined BACS は，AI やビッグデータ処理など，ローカルでは構築が難しいアプリケーションのユースケースを対象とし，それらのサービスを，データ・プラットフォームを介して自由に組わせることで，BACS の機能強化，または機能の再定義が可能になる．サービスの機能や構造，手順などを，DSL (Domain Specific Language) や RDF などで記述・定義することで，より汎用性と柔軟性が高めることができるだろう．加えて，IoT などの新たなデバイスが追加された際も，クラウドから GW の機能更新を行うことで，サードパーティのサービス連携が可能になる．これら柔軟なアーキテクチャにより，ロボットや音響設備など，ビル設備以外との連携といったユースケースの拡張も容易になるといえる．

以降では，このような Software Defined BACS 適用時のインフラ要件，構成要素について述べる．

3.2 適用時の要件

Software Defined BACS は，既往の BACS にレトロフィット可能な構成であるが，適用に当たって前提となる要件がある．BACS が単独動作可能であること，動作モードの指定が可能であること，統合ネットワークが構築済みであること，サイバーセキュリティに考慮することである．これらは，ビル設備が持つべき高可用性を確保し，クラウドからの遠隔制御を実現するために，柔軟性・汎用性を得る上で重要と考える．

3.2.1 単独動作が可能

クラウドシステムは，必然的に通信用の外部回線に依存する構成となる．インターネット用回線の可用性は高水準になってきており，見える化，BEMS (Building Energy Management System) などの一部機能をクラウドから提供するサービスは既にあるが，BACS の機能をすべてクラウドに配置する構成はほとんどない．ビル設備は可用性が重要要件であり，B-OWS サーバも一般的には冗長構成をとっているためで，これらと同等の可用性を得ようとすると，外部回線やゲートウェイ・ルータの冗長化も必要になり，構築・運用コストの観点から現実的ではなくなる．ただし，これらは大型ビルにおいてであり，小型の B-OWS がないビルであれば適用の可能性はある．しかしながら，どのような場合においても，回線切断時において自律的に動作を続けられるよう，BACS の機能設計を行う必要がある．

例えば，大型テナントビルにおいて無線の人感センサ，温湿度センサなどの IoT を使って照明・空調システムを制御する場合を考えてみる．IoT による大量のポイントの管理のためにクラウドにモニタリングシステムを構築し，図 3.2 のようにローカルシステム用へのインタフェースとして，BACnet で通信するための B-OWS の機能を持った GW を配置する．空調・照明などのサブシステムは，クラウドからの遠隔制御指令を受けて環境制御を行うが，GW からの制御のみに依存すると，回線切断時に自律的な制御ができなくなってしまう．そのためローカル側では，クラウドとの接続状況を常に確認し，切断時は自身のサブシステムが持つセンサーのみでの動作に切り替えるといった，設計上の工夫が求められる．このようなクラウドシステムの構築・試験は，建設プロジェクトの最後期であることが多く，ローカル

システムの検査対応のためにも機能的に切り離しが可能な構成としておくことが望ましい．

3.2.2 動作モードの指定

BACS の運用時，外部からの制御を抑制したい場合がある．例えば，AI などを用いた自動制御を適用する際，思いもよらない制御を排除した運用が必要な場合や，設備点検の際などがあたる．他にも，重要設備の遠隔制御などは，管理員が必ず確認してから実施するというフローが義務付けられる可能性がある．こうした場合に備え，ローカルの BACS に制御モードのポイントを導入することが必要である．それによって，例えば強化学習による制御の際に，ノイズとなる不要な学習を防ぐこともできる．図 3.3 に B-OWS である中央監視システムにおいて，制御（指令）モードをポイントとして用意した例を示す．指令モードが「中央監視モード」となっており，ローカルでの制御を優先する状態になっていることが分かる．

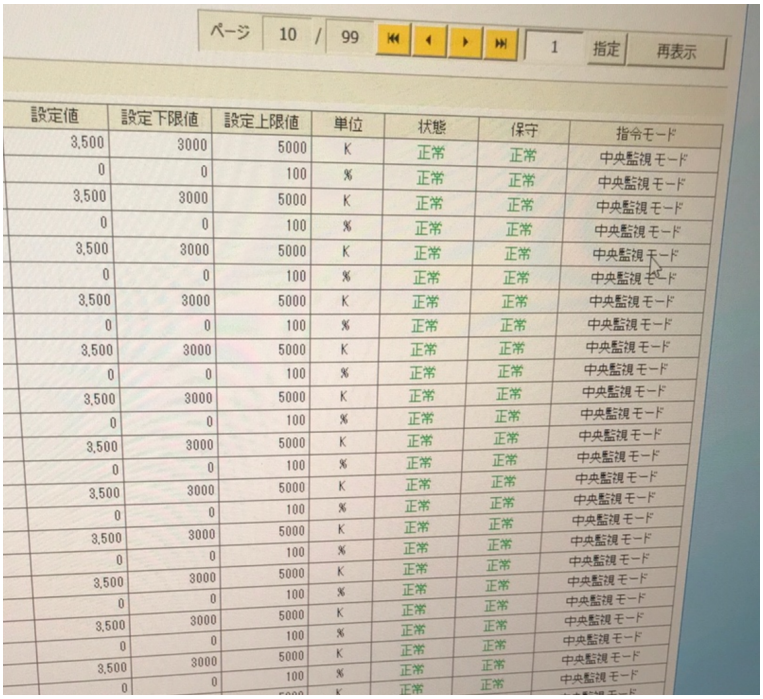


図 3.3: BACS での制御（指令）モード

3.2.3 統合ネットワークの構築

BACS を構成するネットワークは IP 化が進んでいるものの，ネットワークの設計については専門的な知見をもとに行われることが少なく，それによって柔軟な通信や制御ができないことが，しばしば問題になっている．そのため，図 3.4 のような統合ネットワークと呼ばれる，BACS を構成するサブシステムに共通のネットワークを構築する事例が増えている．これは BACS が閉域網での運用を前提としており，ルーティングやフィルタリングなどの設計が不要であったこと，BACS に関わるエンジニアにネットワークの専門家が少ないことに起因している．

統合ネットワークを導入しない場合，責任範囲を明確化するため，照明・空調・計量などのサブシステムは個別に IP ネットワークを構築し，B-OWS との通信は，サブシステム毎に構築した B-BC と，計装ベンダーが構築した幹線ネットワークを介して接続する構成となる．その場合，B-BC となるデバイス・サーバにはネットワーク・インターフェースを 2 つ有していることが多く，そこでルーティングするために，サブシステム内のデバイス，BACS は直接通信でき

ない構成になっている．これらは責任範囲の明確化という意味においては都合がよいといえるが，BACS の機能拡張や，クラウドからの遠隔制御，更にはファームウェアのアップデートやセキュリティパッチの適用が難しくなるなど，スマートビルの運用においては課題が多い．他にも BACnet では，ブロードキャスト通信が頻発するため，大型ビルでセグメントを切らない幹線ネットワークとなっている場合，B-BC の処理能力の問題で適切に動作しないといった事象も発生する．

統合ネットワークでは，ネットワークスイッチの冗長化や，サブシステム毎の通信内容に応じたルーティングやサブネットなども考慮した設計を行う．BACS の幹線ネットワークは照明や空調工事を請け負っているサブコンや，計装ベンダーなどが構築を請け負うことが多いが，なお，統合ネットワークは専門的なエンジニアリングが必要であるため，ネットワーク専門業者が入って構築されることが多い．そのため，サブシステム毎にネットワークを構築する場合に比べ，コスト増になる可能性もあるが，後述するサイバーセキュリティ対策においても相性が良い．なお，BACS の専門業者によっては，製品の性能保証の観点からネットワーク構成について規制がある場合があるが，３段構成（コアスイッチ，ディストリビューションスイッチ，アクセススイッチ）と呼ばれる一般的なネットワーク構成であれば問題になることは少ない．なお，近年では幹線ネットワークにリング型の構成を採用する事例もある．

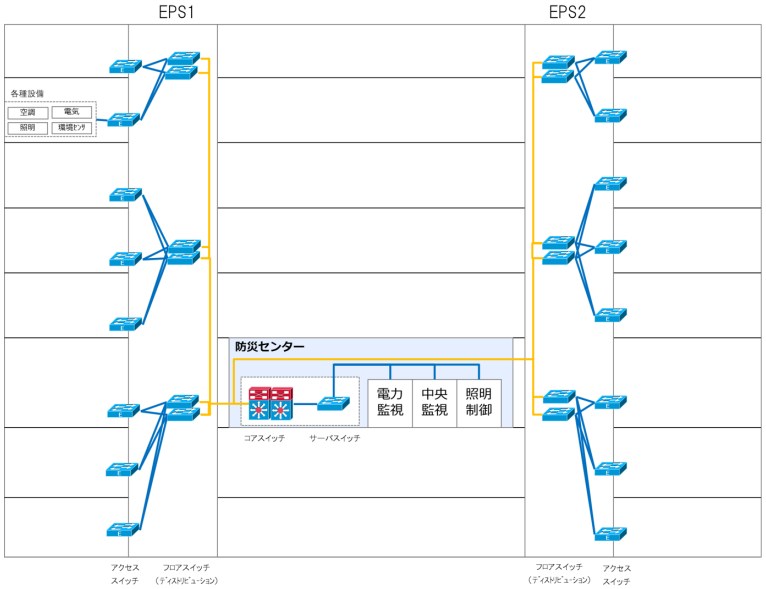


図 3.4: 統合ネットワーク例

3.2.4 サイバーセキュリティ

BACS では閉域網運用を前提としていたため，サイバーセキュリティに関する検討はほとんどなされていなかった．それらが機能のアップデートやセキュリティパッチを拒絶する業界の慣習に繋がっている．しかしながら，IoT やクラウドの普及によって，ビルにおいてもサイバーセキュリティ脅威が顕在化してきた．

そのため我々は，JDCC (日本データセンター協会) のファシリティインフラ WG において，データセンターを題材に，建物設備のあるべきセキュリティ要件について検討を行ってきた．その成果物である「建物設備システムリファレンスガイド」では，現状のできることを中心に管理策を提言し，仮想のユースケースを用いてデータセンターのサイバーセキュリティ対策のあるべき姿についてまとめた [82]．また，それらを発展させて，産業サイバーセキュリティ研究会で「ビルシステムにおけるサイバー・フィジカル・セキュリティ対策ガイドライン」の策定に協力している[*1]．こ

[*1] https://www.meti.go.jp/press/2019/06/20190617005/20190617005.html

のガイドは，ビル内の場所とライフサイクルに注目し，リスクに対するセキュリティポリシーの案を提示しており，建物設備の実務者がより取り組みやすい構成となっている．

Software Defined BACS はクラウドや外部ネットワークからのデータ連携を前提としているため，サイバーセキュリティに関する考慮は必須といえる．ただし，既往の BACS に対してエンドポイントセキュリティの適用は，製品保証の観点から難しいことが多く，現段階においては物理的にネットワークやシステムを構成するサーバ等にアクセスできないような措置を行うこと，ファイアウォールの設置やルーティングの設定などで論理的にアクセスを抑制することが現実的な対応となっている．

3.3 構成要素

Software Defined BACS の構築には，上記の要件を満たしたうえで，遠隔更新可能なゲートウェイ，データプラットフォームの配置が必要である．以下ではそれぞれについて述べる．

3.3.1 遠隔更新可能なゲートウェイ

2.3.5 節でも述べたように，BACS では多様なプロトコルが存在し，IoT の普及によって対応すべき通信仕様も増えてきている．これらの多様なプロトコルに対応するためのフレームワークとしては，Niagara Framework や[*2]などがある．プラグイン可能な通信モジュールをインストールすることで，多様なプロトコルに対応することが可能になっており，主として B-BC として採用されてきた．他にも，遠隔で更新可能なゲートウェイとしては，Java ベースの OSGi (Open Services Gateway initiative) が古くから提案されており，組み込み系のシステムにも多く利用されてきた．先行研究 [118] においても，OSGi による BACnet ゲートウェイを用いている．近年では，エッジデバイスの高スペック化に加え，Docker[*3]などのコンテナ・仮想化技術が発展してきたことにより，Azure IoT Edge[*4]のような，クラウドに登録されたコンテナをエッジ側にデプロイすることで，遠隔での機能更新が容易に行える仕組みも提供されている．Azure IoT Edge では，エッジデバイス上で動作するランタイムによって，コンテナ化された各種の機能モジュールによる，モジュール，クラウド間の容易な通信を実現している．図 3.5 に Azure IoT Edge によるゲートウェイの構成例を示す．管理者がクラウド上の Container Registory に登録した機能モジュールである Docker Image をローカルシステム側にデプロイし，それらがエッジデバイス上のランタイム (Edge Agent) を介して，クラウドの Data Ingest (Azure IoT Hub) と連携するイメージとなっている．

こうした遠隔更新可能なゲートウェイを用いることで，機能更新のために現地に赴いて作業を行う必要がなくなり，より柔軟なシステム運用と管理が可能になる．更には，クラウドでの機能モジュールの登録時に，必要なサイバーセキュリティの要件の確認も可能になる．例えば ID，パスワードを適切に変更しているかといったことであり，これらが実現することでセキュアな Software Defined BACS の運用が可能と考える．

3.3.2 データ・プラットフォーム

ゲートウェイから発出されたデータを収集，保存するデータ・プラットフォームでは，ハードウェア抽象化と，それらのデータを保存し永続化する機能を持つ．ここでは 2.3.8 節でも述べたように，抽象化のためのセマンティクスの定義と，安価に保存可能のためのアーキテクチャの研究も必要である．また，BACS 以外のアプリケーションにおけるデータ利活用のための知識ベース，サードパーティのための容易な API の構築も必要といえる．それら知識ベースの情報や，サービス定義と実行のための管理機能を実装することで，ソフトウェアによるプログラマブルな機能・手順の

[*2] https://www.ccontrols.com/tech/niagara.htm
[*3] https://www.docker.com/
[*4] https://azure.microsoft.com/ja-jp/services/iot-edge/

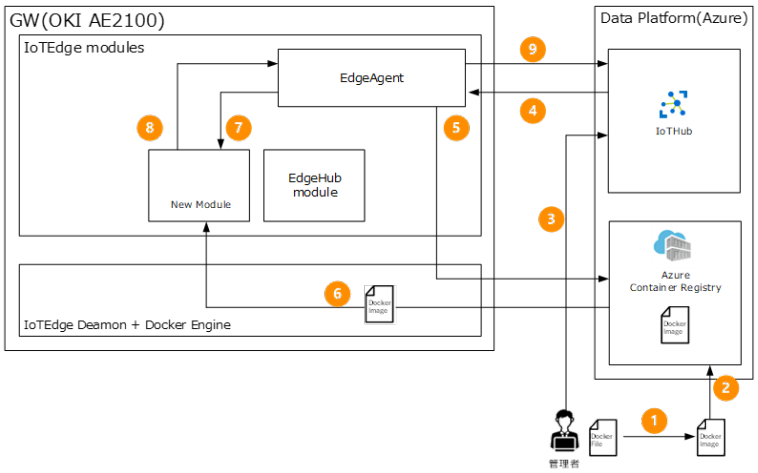


図 3.5: Docker / PaaS によるゲートウェイ構成

定義も可能になる．更には，知識ベースを介して BIM から抽出したジオメトリ，音声・動画といったコンテンツと連携することで，BACS 以外の多様なアプリケーション構築が可能なコモングラウンドの実現にも近づくと考える．

3.4 まとめ

本章では，Software Defined BACS の定義を行い，先行研究である SDB との比較によって，そのシステム構成や機能の説明を行った．また，それらを実現するため，ローカルシステムで考慮すべき要件と，必要な構成要素を述べた．Software Defined BACS は，既往の BACS にレトロフィット可能な構成を特徴としつつも，それらの機能的な限界を超えるために，IoT，AI，ビッグデータ処理技術などの適用が可能なアーキテクチャを有する．我々は，幾つかの新築及び改修プロジェクトにおいて，Software Defined BACS を部分的に実現してきた．以降の章では，それらの実践の詳細を説明するとともに，構築する上での研究課題解決について述べる．

第 4 章

スマートビルのセマンティクス

本章では，スマートビルで扱われるデータのセマンティクスについて考察する．セマンティクスとは，システムに対してデータの持つ意味を正確に解釈させ，データへの関連付けや情報収集などの処理を自動的に行わせるための技術を示し，オントロジも含んだ意味で用いる．そのための必要な技術について説明するとともに，BIM と BACS のポイントリストを使ったセマンティクスの抽出とデータモデル生成手法の提案と評価を行う．

これらの技術提案によって，Software Defined BACS のためのハードウェア抽象化の自動化と，多様なシステム連携のための知識ベースの構築が可能になった．具体的には，BIM のデータ構造とジオメトリを抽出するツールを開発するとともに，BACS のポイントデータとの突合の手法を提案した．データ・プラットフォームやサードパーティ・アプリケーションなどは，提案手法によって生成したデータモデルを読込み，サービス用に再解釈して活用することで，BIM や BACS から抽出した情報を効率的に利用することができる．

4.1 はじめに

2.3.8 節で述べたように，高度な機能を有するスマートビルの構築においては，AI や IoT といった専門業者とのコラボレーションが必須といえる．それらの参入を促し，構築後のアプリケーションの移植性と再利用性を高めるとともに，Software Defined BACS を実現するためには，実践的なセマンティクスの特定と，それらを用いたデータモデルの生成方法の研究が必要である．また，スマートビルで扱われるポイントは，大量かつ専門性・個別性が大きいので，モデリングが困難という課題もある．対象のビルに精通しているエンジニアやファシリティマネージャー以外では難しく，自動化の手段が必要といえる．

そのため我々は建設ドメインにおいて近年提案されているオントロジを利用し，BIM (Building Information Modeling) と BACS のポイントリストを用いたデータモデルの生成手法について提案する．

BIM とはコンピュータ上に作成した主に 3 次元の形状情報に加え，室等の名称・面積，材料・部材の仕様・性能，仕上げ等，建築物の属性情報を併せ持つ建築物情報モデルを構築するものである．BIM の導入によって，建築物に関するあらゆる情報が一元化され，情報の重複入力による手間や不整合が削減されると共に可視性が高まり，とりわけ建築生産における品質の向上，工期の短縮，費用の低減，さらには適正な維持管理の実現などに対する効果が期待されている [94]．BIM ライブラリーコンソーシアム[*1]による BIM オブジェクト標準の制定も進んでおり，BIM モデルに入力すべき設備機器の標準的なプロパティセットの定義も進んでいるが，現状ではツール間でのデータ受け渡しのために整備されているのがほとんどで，BACS では活用されていない．しかしながら，BIM は 3 次元の形状データに加え，空間構成や部材，設備設備機器の属性データなどを含めることができる総合的なデータベースである．従って，BIM の活用により，実用的な BACS のデータモデル生成も可能と考える．また，2.2.3 節で述べた，SDM をはじめとする建

[*1] https://www.bmmc.or.jp/blc/

設以外のシステムへのデータ応用も容易になるだろう．

以降では，4.2 節にてデータモデル生成のための要素技術として，BACS におけるセマンティックウェブ技術の適用動向およびセマンティックウェブの中心的な技術であるオントロジの定義について述べ，既往研究で提案されているオントロジを紹介する．続いて，4.3 節にてスマートビルにおける要件分析を元に必要なセマンティクスを特定し，4.4 節にて BIM とポイントリストを用いたデータモデルの生成手法の提案と評価を行う．更に 4.5 節にて提案手法により抽出した形状とデータモデルを用いて，SDM アプリケーションの再構成を試行し，その汎用性について確認する．

4.2　BACS へのセマンティクウェブの技術適用

ASHARE の専門委員会 (Standing Standard Project Committee 135) では，ASHRAE Standard 223P: ”Designation and Classification of Semantic Tags for Building Data”と呼ばれる BACS におけるセマンティクス活用のための提案を行っている[*2]．ここでは，BACnet の Semantic Tags に対して，Project Heystack のタグ[*3]や Brick のデータモデル[*4]を使うことが述べられている．具体的には，図 4.1 に示すように，BACS 用のセマンティクスやデータモデルを定義し，それらを格納する知識ベース (Knowledge Base) を中心に，BACS のためのアプリケーションを構成するものである[*5]．

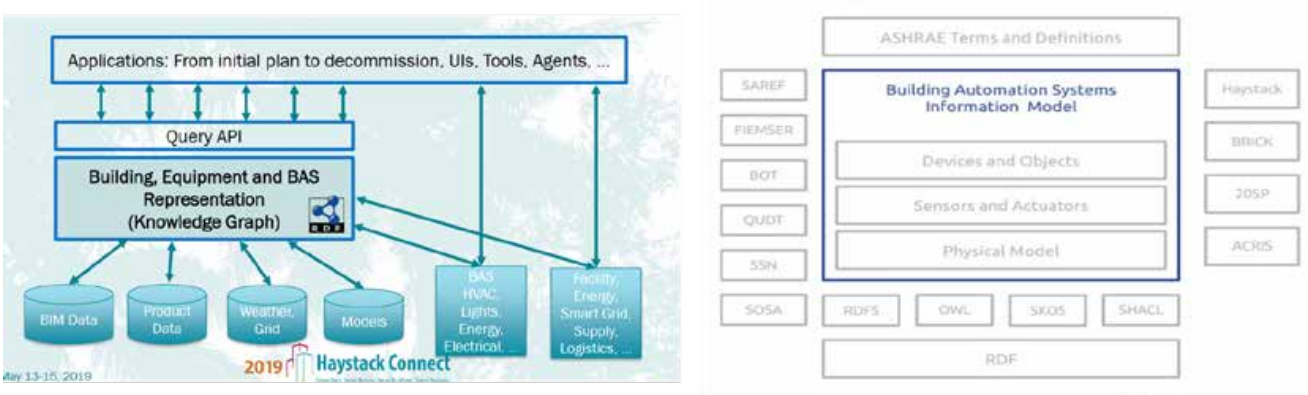


図 4.1: ASHRAE Standard 223P でのセマンティクス概要（発表資料より抜粋）

上記の提案は提案段階ではあるが，ISO 化も考慮に入れており，今後展開が期待される．しかしながら，Semantic Tags 自体がほとんど使われておらず，国内で BACnet の普及・展開活動を行っている電気設備学会も 223P の正式発効を待っている状態である．また，Brick で提案されているデータモデルは，BACS 構成要素の正確なモデリングには有効といえるが，AI/IoT などのドメインの異なる業界とのコラボレーションにおいては，その特殊性・専門性と複雑性から参入障壁が高くなり，かつ定義されているセマンティクスも有効に利用されることは少ない．この課題については，4.3.2 節で改めて考察する．

なお，建物に関するオントロジ定義は W3C の Linked Building Data Community Group[*6]でも行われている．後述する ifcOWL [60] や BOT [64] などを使った実際のシステムやオントロジについての議論が活発に行われている．

[*2] https://www.ashrae.org/about/news/2018/ashrae-s-bacnet-committee-project-haystack-and-brick-schema-collaborating-to-provide-unified-data-semantic-modeling-solution
[*3] https://project-haystack.org/tag
[*4] https://brickschema.org/ontology/
[*5] https://www.haystackconnect.org/wp-content/uploads/2019/05/Proposed-ASHRAE-Standard-223P-Bernhard-Isler.pdf
[*6] https://www.w3.org/community/lbd/

4.2.1 オントロジ

定義

オントロジは「概念 (Concept) と関係性 (Relation) の組を持ち，形式化されているためにコンピュータによる推論 (Reasoning) が可能であるもので，特定のドメインに対して有用にデザインされている」と定義されており，以下のような特徴を持つ [5].

1. **よく定義されたシンタックス**
2. **効率的な推論サポート**
3. **十分な表現力**
4. **便利な表現**

オントロジはセマンティックウェブにおいて中心的な役割を果たす技術である．情報の一元的な表現が可能になり，ウェブ上でデータを共有したり処理したりすることが容易になる．また，オントロジの持つ関係性などのセマンティクスは，ウェブサイトの組織化やナビゲーションに有用であるため，検索の精度を向上させることができる．オントロジやデータモデルを記述するための技術仕様としては，RDF がある．

RDF(Resource Description Framework)

W3C の仕様であり，ウェブ上のリソースの情報記述のために定義された．トリプルと呼ばれる情報単位をベースに構成されており，主語・述語・目的語の関係性がある．RDF は URI (Unified Resource Identifier) を曖昧性のない識別子として利用する．RDF の集合はグラフ構造をなし，それが知識表現・知識ベースとなる．記述されたデータ（リソース）は，知識ベースとなる関係データベースや SPARQL Endpoint と呼ばれる専用のトリプルストアに保存される．また，それらのデータがウェブ上で公開され，連結されていくことで Linked Data [43] が構成される．

RDFS (RDF Schema) は RDF のための Vocabulary　Description　Language であり，RDF に利用するクラスやプロパティから構成されており，より表現力のある OWL (OWL Web Ontology Language) をベースとしている．例えば，rdfs:subClassOf，rdfs:subPropertyOf などが定義されており，これらの語彙を使うことで，リソースやクラスの継承関係の記述，その関係性を用いた推論などが可能になる．なお，RDF で記述されたトリプルは，対象に関するメタデータの構造を示しているともいえる．そのためメタデータ・スキーマと呼ばれることもある．

建設ドメインのオントロジ

本節では，建設・維持管理 (AEC/FM) ドメインで提案されているオントロジの説明を行う．なお，これらは Bhattacharya [14] や Butzin [18] の調査に詳しい．

■ifcOWL [60]　BIM のデータ交換用のファイルフォーマットである IFC (Industry Foundation Classes) の語彙を，OWL を用いて表現したオントロジである．IFC から ifcOWL を用いた RDF への変換ツール[*7]も公開されているが，IFC との等価な情報変換を目指しており，IFC での記述情報のほとんどを RDF に変換していることから，冗長性が高く，ファイルサイズも非常に大きい．

■BOT (Building Topology Ontology) [64]　建物内の空間構成（トポロジー）を定義するために提案されたオントロジであり，2017 年に W3C Linked Building Data Community Group によって提案された．ifcOWL と比べて限定された語彙のみを有する．例えば，空間要素であるクラスとしては，敷地を表す bot:Site, 建物を表す bot:Building，フロアを表す bot:Floor，部屋など表す bot:Space があり，物理的な構成要素として bot:Element が定義されている．またそれらをつなぐ関係性として，bot:containsElement などの推移的なプロパティがある．

IFC から BOT を用いた RDF を出力するツールも提案されているが [17]，ifcOwl へ一度変換して，対応する BOT の語彙を置き換えをしているため，変換に時間がかかるとともに，独自に定義したプロパティに関するメタデータを付加するために冗長性も高い．

■Project Heystack　空調機器をはじめとする建物の中の多様なサブシステムに対して，タグ付けをするための語彙とシステムを提供している．ビル設備に特化しているため空間的な語彙が不足しているといわれており，Heystack のタ

[*7] https://github.com/pipauwel/IFCtoRDF

グを OWL によるオントロジに変換した Haystack Tagging Ontology (HTO)[*8] も提案されているが，空間内に存在するセンサを記述など，複雑なサブシステムの表現は難しい．あくまでもタグ付けのための語彙として利用するのが適切と考える．

■Brick [9] BACS のためのメタデータ・スキーマであり，空間やセンサ，システム間の関係性を記述するための十分な語彙を備えている．Balaji らは，17,700 データポイントに及ぶ 6 つのビルに適用し，その有効性を確認している．BACS のポイントのための Point クラスを定義し，そのサブシステムとして Sensor，SetPoint，Command などの BACS で汎用的なサブクラスを定義している．他に中心的なクラスとして，Location (場所)，Equipment (装置・デバイス)，Measurement (測定）があり，それぞれに対して適切なプロパティと関係性の語彙が定義されている．Brick によってビルのエンジニアやファシリティマネージャーは一貫したルールによるデータ・モデリングと，それらを用いたアプリケーション開発が可能になるとしている．

■SAREF(Smart Appliances RE-Ference) [20] Daniele らはスマートホーム産業と協力し，省エネを目的とし，スマート家電のための意味的な相互運用性を可能にするために SAREF[*9] と呼ばれるをオントロジを提案した．スマートホームやオフィスで利用可能なデバイスに関する語彙が提供されているが，BACS に関する記述や多様なデバイスが存在するスマートビルには向かないといわれている．

■Echonet Ontology [61] ECHONET はネットワーク化されたスマートホームにおける標準プロトコルを目指して提案されていたが，その複雑さと参考実装の少なさから普及には至らなかった．2011 年に提案された ECHONET Lite は，ECHONET を簡略化したもので，既に日本のスマートホームにおける機器制御の代表的なプロトコルになっている．多くのスマート家電が ECHONET Lite に対応しているが，SAREF のようなオントロジは提案されていなかった．Echonet Ontology (eOnt) は，ECHONET Lite で定義された 108 種類のデバイスサポートしたオントロジであり，Pham らは，それらを用いたオントロジ駆動型のシステムを実現するアーキテクチャを提案している．ECHONET はあくまでも，スマート家電の制御に特化しているため，マンションなどのユースケースには有効といえるが，スマートビルへの適用は難しいといえる．

4.3 スマートビルの要件分析

本節では，本研究が対象とするスマートビルのプロセス分析を行い，必要とされる実践的なセマンティクスについて検討を行う．

4.3.1 プロセス分析

必要なセマンティクス検討のために，クラウドを用いた BACS 構築のプロセスの分析を行った．これらは自身のプロジェクト実践の経験と，対応した専門業者へのヒアリングによる整理である．

図 4.2 に BPMN (Business Process Model and Notation) で記述した構築プロセスを示す．まずは設計仕様に基づき，各設備システムベンダーは通信仕様やポイントリストを作成する．それぞれの仕様をネットワークベンダーやクラウドシステムベンダー（データ・プラットフォームや AI/IoT を用いたアプリケーションの構築ベンダーを含む）に引き渡し，システム構築や単体試験の準備を行う．ネットワークベンダー，クラウドシステムベンダーの構築，単体試験が終了し準備が整った時点で総合連動試験が行われる．

ポイントリストを受領した後のデータモデル生成のためのプロセスを図 4.3 に示す．クラウドシステムベンダーはポ

[*8] http://www.vcharpenay.link/hto/doc.htm
[*9] https://ontology.tno.nl/saref/

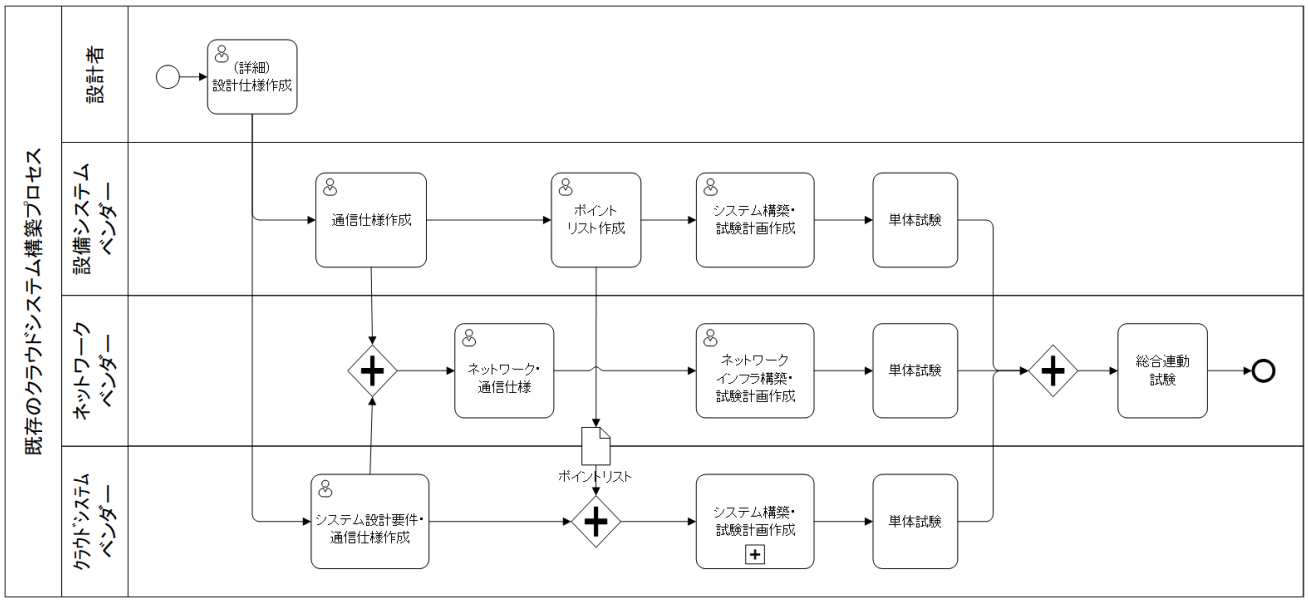


図 4.2: クラウド型 BACS の構築プロセス

イントリストを受領し，ポイント名（オブジェクト識別子，オブジェクト名，デバイス名）をもとに，制御・監視対象の特定を行うとともに，システムで利用するポイント表示名の変更・登録を行う．クラウドベンダーは独自にメタデータの定義を行い，例えば見える化における計量データのグルーピングなどを実現している．この際，受領したポイントリストのメタデータ欠如によって，デバイスとポイントの不整合といったヒューマンエラーが起きやすいことが課題となっている．この課題については，7 章で再考する．

新築ビルにおいて，毎回このようなエンジニアリングが行われるが，ベンダー毎にメタデータやそのスキーマ，利用される語彙などが異なるために，構築したアプリケーションの再利用や移植が難しい．また，建設プロジェクトの特殊な慣習などの問題もあり，対応経験がない専門業者だけでのシステム構築は対応は難しい．そのために，セマンティクスの共通化と 6 章で述べるデータ・プラットフォームが必要である．

4.3.2　採用するオントロジの検討

必要なセマンティクス

4.2.1 節で述べたように，AEC/FM ドメインにおいて多様なオントロジが提案されている．そこで定義されているクラス・プロパティといった語彙やデータモデルは，同じドメインのエンジニアであれば有用といえるが，IoT/AI などの専門業者とのコラボレーションを前提とした場合，語彙が不十分だったり，過度に複雑であったりと有効に活用されない．

例えば，Brick による BACS の記述能力は必要十分といえるものの，ここで提案されているセマンティクスを活用するユースケースはほとんどない．理由としては，代表的なスマートビルのユースケースである遠隔制御やモニタリング，見える化システムで必要となるのは，設備機器間の関係性ではなく，対象の空間やフロアなどの包含間関係であることがほとんどであるためである．例えば，空調機 (AHU) がどの VAV (Variable Air Volume) に対して空気を提供しているといったことは feedsAir という関係性語彙で記述できるが，それら静的な関係性は，AI などの適用シナリオにおいては利用されることはない．

より具体的には，AI 適用のユースケースでは，離散化されたベクトル表現の特徴量としての入力と，AI モデルを介して出力された制御コマンドなど，事前に定義したデータセットに入出力が固定されてることがほとんどである（図

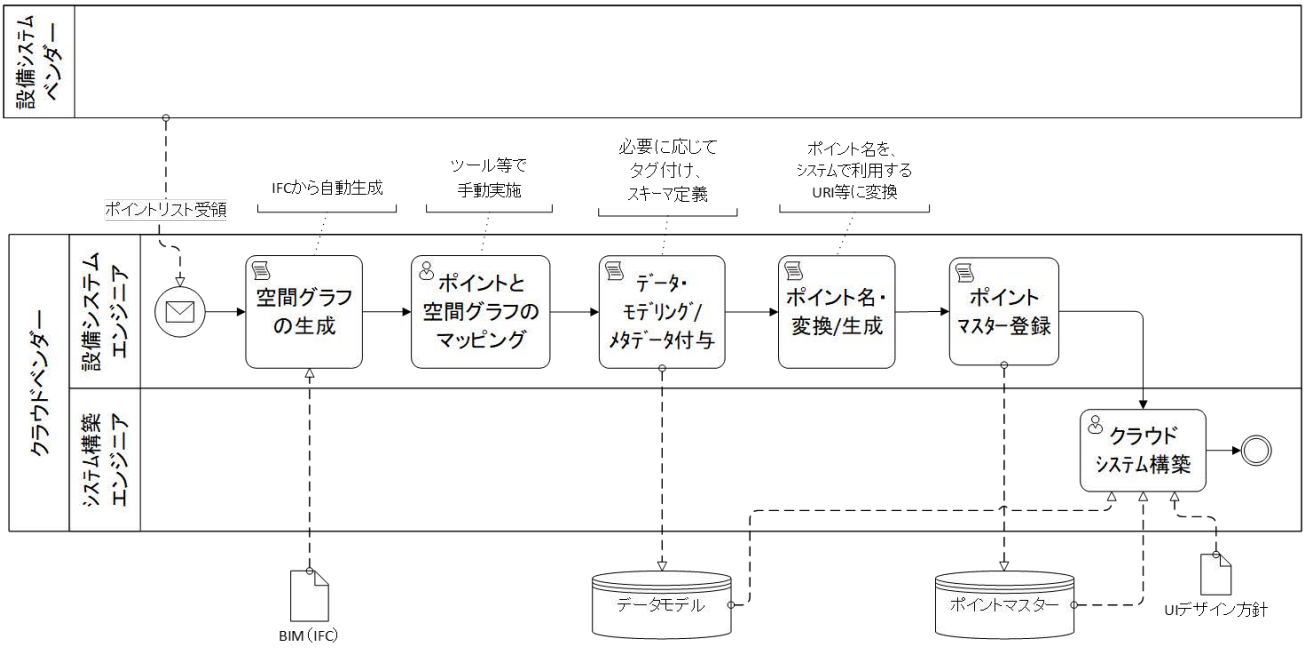


図 4.3: クラウドシステムにおけるデータモデリングのプロセス

4.4）．知識ベースを介して，AI が制御対象を動的に選ぶことは現状のユースケースではほとんどなく，入力ポイントの同定のために SPARQL によるクエリ発行が想定される程度である．モニタリングや見える化システム（図 4.5）においては，ポイント名や URI を元に，該当リソースのメタデータを検索したり，リソースグループを定義したりするユースケースが想定される．リソースグループは，例えばビル，フロア，設備毎に電力消費量をまとめて表示する際に利用する．

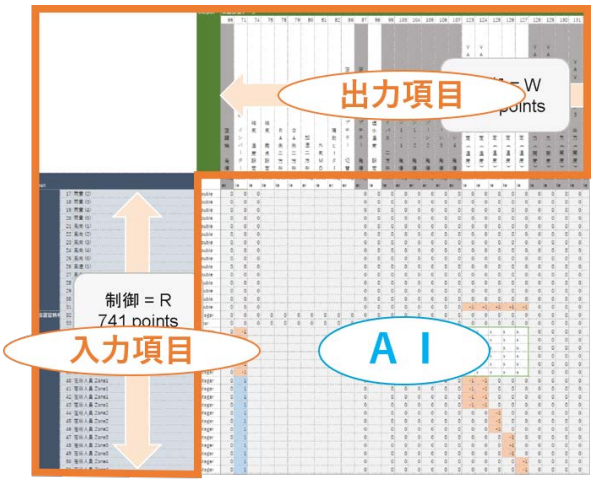


図 4.4: AI モデルによる入出力の関係性

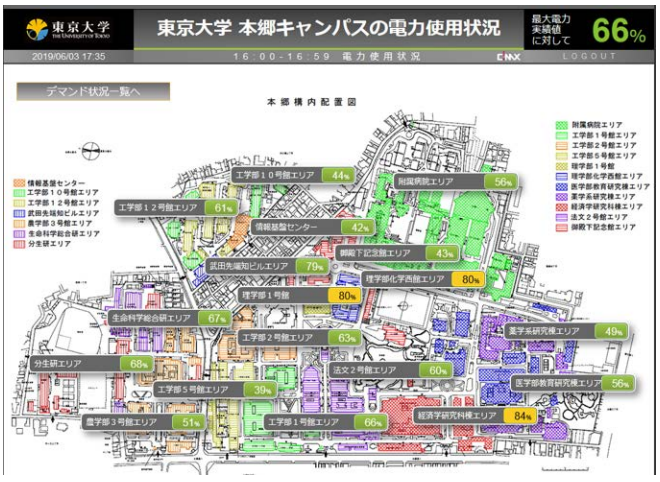


図 4.5: 見える化システム（東京大学の事例）

また，Brick でのモデリングは設備エンジニアが行う必要があるが，同時にオントロジに精通している必要があり，大型ビルの場合はポイント数も多くモデリングも負荷もかなり大きい．現状では BACS の専門業者はセマンティクスを付与するインセンティブはなく，BACnet の Semantic Tags もほとんど用いられておらず，自動化する手段がないために，モデリングを行う設備エンジニアやクラウドシステムの専門業者の生産性は上がらず，ヒューマンエラーも起

きやすくなっている．設計者や専門のエンジニアが IoT/AI と BACS の専門業者の間に入って解説などを行っているが，アプリケーション開発の負荷低減のために，セマンティクスの効率的な付与方法が必要である．

採用するオントロジ

前節で述べたように，必要なセマンティクスは，空間階層とデバイスやセンサ，ポイントの包含関係である（図 4.6）．それにより，クラウドのアプリケーションは制御対象の空間を指定することで，内包されるデバイス等の取得が可能になる．利用するオントロジは，空間階層の記述に適した BOT と BACS のデバイスの記述に適した Brick を採用する．

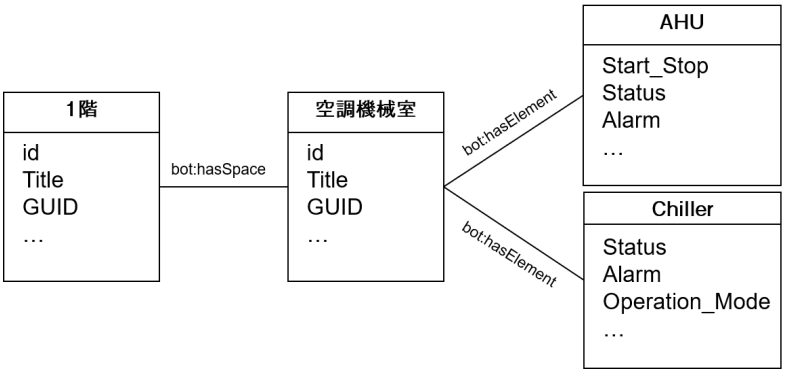


図 4.6: 必要なセマンティクス

これらの意味合いとしては，BACS で一般的なポイントによるエンジニアリングから脱却し，空間階層構造によるオブジェクト指向のモデリングによって，サードパーティーのアプリケーション開発者に対してより直感的なインタフェースを提供することにあるといえる．

4.4　提案手法

本節では，BIM と BACS のポイントリストを用いて，BOT や Brick を用いたデータモデル生成を簡易化するための手法を提案する．提案手法では，空間構成（空間グラフ）を得るために BIM を用いる．また，空間グラフと BACS のポイントリストの情報を突合させることによりモデリングの半自動化を実現する．

図 4.7 に，提案手法によるデータモデル生成のパイプラインと，関連システムの構築イメージを示す．ポイントリストにメタデータを付加し，RDF でモデリングしたデータを知識ベースに格納し，BACS から取得する時系列データのメタデータとして利用する流れと，BIM からメタデータと形状（ジオメトリ）を分けて抽出する流れがある．それぞれに抽出・生成したデータは，専用のデータベース（時系列 DB，ドキュメント DB または知識ベース）やウェブサーバなどに配置される．図中では配置されたデータを元に統合された RDF を作成し，デジタルツインのアプリケーションが動作する様子が示されている．インターネットと親和性の高い技術を多く用いており，BACS や IoT のデータに加え，BIM から抽出した形状データも活用可能なデジタルツインのためのアーキテクチャとなっている．

以下では，関連技術である IFC について述べた後，まず BIM からセマンティクスを抽出し，RDF を生成するためのツール開発について述べる．続いて，空間グラフとポイントリストとの突合手法について説明し，実プロジェクトにおける適用について述べる．

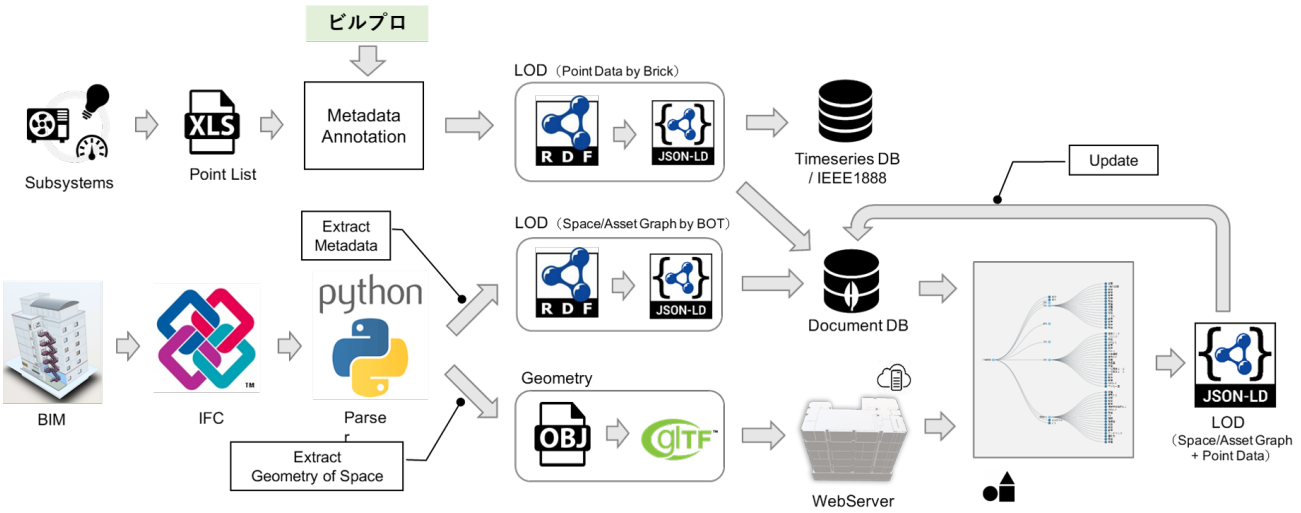


図 4.7: 建物メタデータ抽出のためのパイプライン

4.4.1 関連技術

IFC (Industry Foundation Classes)

本研究では BIM のデータ交換のための標準形式である IFC を対象として，セマンティクスの抽出を行う．IFC (Industry Foundation Classes) は 2013 年に ISO 16739[*10] として国際標準化されており，交換用フォーマットとしてほぼデファクトになっている．IFC には，BIM でモデリングされる形状情報（ソリッドモデル）と，空間階層構造や設備機器や部材の属性情報の双方が記録されており，それらが例えば ISO 10303 で規定されている STEP でエンコードされている．図 4.8 に IFC の例を示す．IfcSpace などの要素定義と，それらの形状情報（IfcPolyline, IfcExtrudedAreaSolid）や属性などが記述されていることが分かる．

IFC を用いることの利点は，ほぼすべての BIM のオーサリングツールで出力が可能であるため，汎用化が容易なことである．BIM から IFC への連携・出力については，そのプロセスやフォーマット含めて IDM (Information Delivery Manual) と MVD (Model View Definition) によって規定されており， buildingSMART Japan[*11]で行われている IFC 検定によってその品質が保証されている．

4.4.2 セマンティクス抽出ツールの開発

我々は IFC から空間グラフと包含される要素を抽出し RDF に変換するツールを開発した．それらの生成フローを図 4.9 に示す．

空間グラフとは，空間的な包含関係を有したグラフ構造のデータモデルであり，例えば「敷地→建物→フロア→部屋」といった階層的な構造を指す．IFC には，IfcSite, IfcBuilding, IfcBuildingStorey, IfcSpace といった，対応するクラスが定義されており，それらを抽出・構造化することで，空間要素の階層化が可能である．また，抽出した空間要素に対して，関係性やクラスを付与するとともに，必要に応じて IFC から抽出した属性などのメタデータを付与したデータモデルを生成する．これらに対して，BACS のポイントを対象となる要素に突合する．例えば，温度センサのポイントを計測対象の部屋やセンサデバイスに紐づける．システムで利用する表示名などを付加するとともに，ユニーク

[*10] https://www.iso.org/standard/70303.html
[*11] https://www.building-smart.or.jp/

図 4.8: IFC の例（抜粋）

```
#2858= IFCSPACE('3q4P0iC9P33P2WOFzvBOXb',#41,'51',$,$,#2826,#2856,'\X2\30C830A430EC\X0\',.ELEMENT.,.INTERNAL.,$);
#2861= IFCPROPERTYSINGLEVALUE('Reference',$,IFCIDENTIFIER('\X2\30C830A430EC\X0\ 51'),$);
#2862= IFCPROPERTYSET('2FNT1K8tLDogltFDZWyAQR',#41,'Pset_SpaceCommon',$,(#2861));
#2864= IFCRELDEFINESBYPROPERTIES('24w_Q7l$9AwOktuDm126SJ',#41,$,$,(#2858),#2862);
#2868= IFCAXIS2PLACEMENT3D(#6,$,$);
#2869= IFCLOCALPLACEMENT(#157,#2868);
#2870= IFCCARTESIANPOINT((540.,125.));
#2872= IFCCARTESIANPOINT((990.,125.));
#2874= IFCCARTESIANPOINT((990.,290.));
#2876= IFCCARTESIANPOINT((-1530.,290.));
#2878= IFCCARTESIANPOINT((-1530.,-415.));
#2880= IFCCARTESIANPOINT((540.,-415.));
#2882= IFCPOLYLINE((#2870,#2872,#2874,#2876,#2878,#2880,#2870));
#2884= IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,$,#2882);
#2885= IFCCARTESIANPOINT((15850.57102928,2272.03669197011,0.));
#2887= IFCAXIS2PLACEMENT3D(#2885,$,$);
#2888= IFCEXTRUDEDAREASOLID(#2884,#2887,#19,3500.);
#2889= IFCSHAPEREPRESENTATION(#102,'Body','SweptSolid',(#2888));
#2891= IFCPRODUCTDEFINITIONSHAPE($,$,(#2889));
#2893= IFCSPACE('3q4P0iC9P33P2WOFzvBOXa',#41,'52',$,$,#2869,#2891,'\X2\90E85C4B\X0\',.ELEMENT.,.INTERNAL.,$);
#2896= IFCPROPERTYSINGLEVALUE('Reference',$,IFCIDENTIFIER('\X2\90E85C4B\X0\ 52'),$);
#2897= IFCPROPERTYSET('3mdLZOPPH8X8BQi8MpoMd8',#41,'Pset_SpaceCommon',$,(#2896));
#2899= IFCRELDEFINESBYPROPERTIES('1gD3lnfW95EAp56XusjpCv',#41,$,$,(#2893),#2897);
#2903= IFCAXIS2PLACEMENT3D(#6,$,$);
```

なリソース識別子として URI を付与し，対象の時系列データへの参照等を保持することで，ここでは建物メタデータと呼ぶ RDF を作成する．建物メタデータを元に，システムで利用するデータモデルに変換し，ポイントマスターとする．これによって，クラウド側のシステムで BACS のポイントを利用する準備が完成する．

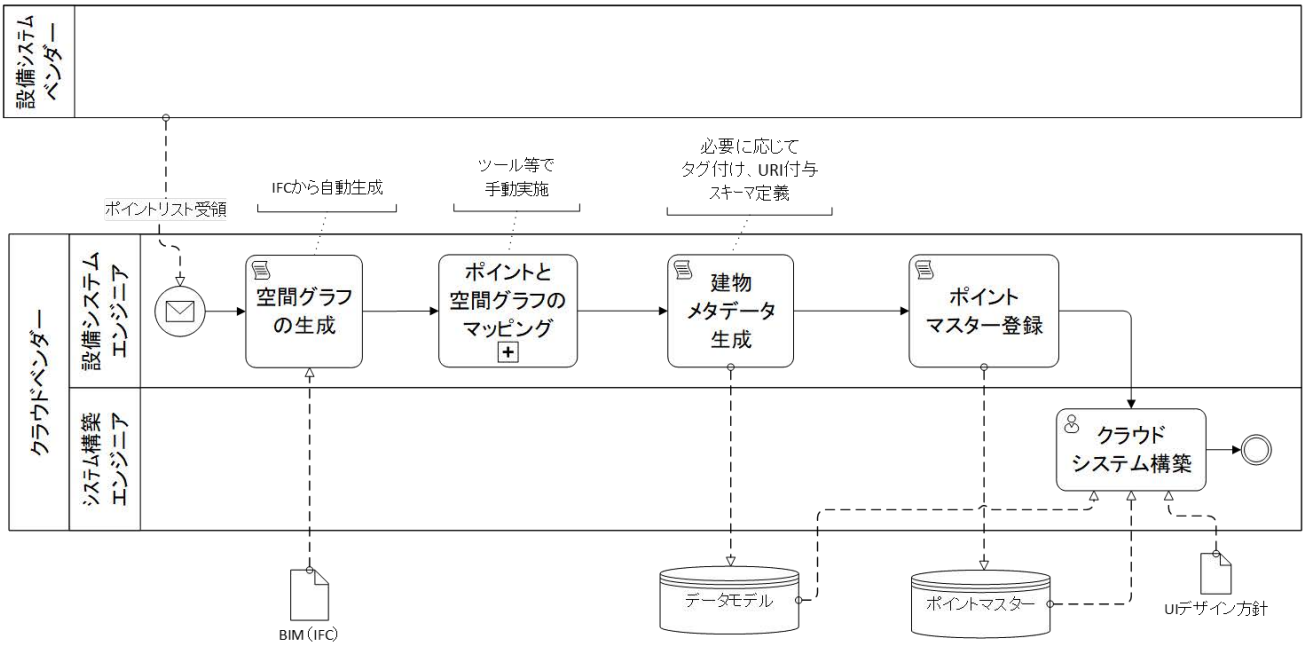


図 4.9: BIM とポイントリストを用いたデータモデリングのワークフロー

ツール開発

IFC から RDF への変換ツールとして IFCtoLBD[*12]があるが，日本語対応などに課題があったり，冗長な情報が含まれていたりするため，IfcOpenShell[*13]を用いて独自に開発する方針とした．

IfcOpenShell はオープンソースの IFC 操作用のライブラリであり，3DCAD 用のライブラリである OpenCASCADE[*14]を内包しているため，IFC に表現されたソリッドモデルを扱うことができる．IfcOpenShell では任意のフォーマットで BIM モデルの形状を抽出することができるが，BACS においては，壁や機器といった要素の形状よりも，それらを包含する空間を対象とすることがほとんどであるため，IFC に表現された空間形状のみを抽出することとした．なお，同様の機能は Autodesk Forge の Model Delivertive API[*15]を用いて実現することが可能であるが，IfcSpace などの抽象的な要素の形状抽出はできず，また実装も煩雑になるために，Python の 3 次元メッシュのためのライブラリである trimesh[*16]を併せて用いることで実装した．具体的な処理手順を以下に示す．

1. IfcSite/IfcBuilding を特定し，BOT のクラスとプロパティを当てはめるとともに GlobalId (GUID)，Name といった IFC の属性を抽出し，それらをプロパティとする RDF リソースを生成
2. IfcBuilding に含まれる IfcBuildingStorey を抽出し，同様に RDF リソースを生成
3. IfcBuildingStorey が内包する IfcSpace を抽出し，同様に RDF リソースを生成
4. IfcSpace に IfcExtrudedAreaSolid や IfcFacetedBrep といったソリッドモデルの定義がある場合，それらを 3 次元メッシュの一般的なファイル形式である OBJ 形式や，ウェブへの親和性向上を意図した GLTP に変換
5. IfcSpace が IfcBuildingElementProxy などのデバイス等を表す物理要素を持つ場合，包含する要素として RDF リソースを生成

表 4.1 に開発したツールの機能比較を示している．抽出する要素数の比較を行っているが，ここでは東京大学の I-REF 棟の IFC データ（図 4.10）を用いた．空間要素 (bot:Zone) の抽出数としては，IFCtoLBD と等しいが，空間以外の要素 (bot:Element) は少なくなっている．これは不要と思われるノードをプログラムの中でフィルタリングをしているためである．

Dynamo-bot-exporter（図 4.11）は，Dynamo[*17]と呼ばれる，ビジュアルプログラミング環境において要素を抽出し，RDF に整形するものであるが，IFC の読み込みが難しく，一度 BIM オーサリングツールである Autodesk Revit を経由する必要がある．直感的なプログラミングが可能であるが，RDF の出力のロジックは複雑で，Revit の API に依存してしまうためにメンテナンスも難しい．抽出している数量が異なるのは，Revit による空間や要素の定義と，IFC による定義が異なるためである．

[*12] https://github.com/jyrkioraskari/IFCtoLBD
[*13] http://ifcopenshell.org/
[*14] https://www.opencascade.com/
[*15] https://forge.autodesk.com/en/docs/model-derivative/v2
[*16] https://github.com/mikedh/trimesh
[*17] https://www.dynamoprimer.com/ja/

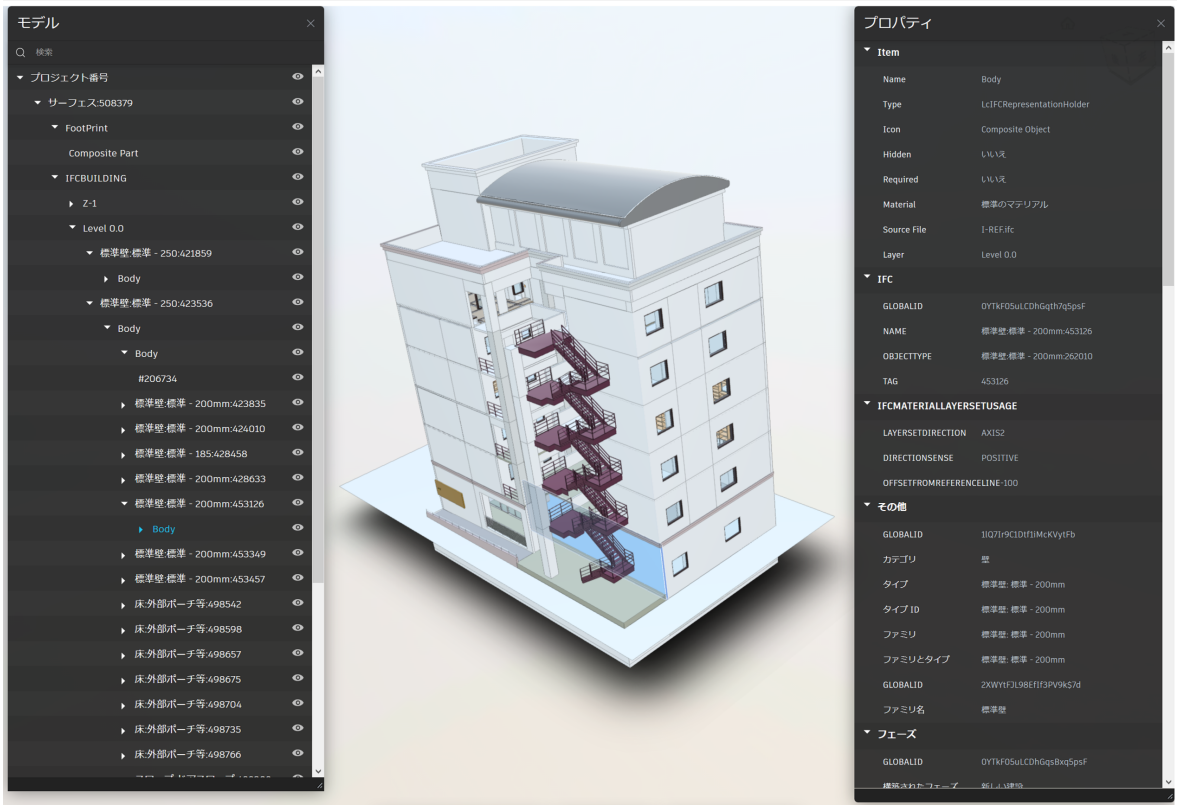


図 4.10: I-REF 棟の IFC データ（Autodesk Forge による可視化例）

表 4.1: IFC 変換ツールの比較

	IFCtoLBD	Dynamo-bot-exporter	IfcOpenShell（自作）
特徴	IfcOWL を介して BOT を生成する．空間要素以外も独自の PRODUCT オントロジを使って記述される．Java のプログラム．	要素の抽出は Dynamo で行い，RDF の生成は Revit のアドインで行う．BIM の機能を使って，面積や形状情報などの付加的な情報も取得することができる．	IfcOpenShell を使って，空間構成に沿って BOT オントロジで直接 RDF を生成する．Python のプログラム
日本語対応	×	×	○
bot:Element の数	2852	797	2533
bot:Zone の数	138	150	138
形状の抽出	IfcConvert を使って，GUID(または Name)から抽出．別ファイルとなるため，bot:3DModel を使って参照を記述．	C#で Revit 要素から直接 OBJ を出力している．bot:simple3DModel を使って参照を記述できる．Area 情報の形状出力もできるのが強み．	空間構成の解析と同時に OpenCASCADE を使って空間形状を出力．IfcSpace に定義された空間形状を OBJ または，Trimesh を使って GLTF を直接出力．
課題	面積やエリア情報のなどの出力ができない．（ライブラリを拡張すれば可能）	Autodesk Revit に限定されてしまうため汎用性が低い．データ取得も Revit API の仕様に依存する．	-

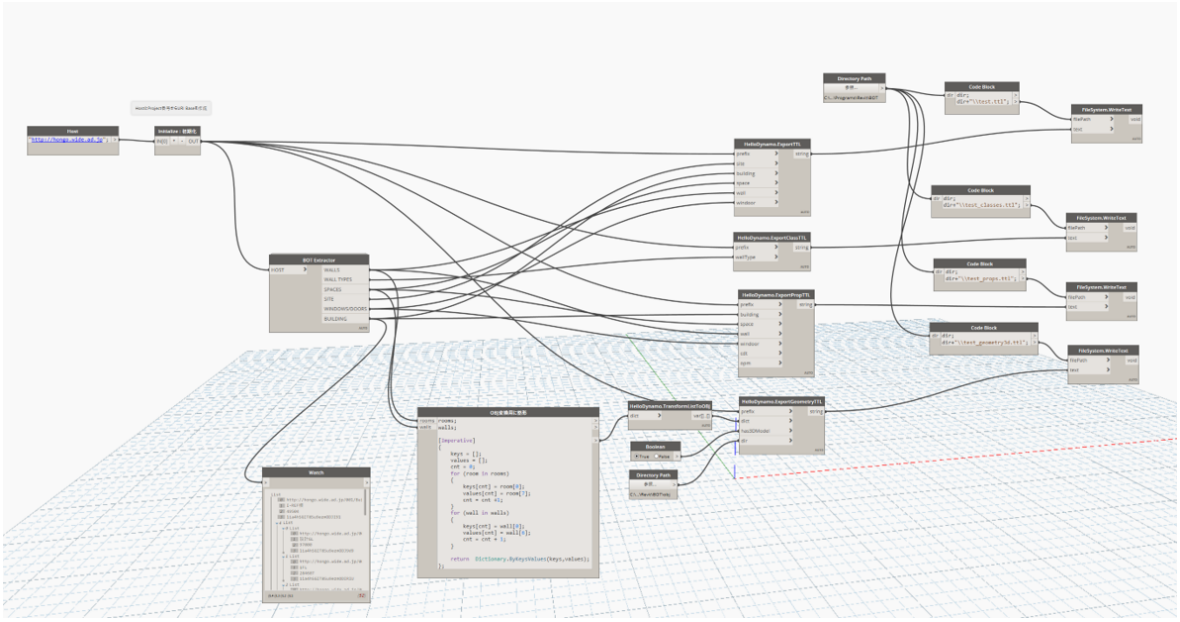


図 4.11: Dynamo-bot-Exporter

図 4.12: 出力後の RDF（抜粋）

```
inst:I-REF 棟_115 a bot:Building ;
    ifc:Address "東京都文京区弥生 1-1-1" ;
    ifc:GlobalId "1ia4hSGIT8Su9ezmRoiXpG" ;
    ifc:LongName "I-REF 棟" ;
    ifc:Name "I-REF 棟" ;
    ifc:NumberOfStoreys 10 ;
    ifc:id 115 ;
    ifc:type "IfcBuilding" ;
    bot:hasStorey inst:2FL_152,
        inst:3FL_158,
        inst:4FL_164,
        inst:5FL_170,
        inst:6FL_176,
        inst:Level_0.0_132,
        inst:RF1_182,
        inst:RF2_188,
        inst:Z-1_126,
        inst:設計 GL_142 .

inst:Level_0.0_132 a bot:Storey ;
    ifc:Elevation -5e+02 ;
    ifc:GlobalId "1ia4hSGIT8Su9ezmODIdJ5" ;
    ifc:LongName "Level 0.0" ;
    ifc:Name "Level 0.0" ;
    ifc:id 132 ;
    ifc:type "IfcBuildingStorey" ;
    bot:containsElement inst:スロープ_559837,
        inst:床_559610,
        inst:床_559795,
        inst:標準壁_158919,
        inst:標準壁_159060,
```

出力後の RDF データ（Turtle フォーマット）の一部を図 4.12 に示す．IFC の状態では分かりにくかった属性データが構造化されている．例えば，「I-REF 棟」という名前のリソースは 10 個の階 (bot:Storey) を持つことや，「Level 0.0」という名前を持つ階のリソースは「スロープ_ 559837」「床_ 559610」などの要素を含むことをが確認できる．図 4.13 に生成した RDF の可視化例を示す．建物，フロア，部屋といった空間階層が抽出されていることが見てとれる．

図 4.14 に抽出した形状データを示す．IfcSpace で定義された形状のみを抜き出すようになっており，デバイスが包含する空間を示すことができる．図 4.15 に出力した形状データにおける OBJ 形式のファイル抜粋を示す．g(グループ) に IFC から出力した GUID が確認できる．なお，ウェブ系のアプリケーションでは，3 次元ファイルフォーマットとして GLTF が標準になると言われている．GLTF では GUID だけではなく階層構造や名称といったセマンティクス，メタデータも内包することが可能であるため，形状を動的に扱うようなアプリケーションには有効といえる．

これらの空間形状を用いたアプリケーションとしては，例えば図 4.16 のように空間形状と，対象のメタデータを表示するようなデジタルツインのアプリケーションが考えられる．同様のアプリケーションとしては，Ji らの研究がある [45]．空間形状に対して，人数や環境指数を元に色付けすることで，ビル内のエネルギーの無駄を可視化している．

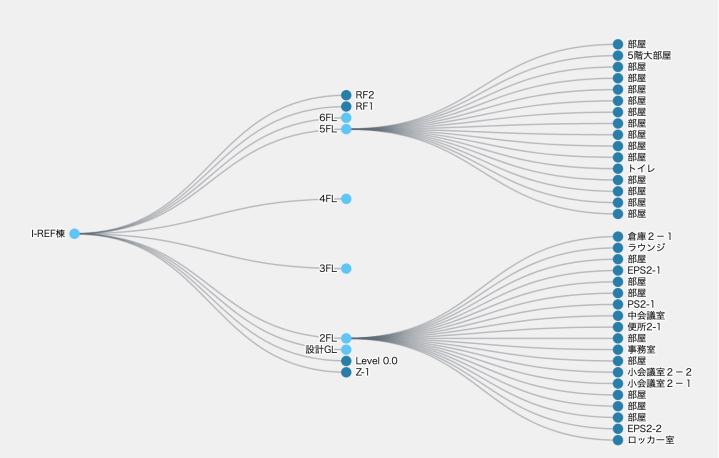


図 4.13: IFC から抽出した空間階層構造

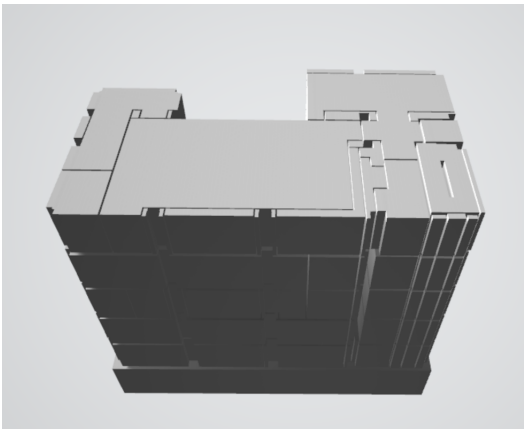


図 4.14: 抽出した IfcSpace の形状

図 4.15: IFC から出力した形状データ（抜粋）

```
g 3q4P0iC9P33P2WOFzvBOXb
v 10.96057102927976 -6.802963308029869 8.200000000000049
v 10.96057102927976 -6.802963308029869 11.700000000000049
v 10.96057102927976 -7.20296330802987 8.200000000000049
v 10.96057102927976 -7.20296330802987 11.700000000000049
…中略
f 1078 1062 1070
f 1074 1076 1072
f 1076 1078 1072
f 1078 1070 1072
```

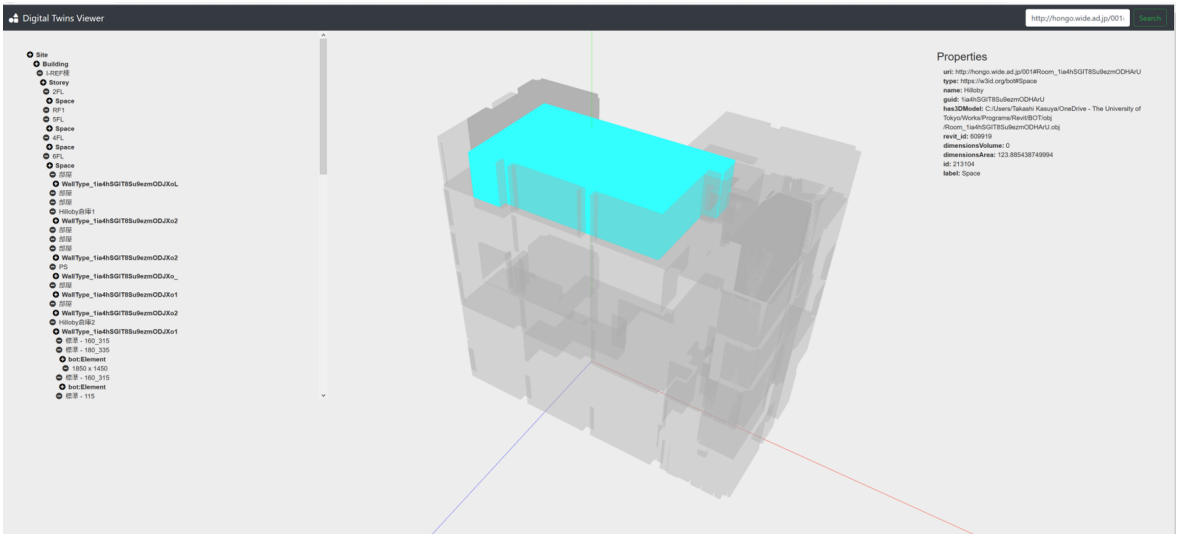


図 4.16: デジタルツインの可視化アプリケーション例

また，IfcOpenShell の機能を利用することで，特定のノード以下に限定した形状抽出も可能である．図 4.17 に IfcBuildingStorey 以下を指定した場合の結果を示す．ここでは，抽出したノードの情報をもとに，IfcOpenShell のコマンドツールを用いて当該形状データを抽出している．図 4.15 と同様に，形状データ内に GUID が保存されており，出力した RDF と合わせて用いることで，デジタルツインのアプリケーション開発に利用できる．

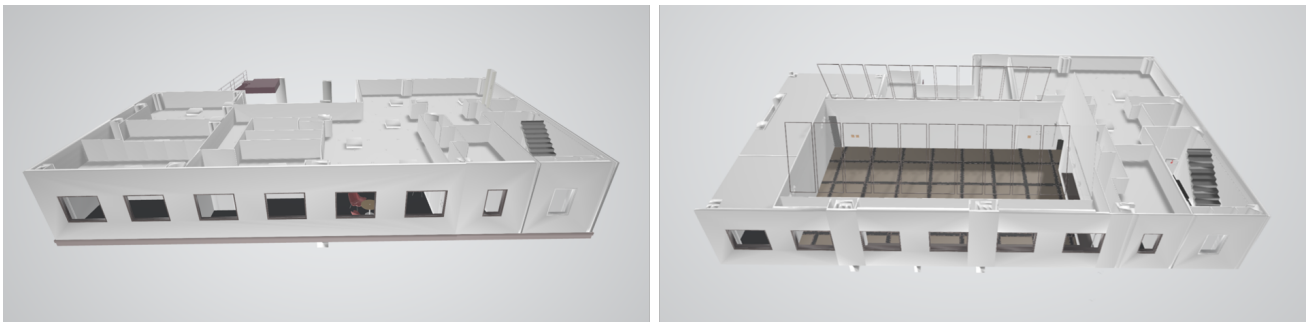


図 4.17: IfcOpenShell で抽出した I-REF 棟のジオメトリ（左：2 階、右：6 階）

空間グラフとポイントリストの突合

BIM から抽出・生成した RDF と，BACS のポイントリストを用いて，それらを突合させた統合 RDF（建物メタデータ）を作成する．具体的には，図 4.9 で示したように，IFC から生成した空間グラフの RDF と，Brick の語彙やクラスを付加した取込用ポイント CSV を用いて生成する．

図 4.18 は生成した建物メタデータの例である．Entrance という空間が 4 つのデバイスを有しており，そのうちの 1 つのデバイスが CO2-W-3 というポイントを有していることが記述されている．

空間グラフとポイントリストの突合については，BIM でのモデリングの粒度によって多少異なるが，概ね BIM の属性として突合用の ID を埋め込むことで自動化が実現できる．突合用の ID を付加した取込用 CSV としては，例えば表 4.2 のようなスキーマのファイルを利用する．ここでは 6 章で述べる WoT でのエンドポイントを生成を前提として付加的なメタデータを追加している．@type が該当するクラスであり，Brick の語彙を用いたり，該当しない場合は独自定義の語彙を用いる．図 4.18 の例でいうと， inst_class:CO2-W-3 が独自定義のクラスである．ここでの突合の処理

図 4.18: 統合 RDF（建物メタデータ）の例

```
inst:Entrance a bot:Space ;
    rdfs:label "Entrance" ;
    bot:hasElement inst_device:CO2-W-3,
        inst_device:MS-W-2,
        inst_device:MS-W-3,
        inst_device:TEP-C-3-RT .

inst_device:CO2-W-3 a inst_class:CO2-W-3 ;
    rdfs:label "CO2-W-3" ;
    rdfs:comment "CO2 濃度" ;
    brick:hasPoint inst_point:CO2-W-3 .

inst_point:CO2-W-3 a brick:Sensor ;
    rdfs:label "CO2 濃度" ;
    inst:topic "takenaka.co.jp/Site_Name/Building_Name/-/Entrance/
                Utility/Sensor/WebCNT3/CO2-W-3/CO2_Level/R" .
```

は容易なものであるが，例えば Python の rdflib[*18]を利用して，空間グラフの RDF から突合対象のノードと ID を検索し，該当する ID の情報を CSV レコードから抽出，当該ノードのクラス名や，それらが保持するポイントを特定していくことで実装できる．

表 4.2: 取込用 CSV スキーマの例

名称	必須	説明
GW	○	ポイントを収容するゲートウェイの識別子
pointId	○	ユニークなポイント ID
title	○	名称 (英語)
titles		名称 (日本語)
@type	○	デバイスのクラス名（Brick など）
property	○	プロパティ名
hasTag		タグ（複数指定可能）
path		BOT パス（BIM がない場合）
description		ポイントの説明 (英語)
descriptions		ポイントの説明 (日本語)
read/write	○	読み取り (R) / 書きこみ（W）
Element	○	突合用のデバイス ID
topic	○	MQTT TOPIC
unit		単位
minimum		最小値
maximum		最大値
telemetryType	○	事前に定義したテレメトリのスキーマ名

4.4.3　適用と評価

提案手法を，実際のプロジェクトで作成した BIM に適用し評価した．ここで用いる BIM は Autodesk Revit でモデリングした竹中工務店技術研究所の設備用 BIM である．元々は建築 BIM モデルと，施工用に制作された照明設備，空調設備などのサブシステム毎の BIM モデルしかなかったが，それらを統合して必要な属性を付加し，制御を伴うデジタルツインのアプリケーション構築のために新たに作成した．例えば，ビーコン発信機などの細かな IoT や設備機器も含めてモデリングしており，スマートビルのアプリケーションでの利活用のために，必要な属性を埋め込んでいる．なお，BIM のモデリング対象の面積は 39,150m^2 であり，BIM から出力した IFC のファイルサイズは 59.8M バイト

[*18] https://github.com/RDFLib/rdflib

であった．

図 4.19 に BIM から出力した IFC の形状情報と，開発したツールで抽出した空間形状を示す．右に示す空間形状は，Autodesk Revit ではスペースや空調用のゾーンとして定義されるものが抽出されている．

図 4.20 にツールで生成した RDF の可視化例を示す．これらの可視化は SPARQL Endpoint である GraphDB[*19] を用いた．赤のノードが敷地 (bot:Site) を示し，そこから hasBuilding というプロパティで定義されているのがビル (bot:Building) である．BIM でモデリングされたエリアは地下 1 階から地上 3 階までとなっており，図中で展開されているのが 2 階を表すノード (bot:Storey) である．黄緑色のノードは空間・部屋 (bot:Space/bot:Zone) を表し，そこから抽出されたデバイス (bot:Element) が確認できる．なお，当該 IFC から抽出されたノードの数量としては，bot:Storey は 4，bot:Space は 179，デバイスの構成要素 bot:Element は 17,196 であった．

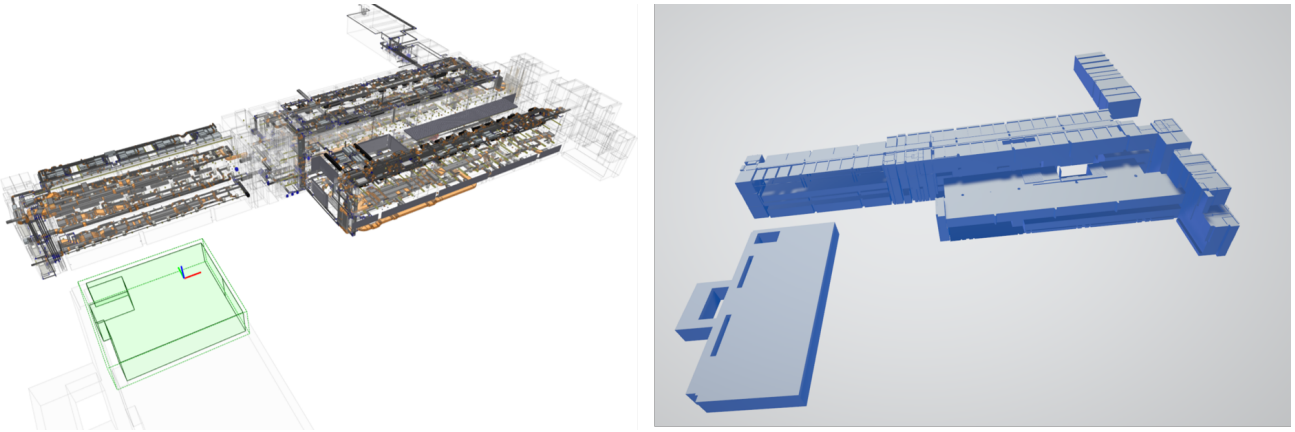


図 4.19: 竹中工務店技術研究所で作成した IFC（左）と抽出した空間形状（右）

図 4.21 に BIM モデルにおける属性の設定例を示す．環境センサを空間上に配置しており（実際は机上であるが，机のモデルは省略されている），そのパラメータとして機器のユニークな ID (BLEaddress) や，ここではデバイスが発出する MQTT の Topic を管理用に埋め込んでおり，今回はこれらを突合用の ID として用いた．

表 4.3 に空間グラフとの突合に利用した取込用 CSV のレコード例を示す．この CSV と図 4.20 の RDF を用いて，建物メタデータを生成している．なお，生成された建物メタデータは次章で述べる，Web におけるエンドポイント生成にそのまま用いられる．

4.4.4 考察

提案手法は，ユースケースをもとにした実践的なモデリング手法でおり，サードパーティにも理解しやすいセマンティクスを提供している．IFC を用いた汎用的な手法となっており，既存手法における日本語対応などの課題を解決している．また，I-REF 棟や竹中工務店技術研究所の BIM データにおいて手法を適用し，十分に大きく複雑なデータにおいても有効に機能することを確認した．

今回，IFC から抽出した空間要素には BOT のクラスを当てはめたが，抽出したノードに「2FL」「3FL」などのタグ情報の付与するという考え方もある．Heystack のようなタグと Brick のクラス定義に関して，Fierro による分析 [31] があり，どちらも利点があるといえるが，それらはアプリケーションに応じて変えるべきであろう．なお，取込用 CSV スキーマとして例示した表 4.2 には hasTag という項目があり，次章で述べるデータ・プラットフォームでは，モデリング後の空間要素やデバイスにタグ情報を加えることができる．

[*19] https://www.ontotext.com/products/graphdb/

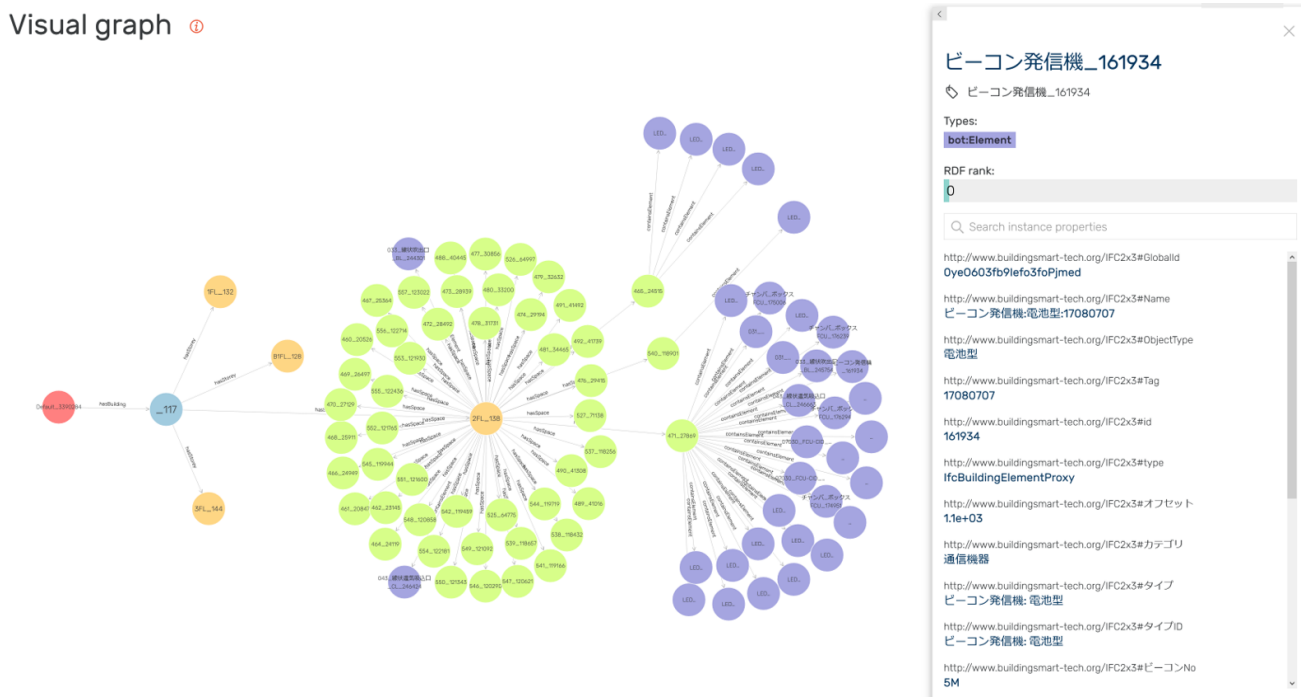


図 4.20: RDF の可視化（抜粋）

採用した空間グラフは直感的で理解しやすいといえるが，施設管理の視点では，例えば設備種別ごとに集約したグルーピング（照明，空調など）が好ましい場合もある．それらを正確にモデリングするとセミラティス構造になるが，モデリング方針は建物種別や管理者ごとに固有であるといえるため，それらは都度手動でのモデリングが現実的といえる．

また，Brick で提案されているセマンティクスについては，AI/IoT のユースケースでは利用されないことを述べた

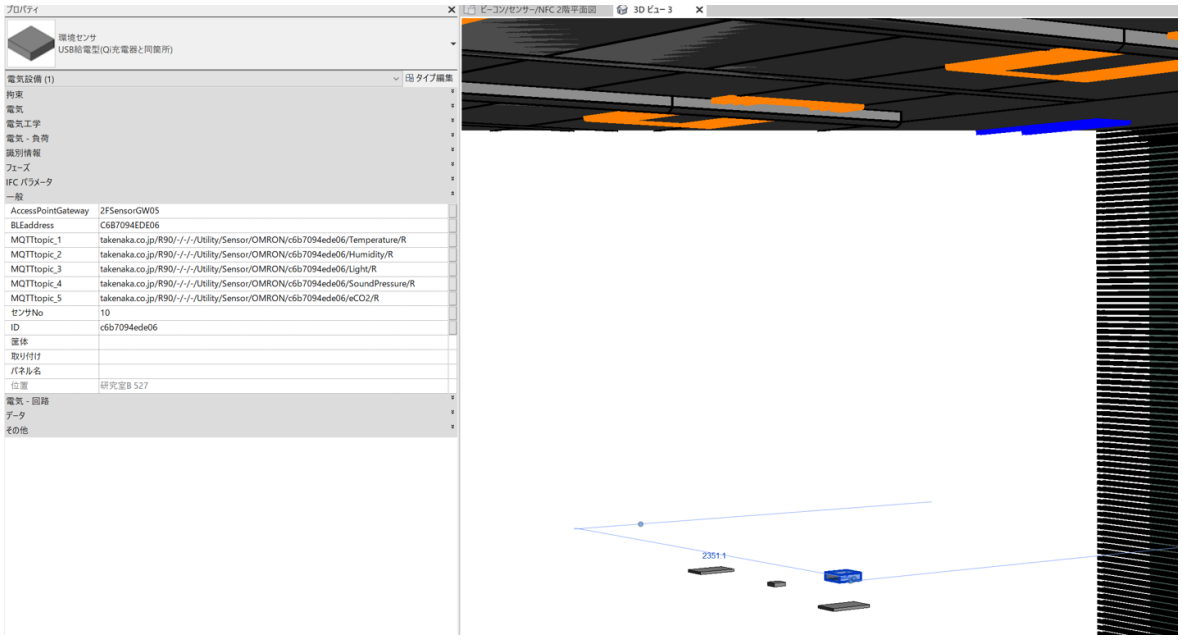


図 4.21: BIM への突合用の ID の埋め込み

表 4.3: 取込用 CSV のレコード例

名称	説明
GWId	R90-research-00001
pointId	R90_000126
title	Wireless_Sensor_c6b7094ede06_Temperature
titles	無線センサ（温度）_c6b7094ede06
@type	Wireless_Sensor
property	Temperature
hasTag	2F&&Temperature
path	
description	Wireless Sensor (Temperature)
descriptions	無線センサ（温度）
read/write	R
Element	c6b7094ede06
topic	takenaka.co.jp/R90/research/2F/5/Utility/Sensor/OMRON/c6b7094ede06/Temperature/R
unit	degC
minimum	-20
maximum	40
telemetryType	Sensor

が，BACS の機能自体をクラウドに構築するユースケースでは有効といえる．これらは BIM を用いるだけでは自動生成が難しいといえるが，例えばクラウドから制御コマンドが発行できる場合，反応する設備を特定することで，ある程度の自動化は可能と考えられる．これらについては，遠隔制御が必要であるため今後の課題としたい．

BIM とポイントリストの突合の際に，属性として ID を埋め込む手法について述べたが，実際のプロジェクトでは，BACS の専門業者が BIM を扱うことはほとんどない．設備用の BIM も施工中に干渉チェックなどで用いられるのがほとんどである．理想的には設計，施工時のデータを維持管理 (FM) に展開できるのが理想であるが，現状では設計・施工 BIM を参考に FM 用の BIM を新たに制作せざるを得ない．FM 用のモデル制作は，オーナー側のコスト負担となるため，受け入れられにくいといえるが，スマートビルのアプリケーションと，BIM-FM と呼ばれる FM における BIM 活用が今後普及・高度化し，その利便性が認められることで，それらは受容されやすくなると考える．また，そのためのプロセスをまとめた IDM や MVD を整備していくのも今後の課題といえる．

4.5　SDM アプリケーション

本節では BIM から抽出した形状とデータモデルを用いて，SDM アプリケーションである web360^2[80, 48] を再構成することで，その汎用性について評価する．

4.5.1　web360^2

web360^2 は，ウェブブラウザやタブレットで立体音響再生を実現する Web アプリケーションである．360 度動画中に配置された SDM オブジェクト（オブジェクトオーディオ）の再生の ON/OFF が可能であり，視聴位置や角度を変えることで，それらの動きに追随した視聴体験の提供が可能である．

先行研究 [80] における実装[*20]では，SDM オブジェクトの設置位置は設定ファイルで静的に指定するため，そのキャリブレーションに時間がかかっていた．これらは IoT を用いたデジタルツインの実現に共通の課題であるといえるが，仮想空間での 3 次元モデリングは BIM が得意とすることであり，その情報を利用することで開発効率の向上が可能になるといえる．

4.5.2　開発方針とシステム構成

開発方針としては，前節までに述べた BIM から抽出した形状とデータモデルを用いて，簡易にアプリケーションを再構築することとする．コンテンツに利用する BIM モデルは，東京大学での I-REF 棟とし，6 階大部屋の Hilloby に配置されているスピーカー位置に SDM オブジェクトを配置することを目標とする．

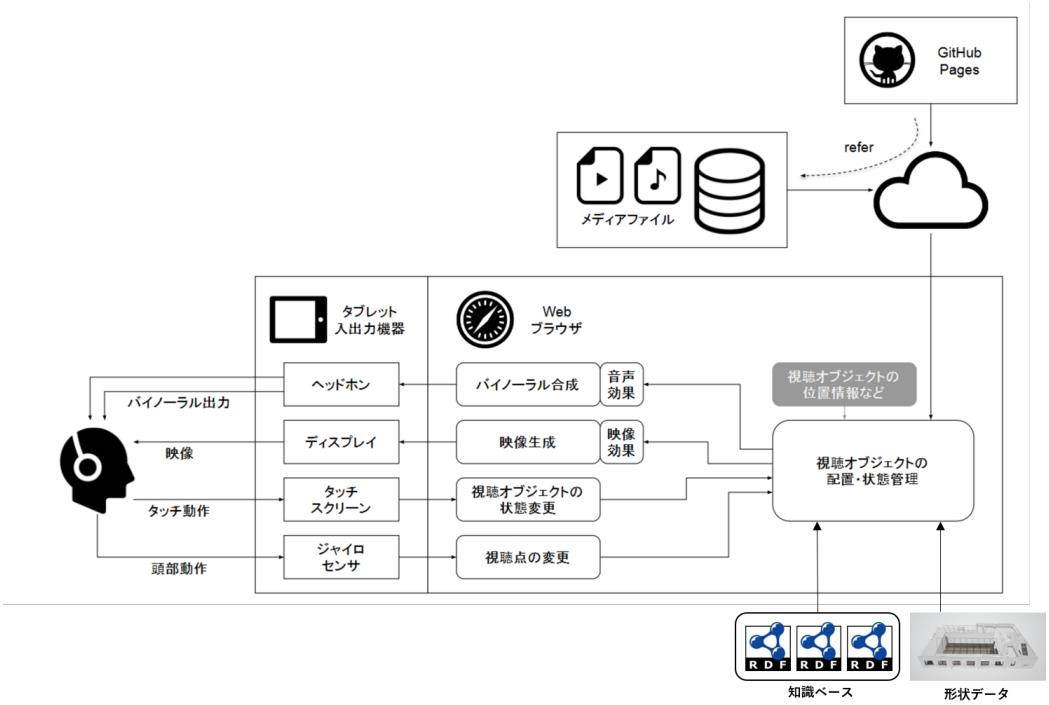


図 4.22: web360^2 のシステム構成

図 4.22 に今回計画したシステム構成を示す．システムの初期化時に SDM オブジェクトの配置を行うが，ここに

[*20] https://github.com/sdm-wg/web360square

BIM から抽出した空間グラフの RDF と形状データを用いる．具体的には，空間グラフにおける bot:Element 要素に含まれる「iref_円形型スピーカー」を SDM オブジェクトの設置対象とする．

IfcOpenShell を使って抽出した形状データ (図 4.17) には，RDF において対応するリソースと同一の GUID が保存されている．そのため初期化時にアプリケーションで RDF から抽出した GUID を指定することで，位置情報のキャリブレーションが不要になる．

4.5.3 オントロジの検討と実装

SDM では，セマンティクスの記述に 2.2.2 節で述べた SDM オントロジを用いる．SDM オントロジでは SDM オブジェクトの位置などを記述する語彙などが存在する．例えば，sdmo:localX，sdmo:localY，sdmo:localZ で 3 次元位置を特定するが，BIM を用いた場合，GUID によって出力された形状位置を特定するのが合理的である．具体的には，図 4.23 に示す Linked Data を構築することで，それぞれの語彙を用いた記述が可能である．

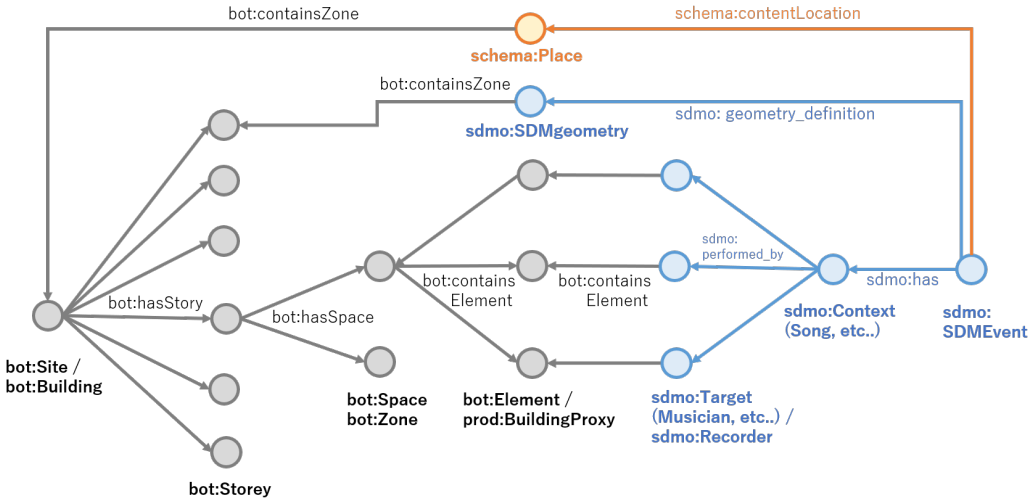


図 4.23: BOT と SDM オントロジによるリソース記述の融合

今回のシステムであれば，形状を意味する sdmo:SDMgeometry によってクラス定義されたリソースが，BOT の空間包含関係を表す bot:containsZone を用いて対象の空間リソース (bot:Space) を指定することで，形状データの指定が可能である．具体的には，bot:has3DModel の語彙を用いることで，形状データのファイルパスを指定したり，また bot:hasSimple3DModel によって RDF に直接テキストで形状データを埋め込んだりすることができる．

SDM オブジェクトを表すのは sdmo:Target であり，そこに今回対象とした 3 つの音源への配置が示される．具体的には sdmo:Target リソースが，空間内に配置されたスピーカー（bot:Element）への参照を持つことが，bot:containsElement で記述される．また，スピーカーを表すリソースには，IFC から抽出した GUID(ifc:GlobalID) が保存されており，形状データに保存された GUID との突合が可能である（図 4.24）．

これらが記述された RDF を初期化時に知識ベース (SPARQL Endpoint) 等から読み込み，記述に基づいてスピーカー位置を特定し，その GUID に合致した仮想空間上のオブジェクト位置に SDM オブジェクトを配置する．

本システムを再構築するにあたって，ソースコードに手を入れたのは初期化時の形状データ配置と，RDF の読み込み，GUID の突合と SDM オブジェクトの位置情報の特定部分のみであり，軽微な追加のみで容易に実現することができた．図 4.25 に，アプリケーション再生時のキャプチャ画面を示す．Hilloby の隅に配置されているスピーカーに，演出を伴った SDM オブジェクトが正確に配置できていることが分かる．

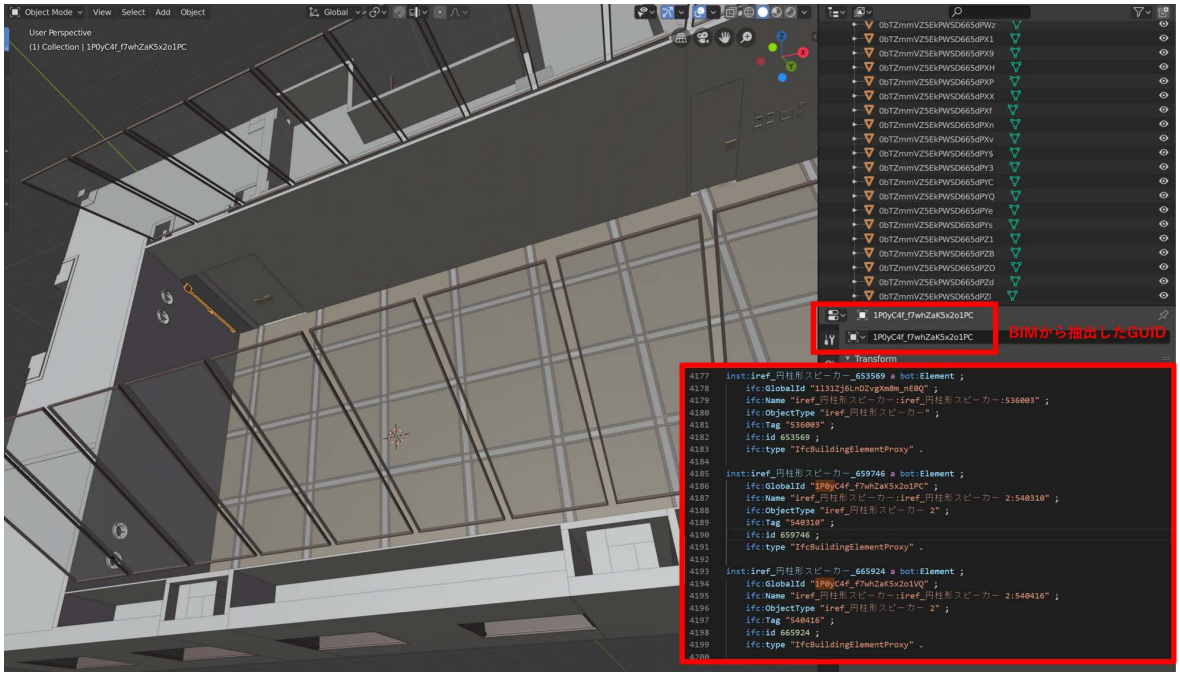


図 4.24: BIM から抽出した GUID と SDM オブジェクトのマッピング

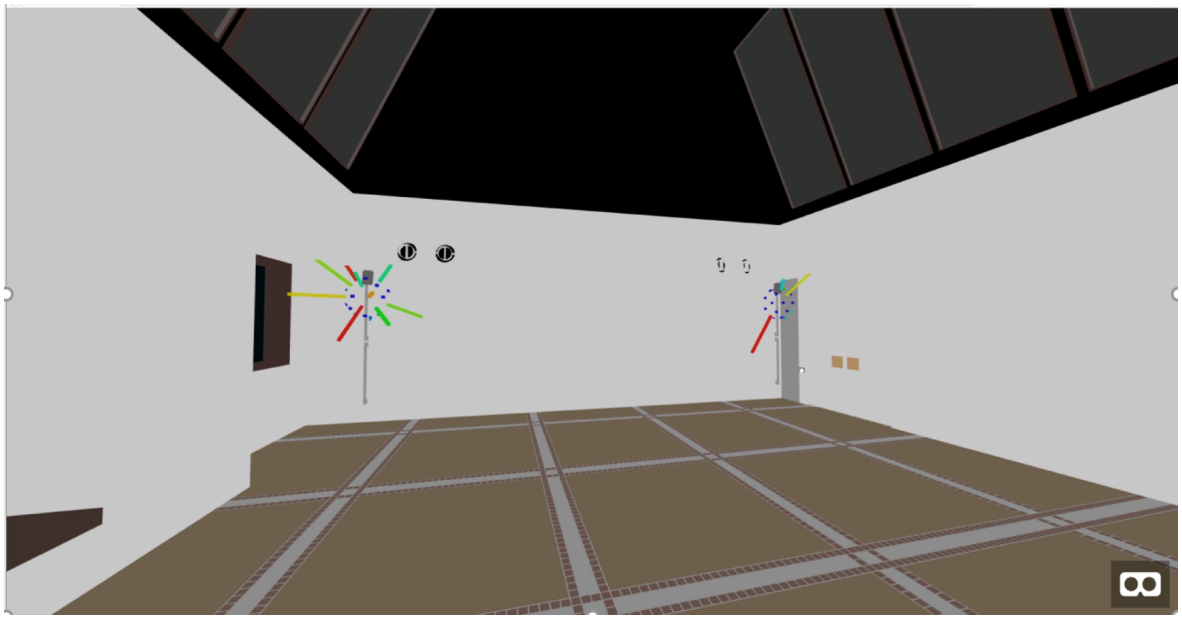


図 4.25: BIM を用いた web360[2]

4.6 まとめ

本章では，AEC/FM で用いられるオントロジを俯瞰し，スマートビルにおけるユースケースと BACS の構築プロセスを分析することで，スマートビルに必要となるセマンティクスを特定した．それらのセマンティクスを用いたデータモデル生成のために，BIM と BACS のポイントリストを用いてモデリングする手法について提案，評価を行った．

提案手法は，BIM の汎用的なデータ交換フォーマットである IFC から，空間構造やデバイスの包含関係などのセマンティクスを含む空間グラフを抽出し，そこに BACS のポイントリストを突合させることで，建物メタデータと呼ぶ統合 RDF を得る．突合方法としては，ポイントリストを元にした取込用 CSV に共通 ID を埋め込むことで実現できるが，合理的な手法として，BIM に属性として埋め込む手法を提案した．I-REF 棟や竹中工務店技術研究所で製作した設備用 BIM で同手法を適用し，期待する結果を得た．

また，BIM から抽出したセマンティクスを SDM アプリケーションである web360[2] に適用し，BIM から抽出した形状データの活用と、SDM オブジェクトの位置指定の自動化を実現した．これにより，スマートビルのアプリケーション以外への汎用性を確認することができた．6 章では，ここで特定したセマンティクスを用いたデータ・プラットフォームの構築について述べる．

第 5 章

エンターテインメントシステムの構築

本章では，VR を用いた立体音響システムを実現する SDM (Software Defined Media) のプラットフォームとアプリケーションについて述べる．建物の形状データを用いるデジタルツインの 1 つであり，4 章で述べたセマンティクスやメタデータの利用によって，アプリケーションの開発効率と汎用性の向上が期待できるアプリケーションである．

5.1 はじめに

インターネットからの配信される VR コンテンツの視聴が可能になってきたが，配置されている音源は静的なものがほとんどで，高品質なストリーミング再生は困難であった．こうした課題解決のため，我々は VR 技術とロスレス配信技術，2.2.2 節で述べた SDM オントロジやオブジェクトベースのオーディオを用いて，自由視点でのライブ体験を実現する「LiVRation」を試作・検証した．

以下では，5.2 節で関連研究について述べ，5.3 節で本研究の目的を述べる．5.4 節でシステムの設計，実装，コンテンツ収録についての述べ，5.5 節でそれらの評価と各所で実施したデモンストレーションの報告を行う．5.6 節で結論と今後の課題について述べる．

5.2 関連研究

著者らの先行研究 [109, 110] は，LiVRation と同様に仮想空間内での視聴位置に応じた音声ミックスを行う，タブレット端末での視聴を前提としたアプリケーションである．アンケートによる主観評価によると，仮想空間上に配置された音声オブジェクトの操作により，90 %以上の被験者が特定の音のみを聞くといった体験の有用性や，音声の立体感が感じられるという結果を得た．一方で映像の立体感やインタラクティブ性にかかわる設問については，比較的低い結果が出ていた．LiVRation では VR 技術を導入することで，インタラクティブ性の改善を図るとともに，コンテンツの配信機能とメタデータ記述を付加することで，インターネット配信時のへの親和性を向上し，コンテンツ配信のためのプラットフォーム化を目指した．

音響の録音・再生システムは，チャンネルベース，オブジェクトベース，シーンベースの 3 つに大別されることがある [66]．チャンネルベースのシステムは，ステレオサウンド（2 チャンネル）から始まりサラウンドサウンド（多チャンネル）へと発展し，2016 年に試験放送の始まったスーパーハイビジョン [58] では，22.2 マルチチャンネルの立体音響システム [125] を採用している．チャンネルベースのシステムでは，収録においては一般的なマイクを利用できる利点があるが，最終的に出力する音声の情報をそのまま記録するため，再生環境に合わせたチャンネル数の音声情報を記録しておく必要がある．また，最終出力の形で音声データを記録するため，視聴者の動きに追従した音声の提示することは難しい．

オブジェクトベースのシステムは，音源の音色のデータとその 3 次元の位置をメタデータとして記録し，再生環境

においてスピーカの位置から音場をレンダリングする方式である．例えば，映画館やホームシアターでの採用が進む Dolby Atmos [2] や，AuroMax [1] などがある．また，オブジェクトベースの方式は，国際標準化機構（ISO）と国際電気標準会議（IEC）の Moving Picture Experts Group (MPEG) において，MPEG-H [15, 36] の標準化が進んでいる．オブジェクトベースの方式では，音源の位置を記録する必要があるが，収録に使うマイクは一般的なものを利用できる利点がある．また，再生する音源と視聴者の相対的な位置関係から音場を計算できるため，視聴者の位置移動や頭部の回転に追従した音声の提示が可能である．

シーンベースのシステムは，ある受音点に到来する音を指向性を持った複数のマイクを組み合わせて，全周 360 度で空間の音全体を録音し，到来する音の方向を再現する技術である．アンビソニックス (Ambisonics) [34] の収音では，アンビソニックマイクという特殊なマイクを利用し，収音したデータは B-フォーマットと呼ばれる信号として記録される．このデータをもとに視聴者の聴取位置における，頭部の回転に追従した音声の提示が可能である．ただし，視聴者の位置移動に追従した音声の提示は難しい．Ricoh Theta V など近年発売された民生用収録機器にもアンビソニックマイクが搭載される例も多く，こうして収録された収録データは YouTube や Facebook の動画共有サイトでも再生に対応している．さらに，アンビソニックでは再現が難しい複数の音源から到来する音を正確に再現する高次アンビソニックス (Higher-order Ambisonics, HOA) [62] の研究も行われている．

映像音声の遠隔配信は IP ネットワークの高速化に伴い，4K60p 映像を複数同時に配信する実証実験が開始されている．高精度な映像音声の双方向での遠隔配信を行うことで，遠隔コラボレーションシステムを構築すると，立体感や情感など，臨場感を高めることができる [84]．さらに，同様の技術を利用してディジタルシネマ劇場へのライブ配信することで，劇場をパブリックビューイングの会場として転用することが可能になる．これにより，同じく高精度化するホームシアターから劇場を差別化し，劇場施設の提供できる価値を大きく向上させると考えられ，ビジネス的な展開が期待されており，商用実証実験が行われた [113].

遠隔のユーザ同士が快適に会話し，効率的に協調作業するためのコミュニュケーションシステムの開発が進んでいる．NTT コミュニケーション科学基礎研究所では，同じ部屋にいる感覚を同室感と名付け，同室感コミュニケーションシステム「t-Room」を開発した [37][122]．t-Room では，ユーザ全員が周囲の音や映像に関して同じ認識や知覚を対称的に共有することを目指し，同じ大きさの部屋に複数の背面スクリーンを囲い込んで設置している．

バーチャルリアリティの視覚ディスプレイとしては CAVE [19] や多面型全天周ディスプレイ (CABIN)[124] を始めとする没入型多面ディスプレイの研究が行われて来た．こうした 4 面の壁と地面を含めた没入多面ディスプレイは HMD とは違い，その場にいる多人数のユーザに同時に仮想世界の体験を提供できる．したがって，こうしたディスプレイに追従する音響の提供はヘッドフォンではなく，複数のスピーカによるインタラクティブ高臨場感音場提示手法が検討されて来た [97]．また，NHK 放送技術研究所は，2 眼立体の 3 次元コンピューターグラフィックス再生と，映像に追従してインタラクティブに 3 次元の音場をスピーカアレイを用いて再生するシステム [104] を組み合わせインタラクティブ 3D 映像音響再生システムを開発した [105].

インターネット上で構造化されたデータを公開する手法としては，RDF (Resource Description Framework) が推奨されている．RDF は，データ（リソース）の関係を主語，述語，目的語という 3 つの要素（トリプル）で表現し，それらを連結させていくことで意味表現を拡充させていく．RDF によって記述されたリソースは IRI (Internationalized Resource Identifier) によって参照可能であり，それらの記述・語彙のルールであるオントロジを利用することで，コンピュータによる自動処理に適した形式となる．この RDF が外部と連携し，相互リンク可能になったものは LOD (Linked Open Data) と呼ばれ，一般的にはインターネット上に配置された SPARQL Endpoint と呼ばれるサーバに公開する．SPARQL Endpoint は SPARQL クエリを処理して，RDF 等でデータを返すサービスであり，Virtuoso[*1]や GraphDB[*2]など，OSS の実装も多く存在する．

[*1] https://virtuoso.openlinksw.com/
[*2] https://www.ontotext.com/products/graphdb/

RDF や LOD は，現在例えば，人文社会系大規模データベース [89] や，ノックアウトマウスの表現型のデータベース [24] に使われ，データの横断的な利用を促進している．さらに，データ間の関係を示す語彙やデータの種類（クラス）を表す語彙は RDF Vocabulary と呼び，音楽データを記述する RDF Vocabulary は Music ontology [63] として定義されて，英国放送協会（BBC）などで広く利用されている．

以上，立体音響，映像伝送，VR，オントロジの研究を概観したが，LOD によってインターネットからの映像・音響データのメタデータ参照を可能とし，かつ立体音響による VR コンテンツ配信のプラットフォームを目指した研究はこれまでなかった．

5.3 本研究の目的

本研究の目的は，音楽イベントの遠隔配信を対象とし，自由視点映像音声のインタラクティブ再生を行うアプリケーションとそのためのプラットフォーム構築である．具体的には，収録したライブを自由視点で視聴するとともに，収録された音声を自由にコントロールし，かつ演奏者や楽器のメタデータ，ソーシャルメディアなどで発信されるコンテンツについての情報も動的に扱うことができる SDM アプリケーションである．そのために，我々は LiVRation[47, 120] を試作し評価を行った．システム要件として以下を設定した．

3 次元の映像・音声の演出をソフトウェアで制御： 視聴オブジェクトを 3 次元表現を持った情報空間上で管理しながら，ソフトウェアの演出によって再生環境に適した，またはユーザがカスタマイズした映像と音声をソフトウェアレンダリングによって作り出すことができる．

自由な視聴体験： 利用者が視聴位置を自身で決めることができる．位置に基づいて，収録音声がリアルタイムで自動合成され，ヘッドフォン等で視聴することができる．

遠隔配信： ライブをリアルタイム配信することが可能であり，再生環境を整えれば，ネットワークを介してどこでも視聴することができる．収録済みの音源に対しても，同様に配信することができる．

高臨場感・没入感： 4K やロスレスなどの高音質な映像・音声を扱うことができる．より臨場感の高いライブ体験をするために，VR や振動伝達（ハプティクス）を利用する．

インタラクティブな体験： コントローラを使って特定の音を強調したり，不要な音を消したりすることができる．ソーシャルメディアと連携して，コメントなどを共有することができる．

上記を実現するため，我々は収録した音源をオブジェクトオーディオ化（SDM オブジェクト化）し，多拠点映像とともに配信する仕組みを構築するとともに，SDM オントロジを使って，それぞれの SDM オブジェクトの情報を定義した．それらを受信し，VR デバイスで再生する LiVRation の詳細について，以下で述べる．

5.4 LiVRation

本節では LiVRation で用いた収録コンテンツの詳細と，システムの設計指針，実装の詳細について述べる．なお，LiVRation は CiP 協議会[*3]が主催しているハッカソンイベントである「Billboard LIVE MUSIC HACKASONG 2017」のために作成されたシステムで，「Live で Vibration を伝える VR 配信」というコンセプトをもとに開発されている．

[*3] https://takeshiba.org/

5.4.1　システム概要

LiVRation は HMD を使った没入環境において立体音響環境を体験することができる．また，MPEG4-ALS を使ったハイレゾ音声のストリーミング再生を実現している．視聴者は，HMD を用いて仮想空間を自由に動き回るとともに，視聴位置での音響空間を仮想的に体験することが可能であり，特定の楽器・音源のみの抽出や，Twitter からの情報表示，振動伝達デバイスからの振動を感じることもできる．図 5.1 に動作画面を示す．画面上に表示されている球状のオブジェクトに収録された 360 度動画 (Video Objects) や音声 (Audio Objects) がマッピングされている．また，視聴者を囲うように Twitter のコメント (Twitter 3D barrage) が表示されている．



図 5.1: LiVRation の動作画面

以下では，コンテンツの収録環境，システムの設計指針，実装について詳細を述べる．

5.4.2　収録環境

LiVRation で利用するコンテンツは，2018 年 1 月 30 日に Billboard Live Tokyo で行われた「仮谷せいら」のライブリハーサルである．リハーサル中に収録した映像，音声を配信コンテンツとして完成させ，収録から約 4 時間後にはイベント本番での最終発表を行った．

図 5.2 に，バンドの編成，360 度カメラの設置場所，マイクの設置場所を示す．ライブの構成は，ボーカルにシンガーソングライターの「仮谷せいら」で，バックバンドにドラムを担当する松浦と，キーボード，ベースを担当する川原という構成であった．会場におけるリスナーへの表現手法としては，全ての楽器にマイク或いは電気信号を増幅する機材を接続し，Sound Reinforcement（SR）用ミキシングコンソールで調整されたのちに大規模なスピーカで再生される．音圧レベルは場所にもよるがおおよそ 100dB SPL となる．

舞台内のマイク構成については，楽器単体の他にオーディエンス用のマイクを 4 本設置することとした．音源より近

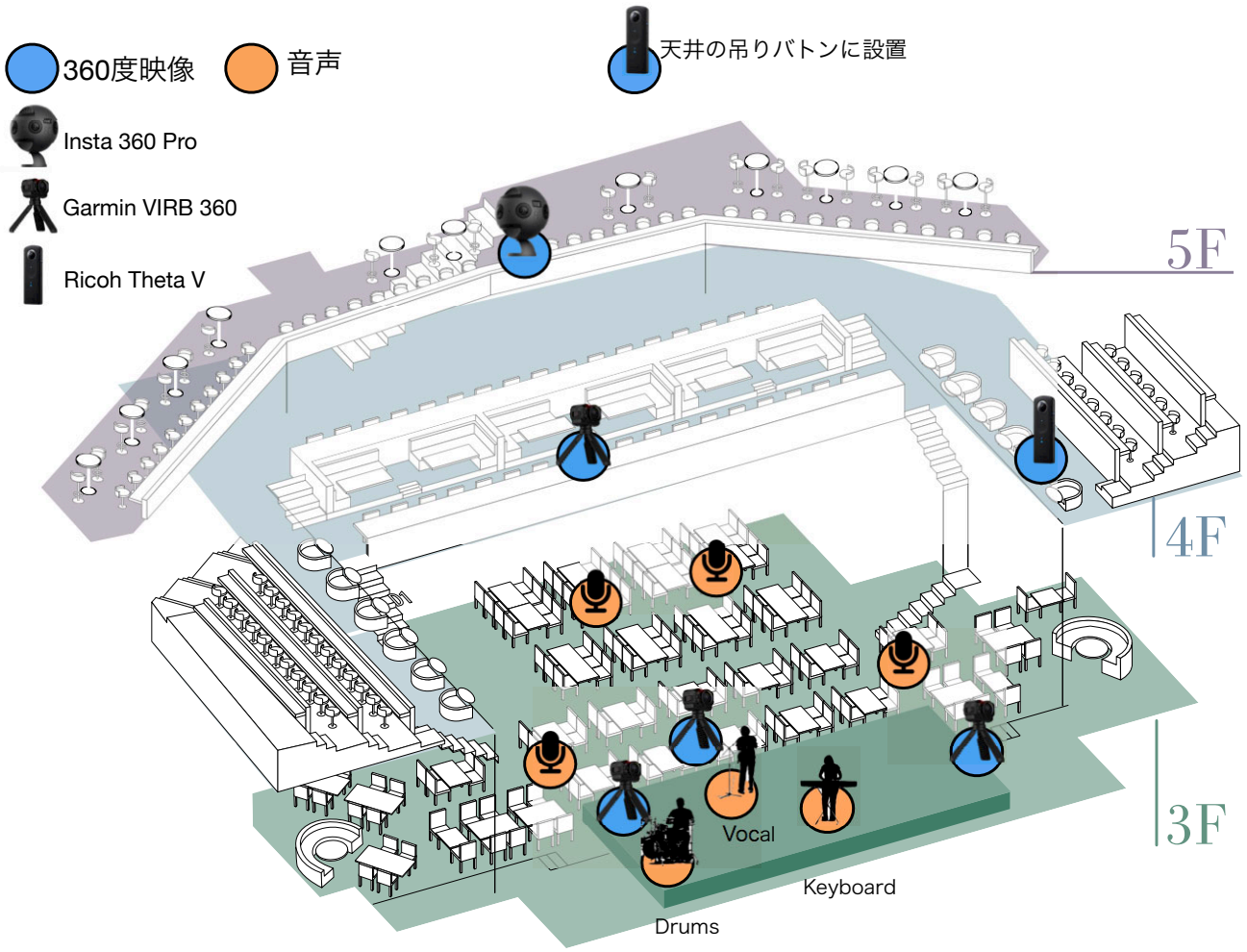


図 5.2: ビルボードライブ収録 (2018) のカメラとマイクの配置

いポジションとしてステージ袖の上手，下手に 1 本ずつ，もう 2 本は天井から無指向性のマイク（DPA4090）を 2 本吊り下げた構成とした．音源より近いマイクはステージ角の位置，高さ 1 mに設置した．音源より遠いマイクについてはステージから奥に 5 m，高さ 5 mのところにセンターを中心に約 1m の幅で 2 本設置した．収録音声は全て SR 用ミキシングコンソールに纏められおり，コンソールの Head Amp 分岐をデジタルで Digital Audio Workstation (DAW) に転送するシステムとなっている．録音される音量レベルは全てミキシングコンソールの設定に依存する為，後に整音作業が必須となる．録音したデータを編集用の DAW にて，それぞれボーカルと楽器ごとの単体で纏め上げた音源（VocalMix，DrumsMix，BassMix，KeyboardMix）に編集し，さらに観客席の左奥，右奥，左手前，右手前のマイクによるアンビエンス Mix を編集した．

映像を収録する 360 度カメラは，「Insta 360 Pro」1 台，「Garmin VIRB 360」4 台，「Ricoh Theta V」2 台を利用し，全てのカメラで 3840x1920@30fps (4K) での撮影を行った．Insta 360 Pro は，6 枚の F2.4 魚眼レンズを備えたハイエンド 360 度カメラであり，ライブ会場の全景を撮影するため，5 階席に設置した．Garmin VIRB 360 は，2 枚のレンズを備えた 360 度カメラであり，設置場所は，ハッカソンの審査員席に利用された 4 階正面，ボーカルの真前，ドラムの真前，ステージ右横の 4 カ所を選択した．Ricoh Theta V は，2 枚のレンズを備えた 360 度カメラであり，もっとも軽いため，ライブ会場の天井に吊ってあるバトンに 1 台設置した．また，もう一台を 4 階席右手に設置し撮影を行った．

5.4.3　設計

LiVRation は 2.2 節，図 2.6 に示した SDM アーキテクチャ [111, 71] に基づいて設計を行った．SDM アーキテクチャは，ネットワーク上に存在するコンテンツおよびその配信システムと，実空間内の各種情報を SDM オブジェクトとして情報空間に取り組み，その結果を自在に視聴空間に投影する SDM 拠点から構成される．

LiVRation も SDM アーキテクチャに準拠した設計となっている．図 5.3 に LiVRation の設計概要を示す．

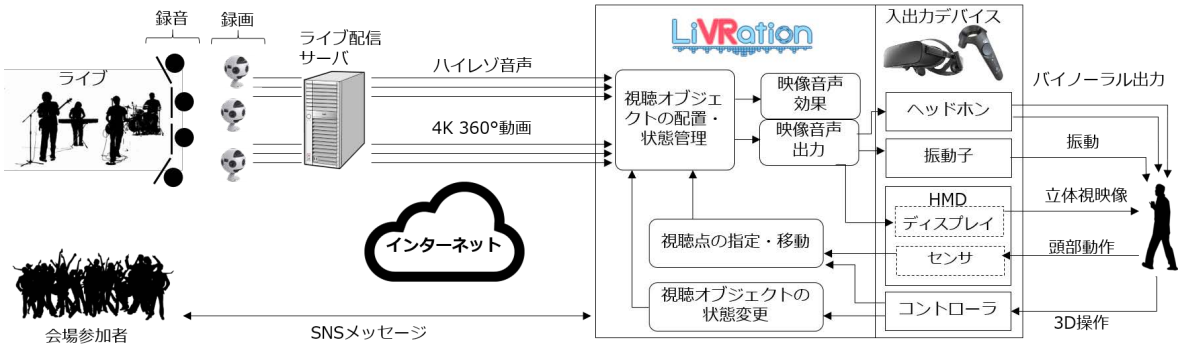


図 5.3: LiVRation 設計概要

前節で述べた収録済みの映像音声は，対象となる 1 曲を切り出して編集され，ライブ配信サーバに格納する．ライブ配信サーバからネットワークを通じて，ハイレゾ音声と 4K 360 度動画を SDM 拠点である LiVRation クライアントへとストリーミング配信する．クライアントでは，これらの映像音声ストリームを音声，動画を表す SDM オブジェクトとして扱い，図 5.4 のようにあらかじめ製作しておいた仮想空間内に配置し，全てのストリームのバッファが一定量蓄積したのちに再生を行う．

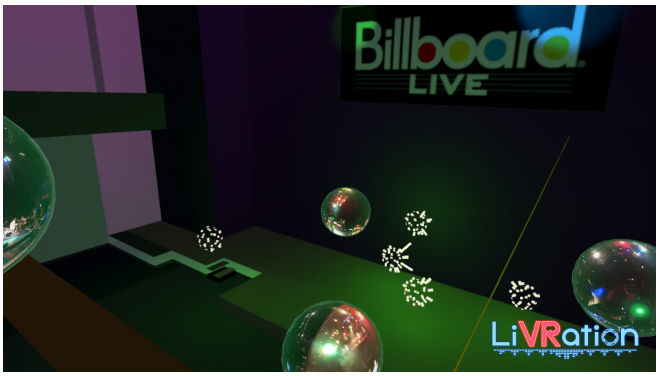


図 5.4: 仮想空間への SDM オブジェクトの配置



図 5.5: 音声オブジェクト操作（音量操作）

音声は，音声オブジェクトと視聴位置との距離関係から，音量と聞こえる方向を決定し，ヘッドフォンからバイノーラル出力される．また，ヘッドフォンジャックからローパスフィルタを通して低音成分に反応する振動子を接続し，身体に装着することでより臨場感の高い体験を提供する．さらに，音声は周波数成分の音量ごとに異なるバーの長さが変化する映像効果を付与することで，可視化を行う．

360 度動画は球体の両面に貼り付け再生を行い，これらの動画や映像効果は HMD の両眼映像を通じてユーザに立体的な映像を提示する．ユーザは，HMD のセンサを通じて頭部の動きを LiVRation クライアントに伝える．これにより自由に視聴したい方向を向く事が可能であり，3 次元の動きを取得できるコントローラを 3D 操作することで，様々

な意図をインタラクティブにシステムに伝える事ができる．

ユーザはコントローラを使って，音声の制御や視聴位置の移動などを行うことができる．具体的には，コントローラで音声を掴んだ状態で，自分の方に引く操作により，その音声だけをソロで視聴すること可能であり，歌声をアカペラで聞いたり，楽器の音をインストルメンタルで聞いたりすることができる．反対に音声を掴んだ状態で，向こう側に押し込む動作で，全ての音声を有効化して通常通り音声が混ざった状態で音声を聞く事ができる．加えて，音声を掴んだ状態で手首をひねると音量の強弱を調整できる（図 5.5）．また，コントローラから照射されるポインタを映像を提示している球体に当てた状態で，選択するとその場所への移動が可能である．

近年，ライブ放送において一体感を味わうため，Twitter などの SNS を通じて，他の視聴者とインタラクションを行う事で，ライブを楽しむ事が一般化している．LiVRation では，Twitter の関連ツイートを仮想空間内に提示する事で，こうした他の視聴者とのインタラクションを実現する．

5.4.4 実装

上記の設計指針に基づき，表 5.1 に示す開発環境，フレームワークを用いて LiVRation を構築した．図 5.6 にシステムの実装概要を示す．

表 5.1: LiVRation 実行環境

実行マシン	CPU： メモリ： グラフィックス： SSD：	Intel®Core-i7-8700K 32.0 GB GeoForce®GTX 1080 Ti 480GB
HMD		Samsung HMD Odyssey
クライアント開発・実行環境		Unity 2017.2.0f3(64-bit)
配信サーバ		Wowza Streaming Engine 4.7.5
サービス		NodeRED v0.17.5
SPARQL Endpoint		graphdb-free-8.3.0
時系列データベース		Elasticsearch v5.6.2

コンテンツ配信の仕組みとしては，配信サーバである Wowza[*4]に mp4 にエンコードした収録動画，音声を配置することで実現している．それぞれの配信コンテンツには URL が自動的に付与され，MPEG-DASH 等で配信することができる．なお，Wowza は MPEG4-ALS にも対応している．

これらのストリーミングデータを Unity[*5]で実装した LiVRation クライアントが受信する構成となっている．構築環境としては，今回の実装では Wowza はローカルネットワーク内に配置し，そこからクライアントがストリーミング配信を受ける構成とした．技術的にはインターネット上からの配信も可能ではあったが，今回デモンストレーションを行ったライブ会場等では一定品質の通信環境の調達が難しかったためにローカルネットワークの構成とした．また，インターネット上から配信する場合，コンテンツ調達のための認証等の仕組み構築が課題になると考えらえるが，本研究では今後の課題とした．

LiVRation クライアントからストリーミングデータ以外を取得するサービスは，NodeRED[*6]を使って，HTTP のサービスとして実装した．具体的には，NodeRED の Twitter API を使って配信コンテンツに関係する Tweet を収集するモジュールと，クライアントの初期化時に SDM オントロジを格納している GraphDB[*7]に対して SPARQL クエ

[*4] https://www.wowza.com/products
[*5] https://unity.com/ja
[*6] https://nodered.org/
[*7] http://graphdb.ontotext.com/free/devhub/sparql.html

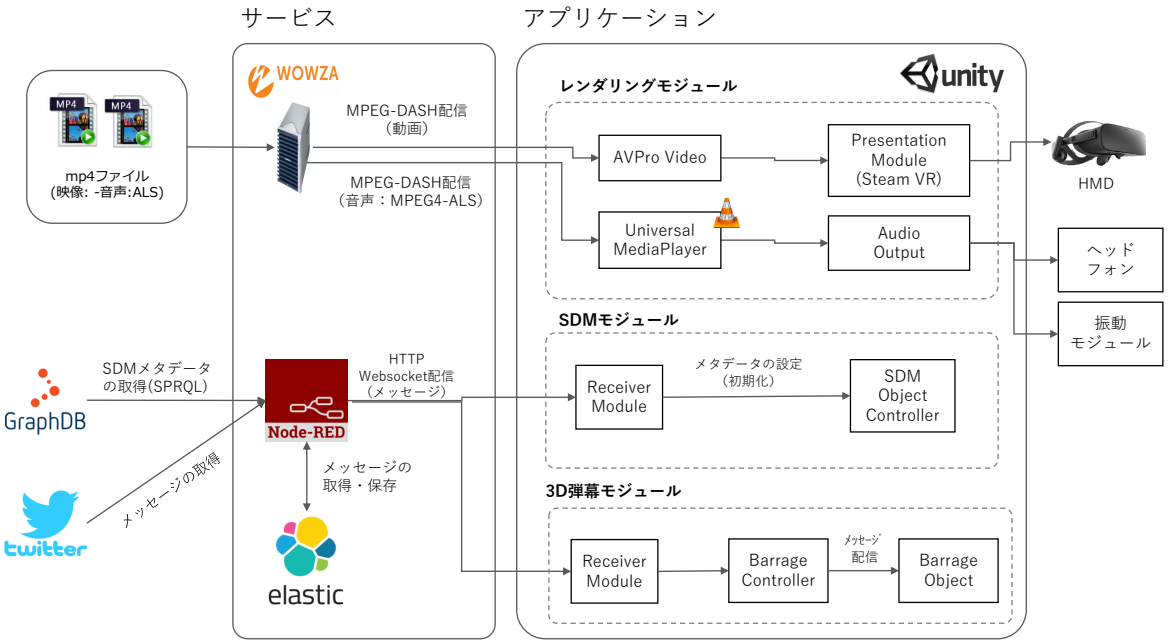


図 5.6: LiVRation 実装概要

リを投げ，その結果を LiVRation クライアントに送信する機能を持つ．前者については，例えば「仮谷せいら」といったキーワードを予め設定しておくと，それに対応したメッセージが websocket で継続的にクライアントに送信される．過去データについては，時系列データベース（Elasticsearch[*8]）に保存され，初期化時に送られる実装となっている．

LiVRation クライアントには，コンテンツの受信や映像・音声のエンコード，レンダリングするモジュールをや各種サービスの情報を受けて画面内に反映，表示する機能モジュールを配置している．具体的には，以下のモジュールを有する．

レンダリングモジュール:　Wowza からのストリーミングデータは Unity のプラグインによって処理されて表示される．動画は AVPro Video プラグインによって球体オブジェクトのテクスチャにマッピングされ，360 度動画として再生される．この際，360 度カメラで収録した音声も同時に取得できるが，MPEG4-ALS のストリーミングデータを利用するためにミュートしている．音声は UniversalMediaPlayer プラグインを使って音声オブジェクトにマッピングされ，周波数成分の音量によって変化するエフェクトが加えられる．UniversalMediaPlayer は VLC と FFmpeg をベースとしたプラグインであり，MPEG4-ALS の再生も可能である． エフェクトについては Rhythm Visualizator Pro プラグインを用いて実現している．前節で述べた距離による音声ミックスは，Unity によって自動計算され，出力された音声はヘッドフォンや振動デバイス [99] で再生することができる．なお，動画・音声それぞれに対して，現状では特定の同期処理を行っていない．動画は配信コンテンツがプラグイン内ですべて再生可能となった時点で再生される．音声は音声オブジェクトが個別に再生可能と判断された時点で再生が自動再生される．再生のタイミングについては，UniversalMediaPlayer プラグインのバッファに関するパラメータによって多少は調整することが可能であるが，Wowza からの配信タイミングに依存するので，映像との完全な同期は Unity だけでははは実装が難しい．これらの高度な同期処理については今後の課題であるが，

[*8] https://www.elastic.co/jp/elasticsearch/

ローカル環境での再生においては，ネットワーク速度が十分であるため，再生のズレが気になるようなことはほとんどない．

上記に加えて，HMD のセンサやコントローラの操作に応じて，視聴位置である球状オブジェクトの中を旋回・移動したり，音声オブジェクトへの操作によって音量を調節する機能を有する．

SDM モジュール: SDM オブジェクトのメタデータ（音源，演奏者，位置情報など）は，クラウド上に構築された SPARQL endpoint である GraphDB から，サービス層を介して LiVRation クライアントに取得される．クライアントの初期化時に Unity から，サービスプラットフォームに対して HTTP でアクセスすると，予め用意された SPARQL クエリで GraphDB に問合せを行い，その結果がクライアントに送られる．そこで取得したメタデータをレンダリングモジュール等に渡す実装となっている．今回実装した LiVRation のために定義した RDF とそのリソース構成（図 5.7，図 5.8）を以下に示す．

図 5.7: LiVRation の RDF によるリソース記述（抜粋）

```
@prefix sdmo: <http://sdm.hongo.wide.ad.jp/sdmo/> .
@prefix sdm: <http://sdm.hongo.wide.ad.jp/resource/> .

###  Define SDMEvent
sdm:hackasong rdf:type sdmo:SDMEvent ;
      s:contentLocation sdm:billboard ;
      s:name "Billboard LIVE HACKASONG 2017"@ja ;
      s:startDate "2018/1/30 17:30" ;
      sdmo:has sdm:song1 ;
      sdmo:recording sdm:bass_mic ,
                     sdm:drum_mic ,
                     sdm:far_left_mic ,
                     sdm:far_right_mic ,
                     sdm:keyboard_mic ,
                     sdm:near_left_mic ,
                     sdm:near_right_mic ,
                     sdm:vocal_mic .

###  Define Context
sdm:song1 rdf:type sdmo:Song ;
      s:name "Colorful World" ;
      sdmo:performed_by sdm:bass ,
                        sdm:drum ,
                        sdm:keyboard ,
                        sdm:vocal ;
      sdmo:program_number "1" ;
      sdmo:recorded_by sdm:bass_mix ,
                       sdm:drum_mix ,
                       sdm:far_left_mix ,
                       sdm:far_right_mix ,
                       sdm:keyboard_mix ,
                       sdm:near_left_mix ,
                       sdm:near_right_mix ,
                       sdm:vocal_mix .

###  Define Target
sdm:vocal rdf:type sdmo:Musician ;
        s:name "仮谷せいら"@ja ;
        sdmo:attend_to sdm:hackasong ;
        sdmo:localX "0.0"^^xsd:float ;
        sdmo:localY "0.5"^^xsd:float ;
        sdmo:localZ "4.5"^^xsd:float ;
        sdmo:perform sdm:song1 .
```

SDMEvent である sdm:hackasong や Context である sdm:song1 を中心に sdmo:performed_by でリンクされる演奏者（Musican クラス），sdmo:recorded_by でリンクされる音源（AudioRecorder クラス）がある．演奏者である sdm:vocal には音源を設置するそれぞれの座標位置 (sdmo:localX〜Z) が記述されている．同様に音源に対しては，メディア配信用の URL が記述される．

3D 段幕モジュール: Twitter 等のソーシャルメディアから取得したライブに関するコメントなど，3D コンテンツ上

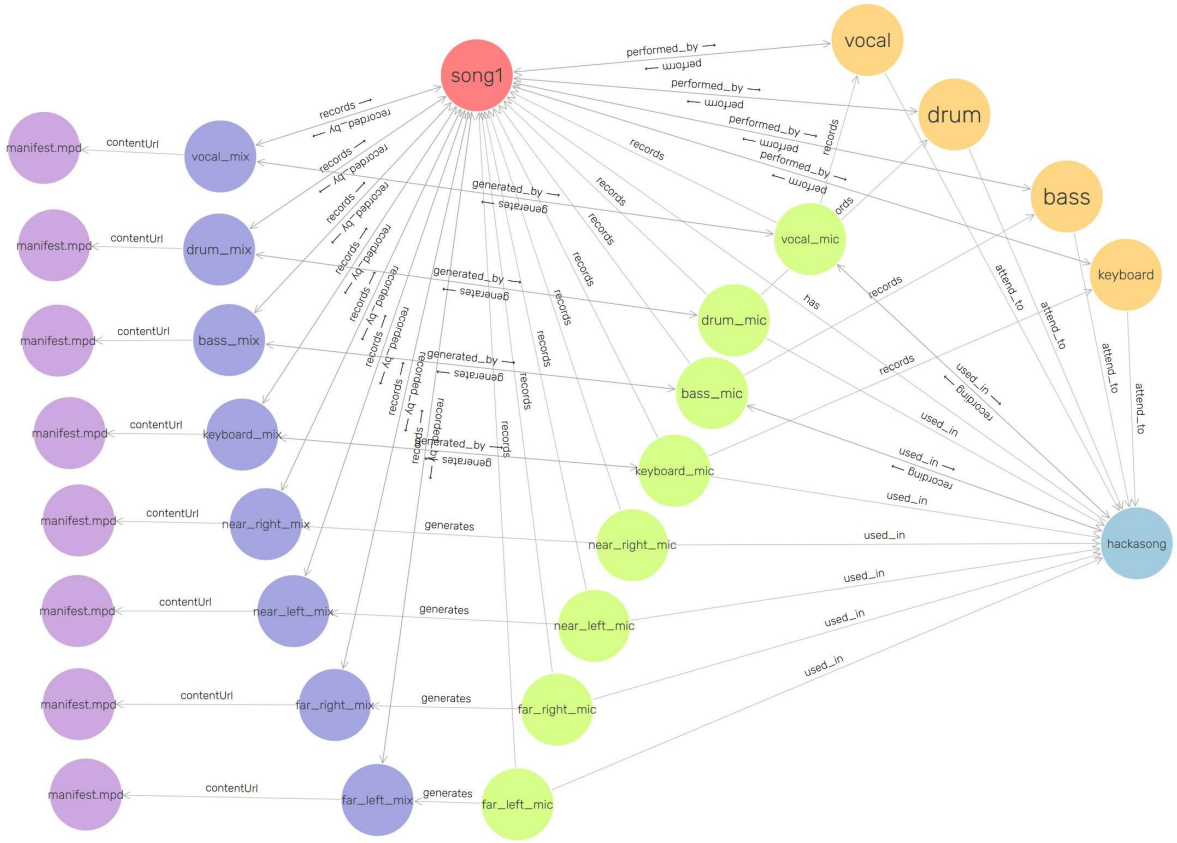


図 5.8: LiVRation のインスタンス構成

に表示するためのモジュールで，Twitter からのタグ付きコメントやプロフィールイメージなどを取得・取得することができる．SDM モジュールと同様に，アプリケーションの初期化時にサービスプラットフォームの特定の URI にアクセスすることで，websocket によって継続的にデータが送られてくる実装となっている．視聴位置の周りをアイコン化された Twitter メッセージが，回りながら次々と表示される演出となっている．なお，弾幕の表示はコントローラで制御（表示／非表示）することが可能である．

5.5 評価

LiVRation についてコンテンツ配信含めたネットワーク評価と主観評価を行った．加えて，Billboard LIVE MUSIC HACKASONG におけるデモンストレーションの様子について述べる．

5.5.1 ネットワーク性能評価

今回のコンテンツでは，8 つの 360 度動画と 7 つの音声をそれぞれストリーム配信しており，コンテンツ内で同時再生している．動画と音声の再生時間は 2 分 14 秒で，データ容量はそれぞれ，160M バイト，5M バイト程度であった．コンテンツ開始時からキャプチャしたネットワークの状態を図 5.9 に示す．

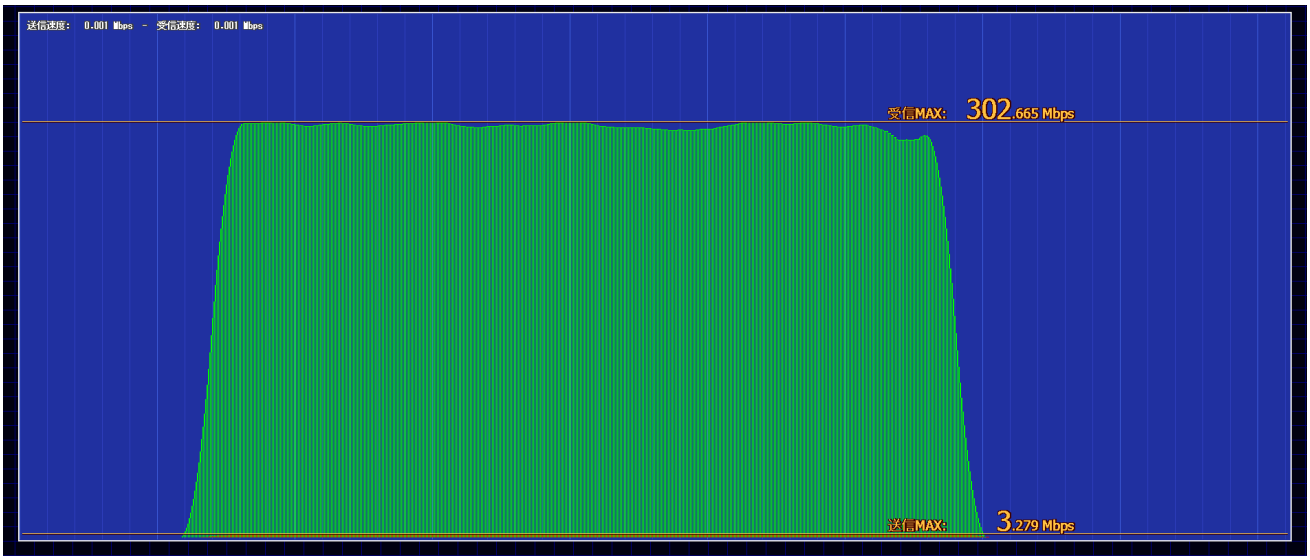


図 5.9: 起動時のトラフィック計測

ローカルネットワークでの配信環境では，概ね 300Mbsp 程度の受信速度であった．それぞれの Unity プラグインの仕様だと考えられるが，動画については最初にコンテンツを全てダウンロードしており，ダウンロードが終わると，音声ストリームのみの 15Mbps 程度となる．音声ストリームについては，コンテンツの終わりまで継続的にデータを受信しているのが確認できる．なお，インターネット上の配信サーバを指定した場合，最大で 100Mbps 程度の速度であり，動画の再生に 30 秒程度かかった．中庭らの研究 [107] によると，家庭までの 4K 映像をネットワーク配信する場合，ビットレートは 100Mbps 以下に設定する必要があるとあり，ハードウェアやプレーヤーのスペックに依存するものの，主観評価では 50Mbps であれば視聴者は十分に満足できるとある．将来的に，動画についても，キャッシュせずにリアルタイム配信できることを考えると，音声ストリームが 15Mbps であり，差分の 75Mbps 程度は動画に充てることができるといえるため，十分に高品質なコンテンツを配信できる可能性がある．しかしながら，家庭においては，一般的にベストエフォート型の回線であることに加え，外乱も多いと考えられるため，ローカルネットワーク環境内にキャッシュ用のサーバやミドルウェアが必要と考えられる．

5.5.2　主観評価

アンケート収集

2018 年 5 月 13 日〜15 日に行われた Interop Tokyo 2018[*9]の展示ブースにおいて，LiVRation の体験者を対象にアンケートを使った主観評価を実施した．評価においては，事前に体験者に LiVRation の使い方を提示し，自由に体験をしていただいた上で，体験後にアンケートに記載いただいている．体験者の総数は 211 名で，内訳としては男性 181 名，女性 23 名であった．年齢別の内訳は，10 代 12 名，20 代 74 名，30 代 50 名，40 代 48 名，50 代 24 名，60 代 2 名である．また，映像音響の専門家は 21 名，非専門家は 163 名であった（27 名は無回答）．なお，Interop Tokyo 2018 の主催者によると，イベントの参加者は 143,806 名であり，主要な訪問者は情報システム，ネットワーク関連会社のエンジニアやセールス，研究者である．

アンケート項目

アンケート項目は，以下に示す設問 Q1 から Q7 までの 7 つを，それぞれ 1 から 7 までの 7 段階のリッカート尺度を用いて設定した．それぞれの回答に関して，最低の 1，中間の 4，最高の 7 の回答の目安を括弧内に記載した．

Q1　映像の立体感は感じられましたか？
Q2　音声の立体感は感じられましたか？
Q3　音声は映像の方角と比べて正しい位置で鳴っているように聞こえましたか？
Q4　映像が動いた時，音声も追従して動いたと感じられましたか？
Q5　インタラクティブな視聴体験の操作は簡単にできましたか？
Q6　音量可視化による音声オブジェクトの有効化・無効化は直感的でしたか？
Q7　音声オブジェクトを有効化・無効化することで個別の楽器の音色を聞くことができましたか？

設問 Q1 と Q2 は映像と音声の基本的な立体感を問う設問であり，設問 Q3 と Q4 はその組み合わせが正しく一致しているように知覚されるかを問う設問である．設問 Q3 では静止時の音声の聞こえる方角と位置を問い，Q4 では動いた時の映像と音声の追従性についての問いを設定した．設問 Q5 と Q6 はコントローラによるインタラクティブな視聴に関する問いであり，Q5 は全般的な操作の容易さについての問いで，Q6 では音声オブジェクトの可視化および操作についての問いを設定した．設問 Q7 は音声オブジェクトに分解された個別の音声オブジェクトの音色が聞けたかを問う．加えて，システム改修の際に要求する重要視する機能についての設問を設け，アンケートの末尾には，「感想・要望・その他」という自由記載項目を用意して，視聴体験のコメントを得た．

結果

主観評価の結果を図 5.10 に示す．X 軸は 1 から 7 までの 7 段階の回答の比率をパーセントで表し，棒グラフの位置は，尺度の中間である 4 を X 軸の 0 の中央に配置し，左に行くほど低評価，右に行くほど高評価という配置で描画した．

Q2 のみ高評価の割合が低いが，ほぼすべての項目について，6 または 7 という高評価となった．これだけでは傾向が見えないため，性別および年齢別の評価も実施した．なお，比較においては，全体の平均値とそれぞれの項目を比較するクロス集計による評価を行っており，設問については先行研究 [110] と同一である．結果としては，似た傾向を示しているが，先行研究で平均以下の比率が 20 %を超えていた設問 Q1，Q5 については改善がみられる．先行研究では，タブレットや指による操作であったのが，VR デバイスと専用コントローラーにより，没入感とインタラクティブ

[*9] https://www.interop.jp/

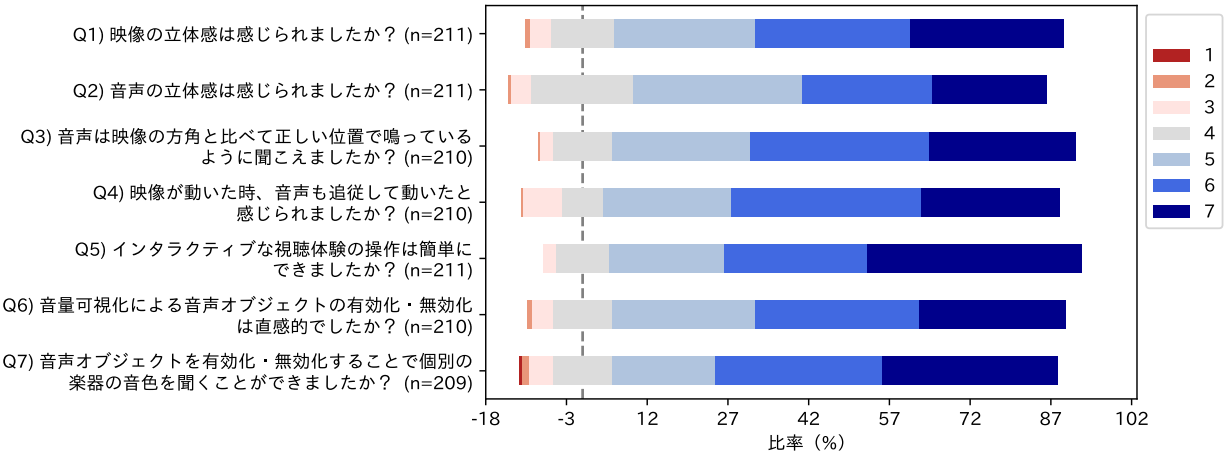
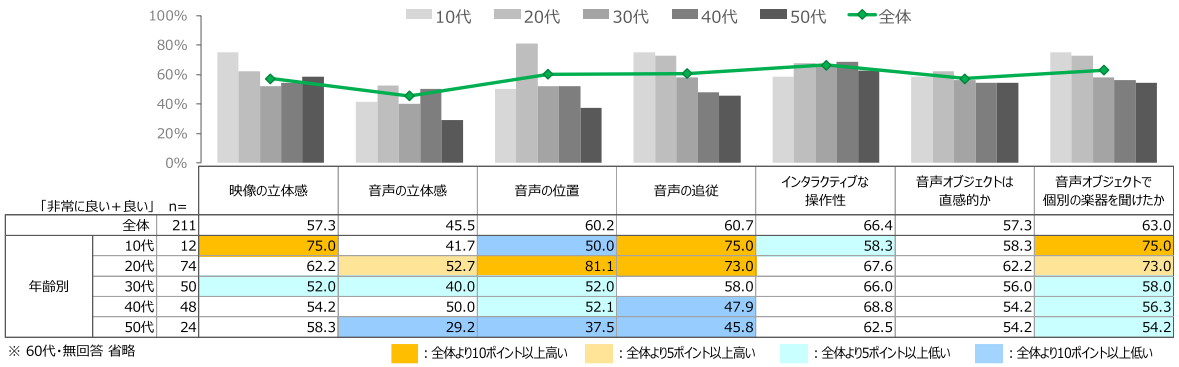


図 5.10: 主観評価の集計結果

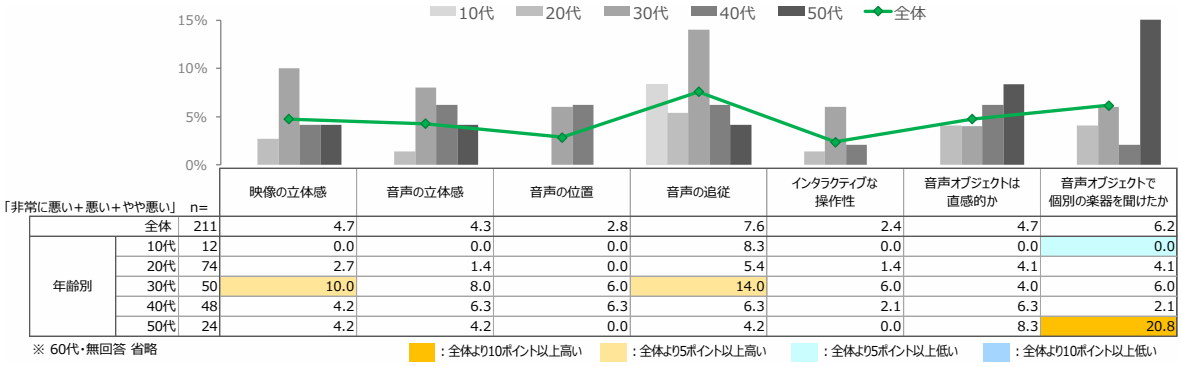
性が改善されたといえる．



「非常に良い＋良い」 n=			映像の立体感	音声の立体感	音声の位置	音声の追従	インタラクティブな操作性	音声オブジェクトは直感的か	音声オブジェクトで個別の楽器を聞けたか
全体		211	57.3	45.5	60.2	60.7	66.4	57.3	63.0
年齢別	10代	12	75.0	41.7	50.0	75.0	58.3	58.3	75.0
	20代	74	62.2	52.7	81.1	73.0	67.6	62.2	73.0
	30代	50	52.0	40.0	52.0	58.0	66.0	56.0	58.0
	40代	48	54.2	50.0	52.1	47.9	68.8	54.2	56.3
	50代	24	58.3	29.2	37.5	45.8	62.5	54.2	54.2

※ 60代・無回答 省略

図 5.11: 年代別分析（高評価のみ抽出）

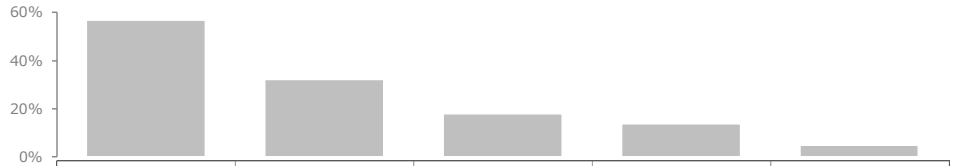


「非常に悪い＋悪い＋やや悪い」 n=			映像の立体感	音声の立体感	音声の位置	音声の追従	インタラクティブな操作性	音声オブジェクトは直感的か	音声オブジェクトで個別の楽器を聞けたか
全体		211	4.7	4.3	2.8	7.6	2.4	4.7	6.2
年齢別	10代	12	0.0	0.0	0.0	8.3	0.0	0.0	0.0
	20代	74	2.7	1.4	0.0	5.4	1.4	4.1	4.1
	30代	50	10.0	8.0	6.0	14.0	6.0	4.0	6.0
	40代	48	4.2	6.3	6.3	6.3	2.1	6.3	2.1
	50代	24	4.2	4.2	0.0	4.2	0.0	8.3	20.8

※ 60代・無回答 省略

図 5.12: 年代別分析（低評価のみ抽出）

図 5.11 は，年齢別に高評価のみを抽出したグラフである．10 代および 20 代について，Q4 および Q7 の割合が高いのが分かる．また，40 代，50 代については，音声の定位・追従に関わる Q2，Q3，Q4 の割合が低い．低評価（1 または 2）を抽出したグラフ（図 5.12）について分析をしてみると，50 代の Q7 に関する評価が低かった．より VR などに



(複数回答可)		n=	映像の高画質化	音声の高音質化	ネットワーク越しに他の人とチャットしながら見られる	タブレット等のVRゴーグル以外で見られる	無回答	%計
全体		211	56.9	32.2	18.0	13.7	4.7	125.6
性別	男性	181	55.8	32.0	18.2	13.8	5.0	124.9
	女性	23	56.5	39.1	21.7	17.4	0.0	134.8
年齢別	10代	12	50.0	25.0	41.7	0.0	0.0	116.7
	20代	74	60.8	36.5	23.0	10.8	2.7	133.8
	30代	50	56.0	24.0	12.0	20.0	4.0	116.0
	40代	48	56.3	39.6	16.7	12.5	4.2	129.2
	50代	24	45.8	29.2	8.3	20.8	16.7	120.8
	60以上	2	100.0	0.0	0.0	0.0	0.0	100.0
職業	学生	23	56.5	21.7	30.4	0.0	0.0	108.7
	教員	3	33.3	0.0	0.0	0.0	66.7	100.0
	社会人	180	57.2	33.3	17.2	16.1	4.4	128.3
専門性	専門家	21	47.6	47.6	19.0	14.3	4.8	133.3
	非専門家	163	57.1	30.1	18.4	14.1	4.3	123.9

※ 無回答の属性は省略　：全体より10ポイント以上高い　：全体より5ポイント以上高い　：全体より5ポイント以上低い　：全体より10ポイント以上低い

図 5.13: バージョンアップで一番重要な項目

親しんでいる若年層の方が，立体音響を意図通りに受容することができているといえる．

図 5.13 は，システム改修時に重要視する機能を年齢別に取りまとめたものである．一番多いのは映像の高品質化であったが，専門家に対しては高音質化が多いことが分かる．10 代については，ネットワーク越しにチャットをするニーズが大きいことも分かる．

最後に自由記載項目について述べる．「より自由な視点でライブを追体験できた」「生で見る以上の体験ができる」「お金を払ってでも使いたい」というポジティブな意見も見られたが，インタフェースの難しさ，分かりにくさを指摘するコメントもあった．具体的には．コントローラで音声を掴んだ状態での押す，引くといったジェスチャーによって個別の音声オブジェクトの音のみを抽出する動作について，直感的でない，反応が悪いといった意見も多かった．HMD のコントローラを使った操作方法については改善の余地があるといえる．また，ヘッドフォンの性能，360 度動画の画質向上や軽量化がなされれば，より没入感が高まったという指摘もあった．これはロスレス音源で配信を行ったものの，騒がしい会場の中での視聴に加え，HTC VIVE に付属する耳を囲わないオンイヤータイプのヘッドフォンを利用した影響があると考えられる．今後は音質も考慮した，より適切な VR デバイスの採用も検討したい．ユニークな意見としては，360 度動画を使った際に観客がそのまま映り込んでしまうことに対してのプライバシーに考慮が必要という意見があった．リアルタイム配信，ビジネス化などを行う際は，考慮が必須となるだろう．その他の意見について，以下の表 5.2 に列挙する．

表 5.2: 自由記述欄のコメント

No.	コメント
1	上空からの Viewing は感動した.
2	3D 映像にしてほしい.
3	カメラ 2 台で撮影して 3D で見られるようになれば，よりリアルな体験になると思いました.
4	音声にも立体感があるとよいと思います.
5	ボリューム等，音の操作が難しかった．ポジションチェンジのボタンが分かりにくかった.
6	音量調整するときに，ポインタを当て続ける必要があるので，設定値を見ながら操作をするときに難しさを感じた.
7	観客の視点，チケットの入手が難しい最前列等の映像が見られるなら，有料でも体験したいと感じました.
8	ぜひスポーツ戦での利用を実現してほしい.
9	ライブ以外にもいろいろな使い方ができそうな VR 体験でした．もっと多ジャンルに広がったら面白そうだと思いました.
10	眼鏡に圧迫感がある.
11	メガネに優しいゴーグルが欲しいです．自分のメガネで見れたりできたら嬉しい.

5.5.3 デモンストレーション

2018 年 1 月 30 日にビルボードライブ東京で行われた「Billboard LIVE MUSIC HACKASONG 2017」の審査において，LiVRation のデモンストレーションを行った．5.4.2 節で述べたように，事前に行ったリハーサルにて収録を行い，そのデータを用いてデモンストレーションを行った．デモンストレーションでは，HMD を装着したステージ上の操作者の視聴音声・映像が，会場のプロジェクターやステージの左右に設置されているのスピーカーで再生されるというものであったが，Line Live によるライブ中継により 3,600 人以上に視聴され，その後も様々なメディアで記事が発表されるなど，非常に反響が大きかった．審査員のコメントとしては，「完成度が高い」「すぐにでも使いたい」などのコメントがあり，結果として審査員による最優秀賞，および観客の投票数で決まる会場賞をダブル受賞した．先行研究 [110] は，昨年度行われた同イベントで次点の優秀賞にとどまっていたが，VR による配信機能を付加したことで，専門家にも一般の方にも，より魅力的なアプリケーション・プラットフォームとして認識されるようになったといえるだろう．また，今回の受賞によって，ビジネス的なポテンシャルについても確認することができたと考えている．

5.6　まとめと今後の展望

本稿では，収録したライブを HMD を利用して自由視点で視聴するとともに，収録された音声を自由にコントロールすることが可能な LiVRation について述べた．

本研究の目的として，「3 次元の映像・音声の演出をソフトウェアで制御」，「自由な視聴体験」，「遠隔配信」，「高臨場感・没入感」，「インタラクティブな体験」を設定したが，それぞれについて，1)SDM アーキテクチャによる設計，2) ゲームエンジン（Unity）での実装，3) ライブ配信サーバの構築及び SDM オントロジによるメタデータ記述，4)VR，ロスレス配信，振動伝達デバイスの導入，5) コントローラによる音源（SDM オブジェクト）操作及び SNS 連携，といった技術要素を取り込むことで達成できたと考える．しかしながら録音環境，再生環境，HMD によるインタフェースについて課題が顕在化した．

5.4.2 節で述べた録音環境においては，スポットライトが当たった部分の映像が白飛びする現象が確認された．一般的にライブ会場は暗く，撮影中にコントラストの調整が困難であったためであるが，こうした環境での撮影手法の確立も課題だろう．音声についても，高品質なコンテンツとするためには，プロフェッショナルによるミックスが必須である．今回の試作環境では，音声・映像それぞれの再生タイミングは Unity に依存する構成となっていたが，ネットワーク環境によっては，再生タイミングが著しくずれ，それによって没入感が阻害されるという結果になった．対策としては，音声・映像の同期のためのミドルウェアや，映像キャッシュ用のプロキシ・サーバなどをローカル環境に構築することが考えられる．しかしながら，実際のライブ環境においても距離によっては音声に遅れが発生するため，多少のずれであれば，没入感の阻害にならない可能性はあるため，SDM オントロジなどを参照しながら，距離によって柔軟な補正を行う機構の導入が必要とである．

今後は，5.5.2 節で述べた，ユーザインタフェースや没入感の改良に加えて，4.5 節でも述べたように，SDM アプリケーションの BIM 連動について検討と実践を行っていく計画である．LiVRation では，3D モデリング用のソフトウェアである SketchUp で作ったモデルを Unity に取り込んだが，前章で述べた BIM から抽出した形状とセマンティクスと SDM オントロジと連携させることで，より簡便なアプリケーション制作が可能になると考えている．

なお，2019 年 12 月に LiVRation による成果の一部を基にした REALIVE360[*10]が，NTT 西日本でサービス化された．本研究と同様に 4K/8K の 360 度動画を配信し，没入感の高いライブ体験を提供している．また，SDM コンソーシアムでは，SDM オントロジの拡張 [79] や，LiVRation のウェブ版である web360^2[80, 48], IoT や AR デバイスを用いて物理空間における立体音響のインタラクションを実現するシステム [119] など，SDM の高度化のためのプラットフォームとアプリケーションの研究を続けている．

[*10] https://realive360.jp/

第 6 章

スマートビルのデータ・プラットフォーム

本章では，スマートビルのための実践的なプラットフォームである futaba (Facility Unified digital-Twin Architecture for Building Automation) について述べる．実践的というのは，実際の建設プロジェクトに導入可能かつサステナブルな運用が可能であることを意図する．現実的なランニングコスト，スケーラビリティ，保守性などを考慮したデータ・プラットフォームを目指した．

futaba は Software Defined BACS 実現のためのデータ・プラットフォームであり，4 章のセマンティクスを取り込むことで，半自動的なハードウェア抽象化とエンドポイント生成が可能となっている．また，PaaS を前提としたビッグデータ処理基盤を有し，WoT (Web of Things) による API を提供することで，サードパーティとの柔軟な連携を可能にした．加えて，我々は futaba の参考実装を用いることで，実際のビルデータの収集と遠隔制御を実施し，それらの機能と汎用性を検証した．

6.1 はじめに

2.3.8 節で述べたように，スマートビルにおけるクラウド型の BACS (Building Automation and Control System) では，イニシャル及びランニングコスト低減のため，データ・プラットフォームを介した疎結合なシステム構成が望ましい．また，IoT を用いたユースケースを考慮した多様なデータタイプへの対応が必要である．

そのため我々は，汎用性の高いスマートビルのデータ・プラットフォームである futaba を提案する．

以降では，6.2 節でスマートビルやそのデータ・プラットフォームに関する関連研究について述べ，6.3 節で目的を述べる．6.4 節で対象とするスマートビルの要件分析を行い，6.5 節で futaba の設計要件について述べ，6.6 節でそれぞれの実装と評価について述べる．6.7 節で futaba を適用したアプリケーション例について述べ，6.8 節で結論と今後の展望を述べる．

6.2 関連研究

Wang らの整理 [74] によるとスマートビルの研究課題としては，以下があるとされている．

1. 異なるベンダーのプロダクト・ソリューションを用いてシステムを統合する際のインタオペラビリティ
2. インターネットおよびエンタープライズアプリケーションと統合手法

これらを解決手法として，一般的な方法としては，通信プロトコルの共通化とハードウェアゲートウェイの設置であると述べている．近年，BACnet や LONWORKS，KNX (The Konnex System Specification)，oBIX (Open Building Information Exchange) などの BACS 用のプロトコルが一般化し，多様なゲートウェイ製品も販売されてきているため状況は変わってきているが，ビルに特化した仕様で高度なドメイン知識が必要であるために参入障壁が高い

ことは変わっていない．

そのため近年では，上記に関する研究以外に，サードパーティの参入障壁を下げ，アプリケーションの再利用性を向上させるための研究や，アプリケーション，データ活用といった研究も行われている．筆者の整理によると関連研究は表 6.1，図 6.1 のように整理できる．

表 6.1: BACS の研究分野

No.	研究領域	説明
1	ミドルウェア（ビル OS）	インターネットの技術を使って異種のシステムを連携させる，汎用 BACS, BEMS を構成するためのミドルウェア（またはフレームワーク）に関する研究．BIM（Building Information Modeling）や環境シミュレーション，エンタープライズシステムとの連携を前提としたミドルウェア．
2	データ・プラットフォーム	BACS や IoT のデータを収集し，提供するためのプラットフォームに関する研究．高い汎用性を目標として，建設ドメインに特有のメタデータやタグを用いる．
3	プロトコル・ゲートウェイ	BACS のデータをより汎用的に扱うためのプロトコル・ネットワークトポロジ，ゲートウェイに関する研究．
4	アプリケーション	モデル予測制御・CAFM（Computer Aided Facility Management）・プレゼンテーションなど，建物固有のアプリケーションに関する研究．
5	データ活用	BAC や IoT データの収集方法論，メタデータ，オントロジーやデータモデルに関する研究．

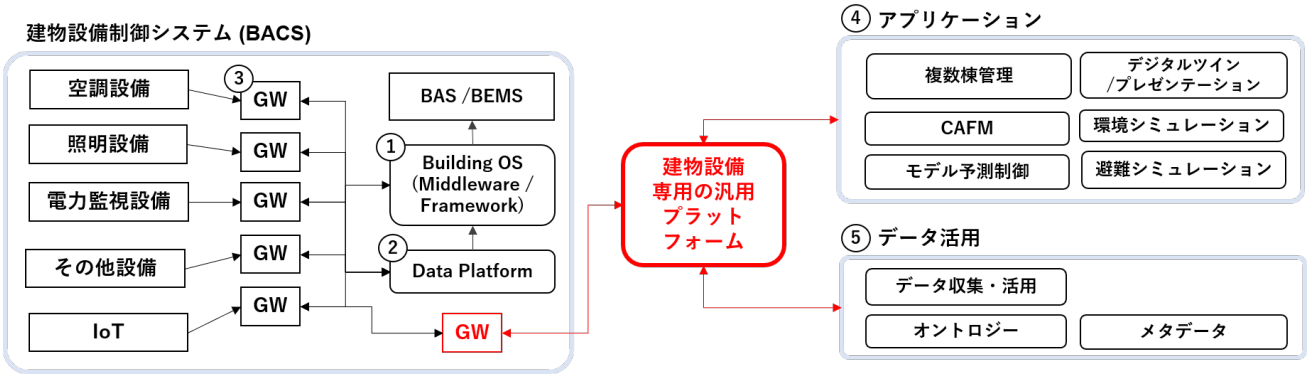


図 6.1: BACS の研究分野

以下では，本研究において関連深いミドルウェアとデータ・プラットフォームの先行研究について述べる．

6.2.1　SDB(Software Defined Building) Project

BACS にインターネット等の先端技術を適用して，スマートビルの高度化を図る研究は U.C.Berkeley の SDB (Software Defined Building) プロジェクト[*1]に多くの実績がある．

sMAP (Simple Measuring and Actuation Profile)

Dawson-Haggerty らは，センサや BACS から収集されるデータといった，物理的なデバイスの情報をシンプルかつ効率的に交換するための技術である sMAP を提案している [21]．sMAP はインターネットを前提とした RESTful な

[*1] http://sdb.cs.berkeley.edu/sdb/boss.php

ウェブサービスであり，データモデルとインタフェースを規定している．sMAP サーバが RESTful API を提供し，クライアントは Schema Server より提供されるスキーマ情報を参照しながら，HTTP メソッドによりデータの取得，および Pub/Sub 通信を実現する WebSub[*2]をによってサーバからの通知を受けることができる．表 6.2 に sMAP に定義されている最小のスキーマ例を示す．組み込みデバイス向けのプロファイルも定義されており，コンピュータリソースが少ない分散環境においてもクライアントは無理なく動作するとしている．

sMAP を用いて多様なデータを収集し，また計量データの見える化や照明などの機器制御を実現しているが，あくまでも多様なデバイスを HTTP による RESTful API にマッピングするデータ提供サービスである．現在値だけではなく，スキーマに定義された各種メタデータの取得や，リアルタイム通知も可能であるが，通知を受けるために，ウェブサーバをクライアント側で起動させておく必要があるなど，実装側の負荷も大きい．また，データタイプとしては，離散値・スカラー・ポイント制御といった基本的な項目のみを含んでおり，多様な IoT のスキーマには対応していない．加えて，データの永続化などは，連携するアプリケーション側の機能となっている．

表 6.2: sMAP のメタデータ・スキーマ

メタデータ	説明
Value	量的な値
Units	単位の解釈
Measured Quantity	計量のタイプ（水，電気など）
Scaling coefficient	計測時のスケール（係数）
Global timestamp	タイムスタンプ
Sequence number	シーケンス番号．欠損値の検出に用いる．
Instrument range(min-max)	計測レンジ
Instrument identifier	機器 ID

BOSS (Building Operating System Services)

Dawson-Haggerty らは，BACS をよりプログラマブルにすることを目的に，ハードウェアの抽象化を行い，アプリケーションの移植性を高め，耐障害性を高めるために BOSS を提案した [22]．sMAP をベースとしたハードウェア抽象化を行い，それらを柔軟に扱えるように独自のクエリ言語を提供し [50]，物理デバイスの障害検知と，リアルタイムデータと，永続化された時系列データを利用するための仕組みを提案している．

BOSS では，命名規則とセマンティクスを用いて BACS のような分散環境におけるリソースの抽象化を行い，データのバッファリングやリソースの発見機能，更には排他的制御を実現するためのトランザクションやアクセス制御機能を提供している．時系列データ処理のためのサービスを有しており，データのクレンジングなどの前処理をあらかじめ組み込むことによって，アプリケーションの開発効率の向上を狙っている．OS をメタファーとして必要機能を分析し，アプリケーションの構築を行っているが，ビッグデータ処理や AI 適用などのユースケースは考慮していない．

XBOS (An Extensible Building Operating System)

Fierro らは BOSS を拡張する形で XBOSS を提案している [32]．XBOS はスマートビルのための，より高度なビル OS を目指しており，要件として以下があるとしている．

ハードウェアの抽象化 (Hardware Presentation Layer) BACnet などのオープンプロトコルを使っているにもかかわらず，ベンダー独自の制御コードなどが用いられることが多く，十分な相互接続性が得られない．ハードウェアを抽象化し，一様なアクセスを可能にする機能が必要がある．

メタデータ定義 (Canonical Metadata Definition, Storage, and Usage) 標準化されたメタデータとセマン

[*2] https://www.w3.org/TR/websub/

ティクスを持つことで，アプリケーションのポータビリティを向上させる必要がある．

制御プロセスマネジメント (Control Process Management)　ビル設備の制御は通常 PLC（Programmable Logic Controller) などのコントローラによって構築されており，制御プロセスの変更などが難しい．デバイス抽象化だけではなく，特定のメタデータ・スキーマを使用して制御プロセスの入力と出力を記述することで，将来的な拡張に対応しやすくする必要がある．

建物の進化管理 (Building Evolution Management)　メタデータなどのビルや制御システム（アプリケーション）のプロファイルを，柔軟・動的に書き換えることで，ビルシステムのライフサイクル管理を容易にする．

セキュリティ (Security: Authorization and Authentication)　ユーザやアプリケーション毎に，アクセス制御を可能にする仕組みの導入が必要である．

拡張性の高い UX と API (Scalable UX and API)　規模によって，BACS におけるモニタリング・監視画面は要件が異なるはずである．それらのインターフェースを開発するための効率的なフレームワークを提供する必要がある．

上記に基づいて，類似研究との比較を行っており，XBOS が全ての要件を備えていると主張している．特徴的なのは，ビルシステムのライフサイクル管理のための「プロファイル (Building Profile)」を定義していることであり，プロファイルは XBOS Kernel と呼ばれるコンポーネントが持つメッセージブローカーを介して動的に管理されている．また，プロファイル定義により，自動的に BACS 監視のためのダッシュボードが生成される．なお，プロファイルは図 6.2 に示すようなビルの場所を示す階層構造と，Key-value によるメタデータの組である．

図 6.2: XBOS の Building Profile の例

```
[/]
Metadata/Site = UC Berkeley Metadata/Building = CIEE
[/spatial/floor/2]
Metadata/Floor = 2
[/spatial/floor/2/room/207]
Metadata/Room = 207 Metadata/Type = Room Metadata/Name = Conference Room
[/hvac]
Metadata/System = HVAC
[/hvac/zones/zone2]
Metadata/HVAC/Zone = Zone2 Metadata/Name = South Offices Metadata/Rooms = [200, 205, 206, 207, 208]
[/hvac/equipment/thermostats/tstat2]
Metadata/Name = Conference Room Thermostat Metadata/HVAC/Zone = Zone2 Metadata/Equipment = RTU Metadata/RTU
              = rtu2 Metadata/Location = Room Metadata/Room = 207 type = smap.drivers.thermostats.CT80
[/hvac/equipment/thermostats/tstat2/temperature]
Properties/UnitofMeasure = C Properties/UnitofTime = s Metadata/Sensor/Measure = Temperature Metadata/Sensor/Type
                         = Sensor
```

上記の Building Profile は空間階層のセマンティクスや，ビルシステムの記述に必要なメタデータを含むが，独自記法を採用しており，柔軟性や拡張性に乏しいといえる．そのため，Balaji らは BACS 特有の語彙と関係性を定義したメタデータ・スキーマであり，オントロジーである Brick [9] を提案した．更に，Fierro らは Brick に最適化された SPARQL Endpoint である HodDB [30] を開発している．

MORTAR (Modular Open Reproducible Testbed for Analysis & Research)

Fierro らは更に，上記の技術を組み合わせ，MORTAR [31] と呼ばれるデータ・プラットフォーム（オープンテストベッド）を開発した．実在するビルのデータ (90 棟) を集めており，それぞれ Brick によるメタデータ・セマンティクスが付与された上で公開されている[*3]．MORTAR は主として SDB プロジェクトの成果である OSS を組み合わせた構成になっている．具体的には，スカラー値の保存の適したスケーラブルな時系列データベースである BTrDB [6],

[*3] https://mortardata.org/

データモデルストレージとして Brick に最適化された HodDB [30]，Go 言語で実装したクエリプロセッサ，アプリケーションの実行環境としては，大容量データでの接続に適した GRPC による API を提供しており，クエリプロセッサを介してデータの取得が可能である．なお，取得されるデータのエンコードにはインメモリの列指向データフォーマットである Apache Arrow[*4]を用いており，アプリケーションが即時にデータを活用できるようになっている．ビッグデータ処理を意識した現代的なアーキテクチャであり，Brick により様々なビル用アプリケーションの適用が検証されているが，あくまでもテストベッドであるため，リアルタイムのアプリケーションのユースケースの適用は難しく，それぞれのコンポーネントの運用負荷・コストもかなり大きい．

SDB プロジェクトにおける課題

以上，SDB プロジェクトによる BACS の高度化の貢献は大きいといえる．しかしながら，BOSS や XBOS はビル用の OS としての性格が強く，利用のために高度なドメイン知識が求められること，リアルタイム処理に加え BTrDB によるビッグデータ処理も可能ではあるが，スカラー値以外は未対応であることなどの課題がある．

6.2.2 ミドルウェア研究

SDB プロジェクト以外にも，BACS 用のシステム・アーキテクチャや，ミドルウェアに関する研究は多くある．それらは，多様かつ異種のサブシステムで構成される BACS の汎用性・柔軟性を，インターネット由来のトレンド技術を使うことで高めるといったアプローチが多い．

Wang らはインターネット上の BACS 間の情報・サービス統合を実現を目的とし，OPC，XML，Web サービスの技術を用いて，スマートビルの統合と総合運用を実現するミドルウェア・フレームワークの提案をしている [74]．同様のアプローチとして，落合らによる SOAP を用いたストレージ中心のアーキテクチャによってエネルギー管理システムを構築した研究 [59] がある．

松岡はメタバース技術を応用し，オープンでスケーラブルであることを特長とする建築設備機器連携制御システムの構築手法を提案している [90]．メタバースとは，インターネットを介して複数のユーザーが，3 次元で表現された仮想空間を共有し，アバター（ユーザーの分身）を用いて他のユーザーとコミュニケーションをとるための通信システムであり，これをプラットフォームとして利用することで，ビル内のセンサや各種設備機器の連携制御を行った．具体的には Second Life[*5]や Open Sim[*6]をプラットフォームとして採用し，仮想空間上に様々な機器と連携するためのエージェントを配置することで実空間の機器制御やモニタリングを実現した．LED 照明制御による検証では，インターネットを介した場合でも平均して 0.33 秒での制御を実現しており，実プロジェクトへの適用も十分可能であると述べられている．実際にこの技術は，ビルテナント用のパーソナル制御（照明，パーソナルファン）技術として，2 件の導入実績があるが，導入した Open Sim によるメタバース技術がプロプラエタリな通信仕様であり，かつ対応可能な専門業者も少ないことから，保守・更新を行うにあたって，大きな障害になっている．

Ji らは時系列データベース (OpenTSDB[*7]) と，知識ベース (Apache Fuseki[*8]) を組み合わせた Hybrid inference system を提案している [45]．時系列データベースから抽出したデータを用いて，CO2 センサを用いた部屋の人数推定や，PMV (Predicted Mean Value) や PPD (Predicted Percentage Dissatisfied) といった環境指標を計算するとともに，SAREF [20] を拡張した ICMBS と呼ばれるオントロジを用いて時系列データのメタデータを RDF で記述し，SPARQL クエリによって参照可能にしている．ICMBS では，エネルギーの無駄を特定するため，OccupancyStatus や，hasNumberOfPeople といった語彙が定義されており，それらは独自のポイント命名規則に従って，半自動的に生

[*4] https://arrow.apache.org/
[*5] https://secondlife.com/
[*6] http://opensimulator.org/
[*7] http://opentsdb.net/
[*8] https://jena.apache.org/documentation/fuseki2/

成されるとしている．これらのデータを Web アプリケーションによって可視化することで，管理者がエネルギーを無駄に消費している室を特定することが可能としているが，あくまでも管理用のツールという側面が強く，サードパーティとの連携などは考慮されない．

Lilis らは，MoM（Message Orientated middle-ware) による，スマートビルにおける BACS のアーキテクチャを提案した [52]．特徴的なのは，省電力を特徴とする分散デバイスが，多様な物理デバイスの抽象化を行い，ミドルウェアを介して BACS のアプリケーションにデータを共有する構成であり，ZeroMQ によるメッセージングと，VPN による暗号化，認証を特徴とする．機能要件としては，相互接続性，非同期・イベントドリブンな通信，ダイナミックなネットワークトポロジー，拡張性・適応性，省コスト，拡張性・開発のしやすさ，耐障害性，セキュリティ，プライバシーを挙げている．

MoM では，メッセージ，メッセージ・キュー，メッセージブローカーの 3 つのコンポーネントが存在しており，それにより，非同期メッセージによる疎結合分散型のシステム構成を可能にしている．こうした出版・購読型（Pub/Sub 方式）の通信で用いられるプロトコルとしては，MQTT (Message Queuing Telemetry Transport)，AMQP，STOMP，CoAP などがよく知られており，以下の利点がある [27]．

同期分離　メッセージを Publish する側のシステムは，Publish 後に Subscribe 側のシステム等の応答のために待機する必要がない．全ての処理は非同期で記述可能で，メッセージおよび他ノードの状態に関係なく処理を続行できる．

時間のデカップリング　メッセージ交換に参加するために，通信関係者（Publisher/Subscriber) が同時にアクティブである必要はない．

論理的な分離　メッセージ交換のために，互いのメソッドを知る必要がない．RPC，SOAP との大きな差異である．

空間の分離　通信関係者は Broker の場所のみを知っていればよく，通信先のシステムについては知る必要がない．

Lilis は上記の利点を認めつつも，1）ブローカーを経由することによるネットワーク通信量の増加，2）ブローカーが単一障害点であることを理由に，P2P 型の ZeroMQ を採用し，そこに独自のディレクトリサービスを加えた構成を提案している．システムの複雑性は増すが，ミドルウェアとディレクトリサービスがそれらを吸収し，トランスポートに OpenVPN サーバによる暗号化通信を加えたとしても，十分な速度性能を持つとしている．

他にも，中小ビルのエネルギーマネジメントシステムの構築を目的とした BEMOSS [49]，独自のドメインモデルに基づいたアーキテクチャを提案している BaaS(Building as a Service) [72] があるが，いずれもセマンティクスの不十分であるため BACS の専門業者以外の新規参入が難しい，ビッグデータ処理や AI 適用のユースケースを想定していないなどの課題がある．

6.2.3　IoT のミドルウェア

Razzaque らは IoT のミドルウェアについて調査と分離を試みた [65]．それによると IoT のミドルウェアのアーキテクチャは表 6.3 のように分類ができる．BACS のそのほとんどは，時系列データ（スカラー値）であるため，同様の分類ができると考える．上記で紹介した研究例でいうと，Event-based [52]，Service-oriented [74, 59, 49, 72], Agent-based [90] のように分類できる．なお，本研究では MQTT 等を使った Event-base と，Service-oriented（RESTful アーキテクチャ）を用いたアーキテクチャを採用する．

また，Tang らは，BIM と IoT の連携という視点で，既往のシステムの調査を行っている [69]．リレーショナルデータベースを用いた連携手法から，セマンティックウェブを用いることで汎用性を高めた手法，更にそれらを組み合わせたハイブリッド手法が紹介されている．Tang らは BIM や時系列データに関わるメタデータなどは RDF として保存し，それらを時系列データが蓄積されたデータベースに紐づけるハイブリッド手法が最も確実な手法であると述べている．また，現状の多く研究はクラウドを前提としていないことを指摘しており，IoT のリアルタイム・ビッグデータ処

表 6.3: IoT のミドルウェアの分類

分類	説明
Event-based	メッセージブローカーを中心とした，Pub-Sub 型のアーキテクチャ．
Service-oriented	アプローチに基づいたアーキテクチャ．企業の IT システムで伝統的に使用されおり，疎結合，サービス再利用やマッシュアップのしやすさといったの特性も IoT 応用にとって有益であるが，デバイスの数が大規模になると管理が困難になる．
VM-based	ネットワーク内のノード（物理デバイス）は VM(仮想環境）を保持し，VM はネットワークを介してデプロイされたモジュールを解釈し動作する．近年の Docker コンテナを用いたアプローチに近い．
Agent-based	ミドルウェアで動作する Mediator（調整役）が，それぞれのノードで動作するエージェントをコントロールし，協調動作できるようにしたアーキテクチャ．可用性が向上するが，リソース発見などの処理が難しくなる．
Tuple-based	ネットワーク内のノードがそれぞれタプル空間を保持するアーキテクチャ．タプル空間は同時にアクセスできるデータリポジトリであり，ミドルウェア上にそれらを統合した統合タプル空間が形成される．
Database-Oriented	センサネットワークを仮想リレーショナルデータベースシステム と見なす．アプリケーションは SQL のようなクエリ言語を使ってデータベースからデータを取得することが可能で，ストアドプロシージャを用いて複雑なクエリの定式化が可能になる．
Application-specific	アプリケーション主導型のアプローチで，アプリケーションまたはアプリケーション・ドメインの要件に基づいて，ネットワークやミドルウェア を微調整することで，特定用途のリソース管理サポート に重点を置いている．

理についてはクラウドの活用が有用であると述べているが，具体的なアーキテクチャについては言及はない．

6.3 本研究の目的

先行研究の多くは，実験的なシステムが多く，構築・保守ベンダーなども想定していないため，そのまま建設プロジェクトへの適用は難しい．SDB による疎結合の RESTful アーキテクチャや，MoM，BIM 連携といった効果的な技術を取り入れつつ，ユースケースを考慮したより実践的な取り組みと研究が必要といえる．

本研究の目的は，IoT や AI，クラウド，ビッグデータを用いた幅広いユースケースに適合する汎用性を持ち，実際の建設プロジェクトに適用可能なスマートビル専用のデータ・プラットフォームの構築である．設計指針として以下を定めた．これらは，スマートビルのアプリケーションが，BACS の専門業者以外によって構築されることが増えることを考慮し，サードパーティが利用・開発しやすいこと，開発成果を展開しやすいこと，建設現場に入らずに遠隔でシステムの調整・試運転が容易であることを念頭に設定した．

多様なユースケース対応する高い汎用性 スマートビルにおいては，即応性が求められるもの，１日毎のバッチ処理が必要なもの，複数ビルを制御するなど，多様なユースケースが存在する．加えて，スカラー値以外の情報も扱えるなど柔軟性や拡張性を持つ，高い汎用性が求められる．

再利用性の高いアプリケーション スマートビルのアプリケーションは，IoT/AI などの専門性が求められることが多いため，BACS の専門会社以外のサードパーティとの協業が多い．BACS では，ポイントの意味や関係性を表すセマンティクスが与えられることがほとんどないため，データの理解が困難で参入障壁が高く，かつ構築したアプリケーションの移植性が乏しい．

建設プロジェクトへ導入可能 既往研究では，コストや保守性などを考慮していないことが多い．例えば，HodDB や BTrDB は拡張性確保のためサーバクラスタを必要とするが，たとえオンプレミス環境であっても保守・運用のコストは高額になる．ビル管理業務は費用削減が常に求められているため，データ・プラットフォームに関わる

コストは可能な限り抑えて提案する必要がある．

6.4　機能分析と設計要件

6.4.1　機能分析

設計要件を策定するにあたり，2.3.7 節で述べた要件やユースケースに加え，多くのプロジェクト対応経験から，まずはデータ・プラットフォームに必要な機能とプロセスの抽出を行った（図 6.3）．

フレームワークとして整理した機能は，デバイスからデータを抽出・発出する「データソース」部，データの収集・前処理や制御信号の発出を行う「複合イベント処理」部，収集されたデータを永続化する「データ保存」部，保存されたデータを抽出し，分析や機械学習・AI の適用を行う「データ加工／分析／機械学習」部，収集・蓄積・分析されたデータを使って動作する「アプリケーション」部に分けられる．

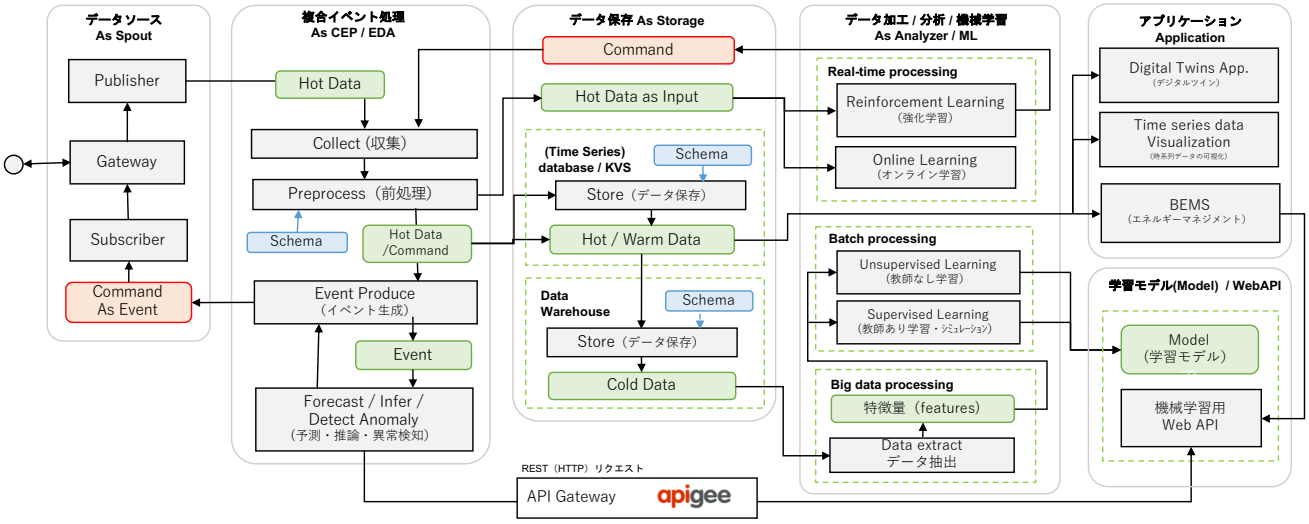


図 6.3: ユースケースを考慮した必要な機能コンポーネントとプロセス

データソース部では，何かしらの Gateway を介して，Publisher がデータ・プラットフォームにデータを発出する．また，複合イベント処理部からの制御信号を Subscriber が受け，それを Gateway が BACnet 等に変換して設備制御を行う．

複合イベント処理部では，発出されたデータ (Hot Data) は，拠点ごとに異なるスキーマを持っていたり，異常値があったりするために，それぞれ前処理 (Preprocess) を施され，データ保存用のコンポーネントに入力される．また，入力値に対して指定のロジックを適用したり，イベント複合処理 (CEP: Complex Event Processing) を行うことでイベントを生成するフローを考慮している．具体的にはルールベースの処理によって，指定の制御信号 (Command) を発したり，異常を検知したりする．また，サードパーティ・アプリケーションからの Command を受信する場合もあり，それらを対象ビルに伝える機能も有する．

データ保存部では，収集されたデータを一時的に保存するためのデータベース（時系列データベースや KVS：Key Value Store）に蓄積する．これらは，多くのデータを保持するとサーバで大量のメモリを消費するため，データ保持 (Data Retention) 期間を設けて，自動的に消えるようにすることが運用上望ましい．また，1 分毎のデータで膨大な量になるために，定期的にデータウェアハウスに移され，Cold Data として永続化される．

データ加工／分析／機械学習部は，AI を用いたユースケースに対応する．Hot Data を用いる場合は，例えばオンライン学習や強化学習の入力に該当する．Cold Data の場合は，データウェアハウスに永続化されたデータをバッチ処理

で定期的に抽出し，教師なし学習や教師あり学習の入力（特徴量）として利用される．ここでの生成された，予測や推論を行う学習済みモデルは，近年ではアプリケーションからウェブの API を介して利用されることも多いため，「学習モデル／ Web API」として配置している．なお，複数イベント処理部にある予測・推論・異常検知のコンポーネントは，これらの Web API を利用することも想定している．その間に API Gateway として配置しているのは，API がどのように活用されているか定量的に把握したり，古いインタフェースのシステムのデータを，REST 等の現代的な API に変換したりするコンポーネントであり，例えば apigee[*9]がある．

最後に，データ・プラットフォームを活用するアプリケーション部は，BIM の形状データ等を用いてデジタルツインを構築し，リアルタイムのデータを KVS 等から取得して表示したり，遠隔制御したりするユースケースを想定している．例として図 6.4，図 6.5 に竹中工務店東関東支店で構築した見える化システムを示す．図 6.4 では，データ・プラットフォームから 1 分毎に取得されるデータを用いて，表示周期である時・日・月・年毎のデータに加工・蓄積するモジュールが継続的に動いている．それら蓄積されたデータを用いて，表示用の Web アプリケーションが動作する構造となっている．図 6.5 の環境情報の見える化システムでは，KVS に保存されたデータをリアルタイム情報として右側のパネルに提示するとともに，別モジュールで 10 分毎のコンタ図を作成し，作成した画像を更新することで見える化を実現している．

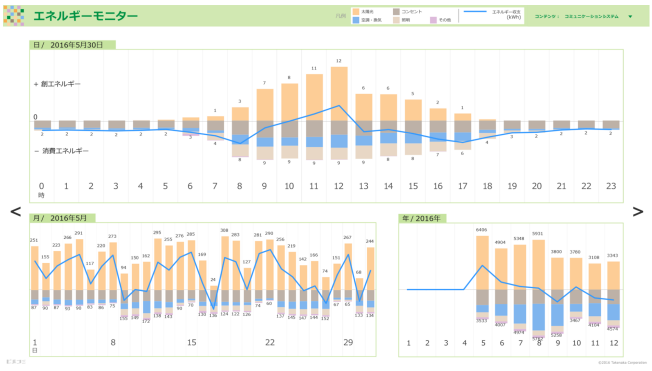


図 6.4: エネルギー見える化システム



図 6.5: 環境情報の見える化システム

例えば，2.3.7 節で示したユースケースは，図 6.6，図 6.7 の通り，本フレームワークによって説明可能である．ここでは赤色の矢印がデータの入力（Ingress）であり，青色の矢印がデータの発出（Egress）である．

図 6.6 は，DR におけるユースケースを想定したものである．リアルタイムで取得したデータに対し，CEP を施すことで DR に必要な制御コマンドに変換している．なお，この処理はアプリケーション側に位置付けることもできる．また，取得されたデータをリアルタイムに可視化・モニタリングするとともに，永続化されたデータを用いて DR のベースライン計算や負荷予測を行っている．

図 6.7 は，強化学習のユースケースを想定したものであり，リアルタイムの強化学習 AI と，バッチ処理による目標値予測 AI の運用を示している．強化学習 AI は目標値予測 AI のデータを利用するため，通常とは異なるパスがあるように見えるが，強化学習のコンポーネントをアプリケーションの 1 つとして見做すことで，説明可能である．

[*9] https://cloud.google.com/apigee/

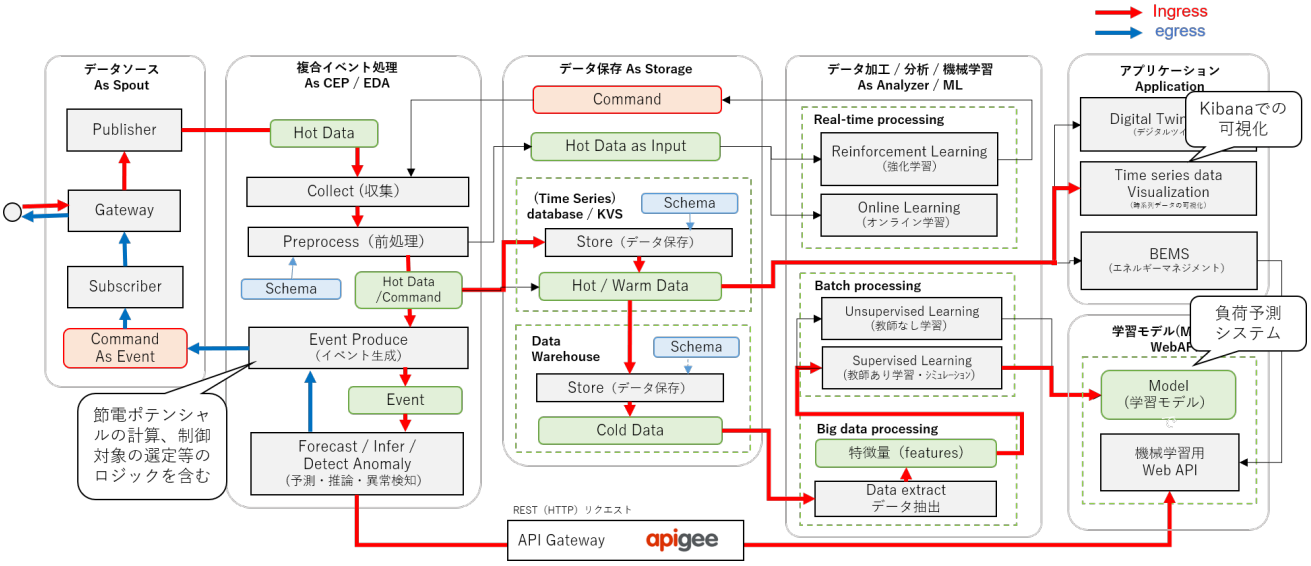


図 6.6: 複数拠点のデマンドレスポンス（VPP）のユースケースにおける分析

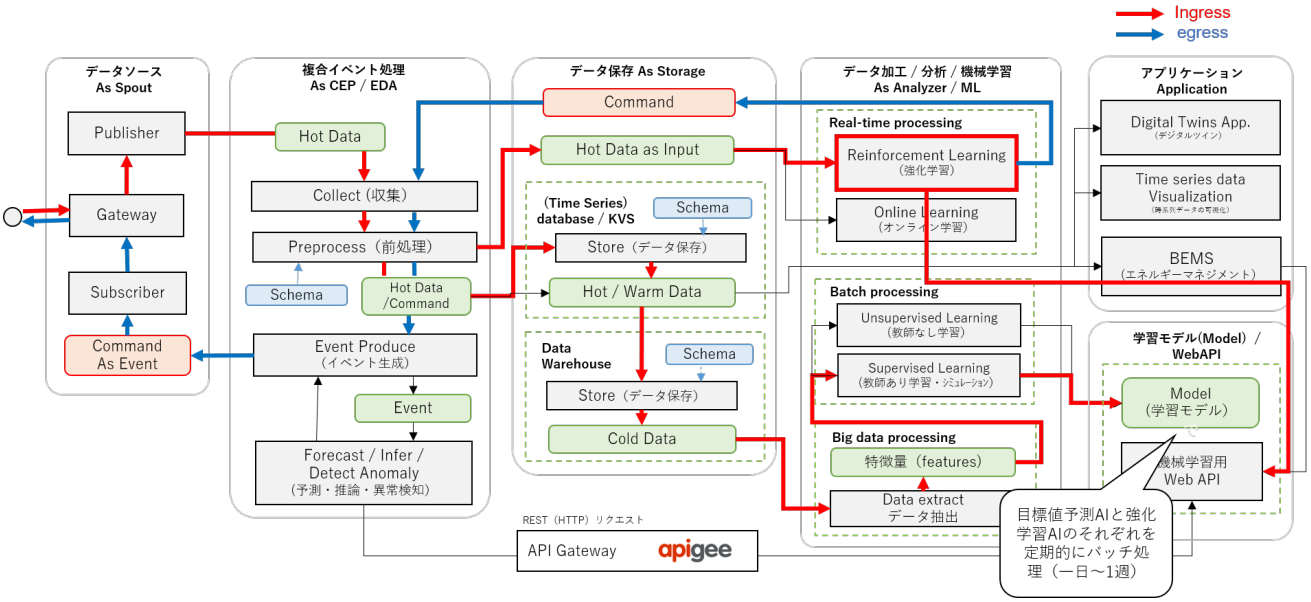


図 6.7: 強化学習エンジンによる遠隔制御のユースケースにおける分析

6.4.2 設計要件

前節で分析した機能要件やユースケースをもとに，データ・プラットフォームに対して以下の設計要件を得た．また，図 6.8 に設計指針と要件，実装にあたって採用する施策（技術）についてまとめた．

制御ロジックの変更容易性 スマートビルのシステムではチューニングのために，頻繁な機能やパラメータの変更が想定される．それらを現地の設備改修で行うことは，コストに加えて事務的な手続きにおいても負荷が大きく，クラウドから容易に変更をデプロイできる構成が望ましい．そのため，遠隔更新が可能なゲートウェイを開発するとともに，RESTful API を持った疎結合なアーキテクチャを採用する．

リアルタイム性の高い通信プロトコル 現地デバイスの遠隔制御のために，Web アプリケーションを提供することが多いが [117]，例えばポーリング型のアーキテクチャを採用した場合，十分なリアルタイム性が確保できずユーザ・エクスペリエンスが著しく損なわれることがあった．このため，IoT に親和性が高い Pub/Sub 型のプロトコルである MQTT を採用する．これは BACS と IoT の連携評価のために行った先行研究 [118] によるもので，少ないリソースで大量のデータを処理することができる．

ビッグデータ保存のための安価なストレージ ユースケースで必要となる BACS のデータは 1 棟当たり数千〜数万ポイントとなり，取得周期は概ね 10 秒〜300 秒である．また複数棟のユースケースも考慮すると，それらは必然的にビッグデータとなる．データ保存には BTrDB のような時系列データベースが考えられるが，ミドルウェアとして動作が必要であり，かつメモリを大量に消費するため，安価な運用には不向きといえる．そのため，ラムダ・アーキテクチャを導入するとともに，Apache Hadoop[*10]用の分散ファイルフォーマット (HDFS: Hadoop Distributed File System) を採用したビッグデータ処理基盤を構築する．

共通機能の提供 電力負荷予測のユースケースでは，日本の電力計測の慣習に合わせて 30 分毎（1 日だと 48 次元）のデータ抽出が多いが，見える化システムなどは 1 時間毎のデータ抽出が多い．こうした多様な粒度のデータ抽出への対応が必要である．また，ビル設備のデータ分析に特有な処理として，積算値差分や欠損値補間がある．積算値は最大値までカウントすると 0 に戻る仕様があり，それらの計算間違いが問題になることもある．これらの共通機能もプラットフォーム側の機能として盛り込むこととする．

セマンティクスの導入 サードパーティ参入のための参入障壁を下げ，アプリケーションの移植性を高めるために必要である．これについては，4 章で述べたセマンティクスを活用するとともに，WoT (Web of Things) (Web of Things) によるデータモデルを採用する．

[*10] https://hadoop.apache.org/

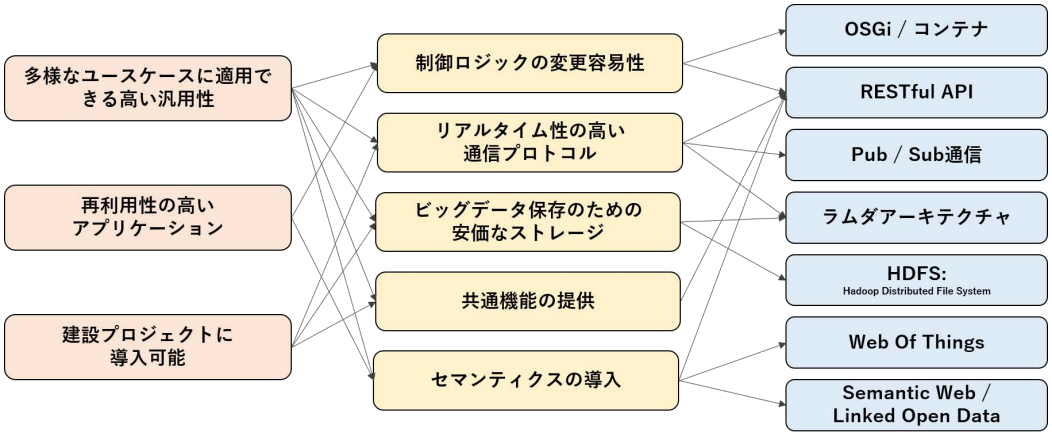


図 6.8: プラットフォームの設計要件と施策

6.5　futaba の設計

前節の要件をもとに futaba を設計した．図 6.9 にシステム・アーキテクチャを示す．ストリームデータの処理に特化したリアルタイム・ビッグデータ処理のためのアーキテクチャーの 1 つとして知られるラムダ・アーキテクチャ [54] を用いている．一般的なラムダ・アーキテクチャと異なるのは，外部からのコントロールを受け入れるパスが存在することである．このようにストリームデータ処理に注目し，ビッグデータ処理までを考慮にしたアーキテクチャは既往研究には類を見ない．

futaba では，ゲートウェイ等を介して取得するデータ（テレメトリ）をリアルタイム処理のためのホットパス，バッチ処理のためのコールドパスに分岐し，それぞれをスピードレイヤー，バッチレイヤーで処理し，サービスレイヤーを介してデータの利活用を行う．オープンシステムと PaaS を前提としており，保守性の向上とランニングコストの低減可能な構成を目指した．サービスレイヤーでは，WoT に準拠した RESTful API を提供しており，リアルタイムデータの取得，メタデータの取得，制御コマンドの発行，蓄積されたデータ抽出などが可能である．

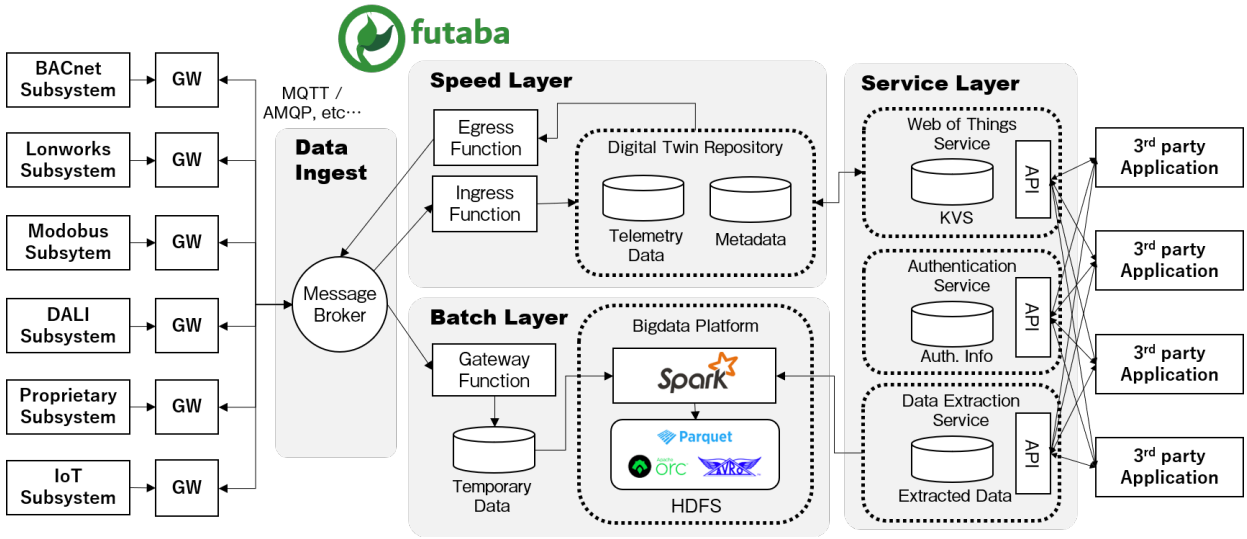


図 6.9: futaba のアーキテクチャ

表 6.4: 関連規格との比較 (Fujitsu WoT 資料より一部追記)

標準化団体 ／規格	標準規定範囲		
	アーキテクチャ	トランスポート	データモデル
IEEE1888	規定あり	SOAP / HTTP	XML
oneM2M	TS-001	TS-0004	TS-0023（家電）
ITU-T SG20	Y.4409/Y.2070	規定なし	規定なし
IIC	IIRA	Connectivity Framework (コンセプトのみ)	
W3C	WoT Architecture	WoT Protocol Binding	WoT Things Description
Echonet Consortium	(規定範囲外)	ECHONET Lite	オブジェクト詳細規定
Device WebAPI Consortium	(規定範囲外)	OMA GotAPU	OMA DWAPI-PCH
IPSO Alliance	(規定範囲外)	OMA LWM2M	Smart Object
KNX Association	(規定範囲外)	KNX	Application Description
BACnet	(規定範囲外)	RS485/TCP	規定あり
BACnet / WS	規定あり	REST/HTTP	規定あり
Oracle IoT Cloud	規定あり	REST/HTTP	Oracle device model
futaba	**Lambda Architecture**	**MQTT, REST/HTTP**	**WoT Things Description**

表 6.4 に関連規格との比較を示す．ここではアーキテクチャ，トランスポート，データモデルで比較している．他規格はデータの永続化やビッグデータ処理に関して，規定していないものがほとんどであるが，futaba はそれらのユースケースを予め取り込んでいる．

なお，futaba は日本語の二葉（双葉）から取られており，特徴的なラムダ・アーキテクチャ，デジタルツインを想起させることを意図して命名した．

以降では，要素技術である WoT に述べた後，スピードレイヤー，バッチレイヤー，サービスレイヤーそれぞれの機能について述べる．

6.5.1 Web of Things

WoT は，センサや設備といった実空間上のオブジェクトを「Thing」というメタモデルを元にモデリングを行い，そこに定義されたプロパティやアクションに対して，HTTP を介してインターネットからアクセス可能にする技術である．

モデリングされた Thing は，Thing Description (TD) と呼ばれる JSON-LD にフォーマットされた RDF で記述される．WoT では，TD の記述に基づいたリソースをサーバ上で管理し，クライアントに対して HTTP のエンドポイントを公開する．クライアントは TD を参照することで，対象の実空間上のオブジェクトである Thing がどのようなデータをセンシングしているか，またどの URL でアクセス可能かを知ることができる．図 6.10 に公式 HP[*11] より引用した TD の例を示す．家庭用照明の例であるが，status と呼ばれるプロパティ (properties) を定義しており，それが toggle と定義されたアクション (actions) によって変更可能である．また，overheating と呼ばれるイベント (events) が定義されており，イベント発生時にリアルタイムでクライアントが情報を受け取る仕組みを提供する．

なお，ビル制御のユースケースを想定すると，プロパティはセンサなどの計測値の収集や，設定温度などの設定値の変更などであり，アクションは例えば，照明の一括点灯，消灯といった単一プロパティの設定だけでは手間のかかる処理などが想定できる．イベントは計測値などの継続的なリアルタイム取得や警報取得などに利用される．

WoT サーバの参考実装としては，Mozilla のもの[*12]があるが，限定的なユースケースにおける実装が示されているのみで，例えばサーバ側で，事前に定義した TD をもとにしてエンドポイントを生成するような機能は提供されていない．また，実空間（フィールド側）のデバイスへのデータのバインドについても参考実装では示されていない．

[*11] https://www.w3.org/TR/wot-thing-description/
[*12] https://iot.mozilla.org/

図 6.10: Thing Description の例（公式 HP より引用）

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1",
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing",
    "properties": {
        "status" : {
            "type": "string",
            "forms": [{"href": "https://mylamp.example.com/status"}]
        }
    },
    "actions": {
        "toggle" : {
            "forms": [{"href": "https://mylamp.example.com/toggle"}]
        }
    },
    "events":{
        "overheating":{
            "data": {"type": "string"},
            "forms": [{
                "href": "https://mylamp.example.com/oh",
                "subprotocol": "longpoll"
            }]
        }
    }
}
```

なお，Jara [44] は，現状の IoT のインタオペラビリティは不十分であり，それらを向上させるための仕組みが，RESTful なアーキテクチャに基づいた WoT であると述べている．データ活用においては対象ドメインに対する共通理解が必要であり，そのためにセマンティクウェブとの融合した Semantic Web of Things(SWoT) が求められるとしている．IoT 用のプロトコルの比較や，データのペイロードについての考察もあるが，具体的なシステムの実装については述べられていない．本研究で目指すのは，スマートビルを題材とした SWoT の実現ともいえる．

6.5.2 スピードレイヤー

データ入力部（ゲートウェイ）

BACS で取得できるポイントデータを，MQTT 等に変換して図 6.9 の Data Ingest に送信するためには，ゲートウェイ (GW) によるプロトコル変換が必要である．BACS では BACnet 以外にも多様なプロトコルが用いられるため，対応プロトコルへを増やしたり，遠隔更新を可能にしたりするため，OSGi[*13]や Docker[*14]を用いて開発する方針とした．

データ処理部

ビル群から受信したテレメトリを事前に定義したルールに基づき前処理を行い，現在値の保存などのリアルタイムデータ処理を行う．データの受信は Data Ingest から行い，例えば MQTT Broker を介してテレメトリを取得する．テレメトリのフォーマットは JSON とし，表 6.5 に示すスキーマを定めた．データは values が格納されるが，多様なデータタイプに対応するため，その中のスカラー値である value のみを必須とし，data 以下のメタデータは自由に決められる仕様とした．図 6.11 に例（ビーコンによる位置情報）を示す．ここでは，value には userId が転記されており，data はそのままの文字列として抽出され，必要に応じてクライアント側で処理される．

Message Broker からの取得したテレメトリは，Ingress Function と呼ばれるコンポーネントによって，必要な前処理が行われる．例えば異常値の判定や，事前に定義されたスキーマ外のデータの除外などである．処理済みのデータ

[*13] https://www.osgi.org/
[*14] https://www.docker.com/

表 6.5: テレメトリのスキーマ

メタデータ	説明
pointID	ポイント識別子
eventTime	イベント発生時刻
topic	トピック名
values	値情報（任意型）
sequenceNumber	シーケンス番号．欠損値の検出に用いる．

は，Digital Twin Repository と呼ぶサービスに保存され，この際に対象のポイントデータの ID と，4.4.2 節で示した建物メタデータをもとにして，サービスレイヤーにおけるエンドポイントでの参照値にマッピングする．公開されたエンドポイントから，ビル内設備の制御も可能であり，ポイントへの制御指示は，Egress Function によって，不正制御などを防ぐためのフィルタリングなどを施され，同様に Message Broker とゲートウェイを経由して，ビル設備の中に伝達される．

図 6.11: テレメトリの例

```
{
  "pointID": "73AC698E-B03F-4F64-B365-8C009EF02777",
  "eventTime": "2020-03-06T18:05:00 999+0900",
  "topic": "takenaka.co.jp/Site/Building/Floor/Space/Location/Beacon/BLE/Area_4/Location/R",
  "values": {
    "value": 31,
    "data": {
      "areaId": 4,
      "userId": 31,
      "locationTime": "2020-03-06T18:00:00+0900",
      "x": "-12.5068",
      "y": "-22.1004",
      "floor": "1"
    }
  }
}
```

6.5.3 バッチレイヤー

Message Broker から取得したテレメトリは Gateway Function によって，一時的保管領域にファイル等で格納される．テレメトリデータから表 6.6 に示す情報を抽出し，格納する．格納データの point 列はテレメトリデータのデータ生成元であり，ゲートウェイにて付加されるポイント ID を格納する．eventTime 列はテレメトリデータに含まれるテレメトリデータのゲートウェイにおける生成日時を格納する．value 列はテレメトリデータの代表値を示し，必須項目である value が格納される．なお，value 列はデータ抽出時に，欠損値補間等の共通処理の対象となる．最後に，rawdata 列は，テレメトリデータ内のペイロード文字列全てを格納する．テレメトリデータ内のペイロードが単一の値のみを含む場合は，value 列と同じ内容となる．

表 6.6: ビッグデータ処理基盤に格納するデータ型

データ列名	内容	データ型 (Apache Spark)
point	ポイント ID	String
eventTime	テレメトリ時刻	Timestamp
value	テレメトリ代表値	String
rawdata	テレメトリ RAW データ	String

その後，Apache Spark[*15]を用いて，１日１回〜数回のバッチ処理で HDFS に変換し，一時ファイルは削除する．Apache Spark Stream を用いる構成も可能ではあるが，サーバクラスタを起動させ続ける必要があり，クラウドでの運用を前提とするとランニングコストが増加するため，Apache Spark の構成とした．データ抽出のリクエスト時のパラメータで抽出周期や期間を指定し，データ補完処理や異常値検出，積算値差分なども行う．

時系列データの保存には BTrDB や Elasticsearch，InfluxDB[*16]のような時系列データベースも考えられるが，ミドルウェアとして動作が必要であり，かつメモリを大量に消費するため，可視化が必要なデータを必要な期間のみ保存することが適切といえる．蓄積されたデータのほとんどは参照されない可能性があるために，HDFS などの静的ファイルとして安価に保存するのが最適と考える．

6.5.4　サービスレイヤー

スピードレイヤーで得られたリアルタイムのデータを公開するための WoT Service と，バッチレイヤーで蓄積されたデータを抽出するための Data Extraction Service，それらのサービスを利用するための認証を行う Authentication Service から構成される．

WoT Service では，4.4 節で述べた手法を用いて，BIM とポイントリストから抽出した建物メタデータと，対応するクラススキーマと組み合わせて，WoT で利用する TD を生成する（図 6.12）．WoT Service が起動時に TD を読み込んで，RESTful API のエンドポイントを生成する．具体的には，WoT で規定されている Property の取得・書込，Action の実行，Event によるリアルタイムデータの取得などであり，BACS の利便性を考慮して一括取得などの API を独自追加している．表 A.1 に設計・計画した API を示す．

図 6.13 にクラススキーマの例を示す．生成したデータモデルのクラスに対応しており，これらのスキーマを参照して，エンドポイントで公開されるデータ型が決定される．クラススキーマで示されるクラスは，継承させることが可能である．id が対応クラス名を示しており，@type は継承元のクラスになる．継承先のクラスには，スキーマに示される Property，Action，Event が継承される．なお，クラススキーマの生成は，取込用 CSV に適切なメタデータを埋め込むことで，ある程度自動化することができるが，継承関係などの細かいモデリングは手動で行う必要がある．

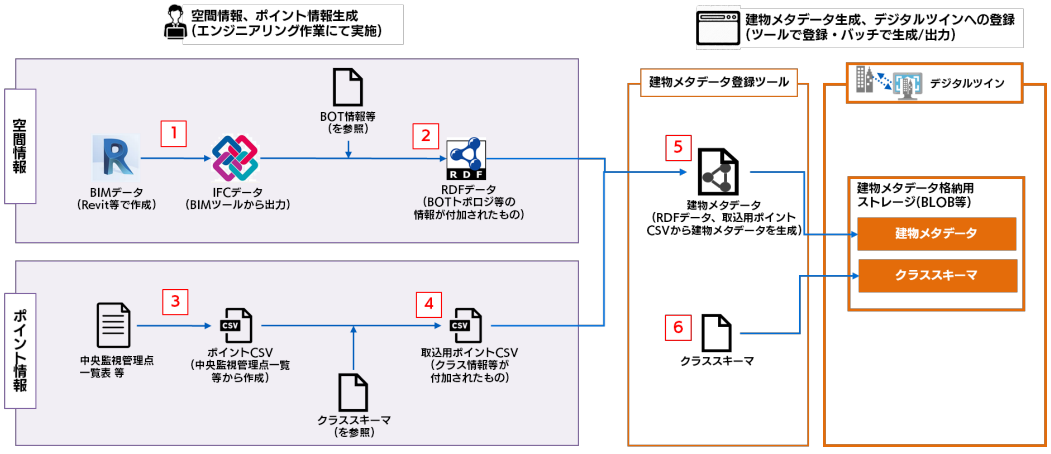


図 6.12: 建物メタデータの生成

Data Extraction Service は，バッチレイヤーで保存された HDFS からデータ抽出のための非同期 API を提供している．リクエストがあった時点で，サーバクラスタが起動するようにすることで，クラスタ稼働時間を最小限にする方

[*15] https://spark.apache.org/
[*16] https://www.influxdata.com/

図 6.13: 参照先のクラススキーマ

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1", "http://www.takenaka.co.jp/R90"
    ],
    "id": "Wireless_Sensor",
    "@type": "BaseMode",
    "tag": ["temphumidity", "light", "sound", "co2", "TVOC"],
    "properties": {
        "Light": {
            "descriptions": "無線センサ（照度）",
            "readOnly": true,
            "type": "number",
            "minimum": 0,
            "maximum": 2000,
            "form": {
                "op": "readproperty"
            }
        },
        "Humidity": {
            "descriptions": "無線センサ（湿度）",
            "readOnly": true,
            "type": "number",
            "minimum": 0,
            "maximum": 100,
            "form": {
                "op": "readproperty"
            }
        },
        "Temperature": {
            "descriptions": "無線センサ（温度）",
            "readOnly": true,
            "type": "number",
            "minimum": -20,
            "maximum": 50,
            "form": {
                "op": "readproperty"
            }
        },
        "eCO2": {
            "descriptions": "無線センサ（eCO2)",
            "readOnly": true,
            "type": "number",
            "minimum": 0,
            "maximum": 100,
            "form": {
                "op": "readproperty"
            }
        },
        "SoundPressure": {
            "descriptions": "無線センサ（音圧）",
            "readOnly": true,
            "type": "object",
            "minimum": 0,
            "maximum": 100,
            "form": {
                "op": "readproperty"
            }
        },
        "eTVOC": {
            "descriptions": "無線センサ（eTVOC)",
            "readOnly": true,
            "type": "number",
            "minimum": 0,
            "maximum": 100,
            "form": {
                "op": "readproperty"
            }
        }
    },
    "actions": {},
    "events": {}
}
```

表 6.7: futaba の API

No.	API 種別	説明	HTTP メソッド	管理権限
1	Authentication	アクセストークン発行	POST	○
2	Authentication	アクセストークン更新	PATCH	○
3	Data Extraction	モデル学習データ要求タスク 作成	POST	
4	Data Extraction	モデル学習データ要求タスク 詳細確認	GET	
5	Data Extraction	モデル学習データ要求タスク 有効状態変更	PATCH	
6	Data Extraction	モデル学習データ要求タスク キャンセル	DELETE	
7	Data Extraction	モデル学習データ要求タスク WebHook 登録	POST	
8	Data Extraction	モデル学習データ要求タスク WebHook 削除	DELETE	
9	Utility	天気予報データ 取得	POST	
10	Utility	建物メタデータ 検索（パス指定）	GET	
11	Utility	建物メタデータ 検索（クエリ指定）	POST	
12	Utility	建物メタデータ 編集	PUT	
13	WoT	TD 取得（パス指定）	GET	
14	WoT	TD 取得（クエリ指定）	POST	
15	WoT	Property 取得（TD 内全取得）	GET	
16	WoT	Property 取得（1Property）	GET	
17	WoT	Property 書き込み	PUT	
18	WoT	Action 実行	POST	
19	WoT	Action 状況取得（TD 内全取得）	GET	
20	WoT	Action 状況取得（1Action）	GET	
21	WoT	Action 詳細状況取得	GET	
22	WoT	Event サブスクライブ	POST	
23	WoT	Event サブスクライブ解除	DELETE	
24	WoT	Event サブスクライブ状況取得	GET	
25	WoT	Property 一括取得（パス指定）	GET	
26	WoT	Property 一括取得（クエリ指定）	POST	
27	WoT	Event 一括サブスクライブ	POST	
28	WoT	Event 一括サブスクライブ解除	DELETE	
29	WoT	Event 一括サブスクライブ状況取得	GET	

針とする．

Authentication Service は，上記の 2 つのサービスを利用するため，OAuth 等による利用者認証 API を提供する．利用者認証には，専用の管理ツールを用いて，サードパーティアプリの登録と ID/シークレットの発行を行う．この際，管理者はサードパーティ毎に，利用可能な API の設定が可能である．

上記の他にも，BACS のユースケースで良く用いられる天気予報の取得や，ポイント (Thing) のメタデータ取得，検索のための API がある．

6.6 実装と評価

設計要件に基づいて，futaba の実装を行った（図 6.14）.

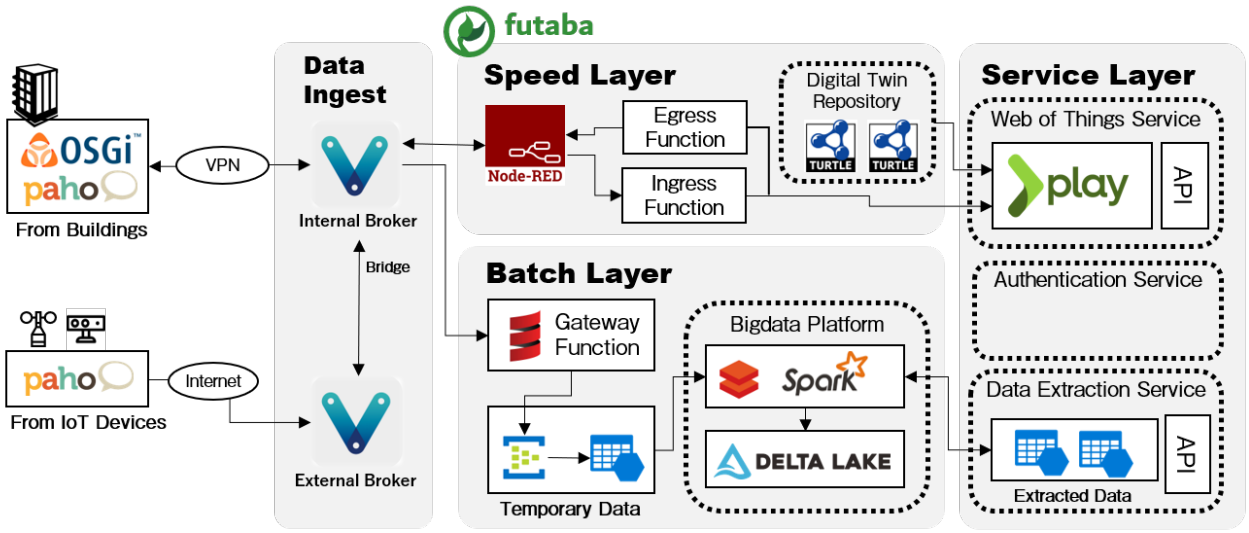


図 6.14: futaba 参考実装

6.6.1 スピードレイヤー

■データ入力部（ゲートウェイ） 図 6.15 は, BACnet ゲートウェイ構築の実装である．ゲートウェイが ReadProperty や WriteProperty といった BACnet サービスを使って，収集対象のポイントのスカラー値を定期的に収集，書き込みを行っており，それらを MQTT に変換して Message Broker に発出している．OSGi で動くバンドルとして設計・実装されているため，1 つのゲートウェイにモジュール化された複数のゲートウェイ機能を実装することができる．

■データ処理部 スピードレイヤーの実装は先行研究 [118] の基盤を利用する．NTT Communications のクラウドである ECL2.0[*17]上で構築しており，MQTT Broker である VerneMQ[*18]により 5 棟のビルデータを集約し，1 分当たり平均で 16,773 件のデータが平均して発出されている．ビルのデータは原則として VPN を介して収集しているが，IoT 機器などは，インターネットを経由する構成があるため，認証・暗号化を行う外部連携用の Broker を経由して連携させている．Ingress Function / Egress Function は簡易的に NodeRED[*19]で実装しており，サービスレイヤーのエンドポイントに公開するために必要な名前空間の変換等を行う．建物メタデータは静的な運用を想定し，事前に生成したものを保存しておくだけとしている．

6.6.2 バッチレイヤー

■HDFS を用いた処理技術の検討 HDFS を使ったビッグデータのバッチ処理は，常時実行する必要はないが，実行時には大量の計算資源を必要とするという特徴がある．このためのフレームワークは，以下の 3 つに大別される．

[*17] https://ecl.ntt.com/
[*18] https://vernemq.com/
[*19] https://nodered.org/

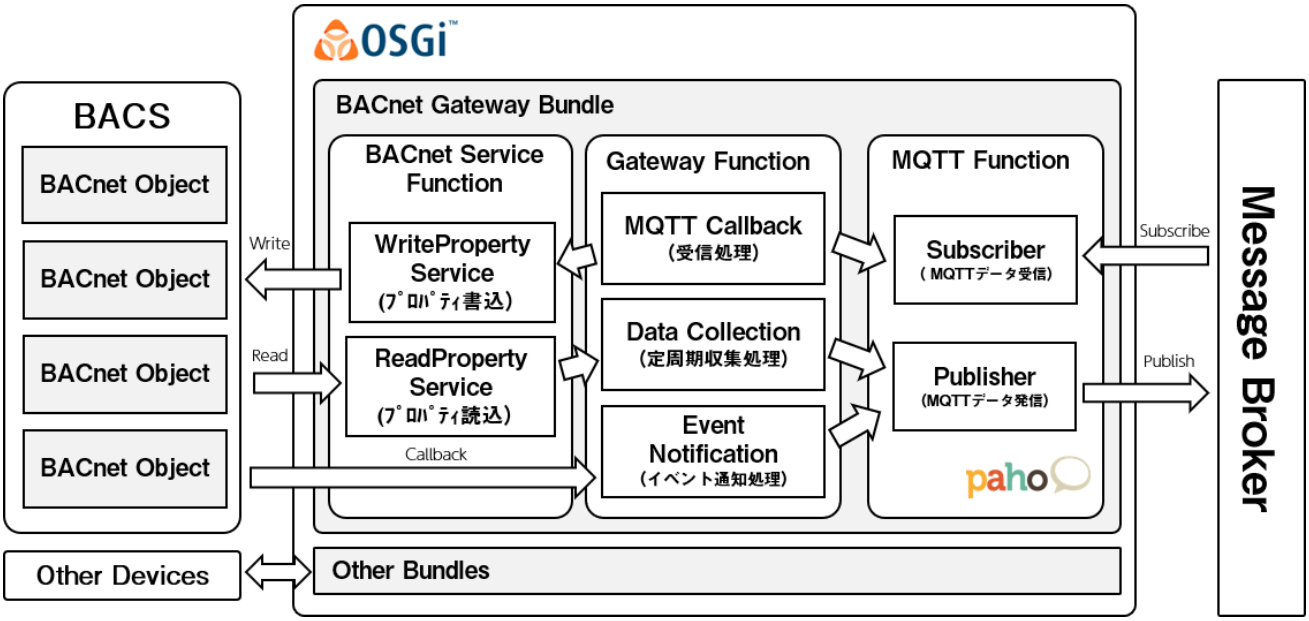


図 6.15: OSGi による BACnet ゲートウェイの実装

OSS のビッグデータ処理基盤　Apache Hadoop での MapReduce や Apache Spark，Apache Hive[*20]等が挙げられ，インターネットのサービス以外でも多く用いられている．これら OSS の基盤技術は，例えば Microsoft Azure 上では仮想マシン (IaaS) によるクラスタ構成，Azure HDInsight，Azure Databricks[*21]によって実現できる．

SQL ベースの分散処理基盤　Microsoft Azure の専用サービス (PaaS) として，Azure Synapse Analytics や Azure Data Lake Analytics といった，SQL ベースの分散処理基盤サービスが提供されている．

サーバーレスな処理基盤　Azure Functions のようなサーバレスアーキテクチャの PaaS を用いる手法である．負荷に応じて高速かつ柔軟に実行ノードを増減出来ることや，実行コストが安価であることから，ビッグデータ処理にも応用ができる．

表 6.8 に，各実行基盤の特徴をまとめる．

表 6.8: Microsoft Azure におけるビッグデータ処理基盤の比較

実行基盤	言語・フレームワーク	マネージド (PaaS)	課金単位	国内での提供	自動スケール
仮想マシン	任意	×	インスタンス時間	×	×
HDInsight	Apache Hadoop 等	○	インスタンス時間	○	×
Databricks	Apache Spark	○	ジョブ実行時間	○	○
Synapse Analytics	U-SQL	○	インスタンス時間	○	×
DataLake Analytics	U-SQL	○	ジョブ実行時間	×	×
Functions	.NET, Python 等	○	ジョブ実行時間	○	○

Azure Databricks は，Microsoft Azure と Amazon Web Services (AWS) で共通のサービスが提供されている Apache Spark の分散実行基盤である．バッチジョブを実行する際に，必要な計算資源を用意し，使用が終了すると解

[*20] https://hive.apache.org/
[*21] https://azure.microsoft.com/ja-jp/services/databricks/

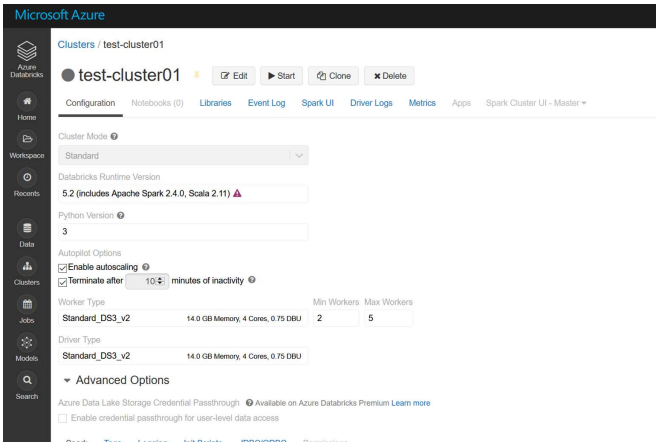


表 6.9: Azure Databrick によるクラスタ生成

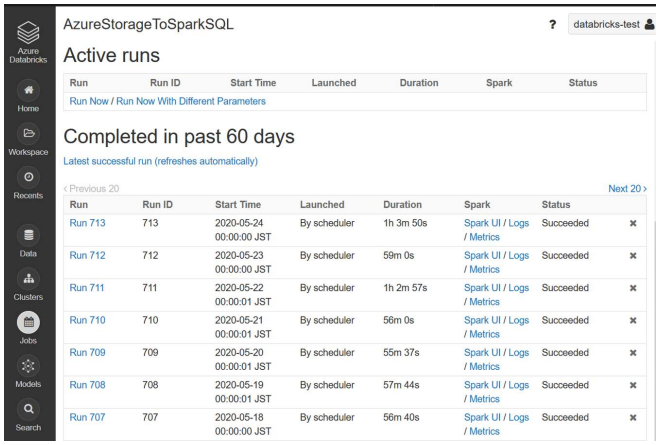


表 6.10: Azure Databrick によるジョブの実行管理

放するため，大量の計算資源を使用しながら，課金額を最小限に抑えることが可能となる．図 6.9 に Azure Databricks により生成するクラスタの設定画面を示す．実行する Apache Spark のバージョンやクラスタを構成する Worker や Driver の仕様や起動数などを細かく設定できる．また，ジョブといわれる Scala や Python などで記述したスクリプトを，PaaS 側の管理でスケジュールに合わせて起動させることができる（図 6.10）．

Azure Functions も Databricks と同様に計算ノードを自動的に増減させることが可能だが，Apache Spark のように 1 つの処理を複数の計算ノードに分散するのではなく，1 つの処理を実行するノードが複数並列起動するという形態であり，使用できるメモリ量が最大でも 14GB と Databricks で使用できるノードに比べて小さいことから，ユースケースで想定される 1 建物 1 日当たり数百 GB のデータを処理できない可能性が高い．以上により，PaaS との親和性と負荷分散性能，ランニングコストから，データ処理基盤として Azure Databricks を採用した．

■HDFS による永続化処理　参考実装では，コールドパスから抽出したテレメトリを Scala で実装したゲートウェイを介して Azure Event Hubs にデータ送り，PaaS の機能で 15 分毎にキャプチャデータをファイル保存しており，それらを Azure Databricks を用いて，1 日毎のバッチ処理で HDFS の一種である Delta Lake に変換して保存する．図 6.16 に，Azure Databricks を用いて，具体的なデータの永続化のイメージを示す．また，図 6.17 にデータ抽出する際の，動作イメージを示す．いずれも参考実装をもとにした最適化後の動作イメージである．最適化後の設計では，キャプチャデータを整形したのち，Delta Cache と呼ばれる高速なメモリ領域に展開し，それらを Delta Lake に日付ごとのフォルダに格納し永続化している．Delta Lake の実態は列指向のカラムナフォーマットである Parquet[*22]となっている．サービスレイヤーからのクエリによって抽出が行われるが，抽出後のファイルは zip で圧縮されて，その後ダウンロードできるようになっている．フォーマットとしては ORC[*23]や Parquet が選べるようになっているが，これらのフォーマットはサードパーティからのヒアリングにより決定した．zip の解凍やその後の処理も含め，python のライブラリで処理可能であることを意識している．

■性能評価　参考実装においては，データの永続化のための毎日のバッチ処理時間が 35 分，月額費用で 2,400 円程度かかっているが，機能最適化検討の結果，Standard_L4s (32GB メモリ，4 コア，29 ノード) での処理で 13 秒程度となり，月額費用は 877 円程度に抑えられることが分かった（表 6.11）．Dalta Lake フォーマットによる圧縮効率も高く，例えば 5 万件/分のデータを想定した場合においても，250MByte 程度となる．Azure Data Lake Storage Gen2 では，1T バイトを保存しても 8,000 円程度といえるため，十分なコスト効率といえる．

[*22] https://parquet.apache.org/
[*23] https://orc.apache.org/

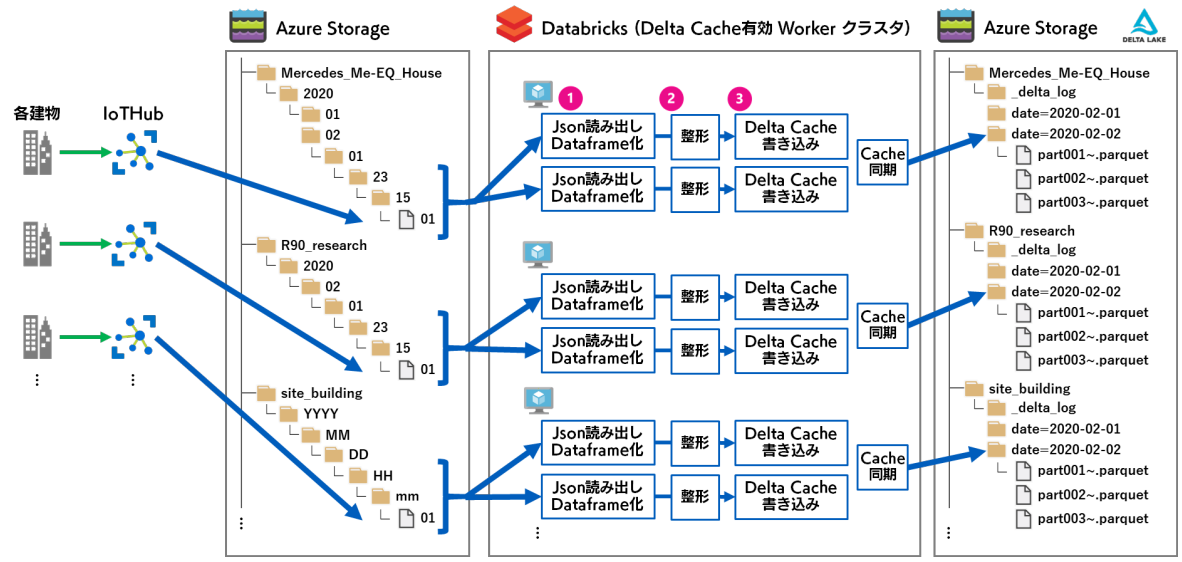


図 6.16: Azure Databricks でのデータ保存（最適化検討後）

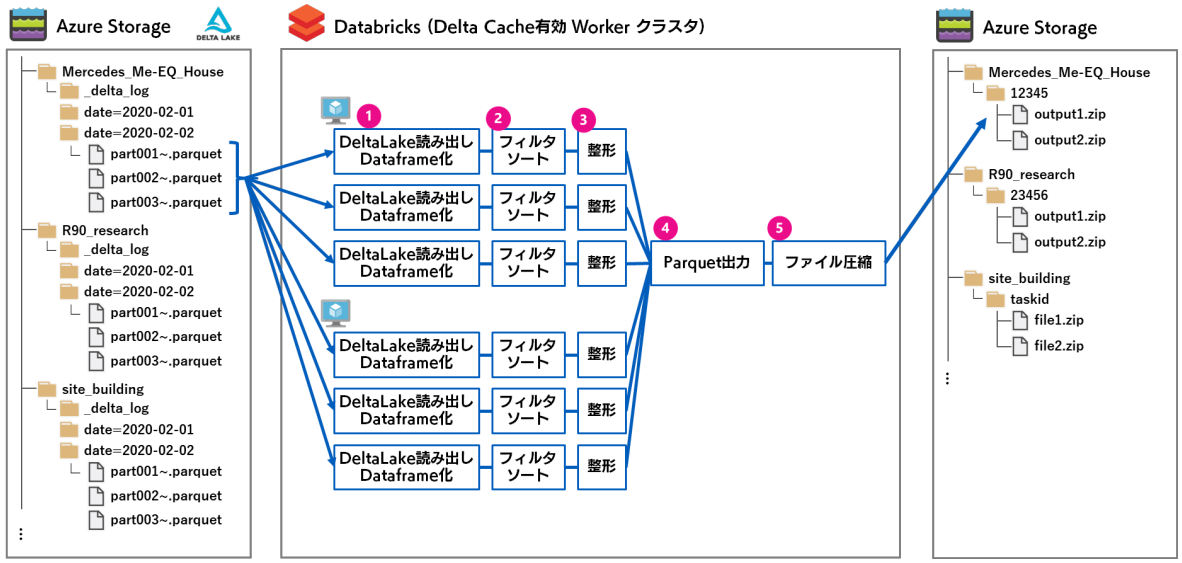


図 6.17: Azure Databricks でのデータ抽出処理（最適化検討後）

表 6.11: ビッグデータ処理基盤の性能評価

項目	現構成	最適化後	備考
DeltaLake バッチ処理時間	35 分	13 秒	VM 起動時間は除く
DataBricks 想定費用/月	2,400 円/月	877 円/月	

6.6.3　サービスレイヤー

サービスレイヤーでは，WoT Service 用のサーバを Mozilla の参考実装を参考に，Play Framework[*24]で実装した（図 6.18）．起動時に建物メタデータを解析し，そこから Thing を生成し，ThingsContiner に保持する構成になってい

[*24] https://www.playframework.com/documentation/2.8.x/Home

る．生成した Thing から TD が生成できるようになっており，サードパーティは TD とクラススキーマを参照しながら，RESTful API のエンドポイントにアクセスする．図 6.19 は，エンドポイントに対してブラウザでアクセスした際に得られる TD の例である．properties 以下に，デバイスが保持するポイントの情報が書かれている．この例では，links を辿って当該プロパティのエンドポイントを特定した後，HTTP の GET メソッドでデータの取得，PUT で書き込み，POST でデータ購読用のエンドポイントを生成が可能である．なお，参考実装においては，Authentication Service の実装は行っていない．

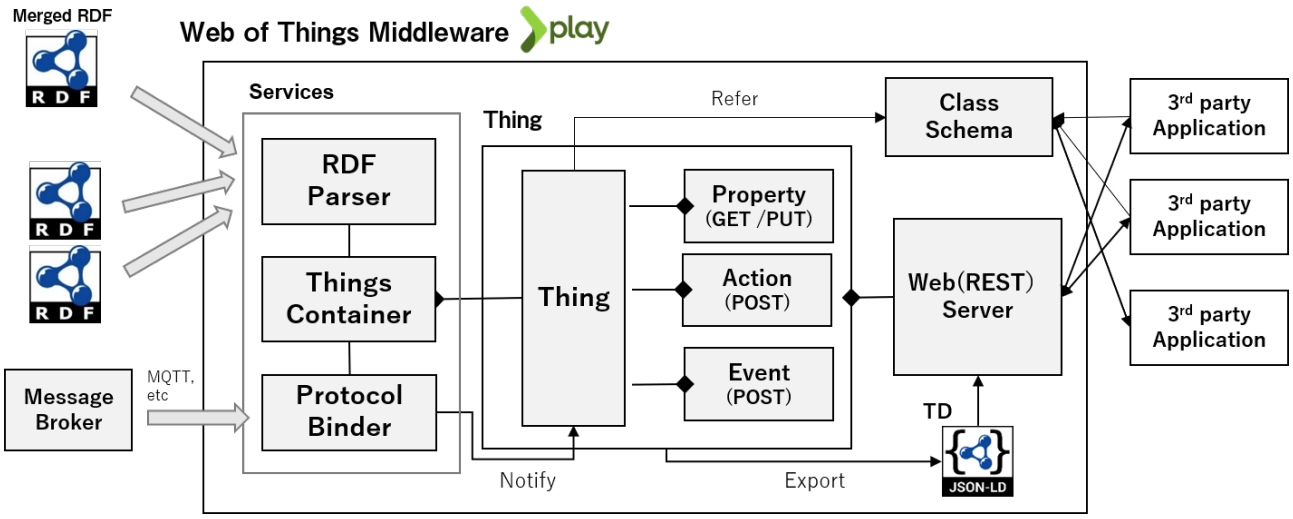


図 6.18: WoT Service の参考実装

6.6.4 評価

6.3 節で述べた設計指針に対する評価を述べる．

多様なユースケース対応する高い汎用性 ラムダ・アーキテクチャを採用したことで，パーソナルファン制御のようなリアルタイム制御 [83] や，負荷予測システムといったバッチ処理型のユースケースにもよく適合する．RESTful な疎結合型のアーキテクチャであることに加え，サービスレイヤーの機能に 6.4 節で示した多様なユースケースの要件を取り込んだことで，多様なサードパーティとの連携も容易になった．また，柔軟なテレメトリのスキーマとすることで，位置情報などスカラー値にも対応し，ビル設備以外のデータ取り込みも可能になった．それらは，同期，非同期の RESTful API によってサードパーティからのデータ取得が可能である．データ補完や抽出する周期の指定など，スマートビルのユースケースで良く用いられる共通機能をプラットフォーム側の機能として実装したことで，開発効率向上も実現している．

再利用性の高いアプリケーション BOT や Brick による簡易なデータモデルを提案・採用し，BIM やポイントリストから生成した RDF をもとに，WoT のエンドポイントを生成している．メタデータとクラススキーマも API で公開することで，BACS の専門業者以外の新規参入が容易となったといえる．スマートビルは個別性が大きいために，アプリケーションをそのまま移植することは難しいが，セマンティクスを元にビル側の空間や設備の構成が把握できるため，それらの構造を理解できれば，アプリケーションの再利用も可能と考える．

建設プロジェクトへ導入可能 本研究ではオープンシステムと PaaS を前提したことで，十分に安価な運用が可能であることが検証できた．前節における検証は，ビッグデータ処理基盤のデータ永続化に注目した検証であったが，サードパーティからの抽出のリクエストも同様の性能で処理できることから，API の運用におけるコストは大きな変動はないといえる．参考実装では，IaaS を使った実装が多かったが，スピードレイヤーやサービスレイ

```
▼ @type:
  ▼ 0:              "http://takenaka.co.jp/Mercedes_Me/EQ_House/classes/CO2-W-3"
▶ links:            […]
▼ id:               "http://kasuya.hongo.wide.ad.jp:9000/wot/Mercedes_Me/EQ_House
                    /Entrance/CO2-W-3"
  title:            "CO2-W-3"
  @context:         "https://kasuya.hongo.wide.ad.jp/schemas"
  actions:          {}
▼ properties:
  ▼ CO2-W-3:
      @type:        "Sensor"
      description:  "CO2濃度"
    ▼ topic:        "takenaka.co.jp/Mercedes_Me/EQ_House/-/Entrance/Utility/Sensor
                    /WebCNT3/CO2-W-3/CO2_Level/R"
      readOnly:     true
    ▼ links:
      ▼ 0:
          rel:      "property"
          href:     "/properties/CO2-W-3"
      title:        "CO2-W-3"
      type:         "number"
```


図 6.19: Thing Description 例（ブラウザからアクセス時）

ヤーに関しても，PaaS による機能の代替が可能である．例えば，スピードレイヤーの Data Ingest であれば，Azure IoT Hub[*25]が利用可能であり，既往のインフラとの適合の都合や，オープンソースを活用したい場合は，Azure HDInsight を用いて Apache Kafka[*26]を選択することもできる．それらはアクセス数や処理負荷によって，自由にスペックを変化させることが可能であり，サービスがスケールするまでは，最小構成での運用も可能である．

また，BACS はビルの基幹設備であるため，遠隔制御の際のトラブルを防ぐために，クラウド側との責任境界を明確にし，不正制御を防ぐための施策も求められる．futaba では BACS との連携はゲートウェイを介して行い，遠隔での更新やメンテナンスが可能な構成にしている．加えて，スピードレイヤーに配置したルールエンジンや認証の仕組みを入れたことで，不正な制御指示を防ぐ構成となっており，実際の建設プロジェクトにおいても十分に適用可能な構成と考える．

6.7　アプリケーション

本節では，futaba の参考実装を用いて構築したアプリケーションについて述べ，ユースケースの適合性について評価する．

2.3.7 節で述べた強化学習による設備制御は futaba の参考実装を使ってシステム構築されている．制御対象ビルの

[*25] https://docs.microsoft.com/ja-jp/azure/iot-hub/about-iot-hub
[*26] https://kafka.apache.org/

EQ House[*27]は延床面積 88m^2 の展示施設・住居で，電気自動車が普及した際の新しい住居の在り方を提案するコンセプトハウスであり，非常に多くのセンサを備えている（図 6.20．表 6.12）．全てのポイント数は 1304，空調関連だけで 854 ある．空調関連のポイントに関連するデバイスは 96，それらに対応するクラスは 20 であった．

図 6.21 は，それらを構造化した RDF である．色によりクラス分類されており，Building が Space を持ち，Space には Device が配置され，Device が Point を持つことが見てとれる．これらのデータは現地のゲートウェイを経由して MQTT でクラウド環境に発出され，また外部からの制御を受け入れることもできる．

表 6.12: EQ House の計測ポイント

センサ種別	説明
気象ステーション	風向・風速・雨量・気圧・日射量
多機能人感センサ	在/不在・人数・表面温度・照度
表面温度計	パネル表面温度
CO2 センサ	CO2 濃度
スマートウォッチ	加速度センサ・心拍・評価など
電力量計	電力量
マグネットセンサ	扉の開閉状況
輝度カメラ	輝度・明るさ感
カメラ	画像解析による人・行動検知
音声認識用マイク	音声認識による会話・語彙抽出
PV・蓄電池関連	充電量，放電量など
空調機関連	風量，送風温度，熱源の効率など



図 6.20: EQ House 外観
(井上登写真事務所)

我々の提示した要件をもとに HEROZ[*28]によって開発された AI は 696 の計測ポイントを毎日抽出し，それらの挙動を予測する予測用 AI の学習に利用している．同様にリアルタイムのデータも取得しながら，179 の制御ポイントを対象とした強化学習によって，省エネと快適性を指標として最適制御を行う．図 6.22 は，快適性の評価指数である PMV を，試験的に環境悪化させた状態から，簡易的な AI 制御によって快適範囲である ±0.5 の範囲まで回復させている様子を示している．なお，この際の制御対象ポイント数は 18 で，空調関連のポイントに限定していた．

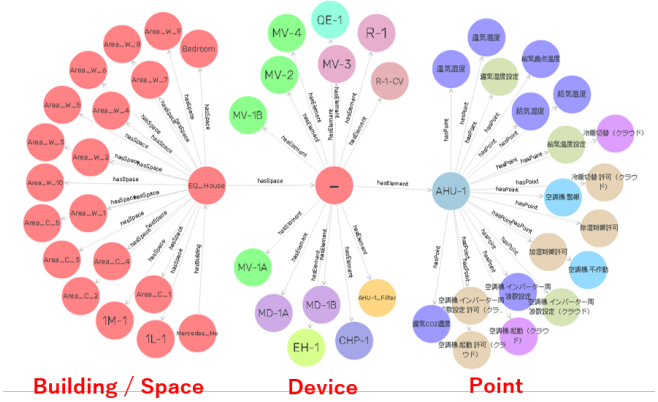


図 6.21: EQ House のセマンティクス（抜粋）

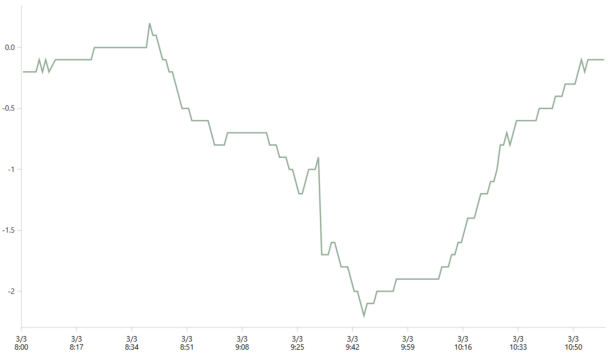


図 6.22: PMV の最適化

本物件のシステムは 2019 年 3 月から継続的に運用され，データも蓄積され続けている．本節で述べた強化学習以外にも，演出等のために外部制御を行ったり，抽出したデータセットを用いたデータ分析に活用したりしている．十分な

[*27] https://www.takenaka.co.jp/eq_house/
[*28] https://heroz.co.jp/

汎用性と可用性を持つといえるだろう．

6.8　まとめと今後の展開

本研究では，スマートビルのデータ・プラットフォームである futaba を提案した．ラムダ・アーキテクチャと WoT の仕様に基づく RESTful API を採用することで，リアルタイム制御やビッグデータ処理など多様なユースケースに対応する高い汎用性を得た．これらはセマンティクスウェブと IoT を融合させた SWoT の 1 事例といえる．また，4 章で特定したセマンティクスを用い，データモデルに組み入れることで，BACS の専門業者以外の新規参入障壁を下げ，アプリケーションの再利用性を向上させた．加えて，PaaS による機能最適化により，例えば 5 万／分のデータを想定した場合でも，バッチレイヤー部が月額 1 万円以下で運用可能であることを確認した．

本章では，参考実装によるアプリケーション例を示したが，2019 年度の国立研究開発法人新エネルギー・産業技術総合開発機構 (NEDO) の助成事業において，futaba の実用化・事業化を目指し，PaaS での再実装を進めている．これらの成果により，高度なスマートビルの普及と，多様な業種の新規参入による新たな市場の誕生と拡大を期待している．

第 7 章

スマートビルのデータ分析

本節では，6 章で述べたデータ・プラットフォームを前提とした，データ分析のアプリケーションについて述べる．それにより，データ・プラットフォームの汎用性について考察するとともに，ビルで収集可能な時系列データの性質について分析する．具体的には，ビルの中で行われる活動やセンサの動きを分析し理解するプロファイリング技術である BAP (Building Activity Profiling)，DR (Demand Response) などのユースケースに重要な負荷予測，および未知のデータ型の推定技術について述べる．

7.1 はじめに

■ビルデータのプロファイリング 2.3.8 節で述べたように，DR などの複雑な動作を伴うアプリケーションに対して，ビルのシステム全体の挙動を表すプロファイリング技術の導入が有効である．また，プロファイリング技術は AI を用いたスマートビルの遠隔制御システムの適用にも有用といえる．例えばデータセンターをはじめとするミッションクリティカルな施設に対して，信頼性の懸念から AI 制御は避けられる傾向があるが，得られたプロファイルによって現在のビルの動作状態が把握できると，そこから逸脱するような動作を避けるような制御も可能になるだろう．こうした技術は，AI や機械学習の説明可能性を向上させるという意味合いで，XAI (Explainable AI)[*1] といわれる．

■負荷予測 プロファイリング技術はビルで消費・発電されるの電力量予測にも有用といえる．正確なプロファイルは，ビルの中で行われる活動やシステムの挙動を記述しているといえるため，それらの予測精度向上にも資する可能性がある．事前に正確な負荷予測（消費電力量や熱量の予測）があると，DR 発動前に節電や増エネのための余力を準備することができる．例えば，上げ DR と呼ばれる電力負荷の増加要請が来た際，蓄電池設備の蓄電量が空に近い状態になっていれば，DR 発動時に蓄電状態にすることで増エネを図ることができる．

現状，建物で運用されている多くの負荷予測システムでは，特徴量に翌日の気象予報（気温，湿度，日射量など）を用いる．気象予測は，気象庁の分析結果をもとに，各社が提供するウェブサービスから API を介して取得することができるが，数時間おきの更新であるため，高い精度は期待できない．図 7.1 は太陽光発電量の予測を行うために，日射量について実測値と予報値を比較したものである．相関はあるといえるものの，分散が多く，予測精度は高いとはいえない．これは天気による影響もあるといえるが，これらの特徴量に依存して負荷予測を行うだけでは，十分な精度は出ないといえる．

また，将来の電力網（スマートグリッド）では，リアルタイムの電力料金の変動や緊急電力抑制措置に応じてクラウドから通達される電力制御指令に対して，分単位で応答する FastADR が検討されており [126]，それらに対応するためのリアルタイムの負荷予測が必要になってくる．

[*1] https://www.darpa.mil/program/explainable-artificial-intelligence

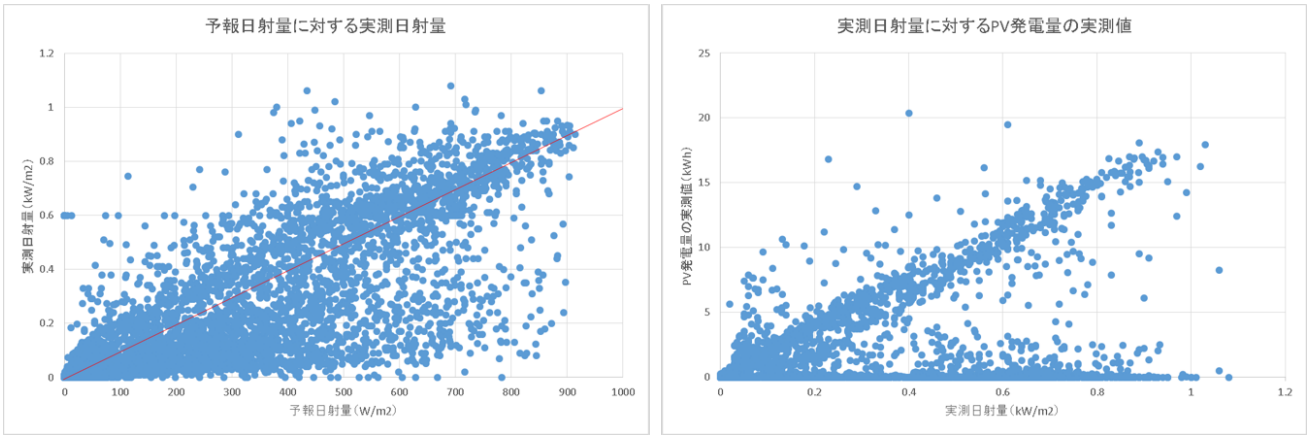


図 7.1: 日射量の予測精度検証と PV 発電量の分析（TAK 新砂ビルでの検証結果）

■BACS ポイントにおけるデータ型推定技術　BACS 構築時のよく知られた課題として，BACS のポイントと実際の機器のマッピングの間違いがある．主として施工時のヒューマンエラーが原因であり，数万ポイントを有するビルの場合，それらの校正には多大な労力がかかる．プラットフォーム側で収集されたデータを用いて誤り検出ができれば，施工側の大きな負荷低減となるといえる．例えば，蓄積されたデータを学習データとして用い，データ型毎の類似度を定義することができれば，誤り検出が可能と考える．これらの検出技術は，適切なメタデータ付与やモデリングができていないビルのデータセットにおける，データ型の推定にも有用といえる．

■目的　本章では，上述の課題解決をデータ分析と BAP と呼ぶプロファイリング手法の応用によって試みる．6 章で述べたデータ・プラットフォームに蓄積されたデータを用いたバッチ型の静的分析であるが，分析結果を用いたリアルタイム制御への応用についても考察する．

以下では，7.2 節で BAP の理論的詳細について述べ，実際のビルのデータセットを用いた分析結果を示す．7.3 節に，BAP を用いた負荷予測に関する各種検討と，その検証結果について示す．7.4 節で，BAP を用いたデータ型の推定手法の提案と，その検証結果を示す．7.5 節にまとめと今後の展望を述べる．

7.2 BAP (Building Activity Profiling)

BAP は高井 [92] と粕谷 [68] によって提案された手法で，ビル内で収集される時系列データを対象とし，それらが離散的に定義できるモード系列から出現すると仮定し，モード遷移と時系列生成パラメータによって定義するハイブリッドシステムによるプロファイリング手法である．本節では，関連研究と BAP の理論的詳細について述べ，実際のデータセットに適用した際の結果とその評価について述べる．

7.2.1 関連研究

ビルを対象としたプロファイリングを目的とした研究は少ない．Wang らは，185 棟の建物の電力量データを収集し，K-means を用いた時系列クラスタリングや，SVM による負荷予測を試みることで，それら時系列データやビルの性質を分析した [73]．人の活動による電力量の違いなど，重要な示唆が含まれているが，人手による静的な分析に留まっている．

ハイブリッドシステムの BACS への応用としては，Fazenda らによる研究がある [29]．ドア，窓，ブラインドの開閉などのコンテクストを含む熱力学的モデリングを行い，それらの遷移状態を記述することによって，シミュレーション空間における最適制御を確認した．

BAP は，教師なし学習を用いており，人手によるシステムのモデリングが不要であるために，大量のデータセットへの適用，自動化も可能である．複数センサを集約した分析も可能であるため，建物のより複雑な挙動も記述できる可能性がある．

7.2.2 モデル定義

単一センサ

BACS やプラットフォームから得られる計測データを x_t $(t = 1, 2, \dots)$ と表記する．これらのデータを適当な区間 $X_i = \{x_{t_a}, ..., x_{t_b}\}$ に区切り（以下，区分化時系列と呼ぶ），その区間に適切なモード $\mathcal{Q}_i = q_j \quad (j = 1, 2, \dots, M)$ を割り当てることによって，単一センサのモデル化を行う（図 7.2 の矢印の左側）．ただし，Mはモード数である．

区分化の方法は，ハイブリッドシステム設計の課題の 1 つであるが，ビルの制御システムにおけるユースケースの多くが 1 日毎に制御ロジックを決めているため，1 日毎に区分化を行う．なお，区分化時系列は同周期で観測またはサンプリングされたデータ列を仮定している．先行研究 [118] では，1 分周期でのデータ取得を前提としているが，日本における建物の負荷抑制（デマンドコントロール）の制御単位である 30 分毎に抽出したデータが扱いやすいため，本研究では 30 分毎（48 次元）のデータを対象として分析を進める．

以上により，1 つのセンサについて，区分化時系列と対応するモードの組み合わせである $(X_i, Q_i) \quad (i = 1, 2, \dots, D)$ の系列が得られる．ここで D はデータ計測日数である．得られたモード系列について，その遷移を確率状態オートマトンで表現し，モード q_m から q_n への遷移を $P(q_n|q_m)$ と表現する．また，モード q_j から生成されるダイナミクス $\hat{X}_j$ を，平均 $\boldsymbol{\mu}_j$，分散共分散行列 Σ_j で表される正規分布 $\mathcal{N}(\boldsymbol{\mu}_j, \Sigma_j) \quad (j = 1, 2, \dots, M)$ で表現する． （図 7.2 の矢印の右側）

複数センサの統合（センサ集合データ）

ビルには多様なセンサが設置されているが，それらを統合的に扱うことによって，センサ集合のモードとその変化によってビル全体の活動が表現できると考える．

ビルに S 個のセンサがある場合，ある 1 日のセンサ集合のモードの組は $\left(\mathcal{Q}_i^{(1)}, \mathcal{Q}_i^{(2)}, \dots, \mathcal{Q}_i^{(S)}\right) \equiv \boldsymbol{Q}_i$ と表現できる．この $\boldsymbol{Q}_i$ に適切なモードを割り当てると，センサ集合のモード $\boldsymbol{Z}_i = \omega_k \quad (\omega = 1, 2, \dots, N)$ を表現できる．

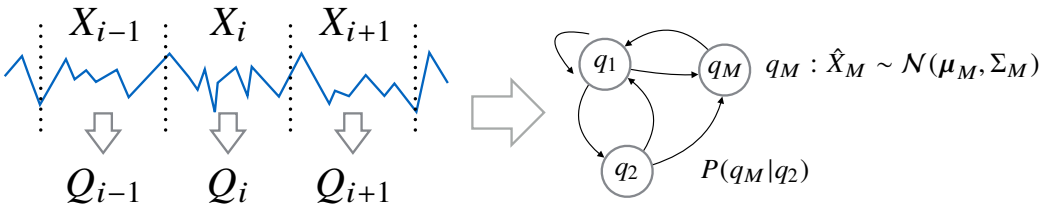


図 7.2: 単一センサデータのモデル化

すなわち，センサ集合においても，モード組とそれに対応するモードの組み合わせ $(\boldsymbol{Q}_i, \boldsymbol{Z}_i)$　$(i = 1, 2, \ldots, D)$ の系列が得られると考える．また，モード遷移についても単一センサの場合と同様に表現し，モード ω_m から ω_n への遷移確率を $P(\omega_n|\omega_m)$ と表現する (図 7.3).

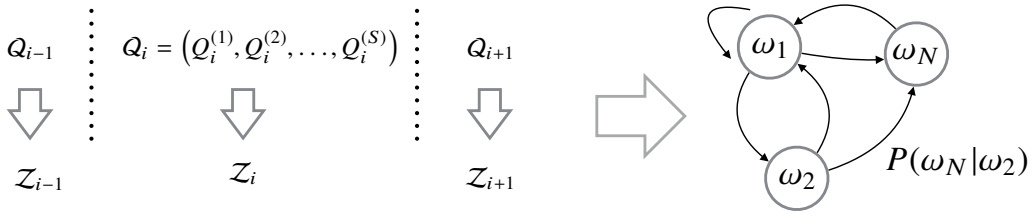


図 7.3: センサ集合データのモデル化

7.2.3　学習アルゴリズム

本節ではまず，単一センサにおけるモード推定や遷移確率，モードから生成されるダイナミクスの生成モデルの学習について述べる．その後，単一センサのモデルを統合することによるセンサ集合のモード学習について述べる．

単一センサ（図 7.4）

各センサのモード学習は以下のステップで行う．

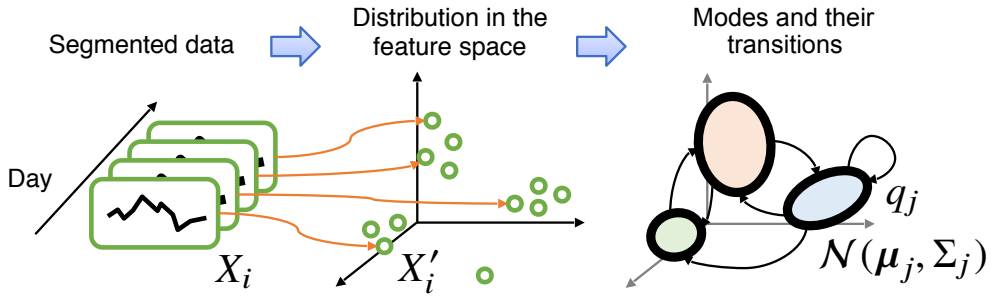


図 7.4: 単一センサのモード学習の流れ

1. **次元削減**　計算量の削減のため，各センサの区分化時系列データセット $\boldsymbol{X} = \{X_1, \ldots, X_D\}$, に対して主成分分析 (PCA: Principal component Analysis) を適用し，次元削減後のデータセット $\boldsymbol{X}' = \{X_1', \ldots, X_D'\}$ を得る．ここで $\boldsymbol{X}$ と $\boldsymbol{X}'$ の同じ添え字の要素は同じ日のデータを示すものとし，縮約する次元数は累積寄与率を用いて決

定する．

2. **クラスタリング**　得られた $\boldsymbol{X}'$ に対して，混合正規分布推定 (GMM: Gaussian mixture model) によるクラスタリングを行う．正規分布の数（クラスタ数）については，あらかじめ指定した範囲で数を変えながら GMM を適用し，ベイズ情報基準（BIC:Bayesian information criterion）を元にして決定する．GMM によるクラスタリングでは，一般的にクラスタ数と共分散の種類をパラメータとして，単純に BIC 最小になる数を選択するが，局所的な最小値の選択を防止するため，BIC をクラスタ数と共分散の種類で変化する関数と仮定し，多項式で近似した際に底となる値の周辺のパラメータを選択している．図 7.5 は，それらの計算結果を示している．縦軸は BIC であり，横軸がクラスタ数を共分散の種類を変えていった際の結果を配列に格納した際のインデックスである．

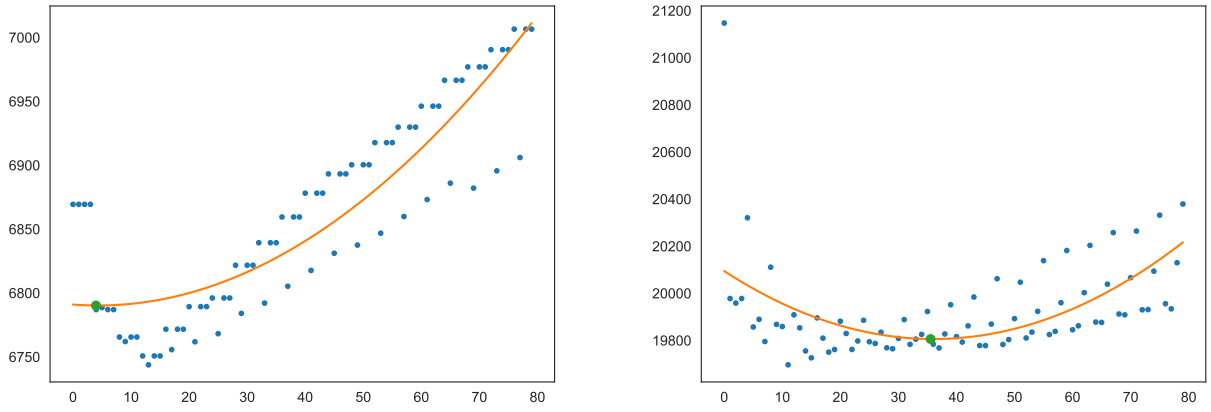


図 7.5: GMM の BIM の計算と多項式近似

ここで得られたクラスタ数がセンサデータに対して出現するモード数であり，モードの集合 $\boldsymbol{Q} = \{q_1, \ldots, q_M\}$ と各モードに属するサブセット $\boldsymbol{X}'(q_j) \subseteq \boldsymbol{X}' \quad (j = 1, \ldots, M)$ が得られる．
得られたモード q_j のダイナミクス生成パラメータである $\boldsymbol{\mu}_j, \Sigma_j$ は，$\boldsymbol{X}(q_j)$ に対応する区分時系列データである $\boldsymbol{X}(q_j) \subseteq \boldsymbol{X}$ を用いて，これらの平均と分散共分散行列を求めることによって得ることができる．ただし，Σ_j については，対角成分のみを用いることとし，その他の成分は 0 とする．これは，データ次元数に比べてサンプル数が少ない場合に σ の逆行列が得られず，後述するカルバックライブラー情報量（KLD: Kullback-Leibler divergence) が求められない場合があるためである．

3. **モード遷移確率計算**　モードの遷移確率 $P(q_n|q_m)$ を以下のように求める．

$$P(q_n|q_m) = \frac{P(q_n, q_m)}{P(q_m)} = \frac{N_{q_n,q_m}}{N_{q_m}}, \tag{7.1}$$

ここで，N_{q_n,q_m} は $\boldsymbol{X}$ の中で連続する区分時系列データ $Xi, Xi+1$ のモードがそれぞれ q_m, q_n となる数であり，N_{q_m} は $\mathrm{X}(q_m)$ の数である．

複数センサの統合：センサ集合データ（図 7.6）

複数センサの統合においても，単一センサの場合と同様の考え方でモデル化を行う．つまり，センサ集合の活動もいくつかのモードに分けられると仮定すると，全てのモード組 Q をある適当な特徴空間に投影してクラスタリングを行うことによって，モード組のモード Z を推定することができる．また，遷移確率についても同様の考え方で導出できる．

ただし，モード組 Q は質的変数であるため，単純に PCA を適用することはできない．そのため，モード間の差異を定義し，差異をもとに分布を求める多次元尺度構成法（MDS:Multi-Dimensional Scaling）を用いて，質的変数である

モード q から量的変数であるモード値 d に変換し，モード組 Q に対応するモード値の組 D を得る．これを用いてクラスタリングを行った後に，モードの遷移確率を求める．以上をまとめると以下に示すアルゴリズムとなる．

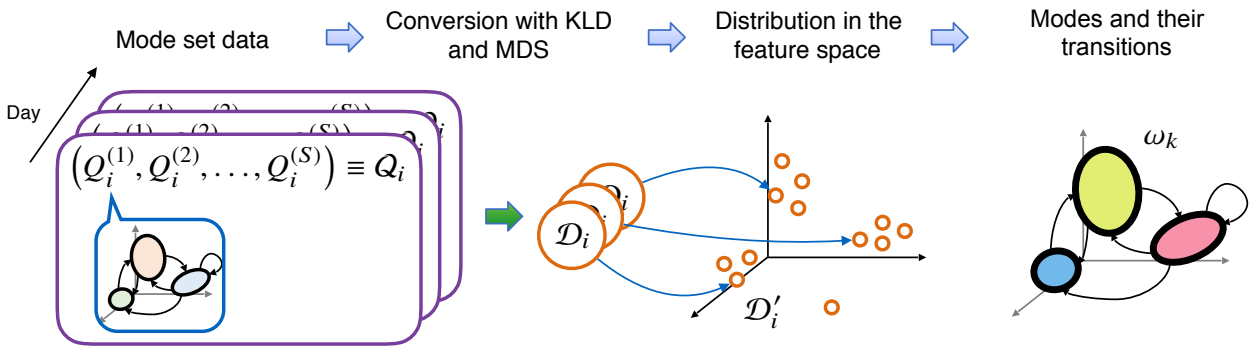


図 7.6: センサ集合のモード学習の流れ

1．各センサのモード差異行列の取得　モード間の差異は，各モードが生成するダイナミクスである $\hat{X}$ を表す正規分布のパラメータを用いて KLD によって求める．ただし，KLD には対称性がないため，モード差異を以下のように表す．

$$\delta(q_a|q_b) = \frac{D_{KL}(q_a||q_b) + D_{KL}(q_b||q_a)}{2}. \tag{7.2}$$

ここで，DKL$(qa||qb)$ はモード q_a とモード q_b の KLD である．各センサについて，モード間の差異を式 7.2 を用いて求めると，これを要素とするモード再行列である $M^{(s)}$ が得られる．

$$M_{ij}^{(s)} \equiv \delta(q_i^{(s)}, q_j^{(s)}), \tag{7.3}$$

ここで $M_{ij}^{(s)}$ は $M^{(s)}$ の i 行 j 列成分である．

2．モードからモード値への変換　式 7.3 で求めた $M^{(s)}$ を用いて，1 次元の MDS を適用し，各モードに対応するモード値を求める（式 7.4）．これにより，質的変数であったモード組 Q_i をモード値の組 D_i に変換することができる．

$$d_m^{(s)} \leftarrow f_{MDS}^{(s)}(q_{MDS}^{(s)}) \qquad (m = 1, \ldots, M). \tag{7.4}$$

3．次元削減　単一センサの場合と同様に，データセット $D = D_1, \ldots, D_D$ に対して $\acute{D}$ を得る．ただし，Dにはあらかじめ白色化を施しておく．また，D と $\acute{D}$ の各要素は対応が取れているものとし，縮約する次元数は累積寄与率を用いて決定する．

4．クラスタリングおよびモード遷移確率の計算　単一センサの場合と同様に，GMM でクラスタリングを行い，モードの集合 $\Omega = \omega_1, \ldots, \omega_N$ と各モードに属する $\acute{D}(\omega_k) \subseteq \acute{D}(k = 1, \ldots, N)$ を得る．ただし，N は得られたモードの数である．また，モードの遷移確率 $P(\omega_j|\omega_m)$ も単一センサの場合と同様に求められる．

以上により，各センサのモードを統合したセンサ集合のモード学習を行うことができる．なお，GMM によるクラスタリングの実装は，Python の scikit-learn[*2]を用いて行った．時系列データデータからクラスタを得る手法としては，

[*2] https://scikit-learn.org/stable/

時系列クラスタリングがあり，Python の ts-learn[*3]でも KShape [42] など多様なアルゴリズムが提供されているが，それらを用いた場合，十分なモード数が得られなかった．PCA による次元圧縮などの工程が重要であるといえる．

7.2.4 評価実験

我々は BAP を，小型のオフィスビルである 2 つのビル（TAK 新砂ビル，竹中工務店東関東支店）に対して適用，評価した．本節ではそれらの分析結果と考察を示す．分析対象のデータとしては，建物全体の電力消費量 (WBP: Whole Building Power) と，外部環境（温度，相対湿度，日射量）である．選定理由としては，WBP は DR の制御対象であり，外部環境は，ビル設備や人の活動に大きな影響を与えると想定されるとともに，天気予報から予測データが得られるためで，予測データと学習モデルによって翌日のモード推定が可能になるためである．

TAK 新砂ビル

TAK 新砂ビルは 2007 年に竣工したオフィスビルで，4 階建て延床面積は 3,918m^2 である．2015 年にデマンドレスポンスと電力コスト最適化への対応のための改修を行い，MSEG (Multi-Source Energy Gateway) と呼ぶ多様な直流電源を統合し，ビル側に交流電源として提供する装置と，照明・空調の遠隔制御機能の追加を行った．図 7.7 は TAK 新砂ビルで構築したモニタリングシステムであり，データ・プラットフォームに発出・蓄積されたデータを表示している．MSEG 含め，設備毎の細かい電力量計測ができていることが分かる．なお，BAP の分析対象のデータは 2017 年 4 月から 2020 年 4 月まで，サンプリング周期は 30 分毎で 1 日 48 次元の区分化時系列データを用いた．

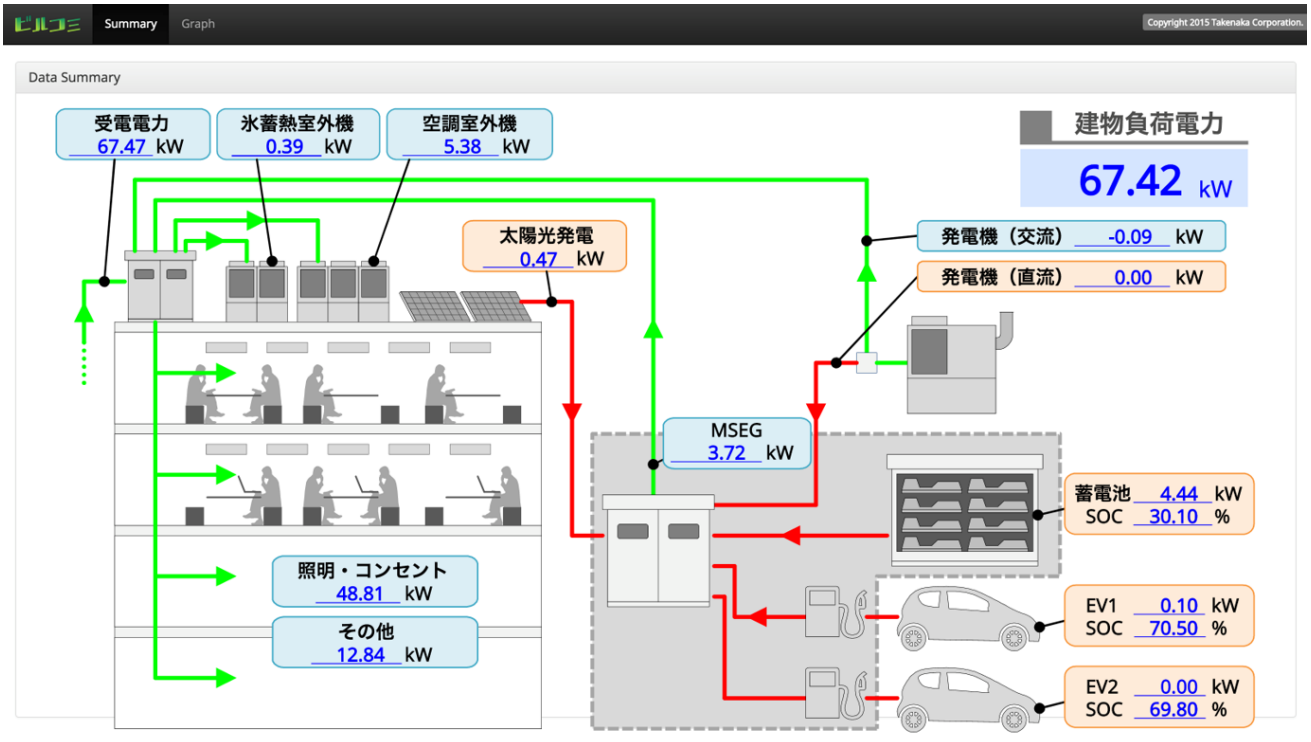


図 7.7: TAK 新砂ビルのエネルギーモニタリングシステム

[*3] https://github.com/rtavenar/tslearn

■単一センサ　BAP による WBP の分析結果を図 7.8 に示す．Distribution は PCA 適用による次元圧縮後のデータを 3 次元空間上に投影したものである．Dynamics はモード毎のダイナミクスを示しており，Histgram にモード毎の区分時系列データの数量，Mode Transition はモード遷移確率，Mode Calendar は垂直軸に曜日を，水平軸に週を置いてモードをプロットしたモードカレンダーである（白い部分はデータ欠損部分）．推定したモードに従ってそれぞれで色を合わせて表示している．

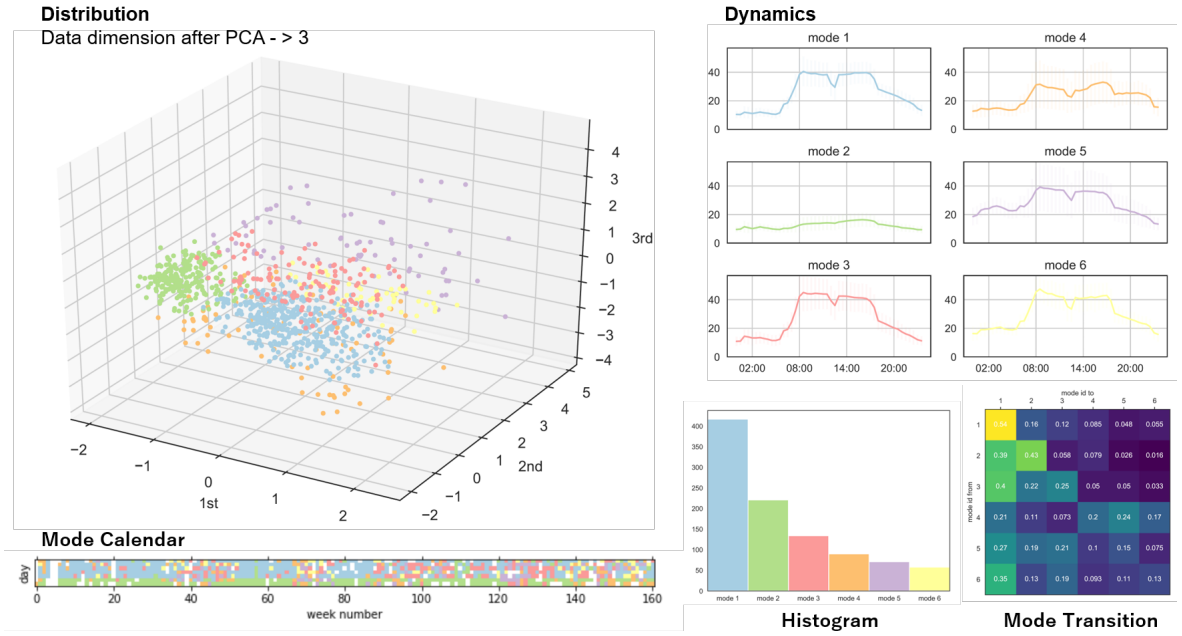


図 7.8: TAK 新砂ビルにおける WBP の分析

ここでは 6 つのモードが検出されており，モードカレンダーでは季節性がはっきりと確認できる．また，休日と平日も分離されており，負荷が少ないため明らかにモード 2 は休日といえる．平日にモード 2 が観測されている部分は，それが夏休みなどの休暇期間であると推測できる．なお，モード 4 やモード 5 がしばしば週末に観測されていることから，このビルの執務者は，週末に働くことがあるということが推測できる．

図 7.9 に外部環境データの分析結果を示す．温度の分布を確認すると，WBP と同様に季節による周期性が確認できる．しかしながら，相対湿度と日射量については，明快な周期性などは確認されず，ランダムにモードが現出しているように見える．これらは，天気による影響が支配的であるためといえる．

上記の補足として，図 7.10 にビル内部の温度センサの分析結果を示す．全部で 9 個の無線の温度センサが建物の 2 階（執務室，廊下等）に設置されており，それぞれの結果を示している．なお，分析期間は計測の都合上，2017 年 4 月から 2019 年 12 月までとなっている．室内環境は，空調（冷房，暖房）の稼働状況による執務室の環境変化が見てとれる．例えば TempSensor2 では，モード 3 やモード 6 が冷房時の挙動，モード 4 が暖房時の挙動，その他が中間期または非空調時の挙動と推測できる．

■複数センサ　図 7.11 に 5 つ（WBP と外部環境）のデータを使った複数センサでの分析結果を示す．PCA による次元圧縮では，いずれの結果でも 2 次元に投影されている．なお，統合モードにおいては，GMM による収束性が悪く，クラスタ数の上限を決める必要があった．単一センサ（WBP）と同様の季節の周期性は確認できるが，例えば休日のデータなどは正しく分解されていない．5 つのデータに加えて，図 7.10 のデータも加えて分析を行ったが，大きな変化はなかった．この問題については別途考察する．

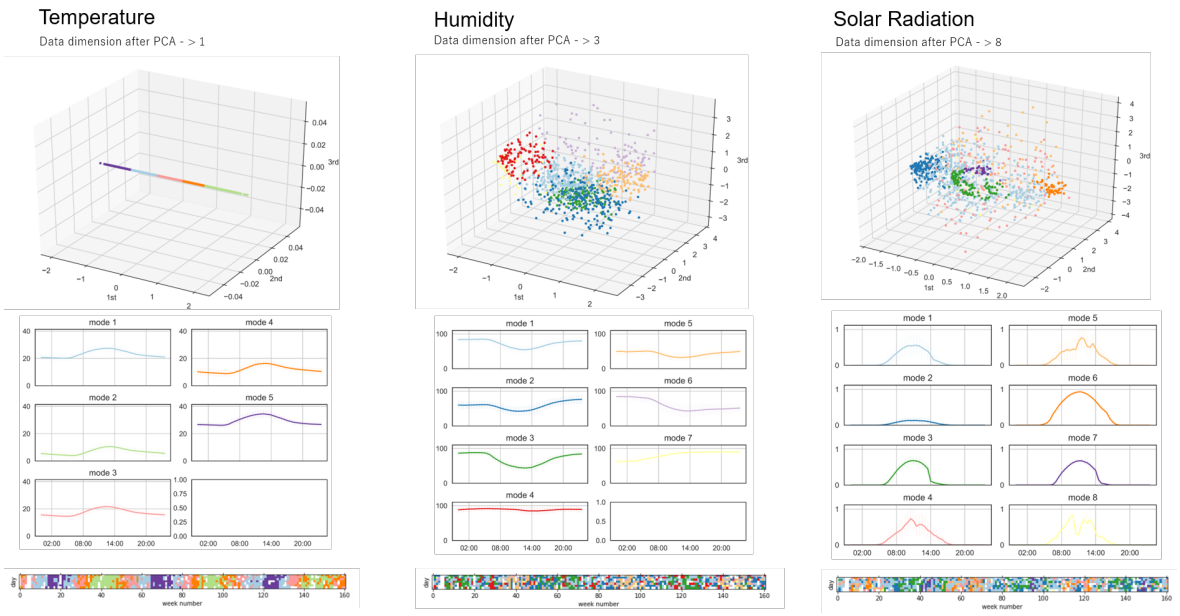


図 7.9: TAK 新砂ビルにおける外部環境データの分析

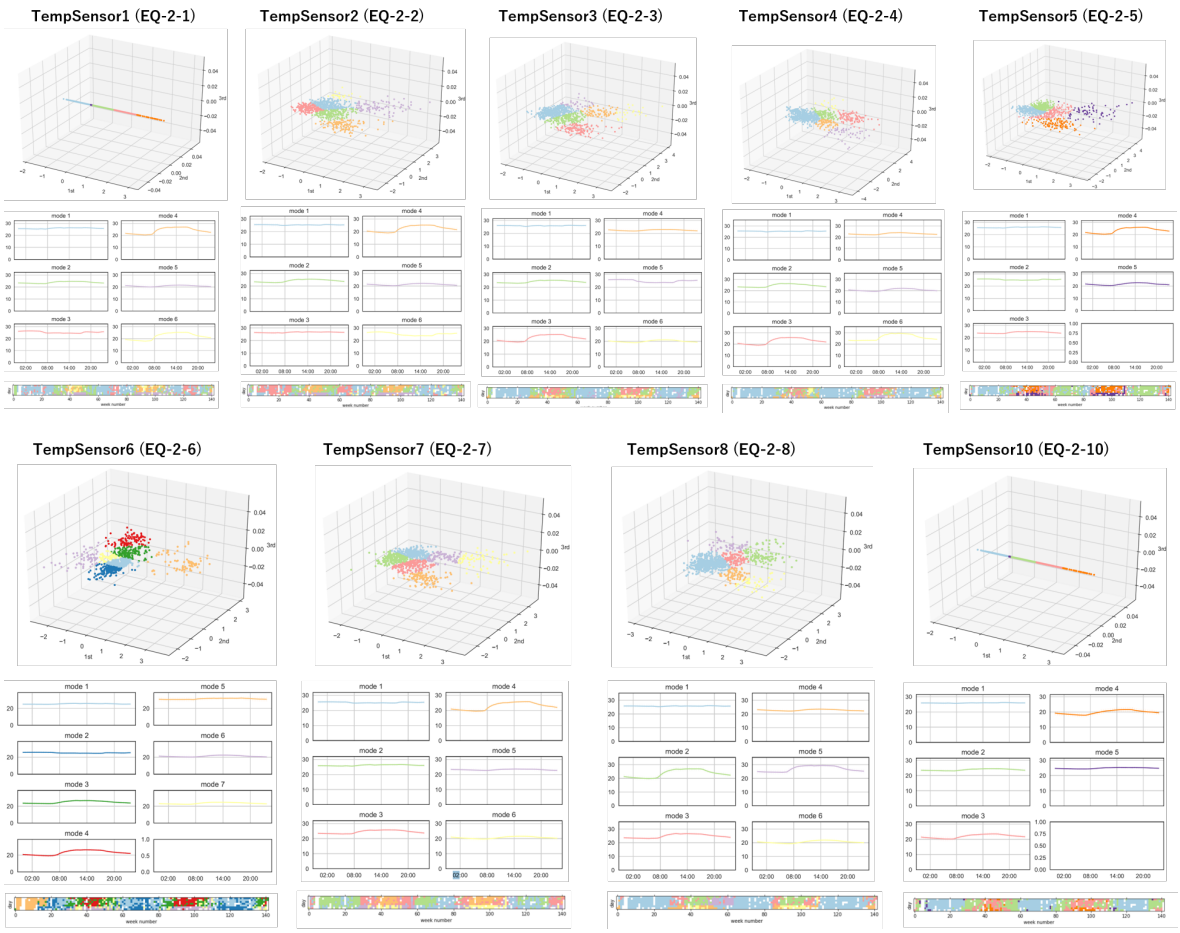


図 7.10: TAK 新砂ビルにおける室内環境データの分析

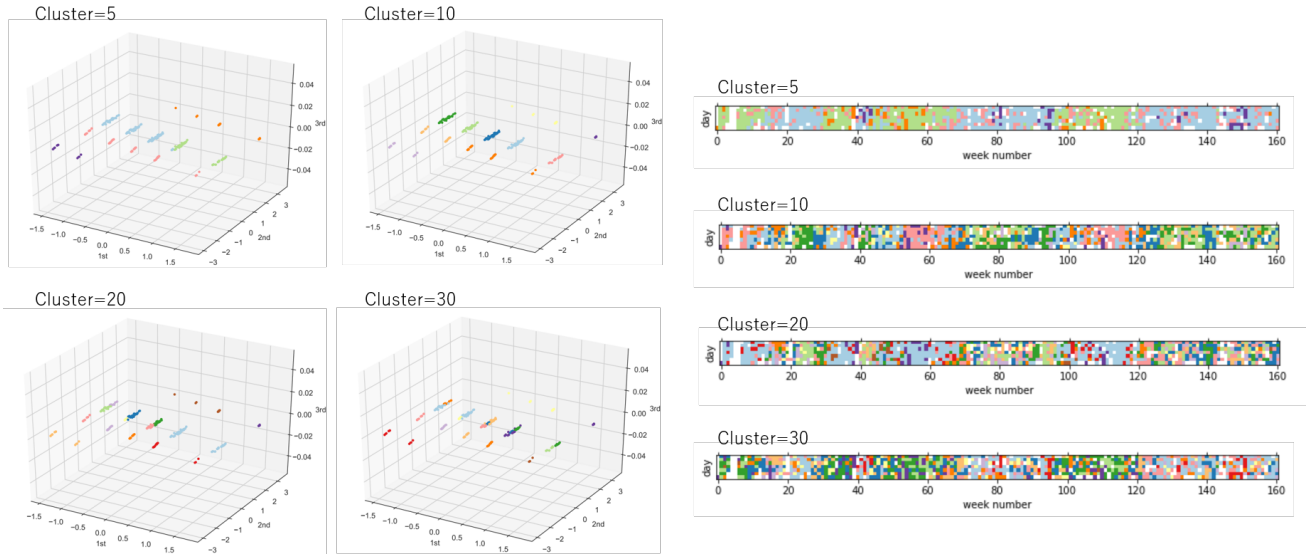


図 7.11: TAK 新砂ビルにおける複数センサ（統合モード）の分析

竹中工務店東関東支店

竹中工務店の東関東支店は，延床 1,318m^2 のオフィスビルである[*4]．2003 年に竣工し，2016 年に ZEB として大規模な改修工事を行った．具体的には，様々な省エネ設備の導入に加え，40kWh の太陽光発電システム (PV) を導入した．PV による発電量は，MSEG に含まれる蓄電池を有した設備に入力された後，そのままビルで消費されるように構成されている．これらの改修によって，東関東支店は年間の電力消費量と発電量をバランスさせる Net ZEB を実現した [67]．なお，本分析における対象データは 2017 年 1 月から 2018 年 12 月までの 2 年間である．

■単一センサ　図 7.12 に WBP の分析結果を示す．Distribution では，PCA による次元圧縮によって 2 次元に投影されていることが分かる．モードは 6 つ検出されており，TAK 新砂よりも明確に季節による周期性や，平日・休日が確認できる．例えば，モード 2 は明らかに休日であり，負荷の少ないモード 1 やモード 4 は中間期であることが推測できる．朝方にピークのあるモード 3 とモード 4 は冬季であり，これは夏場より太陽光発電による負荷低減の効果が薄いためと推測できる．

図 7.13 に外部環境データの分析結果を示す．傾向としては，TAK 新砂と同様だが，検出されているモード数は少ない．

■複数センサ　図 7.14 に上記の 5 つ（WBP と外部環境）のデータを使った複数センサでの分析結果を示す．ここでも，クラスタ数の上限を指定した際の比較について示している．単一センサ（WBP）と同様の季節による周期性は確認できる．TAK 新砂ビルとは異なり，モード数が増えるにしたがって，その解像度は上がっているように見え，平日・休日の分離もうまくいっているように見える．

■複数センサのモード数の分析　ここで複数センサの統合モードにおけるモード数について注目する．図 7.15 は，GMM でクラスタ数の上限を決めた場合のモードに対応する区分化時系列データ数の比較である．クラスタ数の上限を上げるほど，モード毎に検出される区分化時系列データは相対的に少なくなっていき，例えば数個程度だと，モードとしてはほとんど意味をなさない．そのため，ここでは 12 を適正数と仮定した．この際のモードカレンダーを図 7.16 を

[*4] https://www.takenaka.co.jp/needs/energy/service01/index.html

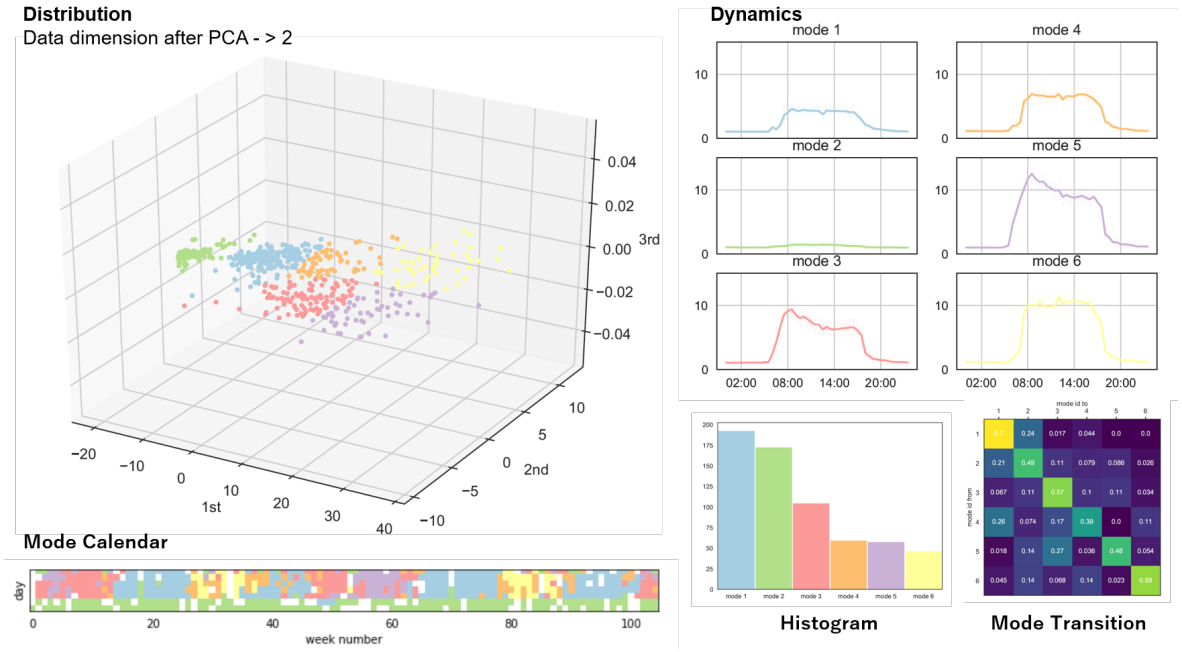


図 7.12: 竹中工務店東関東支店における WBP の分析

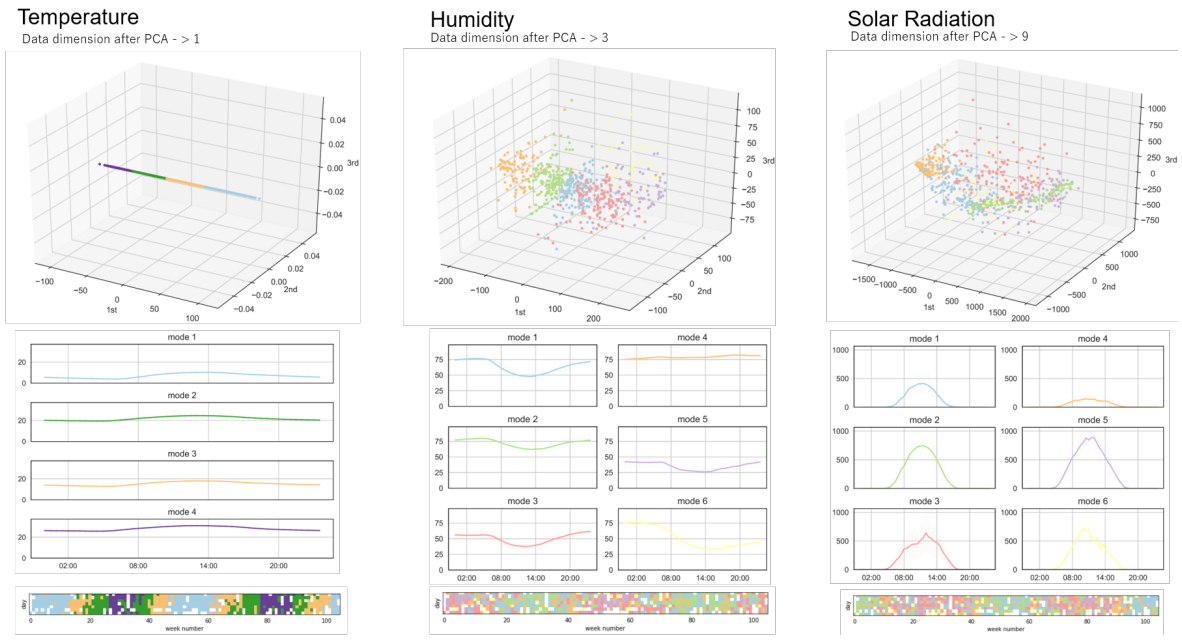


図 7.13: 竹中工務店東関東支店における外部環境データの分析

示す．また，この統合モードを仮定した場合の，WBP の生成モデルによるダイナミクスを図 7.17 に示す．これらのダイナミクスは，図 7.12 と似たトレンドが確認できるが，分散が異なっており，より解像度が高まったように見える．

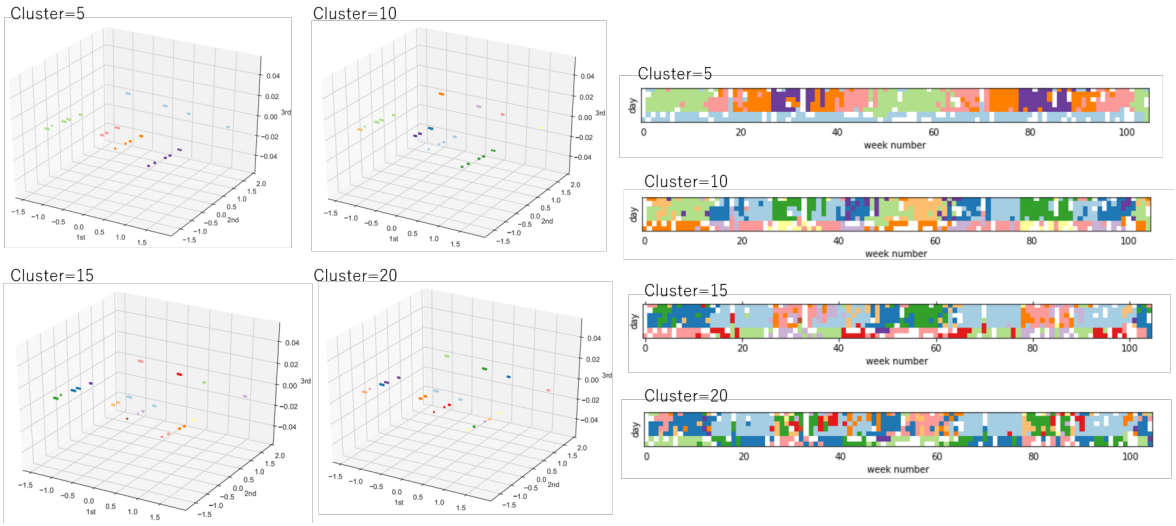


図 7.14: 竹中工務店東関東支店における複数センサ（統合モード）の分析

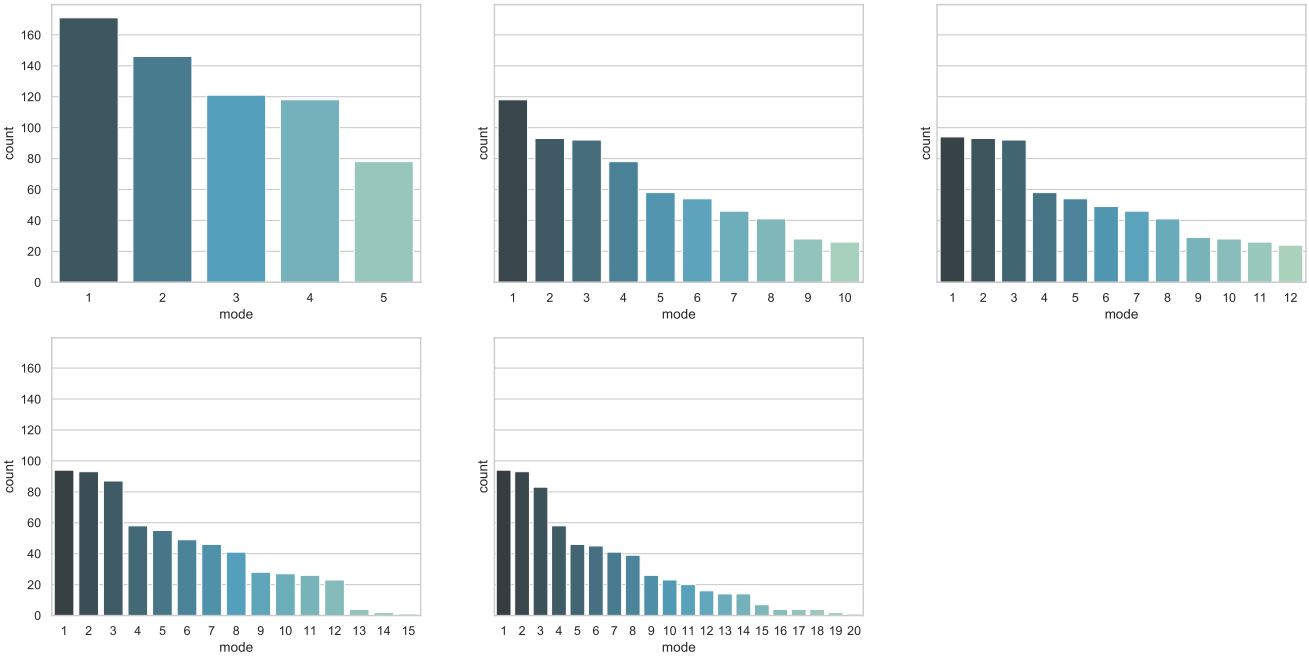


図 7.15: 統合モードにおけるクラスタ数毎のモード数比較

7.2.5　BAP による分析のためのシステム設計と実装

BAP によるプロファイル結果を用いたシステム運用のためには，データ・プラットフォーム (futaba) と連携し，定期的なバッチ処理にてモデルを更新し，天気予報などを用いて翌日や直近のモードを推定する必要がある．そのためのシステムを PaaS を用いて実装した．

■設計　図 7.18 に futaba による BAP 計算モジュールのシステム構成を示す．当該モジュールは，運用コスト低減や可用性向上を意図し，PaaS を前提に設計した．プラットフォーム上に蓄積されたビッグデータを取得できる非同期 API を用いて，定期的に分析対象データの取得を行い，BAP を適用した後，得られたモード（単一、複数）をリアル

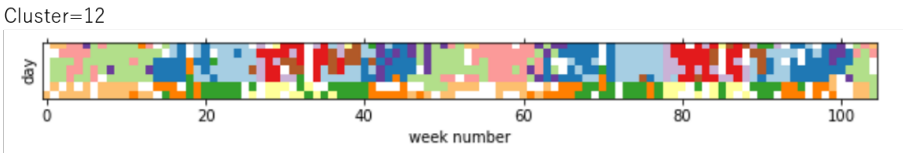


図 7.16: 統合モード（クラスタ数 12）におけるモードカレンダー

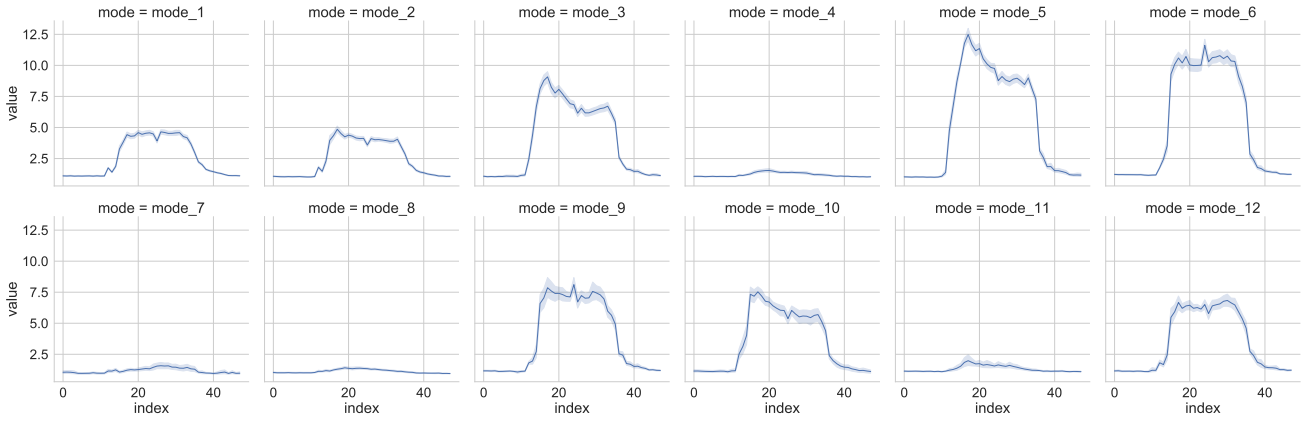


図 7.17: 統合モードの対象から抽出した WBP 時系列におけるダイナミクス

タイムのデータ取得・書込みが可能な WoT API を介して，対象の Thing（センサ，デバイス，空間）に対して書込みを行う．なお，このために書込み先の Thing が持つクラススキーマには，読み書き可能な Mode というプロパティを設定している．実際の運用では，BAP の分析対象の Thing に対応するクラススキーマは，図 7.19 に示す BaseMode を継承して作成する．

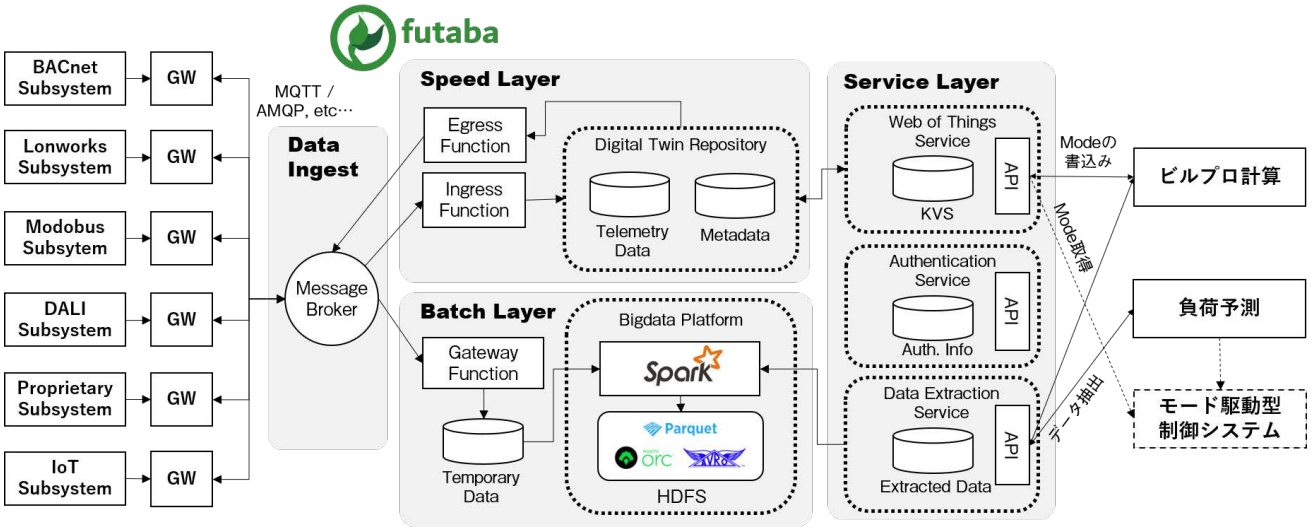


図 7.18: futaba を用いた実装イメージ

図 7.20 に単一センサの場合のデータ処理のパイプラインを示す．Azure Machine Learning Pipeline[*5]を用いて，データの前処理，特徴量抽出と次元圧縮，クラスタリングやモード推定，遷移確率の推定といった処理の流れをコン

[*5] https://docs.microsoft.com/ja-jp/azure/machine-learning/concept-ml-pipelines

図 7.19: BAP 適用を前提としたベースクラスのスキーマ

```
{
  "@context": "https://www.classschema/schema",
  "id": "BaseMode",
  "@type": "Base",
  "tag": ["bap"],
  "properties": {
    "Mode": {
      "description": "モード値",
      "isVirtualProperty": true,
      "readOnly": false,
      "type": "integer",
      "minimum": 0,
      "maximum": 10,
      "forms": {
        "op": [
          "readproperty",
          "writeproperty"
        ]
      }
    }
  },
  "actions": {},
  "events": {}
}
```

ポーネント化することで，将来的なロジック更新を容易にしている．また計算中の中間成果物を Azure Blob Storage, SQL Database などに保存・永続化することで，後の分析を可能にした．複数センサの場合は，単一センサの計算結果を使ってモード差異行列と MDS によるモード値を求める計算モジュールを追加し，それ以降は同様のパイプライン処理を行う．

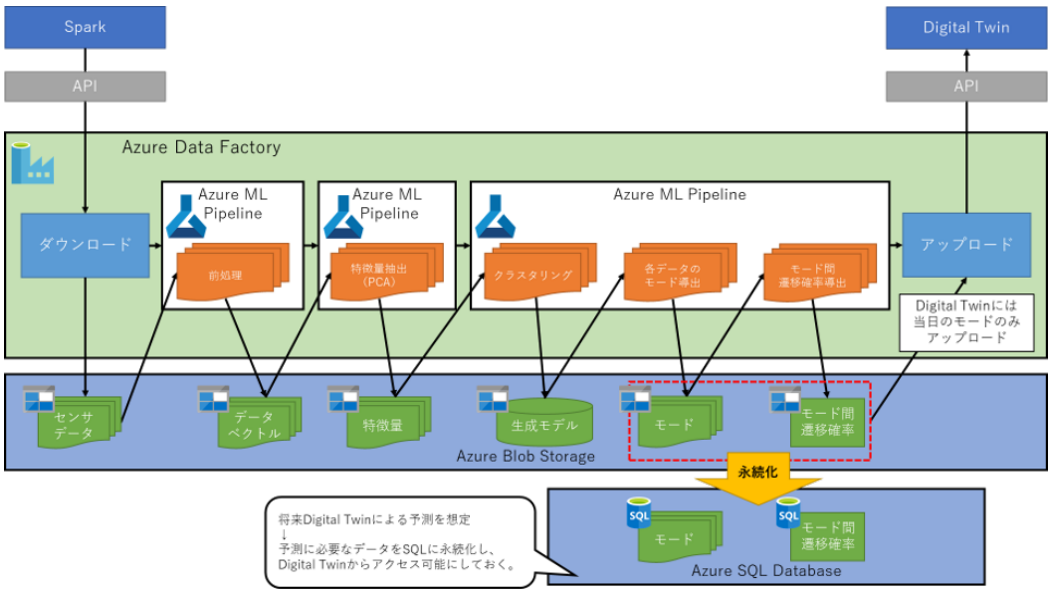


図 7.20: 単一センサの場合のシステム構成

ここでは，Azure Data Factory[*6] と呼ばれるデータの移動・変換を自動化するデータ統合サービスを用いて，RESTful API からの時系列データの取得と，推定した翌日のモード情報の書き込みを実現している．なお，ここでの翌日のモー

[*6] https://azure.microsoft.com/ja-jp/services/data-factory/

ド推定は遷移確率に従って，確率的に導出することにしているが，容易に導出のアルゴリズムを変更することができる．例えば，過去のモード列と天気予報，曜日などを特徴量とした予測などが想定できる．

■実装と評価　構築したパイプラインの計算性能の評価のために，試験的なデータを用いて計算時間の計測を行った（図 7.21，図 7.22）．Azure Machine Learning Pipeline は，実際の処理部分は Python (Jupyter Notebook) で記述し，パイプラインの起動時に実行環境ごとデプロイされる仕様になっている．図中の黄色部分がそれら PaaS の挙動に関わる部分で，赤色部分が Python のコード実行部分である．単一，複数センサそれぞれ一時間程度かかっており，その多くがが PaaS に挙動に関わるところであることが分かる．夜間のバッチ処理を前提とした処理ではあるが，処理対象ポイントが増加すれば朝までに処理が終わらないこともありうる．パイプラインの機能粒度の変更や，生成モデルの計算を毎日行わないといった最適化や運用上の工夫が必要といえる．なお，図中に赤く示された以外は PaaS 動作の準備期間であるため課金は発生しない．そのため運用コストについて十分に小さくすることはできるといえる．

処理内容		所要時間	開始時間	終了時間
MLPipeline	ステップ			
DataFactoryPipeline全体		0:53:14	0:00:00	0:53:14
データダウンロード等		0:01:27	0:00:00	0:01:27
前処理		0:10:47	0:01:27	0:12:14
	不明	＊＊＊	＊＊＊	＊＊＊
	jobをキューへ	0:03:24	0:02:02	0:05:26
	前処理	0:02:54	0:05:26	0:08:20
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:18	0:09:29	0:09:47
	不明	＊＊＊	＊＊＊	＊＊＊
特徴量抽出		0:17:09	0:12:16	0:29:25
	不明	＊＊＊	＊＊＊	＊＊＊
	jobをキューへ	0:10:35	0:12:51	0:23:26
	特徴量抽出	0:03:08	0:23:26	0:26:34
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:19	0:26:44	0:27:03
	不明	＊＊＊	＊＊＊	＊＊＊
モード導出		0:23:10	0:29:27	0:52:37
	不明	＊＊＊	＊＊＊	＊＊＊
	jobをキューへ	0:10:17	0:30:02	0:40:19
	クラスタリング	0:03:19	0:40:19	0:43:38
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:18	0:43:49	0:44:07
	不明	＊＊＊	＊＊＊	＊＊＊
	jobをキューへ	0:00:06	0:44:44	0:44:50
	モード導出	0:00:51	0:44:50	0:45:41
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:19	0:45:54	0:46:13
	不明	＊＊＊	＊＊＊	＊＊＊
	jobをキューへ	0:00:05	0:46:20	0:46:25
	遷移確率導出	0:00:45	0:46:25	0:47:10
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:18	0:47:24	0:47:42
	jobをキューへ	0:00:06	0:47:47	0:47:53
	モード推定	0:00:42	0:47:53	0:48:35
	jobをキューへ？	0:00:47	0:47:48	0:48:35
	永続化情報準備	0:00:54	0:48:35	0:49:29
	データ移動	0:00:19	0:49:15	0:49:34
	不明	＊＊＊	＊＊＊	＊＊＊
	データ移動	0:00:17	0:49:43	0:50:00
	不明	＊＊＊	＊＊＊	＊＊＊
永続化、データアップロード等		0:00:37	0:52:37	0:53:14

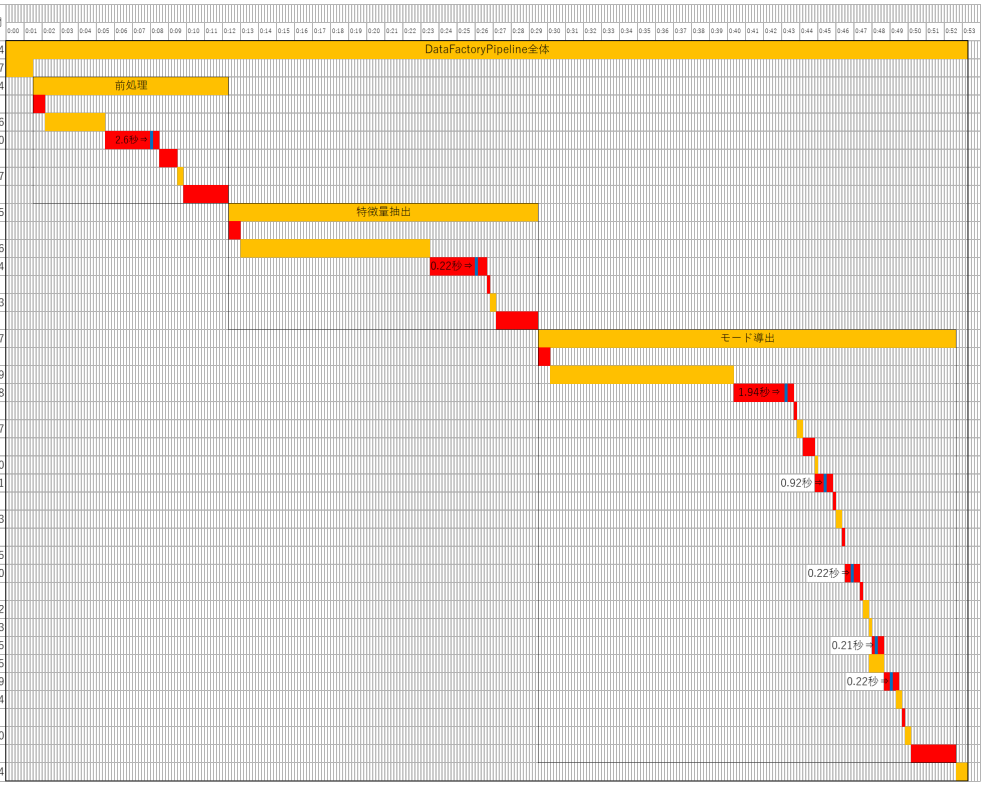
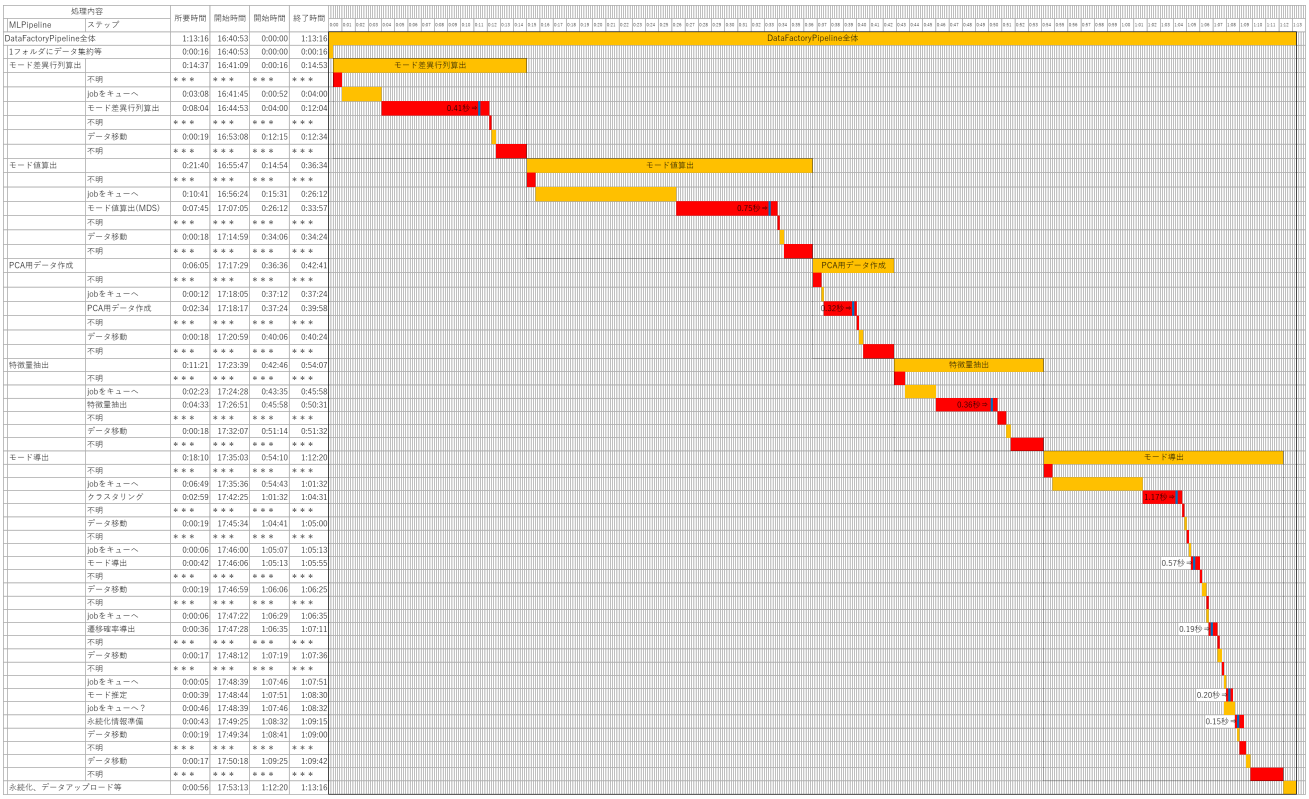


図 7.21: 単一センサの場合の計算時間評価



処理内容		所要時間	開始時間	開始時間	終了時間
MLPipeline	ステップ				
DataFactoryPipeline全体		1:13:16	16:40:53	0:00:00	1:13:16
1フォルダにデータ集約等		0:00:16	16:40:53	0:00:00	0:00:16
モード差異行列算出		0:14:37	16:41:09	0:00:16	0:14:53
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:03:08	16:41:45	0:00:52	0:04:00
	モード差異行列算出	0:08:04	16:44:53	0:04:00	0:12:04
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:19	16:53:08	0:12:15	0:12:34
	不明	* * *	* * *	* * *	* * *
モード値算出		0:21:40	16:55:47	0:14:54	0:36:34
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:10:41	16:56:24	0:15:31	0:26:12
	モード値算出(MDS)	0:07:45	17:07:05	0:26:12	0:33:57
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:18	17:14:59	0:34:06	0:34:24
	不明	* * *	* * *	* * *	* * *
PCA用データ作成		0:06:05	17:17:29	0:36:36	0:42:41
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:00:12	17:18:05	0:37:12	0:37:24
	PCA用データ作成	0:02:34	17:18:17	0:37:24	0:39:58
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:18	17:20:59	0:40:06	0:40:24
	不明	* * *	* * *	* * *	* * *
特徴量抽出		0:11:21	17:23:39	0:42:46	0:54:07
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:02:23	17:24:28	0:43:35	0:45:58
	特徴量抽出	0:04:33	17:26:51	0:45:58	0:50:31
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:18	17:32:07	0:51:14	0:51:32
	不明	* * *	* * *	* * *	* * *
モード導出		0:18:10	17:35:03	0:54:10	1:12:20
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:06:49	17:35:36	0:54:43	1:01:32
	クラスタリング	0:02:59	17:42:25	1:01:32	1:04:31
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:19	17:45:34	1:04:41	1:05:00
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:00:06	17:46:00	1:05:07	1:05:13
	モード導出	0:00:42	17:46:06	1:05:13	1:05:55
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:19	17:46:59	1:06:06	1:06:25
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:00:06	17:47:22	1:06:29	1:06:35
	遷移確率導出	0:00:36	17:47:28	1:06:35	1:07:11
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:17	17:48:12	1:07:19	1:07:36
	不明	* * *	* * *	* * *	* * *
	jobをキューへ	0:00:05	17:48:39	1:07:46	1:07:51
	モード推定	0:00:39	17:48:44	1:07:51	1:08:30
	jobをキューへ？	0:00:46	17:48:39	1:07:46	1:08:32
	永続化情報準備	0:00:43	17:49:25	1:08:32	1:09:15
	データ移動	0:00:19	17:49:34	1:08:41	1:09:00
	不明	* * *	* * *	* * *	* * *
	データ移動	0:00:17	17:50:18	1:09:25	1:09:42
	不明	* * *	* * *	* * *	* * *
永続化、データアップロード等		0:00:56	17:53:13	1:12:20	1:13:16

図 7.22: 複数センサの場合の計算時間

7.2.6　考察

本節では，BAP のモデル化と実装についての理論を示し，ビル 2 棟に対して適用した結果を示した．また，futaba において BAP を自動的計算するシステムの実装とその評価について述べた．

個別のセンサの場合，分析結果であるモードは，分析対象センサの一般化した挙動を示しており，ダイナミクスとモードカレンダーをもとに，施設管理者がビルの挙動を説明するのに有益といえる．従って翌日のモードが適切に予測され，そのモードを前提とした AI 制御であれば，施設管理者としても受け入れやすい．XAI として十分な表現力があるといえるだろう．

複数センサの統合モードは，東関東支店のように意図した結果となる場合もあれば，TAK 新砂ビルのように追加でセンサを足したとしても，平日・休日の分離が十分でないという結果になることもあった．これは TAK 新砂ビルが，東関東支店ほど休日と平日のデマンドの差が大きくなかったり，明快でないなどの理由が考えられる．また，これらの結果は MDS により計算したモード値，すなわちモード間の距離にも大きく依存しているように思われる．従って，KLD 以外の距離関数の導入や，統合対象のセンサ種別についても，今後詳細な検討が必要といえる．次節で検討するが，これらのモードの解像度が上がるほど，以降に示す負荷予測などのビル設備の挙動予測の精度が上がる．

BAP の計算モジュールは，futaba の API のユースケースによく適合しており，毎日のバッチ処理によって，モード計算を自動化し，また分析結果を WoT API を用いて Thing のプロパティとして設定，更新することができる．PaaS を用いているため，従量課金による運用コストの低減も期待できるが，現状では動作時間に課題があるといえる．

7.3 負荷予測技術

本節では，DR などの複雑な挙動を示すスマートビルのアプリケーションに必要なリアルタイム負荷予測技術についての検討を行う．

ビル制御で必要な負荷予測としては，翌日予測とリアルタイム予測の 2 つがある．翌日予測は天気予報から取得した外部環境の予測データなどを特徴量とし，翌日の負荷予測を行うもので，翌日の運転パターンやモード決定のために，前日中に行われる．なお，天気予報サービスは，6 時間ごとに更新されていることが多く，そのため前日 21 時か翌日の 3 時以降に行われることが多い．

リアルタイム予測は，DR などのユースケースに対応するために必要なリアルタイムの負荷予測であり，概ね数分後から数時間後の予測を対象とする．ここでは前日予測よりも，より高精度な予測を実現するため，BAP によるプロファイリングの結果であるモードを潜在変数とした時系列モデルの適用を試みる．

7.3.1 先行研究

電力量の負荷予測については，ビルの種類や予測する期間によって様々な手法が存在している [4]．近年では，サポートベクターマシン（SVM）や ANN (Artificial Neural Network) などの AI・機械学習手法や，自己回帰和分移動平均 (ARIMA) などの従来的な時系列モデルを組み合わせた手法が普及してきている．また，Long Short Term Memory (LSTM) [53, 76] のような深層学習を応用した研究もある．Zheng らは，k-means を用いて得られたクラスターの平均値を特徴量として用いて，結果の精度を向上させた [77]．一方で，BIM データを用いた環境シミュレーションによって，エネルギー負荷を計算する方法もある [33, 56].

また，人の活動に注目した時系列分析としては，Tayler らが提案している手法があり [70]，Prophet[*7]としてライブラリも公開されている．Facebook におけるログ分析ノウハウをまとめたもので，時系列予測の訓練を受けていないアナリストでも容易に使えることを念頭に作られた．トレンド，季節周期性，休日に分解した時系列モデルを採用しており，季節周期性として年，月，週（曜日），日単位での分析も容易に行える．独自に定義したイベントや休日効果を含めることも可能で，データの振る舞い変化点検出やハイパーパラメータの推定なども自動化されている．細かな季節性のモデリングは，ビル内の活動にもうまくフィットするといえるが，同ライブラリを用いて検証を行ったところ，電力量がマイナスになるなど期待した値が得られなかった（図 7.23）．時系列の構成要素を図 7.24 に示す．上からトレンド，休日，曜日，日となっており，例えば中間期は少ない，休日・週末は少ない，日中に負荷が増加するということが読み取れるが，図 7.23 では日中の負荷変動に追従できておらず，週末の精度も悪い．このような周期性に注目する手法は，電力量予測への応用には不向きであることが分かる．

既往研究の多くは，翌日〜数カ月先といった長期的な予測を目的としており，リアルタイム予測に関する研究は少ない．Chae らは，15 分単位でのビルの電力消費量を予測するために，ベイズ正則化アルゴリズムを用いた ANN に基づく短期的なビルのエネルギー使用量予測モデルを提案・評価しており，商業ビルの複合施設を対象としたテストケースにおいて，15 分間隔の電力消費量と 1 日のピーク電力使用量を予測できることを確認している．しかしながら，これらのデータセットは月別や平日・休日毎に分けて学習を行っているのみで，より潜在的なビル内の活動については考慮されない．

[*7] https://facebook.github.io/prophet/

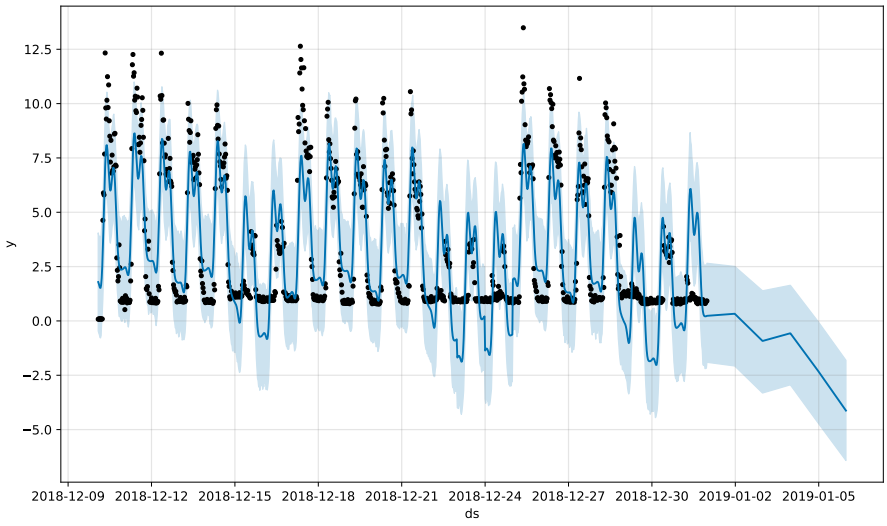


図 7.23: Prophet による時系列分析例（東関東支店の WBP）

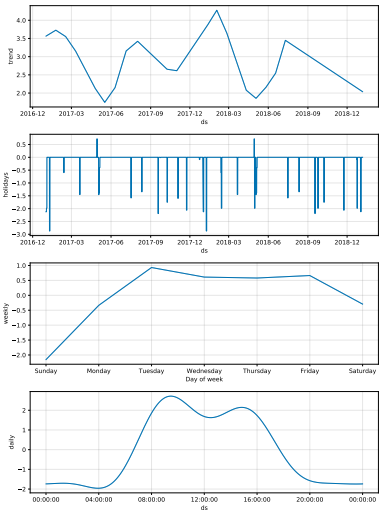


図 7.24: 時系列の構成要素

7.3.2　提案手法

我々は BAP による分析結果であるモードを用いた，時系列モデルを用いた負荷予測技術について提案する．モードは区分化時系列毎のビルやセンサの状態を表しているといえるため，負荷予測にも有用な特徴量となる可能性がある．それらを前日予測，リアルタイム予測それぞれに対して適用する．これらの要素技術を用いて，図 7.17 に示される「モード駆動型制御システム」が構成される．

モード駆動型制御システムの動作イメージを図 7.25 に示す．まずデータ・プラットフォームに蓄積されたセンサ等のデータを抽出し，モード推定のためのモデルを生成し，推定されたモード群を集約した統合モード（Integrated Mode）のための学習モデルも生成する．翌日の天気予報を用いて，翌日の電力消費量を予測するとともに，予測結果と天気予報を用いて翌日のモード群を推定する．推定されたモード群をもとに，翌日の統合モードを推定し，それらのプロファイルを元にリアルタイムのビル制御コンポーネントを動作させる．このコンポーネント中でモードを用いたリアルタイム予測を行い，DR 等に必要な制御を行う．

以降では，提案手法の適用の背景について述べた後，実際のデータを用いて検証を行う．

BAP を用いた前日予測

まずは抽出したモードが，負荷予測の特徴量として有効であるか評価する [68].

データセットとしては，TAK 新砂ビルの 2016 年 11 月から 2018 年 10 月までのデータ，東関東支店の 2017 年 1 月から 2018 年 12 月までのデータで，目的変数を 7.2.4 節でも用いた WBP とし，特徴量として，前日の WBP(ラグ特徴量)，外部環境データ（温度，相対湿度，日射量），曜日インデックス（月曜日〜日曜日），休日フラグ，24 時間を 360° とした際の sin/cos の値を用いた．ここで，当該データセットにおいて推定された WBP モードのダイナミクス（平均値）を加えることで，特徴量として有効か検証した．なお，予測に用いた外部環境データは天気予報ではなく実データである．学習・予測のアルゴリズムには事前の比較検討で最も性能が良かった RandomForest を採用し，ハイパーパラメータのチューニングには Optuna[*8]を用いた．

図 7.26 に予測結果を示す．ここでは 1 週間分のデータを予測対象として評価しており，実データ（Actual Energy），

[*8] https://preferred.jp/ja/projects/optuna/

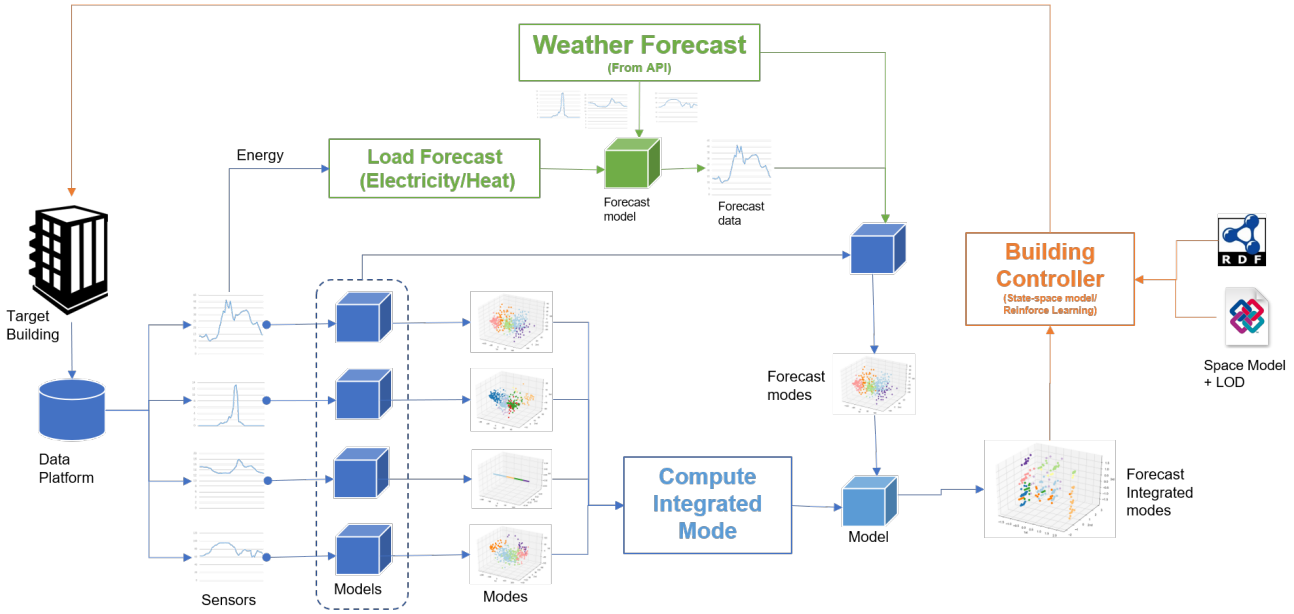


図 7.25: モード駆動型制御システムの動作イメージ

予測データ（Forecasted Energy），モード平均値（Mode Means）についてプロットした．表 7.1 に示すように，いずれの評価においても特徴量としてモードを用いることで，ほとんどの評価指標において精度向上が確認できた．従って，翌日のモードが適切に予測できれば，電力消費量の予測精度の向上が期待できる．

表 7.1: 負荷予測の精度比較

Label	**RMSE**	**MAE**	R^2[%]	**MAPE**[%]
(1) Without Mode	2.73	2.1	90.0	13.31
(2) With Mode	2.29	1.7	93.0	11.24
(3) Without Mode	0.92	0.55	91.0	21.51
(4) With Mode	0.81	0.48	93.0	19.3

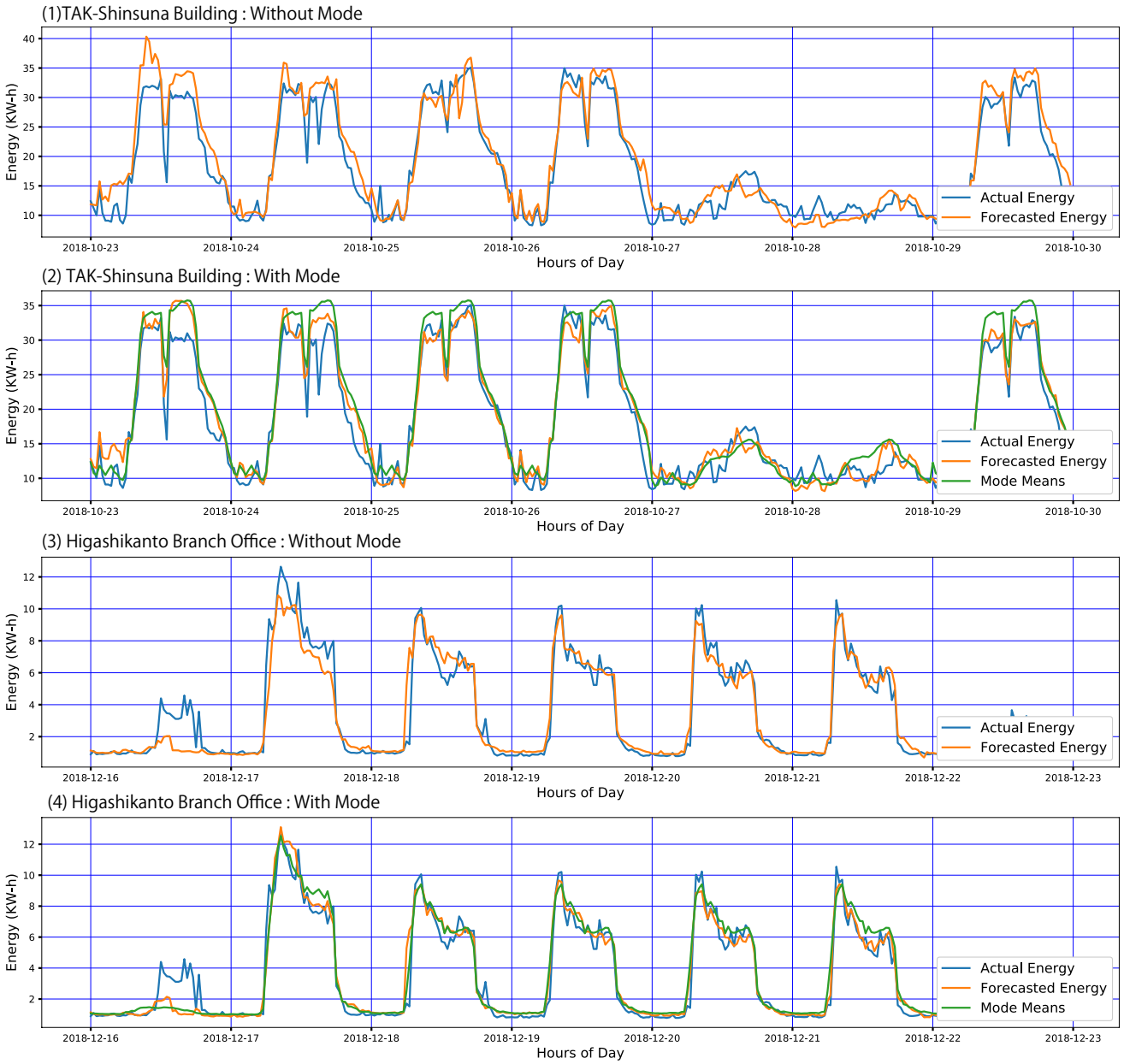


図 7.26: モードを特徴量とした際の負荷予測の精度評価

時系列分析手法の適用

■事前調査　負荷予測のアルゴリズムについて，近年では ANN などの機械学習・人工知能技術の適用が増えてきているが，もし BAP のようにある程度の動作が事前に予想できている場合，時系列の生成過程に関して何らかの性質や構造を仮定する時系列モデルや，状態の時系列変化と，その状態から観測される値に分解する状態空間モデルの適用が有効である．

図 7.27 に，TAK 新砂ビルの WBP を対象とした状態空間モデルを用いた事前検証結果を示す．適用手法は，説明変数に平日と休祝日を考慮したフラグを用いた動的線形モデル (DLM: Dynamic Linear Model) である．想定した季節周期性は 1 日であり，2015 年 2 月 1 日から 2 月 24 日までのデータを用いてモデリング，20 日・21 日，27 日・28 日を予測している．なお，DLM の実装・評価には R の DLM パッケージ[*9]を用いた．

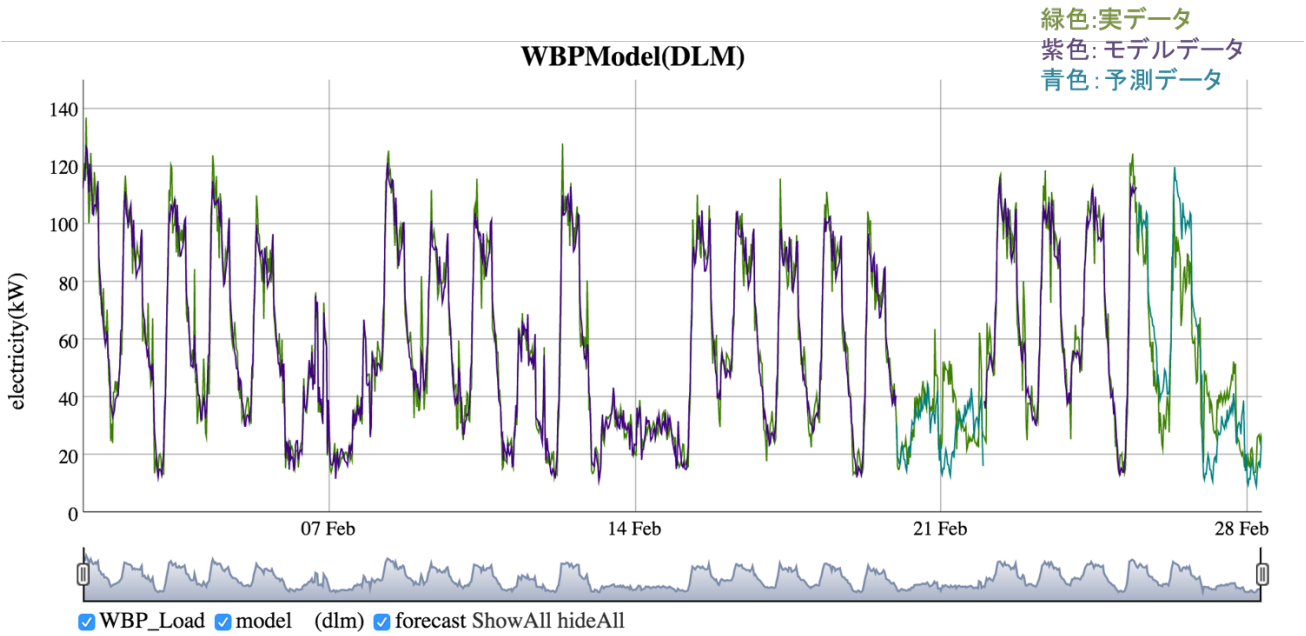


図 7.27: DLM による負荷予測

表 7.2 に DLM と ANN による予測精度の比較を示す．休祝日モデルは図 7.27 と同様で平日・休祝日を説明変数として用いたモデル，休祝日分離モデルは平日とそれ以外を別のモデルとして学習するものである．それらを 2 時間後までと，2〜4 時間後までの結果についての予測誤差を評価した．状態空間モデルは，観測されない潜在変数を更新することで直近の予測精度を上げていくので自明といえるが，直近の予測に関しては ANN よりも予測精度が優れていることを確認した．なお，ARIMA モデルに祝日効果を入れた SARIMAX(Seasonal Auto-Regressive Integrated Moving Average with eXogenous variables model) モデルとの比較も行ったが，上記のデータセットにおいては DLM の方がフィッティングがよかった．状態空間モデルは，時系列モデルの拡張といえるため，以下では簡略化のために時系列モデルでの検討を行う．

モードを用いた時系列成分の分解

BAP のモード（生成モデル）は正規分布 $\mathcal{N}(\boldsymbol{\mu}_j, \Sigma_j)$　$(j = 1, \ldots, M)$ で表現するが，観測された区分化時系列データから，平均 $\boldsymbol{\mu}_j$ を引いた残差は $\mathcal{N}(0, \Sigma_j)$ と表現できる．その中の 1 つの区分化時系列データに注目すると

[*9] https://cran.r-project.org/web/packages/dlm/index.html

表 7.2: 状態空間モデル（DLM）による負荷予測精度比較

二乗誤差	休祝日モデル	休祝日分離モデル	ANN モデル
2/16〜2/26 (〜2 時間)	6712.2	7791.7	20837.5
2/16〜2/26 (2〜4 時間)	16437.6	14752.3	20837.5

$\mathcal{N}(0, {\sigma_t}^2)$　$(t = 1, 2, \dots)$ であり，これらは ${\sigma_t}^2$ の分散を持った正規分布によるデータ列である．ただし，Σ_j や ${\sigma_t}^2$ は統計的に導出されているが，ビル設備やセンサの特徴を鑑みると，区分時系列毎の残差列は環境や執務者の活動の影響を受けた時系列モデルであると仮説が立てられる．つまり，モードの平均値と残差によって時系列を説明できるといえる（図 7.28）．

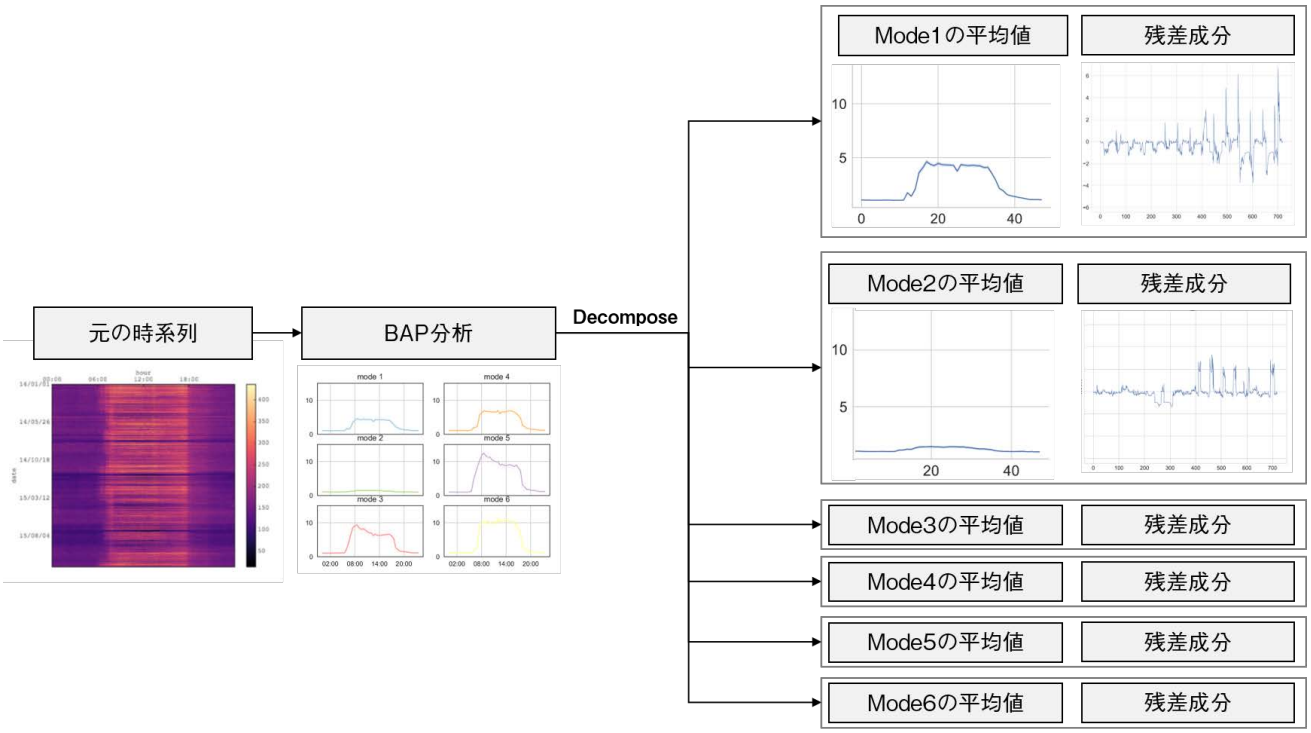


図 7.28: モードによる時系列の分解

7.3.3 残差分析による時系列の性質分析と負荷予測

図 7.28 で述べたモデル検証のため，残差成分に対して時系列分析を行い，それらの性質を明らかにする．対象の時系列は，7.2 節で用いた東関東支店の WBP である．分析手順は以下である．なお，これらの統計的な分析は，python の StatsModels[*10]を用いて行った．

1. **区分化時系列から残差成分を抽出**（図 7.29）
   日中の負荷が大きくなるに従って，残差が大きくなっており，一見すると周期性を持った定常性を持った時系列に見える．
2. **コレログラムによる自己相関および偏自己相関の検定**（図 7.30）
   区分化時系列は 48 次元であるため，自己相関係数（左）は周期的になっていることが確認できる．偏自己相関係数のグラフ（右）を確認すると，それぞれラグ 1〜2 に正の相関があることが見てとれ，自己回帰性があるといえる．なお，図中の青色の帯は 95 %の信頼区間を示しており，有意水準 5 %で無相関という帰無仮説を棄却し有意である．
3. **定常性，正規分布の確認**（表 7.3）
   拡張ディッキー–フラー検定 (ADF 検定) によって，それぞれの時系列が定常過程であるか（または単位根過程であるか）を確認する．その結果，全ての時系列において有意水準５%で定常性を持つという帰無仮説が棄却されなかった．また，Shapro-Wilk 検定も行い，対象時系列は BAP の定義通り正規分布に従うことが確認できた．従って，対象時系列には自己回帰性があり，ホワイトノイズを含み，かつ非定常過程ではないことから，自己回帰移動平均モデル (ARMA: Auto-regressive moving average) によるモデリングが適切であると推測できる．
4. **ARMA モデルによるパラメータ推定**（図 7.31）
   自己回帰や移動平均の次数等の ARMA のパラメータ推定は，赤池情報量基準 (AIC : Akaike Information Criterion) によって決定する．これらは StatsModels の statstools にある自動選択機能を用いることができる．表 7.3 に選択されたパラメータを示す．
5. **ARMA モデルによる負荷予測**（図 7.31，表 7.3）
   推定したパラメータを用いて，ARMA モデルによる予測を行う．対象時系列は残差列のみであるので，そこに対応するモードの平均値を加えたものを予測とする．図 7.31 を見ると，例えば図 7.26 の (4) では大きくずれていた休日の電力量予測も，適切に負荷に追従していることが分かる．表 7.4 に定量的な評価を示す．評価期間は 5 日間で，表 7.1 と単純な比較はできないが，定性的な分析も踏まえると，休日以外についても精度も向上しているといえるだろう．

表 7.3: ADF 検定と ARMA のパラメータ推定

	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
ADF 検定統計量	2.449e-18	2.144e-18	1.975e-26	3.327e-26	8.851e-25	9.114e-28
自己回帰次数	4	4	2	4	3	3
移動平均次数	1	2	2	1	1	0

上記検討により，モード毎の残差列は ARMA モデルによりモデリングが可能であり，提案手法が負荷予測に有用であることが検証できた．

また，リアルタイム予測に対して，統合モードを用いた際の有用性の検証を行った．図 7.32 に，図 7.17 で示した統

[*10] https://www.statsmodels.org/

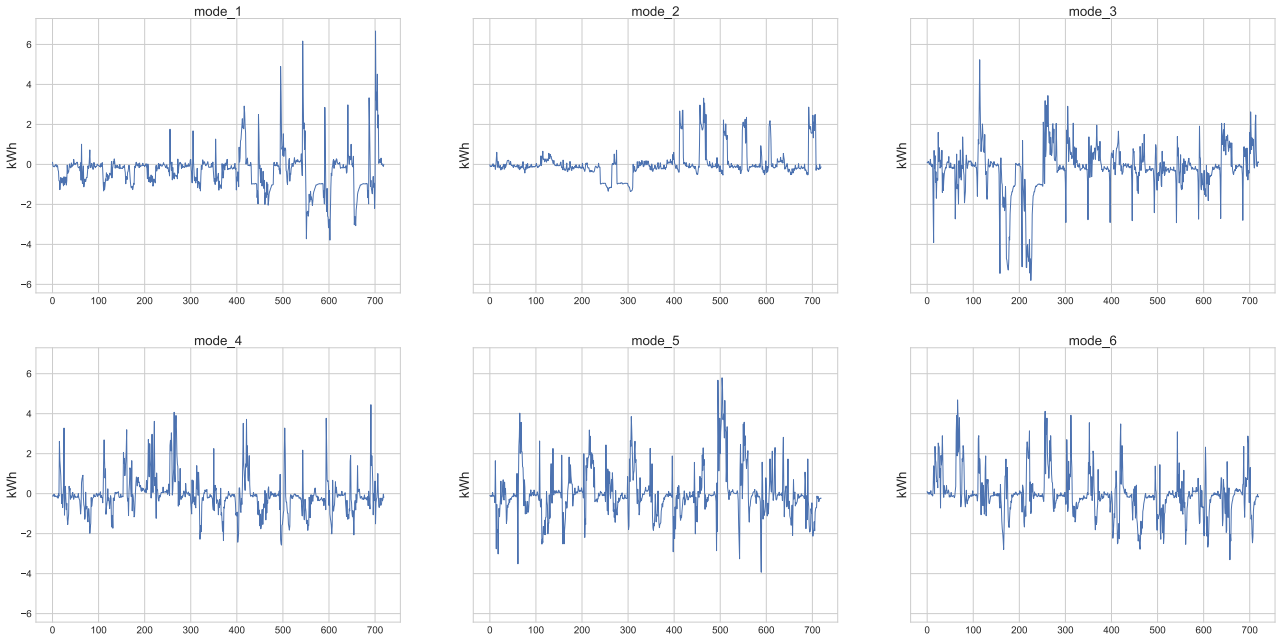


図 7.29: 東関東支店の WBP 時系列におけるモード平均値との残差

合モードから抽出した WBP 時系列に対して，上記と同様の予測を行った例を示す．7.4 に示すように，結果として平均的な精度は向上している．よって，適切に統合モードが推定できれば，より高精度な予測が可能であることが検証できた．

表 7.4: Compare of the prediction accuracy

Mode	**RMSE**	**MAE**	R^2[%]	**MAPE**[%]
WBP Mode	0.73	0.42	89.0	25.22
Integrated Mode	0.55	0.32	86.0	17.81

考察

本節で提案した BAP と時系列モデルを用いた負荷予測手法は，近年多く研究されている ANN などに比べて，挙動の説明がしやすく，また急な負荷変動にもよく追従している．深層学習の枠組みにおいては，RNN (Recurrent Neural Network) やその拡張手法の LSTM のように長期的な依存関係を含めて学習できるモデルも存在するが，ビルの電力負荷は直近の外気温や日射量などの影響が強く，また上記で考察した残差列の性質が明らかであることから，短期的なリアルタイム予測については，本手法が適しているといえる．

本研究では，モードと ARMA を用いた予測のみに留めたが，ARMA のパラメータ推定ができれば，そこから状態空間モデルへの変換は容易である [114]．状態空間モデルは，観測できない潜在的な状態をモデルに組み込むことにより，より複雑な時系列モデルを表現することが可能であり [108]，個別のモデルとして ARMA も組み込むことができる．例えば，曜日やイベント，DR 発動条件など，モードには含まれない多様な潜在変数を含ませることも可能であるため，それらの影響が自明である場合に適用することで，より高精度な予測が可能になるだろう．それにより，高精度なモード駆動型制御システムの構成が実現できると考える．

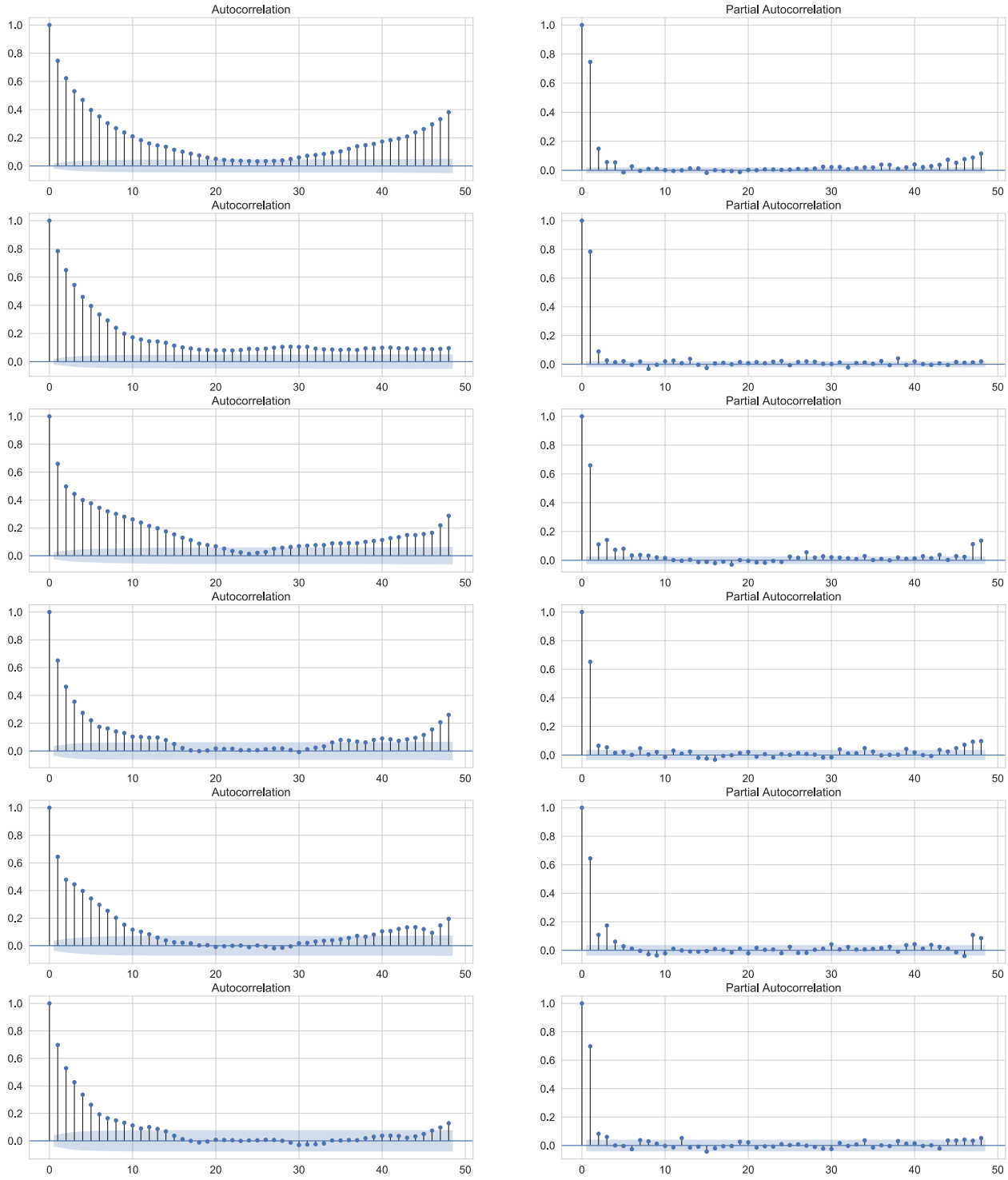


図 7.30: コレログラム（左：自己相関係数，右：偏自己相関係数）

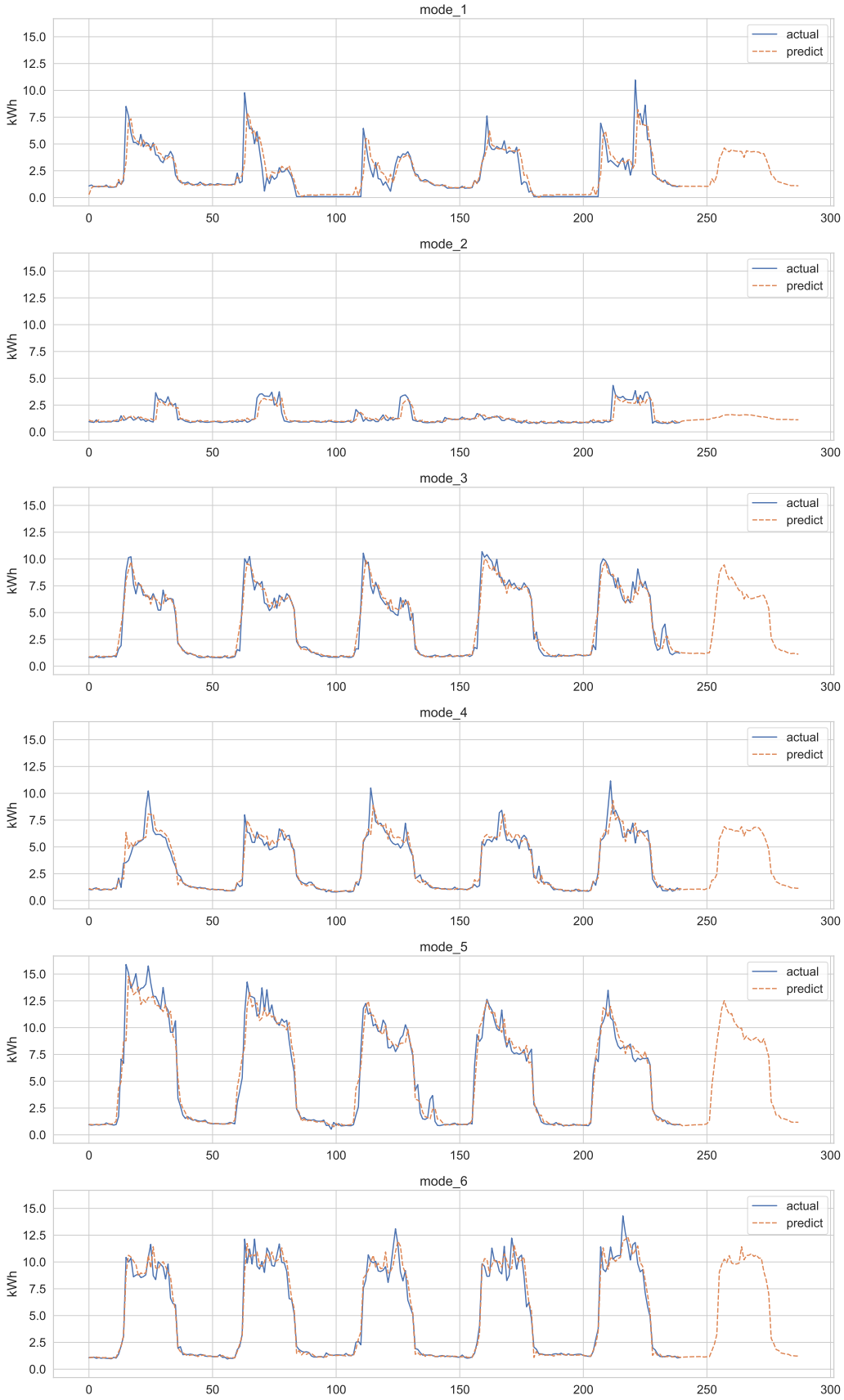


図 7.31: ARMA による負荷予測（WBP モード）

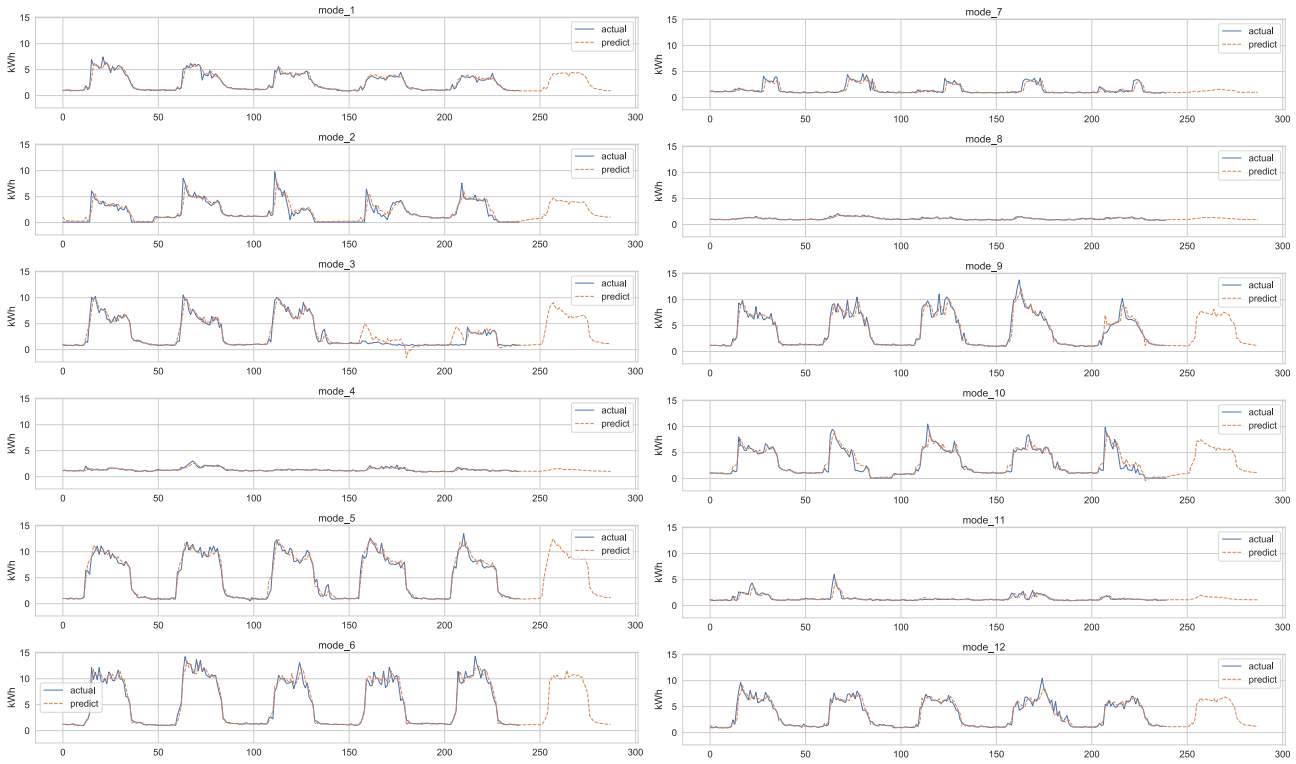


図 7.32: ARMA による負荷予測（統合モード）

7.4　データ型の推定

本節では BAP によるプロファイルを用いて，未知のデータセットのデータ型推定技術の提案と検討を行う [116]. データ型の推定技術によって，BACS のポイントにおけるマッピングの誤り検知も実現できる.

7.4.1　先行研究

BACS のポイントにおける，メタデータ付与または推定する研究としては，多くの研究があり，以下のように分類できる.

1. ポイント名から類推するもの [11, 13]
2. データから類推するもの [39, 40]
3. ツールを使ってメタデータ付与をサポートするもの [38, 75]

1. は，ポイント名から正規表現等を使って類推するものであり，例えば「AHU-1-RAT」は空気調和機の 1 番の還気温度と推定する．1. は日本におけるサンプルが少なく汎用化が難しく，また，3. については推定は行わず，施設管理者によるタグの設定支援やその履歴管理を目的とするものである．本研究では，すでにデータ・プラットフォームがあり，十分なデータが取得可能であることを前提としているため，2. を採用する.

[39, 40] では，BACS におけるポイントデータのクラス分類を，取得した時系列データを一定のタイムウィンドウで区切った際の中央値，上下限値，分散などを特徴量として，RandomForest 等による学習モデルによる分類を行っている．80 %以上の高い推定精度があるが，1 週間程度の短い期間のデータによる試行であり，例えば夏のデータを使って冬のデータの推定をすることは難しい.

本研究では，長期的なデータ使った汎用的なデータ型，クラス分類を行うため，BAP を使った分類を試みる.

7.4.2　提案手法

7.2.2 節で述べたように，BAP によって得られるモードのダイナミクス $\hat{X}_j$ は，平均 $\boldsymbol{\mu}_j$，分散共分散行列 Σ_j で表される正規分布 $\mathcal{N}(\boldsymbol{\mu}_j, \Sigma_j)$　$(j = 1, 2, \ldots, M)$，で表現される．これらは，それぞれのセンサについての典型的な挙動を示すものといえる．そのため事前に得られた分析結果と，新たに得られた未知のセンサデータの距離や類似度を比較することで，データの型推定が可能と考える.

本研究では，距離の定義として DTW (Dynamic Time Warping) を採用する．DTW は 2 つの時系列の各点の距離を総当りで比較した上で，系列同士の距離が最短となるパスを見つける手法であり，周期や次元数が異なる場合でも距離定義が可能である．対象のデータは固定長であるため，一般的なユークリッド距離も有効といえるが，比較を行ったところ，それぞれに傾向の違いはあるが，DTW の方が正しく推定した確率が高かった.

7.4.3　問題設定

前提として，推定対象の BACS のポイントデータはある程度蓄積されていることし，それに対して BAP で分析済みの学習データが十分にある環境を想定する．本検証では，TAK 新砂ビル，竹中工務店東関東支店の約 10 カ月分 (2019/3 - 2020/1) のデータセットを既知のデータとして利用し，推定対象のデータセットとしては，半年分 (2019/8 - 2020/1) の EQ House[*11]のデータを利用する．分析対象のデータは 30 分毎の計測値であり，区分化された 1 日のデータとしては 48 次元の固定長となる．これらに対して欠損値補間などの前処理を行ったデータを利用する.

[*11] https://www.takenaka.co.jp/eq_house/

問題設定として，未知の時系列データセットに対して，Brick [10] のクラスやタグを当てはめることを考え，推定対象のポイントについては，数日分の時系列データのみがあることする．例えば，外気温度のセンサであれば，*Outside_Air_Temperature_Sensor* というクラスが該当するが，既知のデータセットには，これらのクラスを予め当てはめておく．それらのデータセットに対して BAP を適用することで得られたモードのダイナミクスと，未知のデータセットの距離・類似度を定義することでクラスの推定を行う．

7.4.4　評価

表 7.5 に分析に用いた既知のデータセットの概要と BAP の適用結果を示す．先行研究におけるデータ・プラットフォーム [118] には，センサ型を示すメタデータが予め設定されており，それらを仮想的なクラスとしてグループ化し BAP を適用した．なお，対応する Brick のクラスについては，語彙定義が一部しか公開されていないため，名称をもとに設定したが，ZEB を含む特殊な建物のデータセットであるためか不足する語彙も多かった．「データ数」は区分化時系列データの数であり，グループ化された数が多いほど大きくなっている．

図 7.33 に BAP 適用によって得られたモードのダイナミクス例を示す．*Temp*，*Outdoor_Temp* はそれぞれ 13 個，2 個のモードが検出されている．これらダイナミクスの平均値（mean）と，未知のデータセットの時系列の距離を DTW ですべて比較し，距離が一番短いものを該当のクラスとする．

代表的な外部環境データ 7 種類を対象としたクラス推定の結果を表 7.6 に示す．評価に際しては未知のデータセットから 1 日ごとの区分化時系列をランダムに 100 回抽出し，距離比較を行い推定結果とした．なお，正答率は推定された結果が，対応するクラスでない場合も，カテゴリ（親クラス）が同じと判断できるものは正解としてカウントしている．理由としては，親クラスが同一であれば，対象とするデータのスキーマやタグを共通化することができるためで，例えば *Outside_Air_Temperature_Sensor* が *Water_Temperature_Sensor* と検出されたとしても，単位や小数点以下の扱いなどはほとんど変わらないためである．ただし，最大値・最小値などは変わる可能性があるため，それらの特定が必要な場合は施設管理者が適切なサブクラスをつける必要がある．気圧や外部温度・湿度，風向については高い正解率が確認できたものの，感雨量と風速は低い結果となった．対象のデータを分析してみたところ，感雨量は既往のデータセットに対して，1/10 程度の値が検出されており，検出する際のスケールの問題であった可能性がある．風速については，33 %は正確に推定しており，場所による傾向の違いが大きな原因といえる．

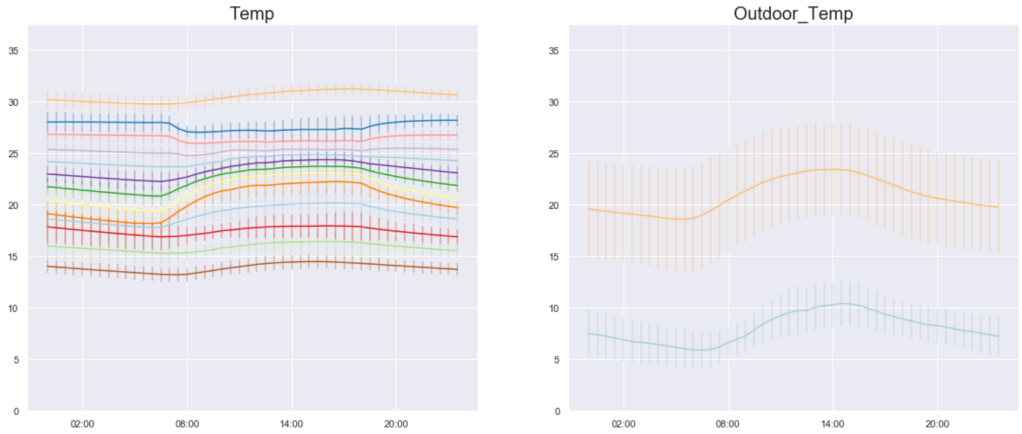


図 7.33: 推定したモードのダイナミクス

表 7.5: 既存のデータセットと BAP 適用結果

センサ型 (Key)	ポイント説明	データ数	モード数
Air_Direction	風向	240	25
Air_Pressure	気圧（無線センサ）	671	1
Air_Pressure-E	E 側屋内外差圧圧力	158	5
Air_Pressure-W	W 側屋内外差圧圧力	137	5
Air_Velocity	風速	108	1
Cal_Cum	チラー冷温水積算熱量	526	21
Cal_Inst	冷温水瞬時熱量	1248	22
CHW_Cal_Cum	冷温水積算熱量	274	13
Cop	成績係数（単体）	770	7
DewPoint	露点温度（無線センサ）	652	8
Downside_Temp	蓄熱槽下層温度	229	1
Downside_Water_Temp	熱交換杭チューブ温度下部	994	25
Downstream_Temp	集熱器下流温度	154	6
Humidity	湿度（無線）	593	11
Illuminance	照度（無線センサ/人感センサ）	6893	24
Indoor_Enthalpy	室内エンタルピ	458	1
Indoor_Humidity	室内湿度	428	4
Indoor_Temp	室内温度	472	1
Input_Return_Water_Temp	再生温水入口温度	190	3
Input_Temp	集熱器往き配管温度	190	5
Input_Water_Temp	冷温水入口温度	2108	10
Loudness	音（無線センサ）	295	22
Middleside_Water_Temp	熱交換杭チューブ温度中部	992	22
Outdoor_Enthalpy	外気温度エンタルピ	225	4
Outdoor_Humidity	外気湿度	224	5
Outdoor_Temp	外気温度	198	2
Output_Return_Water_Temp	再生温水出口温度	197	3
Output_Temp	集熱器還り配管温度	149	5
Output_Water_Temp	冷温水出口温度	2109	7
Radiate_Thermometer-E	東側窓放射温度	234	1
Radiate_Thermometer-W	西側窓放射温度	224	1
Rain_Detected	感雨センサ	212	7
RainFall	感雨量	218	25
Return_Water_Flow	再生温水積算流量	212	22
Solar_Radiation	日射量	78	25
Surface_Temp	放射パネル表面温度	2263	22
System_Cop	成績係数（システム）	769	8
Target_Temp	蓄熱槽温度設定	2667	5
Temp	温度（無線センサ/人感センサ）	51106	13
Toplight_Temp	トップライト温度	208	1
Upside_Temp	蓄熱槽上層温度	261	3
Upside_Water_Temp	熱交換杭チューブ温度上部	999	25
Upstream_Temp	集熱器上流温度	154	6
Water_Flow	冷温水流量	2070	22
Water_Flow_Cum	積算流量	246	5
Water_Level	揚水井戸水位	223	19
Water_Temp	温水送水温度/揚水温度/etc..	429	25

表 7.6: 精度検証の結果

ポイント種別	正解率 [%]	最頻値
気圧	100	Building Air Static Pressure Sensor
外部湿度	69	Outside_Air_Humidity_Sensor
感雨量	36	Cal_Inst (冷温水瞬時熱量)
日射量	75	Solar_Radiance_Sensor
外部温度	88	Water_Temperature_Sensor
風向	90	Wind_Direction_Sensor
風速	33	Rain_Detected (感雨センサ)

7.4.5 考察

本検証においては，クラス推定に関しては平均して約 70 %の正解率となった．ただし，表 7.6 の最頻値で確認できるように，意図したクラスとは別のクラスが一番多く検出されている項目もある．原因としてはスケールの違いや場所性などが考えられるが，より多くのデータセットを得ること，親クラスを付与することで解決される可能性がある．

また，本手法では，事前に定義したクラス群に対して，所属するクラスの確率が定義できるため，それらは，BACS のポイントと実際の機器のマッピングに対する信頼度と定義することもできる．マッピングの間違い検出にも応用可能といえるだろう．

なお，未知のデータが十分にある場合，BAP を適用することで，それぞれの生成モデルが得られるため，類似度の判定の指標に 2 つの確率分布の差異である KLD が利用できる．DTW では分散などを考慮しないために，KLD ではよりデータの性質を考慮した比較が可能である．検証を行ったところ，計算量・計算時間についても，DTW による計算と比較してかなり低減することができた．しかしながら，BAP の計算には概ね 60 日以上のデータがないと適切なモード数にならないため，BACS の稼働後すぐに適用するのは難しい．

7.5　まとめと今後の展開

本章では，futaba を用いたデータ分析のアプリケーション (BAP，負荷予測，データ型推定) の提案とそれぞれの検証を行った．

まずは，ビルの活動を表すためのプロファイリング技術である BAP について説明し，2 棟のビルに適用した．BAP では，プロファイリングの結果としてモードが得られる．単一センサでの結果について述べると，いずれのビルにおいても，電力量 (WBP) については季節性や平日・休日がモードによって十分に説明されており，その他の外部環境データにおいても，季節や天気などを考慮すると，適切に説明されているといえる．統合モードについては，東関東支店の場合には適切なプロファイルが得られたといえるが，TAK 新砂ビルの場合には期待する結果が得られなかった．統合モードの計算手法や，統合対象のセンサの選定については更なる検討が必要である．また，futaba での継続的運用を目的に BAP の計算モジュールを開発した．PaaS を用いて，柔軟かつ可用性の高いデータ処理のパイプラインを構築することができたが，動作時間などに最適化の余地がある．

次に，BAP と時系列モデルを用いた負荷予測技術について述べ，それらを用いたモード駆動型制御システムについて提案した．まず，モードのダイナミクスにおける平均値を抽出し，特徴量として前日負荷予測に用いることで予測精度向上が可能なことを示した．また，モードの平均値と得られた区分化時系列の残差成分について，そこから現出する時系列データのシステム同定を行い，ARMA によるモデリングが可能であることを示し，より高度なリアルタイム負荷予測が可能であることを示した．

最後に，BAP の結果を用いた未知の時系列データのデータ型推定技術について提案した．抽出したデータセットに対して BAP を適用し，得られたモードのダイナミクスに対して，未知のデータセットに対する距離と定義した DTW を計算し，その類似度によってデータ型を判別する．一部データのスケール等の問題で正しい推定が行われなかったが，平均して約 70 %の正答率が得られた．これらは，所属クラスに対する信頼度としても定義可能であり，そこから BACS ポイントのマッピング違いの検出に応用可能といえる．

いずれのアプリケーションも，futaba が想定しているリアルタイムとバッチ処理のユースケースによく適合しているといえる．BAP の分析結果も WoT API を介して，プラットフォーム側にフィードバックすることも可能である．今後は，BAP の計算モジュールを継続的に動作させ，実物件においてモード駆動型制御システムを構築，実証していくことを予定している．

第 8 章

結言

最後に，Software Defined BACS やコモングラウンドが，本研究によって実現に近づいたかを考察し，本研究のまとめとする．

8.1 Software Defined BACS の実現

Software Defined BACS は，可用性の高いローカルシステムである BACS の機能を維持しつつも，その機能的限界を超えるため，ハードウェアの抽象化とデータの管理機能を持ったデータ・プラットフォームを有し，API 連携によって再利用や移植性の高いサービス構築が可能であり，それらのサービス定義（構造や手順）や制御パラメータ等の更新がソフトウェアによって可能なシステムと定義した．本研究では，特にデータ・プラットフォームにおけるハードウェア抽象化と，データ保存や利活用のための API，管理機能について，研究と実践を行ってきた．

まず，4 章で示した BIM と BACS のポイントリストを融合させたデータモデルによって，ハードウェア抽象化の生成について実現した．これらは実際のユースケースを基にしているため，建設ドメイン外のサードパーティから受け入れられやすい実践的なものになっている．BIM から抽出した空間階層とデバイスの包含関係を中心とするセマンティクスを有しており，制御，モニタリング対象の選定が容易である．

また，それらのデータモデルを取り込んだデータ・プラットフォームである futaba は WoT をベースにしたデータモデルと RESTful API を有しており，サードパーティにオブジェクト指向の直感的なインタフェースを提供することができる．ラムダ・アーキテクチャによってリアルタイムだけでなく，AI やビックデータ処理を前提としたバッチ型のアプリケーションにも対応可能である．PaaS を前提に設計しているため，十分に安価で可用性が高く，実際のプロジェクトにも導入可能となっている．遠隔更新可能なゲートウェイも導入し，既往の BACS の機能を保持しつつ，先端的な機能を追加することが可能になった．

更に，7 章で述べた BAP は，建物の活動を表すプロファイリング技術，XAI である．futaba の WoT API によって，Thing としてモデリングしたノード（センサ，デバイス，空間）に BAP の分析結果であるモードを書き込むことが可能であり，それらを用いることで，ビルの状態や活動に即した制御，より高精度な負荷予測システムの構築が可能であることを示した．BAP とその応用技術は，Software Defined BACS の目指す，ソフトウェアによる動的なシステム構成のために必要な情報といえるだろう．

以上，Software Defined BACS のベースとなる機能については実現できたといえるが，サードパーティによるサービス定義も含めてソフトウェア（コード）で記述を行い，それらを組み合わせたサービスの自動生成までは実現できていない．

8.2　コモングラウンドの実現

コモングラウンドは，Software Defined BACS を拡張し，BACS 以外のデバイスやアプリケーション，実空間やシステムの数だけ存在するデジタルツインを内包し，知識ベースを介して，多様なシステムを仲介するプラットフォームといえる．

4 章では，BIM から取得できるセマンティクスやメタデータと，抽出した形状データが SDM のアプリケーションにも応用できること示した．ここでは，セマンティックウェブの技術を用いて，SDM とスマートビルのオントロジを連結する知識ベースを構築した．

5 章では，SDM オントロジを用いたコンテンツ記述，および VR と立体音響を用いたアプリケーションとそのプラットフォームについて述べ，その有効性を示した．

6 章で述べた futaba も，セマンティックウェブを用いたインターネット技術に立脚したシステム・アーキテクチャとなっており，コモングラウンドの全体像の中で説明することが可能である．

これらの技術は，ビル設備の制御に留まらない，多様なコンテンツ制作やシステム連携が可能なコモングラウンドの基盤技術となると考える．

8.3　まとめと今後の展望・展開

本研究では，コンピュータサイエンスの知見に基づいて，BIM や BACS といった建設産業に固有のデジタルリソースの活用について検討した．Software Defined BACS について提案を行い，スマートビルのアプリケーション構築に有用なデータ・プラットフォームの要件を明らかにするとともに，SDM のアプリケーションについても深耕し，BIM の応用について検討した．更には，ビルで収集可能なデータのプロファイリング技術について提案し，その性質について明らかにした．これら本研究の成果は，Software Defined BACS やコモングラウンドを実現するための基盤的技術になるといえる．しかしながら，本研究での実現は，データモデルや API の自動生成によるスマートビルのサービスの再利用や移植性の向上することに留まっており，サービス定義・生成までの機能までは実現していない．これらの実現のためには，以下のアプローチが有効と考える．

1. API の公開によるサードパーティおよびアプリケーションの拡充
2. アプリケーション・サービスに対するメタデータ，オントロジの定義
3. スマートビルのサービスカタログの作成
4. データ・プラットフォームへのサービス管理機能の実装
5. スマートビルの Linked Open Data (LOD) の構築

研究を通じて得た課題意識として，スマートビルのアプリケーションは様々なところで提案されているにも関わらず，プロジェクトの個別性が大きかったり，構築ベンダーが異なったりする理由で，サービス記述などの共通セマンティクスが定義されていないことがあった．それらを集約したサービスカタログが必要いえる．また，それらの管理機能を futaba に実装し，サービスの組み合わせによって高度なシステムを構成していく，いわゆる SOA (Service Oriented Architecture) を実現する基盤にしていくことが求められる．今後，futaba と同様のデータ・プラットフォームが普及すれば，それらのサービス記述やメタデータを結ぶ LOD が構成され，スマートビルの普及を加速する原動力となると考える．LOD には，ビルの中で展開される SDM アプリケーションなども登録され，SDM オントロジなどを使って空間的なリンクが形成されていく．勿論ここには，サイバーセキュリティやプライバシに関する考慮，研究も必要となるだろう．そのためにも建設プロジェクトにそのまま適用可能な実践的なデータ・プラットフォームの研究が必要であったといえる．

本研究で明らかにしたセマンティクスとデータモデルは，NTT グループによるスマートシティのためのプラットフォームの基盤技術として取り入れる検討も進めている．futaba も BACS，IoT の汎用的なデータ・プラットフォームとして，いくつかの実プロジェクトへの適用を検討中である．本研究で得られたノウハウを活用し，今後も様々な領域への展開を進めていきたい．現在，都市 OS やスマートシティのアプリケーションなどが多く提案されているが，我が国において，そのほとんどがコンセプトレベルである．都市を構成するビルのインフラは，スマートシティを構成する重要な要素であり，その意味で，スマートシティの推進にも繋がるといえる．本研究の成果により，スマートビルやスマートシティのサービスに関する新たなエコシステム，新たなビジネスモデルが生まれることを期待する．

発表論文

学術論文

1. 粕谷貴司，塚田学，菰原裕（東京大学），高坂茂樹（エスイーディー株式会社），水野拓宏，野村譲誉（株式会社アルファコード），上田雄太（株式会社 CRI・ミドルウェア），江崎浩（東京大学）．インタラクティブな遠隔ライブ VR 配信プラットフォーム，情処論文誌：デジタルコンテンツ (DCON) トランザクション，Vol.7，No.2，2019.
2. 粕谷貴司．BIM とデータ分析手法を用いた設備システムのモデル生成に関する研究，建築学会技術報告，2020.

国際会議 (査読付き)

1. Takashi Kasuya, Manabu Tsukada, Yu Komohara, Shigeki Takasaka, Takuhiro Mizuno, Yoshitaka Nomura, Yuta Ueda, and Hiroshi Esaki. LiVRation: Remote VR live platform with interactive 3D audio-visual service, IEEE Games Entertainment & Media Conference (IEEE GEM) 2019, Yale University, New Haven, CT, U.S., June, 2019.
2. Takashi Kasuya, Takeshi Takai, Hiroshi Esaki. Building Activity Profiling: Explainable and Predictive Modeling for Building Automation, Proceedings of IEEE ICAIIC 2020, Fukuoka, Feb, 2020.

その他論文・解説

1. 粕谷貴司，田中規敏. IoT を活用した建物制御システム. 電気設備学会誌，Vol.38，No.8，pp.466–469，2018.
2. 粕谷貴司，塚田学，菰原裕，高坂茂樹，水野拓宏，野村譲誉，上田雄太，江崎浩. LiVRation: VR による自由視聴点映像音声のインタラクティブ再生．マルチメディア・分散・協調とモバイル (DICOMO2018) シンポジウム，2018.
3. 粕谷貴司．最新の IoT/AI を活用したスマートビルの実施例: EQ House における AI による無人制御への挑戦 (特集 AI・IoT を活用した建築・設備における新たな価値の創出)．BE 建築設備，Vol.70，No.12，pp.24–30，2019.
4. 粕谷貴司，曽根卓朗，塚田学，安藤亮介，白浜妥知，庄子琢郎，江崎浩. インタラクティブ操作と AR 可視化を実現する立体視聴プラットフォーム. 情報処理学会研究報告 (CG)，Vol.176，No.24，pp.1-8，2019.
5. 粕谷貴司．IoT・BIM を使ったデジタルツインアプリケーションの構築 (特集センシングの現状とビルの運用・保全管理への適用)．BE 建築設備，Vol.71，No.1，pp.68–74，2020.

ポスター発表

1. 粕谷貴司，高井勇志，江崎浩．Building Activity Profiling:建物設備システムにおける説明可能な AI のためのモデリング手法，超知性ネットワーキングに関する分野横断型研究会 (RISING)，2019.

参考文献

[1] AUROMAX© next generation immersive sound system.

[2] Dolby Atmos© Specifications. Technical Report Issue 3, Dolby Laboratories, 2015.

[3] Mohamed Amine Abid and Hermann De Meer. Virtualized Software Defined Buildings: A Key Enabler of the Future Smart Cities. *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2018*, pp. 0–5, 2018.

[4] K. P. Amber, R. Ahmad, M. W. Aslam, A. Kousar, M. Usman, and M. S. Khan. Intelligent techniques for forecasting electricity consumption of buildings. *Energy*, Vol. 157, pp. 886–893, aug 2018.

[5] Ayesha Ameen, Khaleel Khan, and B.Padmaja rani. Extracting knowledge from ontology using jena for semantic web. 04 2014.

[6] Michael P. Andersen and David E. Culler. BTrDB: Optimizing storage system design for timeseries processing. In *Proceedings of the 14th USENIX Conference on File and Storage Technologies, FAST 2016*, 2019.

[7] Jakob Beetz André Borrmann, Markus Koenig Christian Koch, editor. *Building Information Modeling : Technology Foundations and Industry Practice*. Springer, October 2018.

[8] Ray Atarashi, Takuro Sone, Yu Komohara, Manabu Tsukada, Takashi Kasuya, Hiraku Okumura, Masahiro Ikeda, and Hiroshi Esaki. The Software Defined Media Ontology for Music Events. In *Workshop on Semantic Applications for Audio and Music*, Proceedings SAAM '18, October 9, 2018, Monterey, CA, USA, Monterey, California, United States, October 2018.

[9] Bharathan Balaji, Arka Bhattacharya, Gabriel Fierro, Jingkun Gao, Joshua Gluck, Dezhi Hong, Aslak Johansen, Jason Koh, Joern Ploennigs, Yuvraj Agarwal, Mario Berges, David Culler, Rajesh Gupta, Mikkel Baun Kjærgaard, Mani Srivastava, and Kamin Whitehouse. Brick: Towards a Unified Metadata Schema For Buildings. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys '16, pp. 41–50, New York, NY, USA, 2016. ACM.

[10] Bharathan Balaji, Arka Bhattacharya, Gabriel Fierro, Jingkun Gao, Joshua Gluck, Dezhi Hong, Aslak Johansen, Jason Koh, Joern Ploennigs, Yuvraj Agarwal, Mario Berges, David Culler, Rajesh Gupta, Mikkel Baun Kjærgaard, Mani Srivastava, and Kamin Whitehouse. Brick: Towards a unified metadata schema for buildings. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys '16, pp. 41–50, New York, NY, USA, 2016. ACM.

[11] Bharathan Balaji, Chetan Verma, Balakrishnan Narayanaswamy, and Yuvraj Agarwal. Zodiac: Organizing large deployment of sensors to create reusable applications for buildings. *BuildSys 2015 - Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built*, pp. 13–22, 2015.

[12] Maad. Bali, Dietmar A. Half, Dieter Polle, and Juergen Spitz. *Smart Building Design: Conception, Planning, Realization, and Operation*. Birkhäuser, 2019.

[13] Arka A. Bhattacharya, Dezhi Hong, David Culler, Jorge Ortiz, Kamin Whitehouse, and Eugene Wu. Automated metadata construction to support portable building applications. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, p. 3–12, New York, NY, USA, 2015. Association for Computing Machinery.

[14] Arka Aloke Bhattacharya, Joern Ploennigs, and David Culler. Short paper: Analyzing metadata schemas for buildings: The good, the bad, and the ugly. In *BuildSys@SenSys*, 2015.

[15] R. Bleidt, A. Borsum, H. Fuchs, and S. M. Weiss. Object-based audio: Opportunities for improved listening experience and increased listener involvement. *SMPTE Motion Imaging Journal*, Vol. 124, No. 5, pp. 1–13, July 2015.

[16] Stefan Boeykens. Using 3D Design software, BIM and game engines for architectural historical reconstruction. *Designing Together - Proceedings of the 14th international conference on Computer Aided Architectural Design Futures*, pp. 493–509, 2011.

[17] Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter-current status. 06 2018.

[18] Bjorn Butzin, Frank Golatowski, and Dirk Timmermann. A survey on information modeling and ontologies in building automation. *Proceedings IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, Vol. 2017-Janua, No. October 2018, pp. 8615–8621, 2017.

[19] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the cave. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH 1993, pp. 135–142, New York, NY, USA, 1993. ACM.

[20] Laura Daniele, Frank den Hartog, and Jasper Roes. Created in close interaction with the industry: The smart appliances reference (saref) ontology. pp. 100–112, 08 2015.

[21] Stephen Dawson-Haggerty, Xiaofan Jiang, Gilman Tolle, Jorge Ortiz, and David Culler. sMAP - a simple measurement and actuation profile for physical information. pp. 197–210, 01 2010.

[22] Stephen Dawson-Haggerty, Andrew Krioukov, Jay Taneja, Sagar Karandikar, Gabe Fierro, Nikita Kitaev, and David Culler. BOSS: Building operating system services. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pp. 443–457, Lombard, IL, 2013. USENIX.

[23] Stephen Dawson-Haggerty, Jorge Ortiz, Jason Trager, David Culler, and Randy Katz. Energy savings and the software-defined building. *Design and Test of Computers, IEEE*, Vol. 29, pp. 56–57, 08 2012.

[24] Mary E Dickinson, Ann M Flenniken, Xiao Ji, Lydia Teboul, Michael D Wong, Jacqueline K White, Terrence F Meehan, Wolfgang J Weninger, Henrik Westerberg, Hibret Adissu, et al. High-throughput discovery of novel developmental phenotypes. *Nature*, 2016.

[25] Jing Du, Zhengbo Zou, Yangming Shi, and Dong Zhao. Zero latency: Real-time synchronization of BIM data in virtual reality for collaborative decision-making. *Automation in Construction*, Vol. 85, pp. 51–64, jan 2018.

[26] Khaled El Ammari and Amin Hammad. Remote interactive collaboration in facilities management using BIM-based mixed reality. *Automation in Construction*, Vol. 107, p. 102940, nov 2019.

[27] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM Comput. Surv.*, Vol. 35, No. 2, p. 114–131, June 2003.

[28] P. O. Fanger. *Thermal comfort : analysis and applications in environmental engineering.* R.E. Krieger

Pub. Co., 1982.

[29] Pedro Fazenda, Pedro Lima, and Paulo Carreira. Context-based thermodynamic modeling of buildings spaces. *Energy and Buildings*, Vol. 124, pp. 164–177, 2016.

[30] Gabe Fierro and David E. Culler. Design and Analysis of a Query Processor for Brick. *ACM Transactions on Sensor Networks*, Vol. 14, No. 3-4, pp. 1–25, nov 2018.

[31] Gabe Fierro, Marco Pritoni, Moustafa Abdelbaky, Daniel Lengyel, John Leyden, Anand Prakash, Pranav Gupta, Paul Raftery, Therese Peffer, Greg Thomson, and David E. Culler. Mortar: An open testbed for portable building analytics. *ACM Transactions on Sensor Networks*, Vol. 16, No. 1, 2019.

[32] Gabriel Fierro, David E Culler, and Gabe Fierro. XBOS: An Extensible Building Operating System. Technical report, 2015.

[33] Hao Gao, Christian Koch, and Yupeng Wu. Building information modelling based building energy modelling: A review. *Applied Energy*, Vol. 238, pp. 320 – 343, 2019.

[34] Michael A Gerzon. Periphony: With-height sound reproduction. *Journal of the Audio Engineering Society*, Vol. 21, No. 1, pp. 2–10, 1973.

[35] Mengjie Han, Xingxing Zhang, Liguo Xu, Ross May, Song Pan, and Jinshun Wu. A review of reinforcement learning methodologies on control systems for building energy. 2018.

[36] Juergen Herre, Johannes Hilpert, Achim Kuntz, and Jan Plogsties. MPEG-h 3D audio - the new standard for coding of immersive spatial audio. *IEEE Journal of Selected Topics in Signal Processing*, Vol. 9, pp. 1–1, 08 2015.

[37] K. Hirata, Y. Harada, T. Takada, S. Aoyagi, Y. Shirai, N. Yamashita, K. Kaji, J. Yamato, and K. Nakazawa. t-room: Next generation video communication system. In *2008 IEEE Global Telecommunications Conference (GLOBECOM 2008)*, pp. 1–4, November 2008.

[38] Emil Holmegaard, Aslak Johansen, and Mikkel Baun Kjargaard. Metafier — A tool for annotating and structuring building metadata. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pp. 1–8. IEEE, aug 2017.

[39] Dezhi Hong, Jorge Ortiz, Arka Bhattacharya, and Kamin Whitehouse. Sensor-Type Classification in Buildings. 2015.

[40] Dezhi Hong, Hongning Wang, Jorge Ortiz, and Kamin Whitehouse. The Building Adapter : Towards Quickly Applying Building Analytics at Scale Categories and Subject Descriptors. *BuildSys*, pp. 123–132, 2015.

[41] i-Construction 委員会. i-Construction〜建設現場の生産性革命〜. April 2018.

[42] Zachary G. Ives. Technical perspective: k-shape: Efficient and accurate clustering of time series. *SIGMOD Rec.*, Vol. 45, p. 68, 2016.

[43] Krzysztof Janowicz, Pascal Hitzler, Benjamin Adams, Dave Kolas, and Charles Vardeman. Five stars of linked data vocabulary use. *Semant. Web*, Vol. 5, No. 3, p. 173–176, July 2014.

[44] Antonio J. Jara, Alex C. Olivieri, Yann Bocchi, Markus Jung, Wolfgang Kastner, and Antonio F. Skarmeta. Semantic web of things: An analysis of the application semantics for the iot moving towards the iot convergence. *Int. J. Web Grid Serv.*, Vol. 10, No. 2/3, p. 244–272, April 2014.

[45] Youngmin Ji, Woosuk Choi, Kisu Ok, and Jooyoung Ahn. Intelligent building using hybrid inference with building automation system to improve energy efficiency. In *SWIT@ISWC*, 2017.

[46] Gerald C. Kane, Doug Palmer, Anh Nguyen Phillips, David Kiron, and Natasha Buckley. Strategy, not technology, drives digital transformation. *MUT Sloan Management Review*, July 2015.

[47] T. Kasuya, M. Tsukada, Y. Komohara, S. Takasaka, T. Mizuno, Y. Nomura, Y. Ueda, and H. Esaki. LiVRation: Remote VR live platform with interactive 3D audio-visual service. In *2019 IEEE Games, Entertainment, Media Conference (GEM)*, pp. 1–7, 2019.

[48] Shin Kato, Tomohiro Ikeda, Mitsuaki Kawamorita, Manabu Tsukada, and Hiroshi Esaki. Web3602: An Interactive Web Application for viewing 3D Audio-visual Contents. In *17th Sound and Music Computing Conference (SMC)*, 2020.

[49] Warodom Khamphanchai, Avijit Saha, Kruthika Rathinavel, Murat Kuzlu, Manisa Pipattanasomporn, Saifur Rahman, Bora Akyol, and J. Haack. Conceptual architecture of building energy management open source software (bemoss). pp. 1–6, 10 2014.

[50] Andrew Krioukov, Gabe Fierro, Nikita Kitaev, and David Culler. Building application stack (bas). In *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, BuildSys '12, p. 72–79, New York, NY, USA, 2012. Association for Computing Machinery.

[51] T. Liebchen and Y. A. Reznik. Mpeg-4 als: an emerging standard for lossless audio coding. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, pp. 439–448, March 2004.

[52] Georgios Lilis and Maher Kayal. A secure and distributed message oriented middleware for smart building applications. *Automation in Construction*, Vol. 86, No. June 2017, pp. 163–175, 2018.

[53] D. L. Marino, K. Amarasinghe, and M. Manic. Building energy load forecasting using deep neural networks. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, pp. 7046–7051, Oct 2016.

[54] Nathan Marz and James Warren. *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications, 2015.

[55] Manuel Mazzara, Ilya Afanasyev, Smruti Sarangi, Salvatore Distefano, and Vivek Kumar. A reference architecture for smart and software-defined buildings, 07 2019.

[56] Alice Micolier, Franck Taillandier, Patrick Taillandier, and Frédéric Bos. Li-bim, an agent-based approach to simulate occupant-building interaction from the building-information modelling. *Engineering Applications of Artificial Intelligence*, Vol. 82, pp. 44 – 59, 2019.

[57] C. Morse, A. Chernick, Z. Ren, S. Naumovski, and L. Gehron. Sound space: Communicating acoustics through interactive visualization. In *2019 IEEE Games, Entertainment, Media Conference (GEM)*, pp. 1–4, 2019.

[58] E. Nakasu. Super hi-vision on the horizon: A future TV system that conveys an enhanced sense of reality and presence. *IEEE Consumer Electronics Magazine*, Vol. 1, No. 2, pp. 36–42, April 2012.

[59] H. Ochiai, M. Ishiyama, T. Momose, N. Fujiwara, K. Ito, H. Inagaki, A. Nakagawa, and H. Esaki. Fiap: Facility information access protocol for data-centric building automation systems. In *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 229–234, 2011.

[60] Pieter Pauwels and Walter Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, Vol. 63, pp. 100–133, mar 2016.

[61] Van Cu Pham, Yuto Lim, Antonio Sgorbissa, and Yasuo Tan. An Ontology-driven ECHONET Lite Adaptation Layer for Smart Homes. Technical report.

[62] Mark A Poletti. Three-dimensional surround sound systems based on spherical harmonics. *Journal of the Audio Engineering Society*, Vol. 53, No. 11, pp. 1004–1025, 2005.

[63] Yves Raimond, Samer A Abdallah, Mark B Sandler, and Frederick Giasson. The Music Ontology. *Proceedings of the International Conference on Music Information Retrieval*, pp. pp. 417–422, 2007.

[64] Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. In *Semantic Web – Interoperability, Usability, Applicability an IOS Press Journal*, 2019.

[65] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke. Middleware for internet of things: A survey. *IEEE Internet of Things Journal*, Vol. 3, No. 1, pp. 70–95, 2016.

[66] ITUR Rec. Itu-r bs. 2051-0 (02/2014) advanced sound system for programme production. *Int. Telecommun. Union, Geneva, Switzerland*, 2014.

[67] Hiroaki Takai, Koji Tanaka, Kazuki Wada, and Hiroki Kawakami. Transforming an occupied office into a zero energy building. *ASHRAE Journal*, Vol. 61, No. 5, May 2019.

[68] Kasuya Takashi, Takai Takeshi, and Esaki Hiroshi. Building activity profiling: Explainable and predictive modeling for building automation. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pp. 242–247, 2020.

[69] Shu Tang, Dennis R. Shelden, Charles M. Eastman, Pardis Pishdad-Bozorgi, and Xinghua Gao. A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends, 2019.

[70] Sean J Taylor and Benjamin Letham. Business Time Series Forecasting at Scale. *PeerJ Preprints 5:e3190v2*, Vol. 35, No. 8, pp. 48–90, 2017.

[71] Manabu Tsukada, Keiko Ogawa, Masahiro Ikeda, Takuro Sone, Kenta Niwa, Shoichiro Saito, Takashi Kasuya, Hideki Sunahara, and Hiroshi Esaki. Software Defined Media: Virtualization of Audio-Visual Services. *IEEE International Conference on Communications (ICC2017)*, May 2017. Paris, France.

[72] Norbert Vicari, Gunes Kurt, Berna Ors Yalcin, Francisco de Borja Ortiz De la Orden, Christoph Fiehe, Anna Litvina, Darko Anicic, Michael Bahr, Sebastian Käbisch, Christoph Niedermeier, Jan Seeger, Egon Wuchner, Marc-Oliver Pahl, Benjamin Hof, Malte Burkert, Andreas Müller, Martin Neubauer, Nacho Mansanet, Joan Fons, and Björn Butzin. Building as a Service - BaaS Deliverable D05 – BaaS Reference Architecture. pp. 1–144, 2016.

[73] Long Wang, Yong Ding, Till Riedel, Andrei Miclaus, and Michael Beigl. Data analysis on building load profiles: A stepping stone to future campus. In *2017 International Smart Cities Conference (ISC2)*, pp. 1–4, 2017.

[74] Shengwei Wang, Zhengyuan Xu, Jiannong Cao, and Jianping Zhang. A middleware for web service-enabled integration and interoperation of intelligent building systems. *Automation in Construction*, Vol. 16, No. 1, pp. 112–121, 2007.

[75] Thomas Weng, Anthony Nwokafor, and Yuvraj Agarwal. BuildingDepot 2.0: An integrated management system for building analysis and control. pp. 1–8, 11 2013.

[76] Qipei Zhang, Jixiang Lu, Zhihong Yang, and Mengfu Tu. A deep learning based real-time load forecasting method in electricity spot market. *Journal of Physics: Conference Series*, Vol. 1176, p. 062068, mar 2019.

[77] Huiting Zheng, Jiabin Yuan, and Long Chen. Short-term load forecasting using EMD-LSTM neural networks with a xgboost algorithm for feature importance evaluation. *Energies*, Vol. 10, p. 1168, 08 2017.

[78] 冨澤一生. オブジェクトとプロパティ. 電気設備学会誌, Vol. 32, No. 2, pp. 115–118, 2012.

[79] 加藤慎, 曽根卓朗, 塚田学, 江崎浩. 再起的記述を可能とする映像音声メディア・オントロジー. マルチメディア, 分散, 協調とモバイル（DICOMO2020）シンポジウム, 7 月 2020.

[80] 加藤慎, 池田洋, 川守田光昭, 塚田学, 江崎浩. Web360^2: インタラクティブな 3D 視聴体験を提供する web アプリケーション. デジタルコンテンツクリエーション研究会 (CVIM,CGVI 合同開催), October 2019.

[81] 花岡郁哉. リアルとデジタルを総合的に扱う. 月刊建築技術　2019 年 12 月号, No. 839, pp. 88–93, December 2019.

[82] 粕谷貴司. 建物設備システムリファレンスガイドについて (特集エネルギーに関するサイバーセキュリティの現状と対応). エネルギー・資源 = Energy and resources, Vol. 38, No. 2, pp. 95–99, mar 2017.

[83] 粕谷貴司, 田中規敏. IoT を活用した建物制御システム. 電気設備学会誌, Vol. 38, No. 8, pp. 466–469, 2018.

[84] 金順暎, 仲地孝之, 江村暁, 藤井竜也, 羽田陽一. 4K マルチ映像と 6 チャンネルエコーキャンセラを用いた超高臨場遠隔コラボレーションシステム. 電子情報通信学会技術研究報告. CQ, コミュニケーションクオリティ, Vol. 112, No. 10, pp. 87–92, April 2012.

[85] 建築ＢＩM推進会議. 建築分野におけるＢＩMの標準ワークフローとその活用方策に関するガイドライン（第一版）. February 2020.

[86] 建築コスト管理システム研究所・新技術調査検討会. Bacs の基本構成と最新技術動向. 建築コスト研究, No. 100, January 2018.

[87] 戸田勇登, 高橋洋祐, 加戸啓太, 平沢岳人. データベースを介した三次元 CAD とゲームエンジンの連携に関する研究. 日本建築学会技術報告集, Vol. 24, No. 58, pp. 1295–1298, 2018.

[88] 菰原裕, 塚田学, 江崎浩, 曽根卓朗, 池田雅弘, 高坂茂樹, 新麗, 新善文. SDM Ontology: Software Defined Media のメタデータ管理のための Ontology. マルチメディア，分散，協調とモバイル (DICOMO2017) シンポジウム, June 2017.

[89] 後藤真. 人文社会系大規模データベースへの linked data の適用-推論による知識処理. 情報知識学会誌, Vol. 25, No. 4, pp. 291–298, 2015.

[90] 松岡康友. メタバースを用いた建築制御フレームワークの可能性 : 遠隔操作の応答時間検証. 竹中技術研究報告 竹中工務店技術研究所 編, No. 65, p. 7p, 2009.

[91] 伊藤弘. BACnet システムインターオペラビリティ ガイドライン. 電気設備学会誌, Vol. 32, No. 2, pp. 127–130, 2012.

[92] 高井勇志. 建物活動プロファイル：建物内活動のモデル化と学習. 人工知能学会全国大会論文集, Vol. JSAI2016, , 2016.

[93] 渡邊剛. ISO16484-3 基本機能とポイントリスト. 電気設備学会誌, Vol. 33, No. 2, pp. 96–99, 2013.

[94] 三木秀樹, 一ノ瀬雅之, 須永修通, 中野民雄, 市川憲良. 空調・衛生設備部材の IFC による表現手法の明確化. 日本建築学会技術報告集, Vol. 20, No. 44, pp. 375–380, 2014.

[95] 落合秀也. 汎用設備管理向け通信プロトコル. 電気設備学会誌, Vol. 32, No. 2, pp. 147–150, 2012.

[96] 大山俊雄. インテリジェントビルの進化. 人間工学, Vol. 43, No. 1Supplement, pp. 29–33, 2007.

[97] 小木哲朗, 茅原拓朗, 加藤允文, 浅山宏, 廣瀬通孝. 没入型多面ディスプレイのためのインタラクティブ高臨場感音場提示手法. 日本バーチャルリアリティ学会論文誌, Vol. 8, No. 1, pp. 75–83, March 2003.

[98] 経済産業省 資源エネルギー庁 省エネルギー・新エネルギー部新エネルギーシステム課. VPP の実用化に向けた取組み. 電気学会誌, Vol. 139, No. 3, pp. 140–143, 2019.

[99] 仲谷正史, 筧康明, 南澤孝太, 三原聡一郎, 舘 [ススム]. 触感表現の一般普及に向けた方法論とテクタイルワークショップを通したその実践 (特集ハプティックコンテンツ). 日本バーチャルリアリティ学会論文誌, Vol. 19, No. 4, pp. 593–603, 2014.

[100] 西田豊明. AI 時代のコモングラウンド. TASC Monthly 特別シリーズ「情報テクノロジーの進展がもたらす未来社会の姿を考える」, No. 526, 2019.

[101] 赤木正幸, 浅見泰司, 谷山智彦. 不動産テックを考える. プログレス, 2019.

[102] 足達嘉信, 飯島勇, 飯田千恵, 猪里孝司, 石曾根栄之, 井上雅子, 大川英二, 奥村潤, 繁戸和幸, 志手一哉, 高橋将幸, 立石賢太, 田邉邦夫, 土田真一郎, 友景寿志, 松岡辰郎. ファシリティマネジメントのための BIM ガイドライン. 公益社団法人日本ファシリティマネジメント協会, 2019.

[103] 対馬義幸, 山元弘和. 梅田センタービル物語. 彰国社, April 1988.

[104] 大久保洋幸, 大谷眞道, 小野一穂, 正岡顕一郎, 池沢龍, 小宮山摂, 浅山宏, 湯山一郎. CG 同期したインタラクティブ音場再生システムについて. 日本バーチャルリアリティ学会論文誌, Vol. 5, No. 3, pp. 965–973, September 2000.

[105] 大久保洋幸, 中山靖茂, 池永敏和, 小宮山摂. インタラクティブ 3D 映像音響再生システム. NHK 技研 R&D, ('04〔NHK〕技研公開 講演・研究発表 特集号 (1)), No. 86, pp. 72–79, July 2004.

[106] 池田靖史. 建築技術の情報革命. 月刊建築技術　2019 年 12 月号, No. 839, pp. 78–83, December 2019.

[107] 中庭涼, 藤井哲. 4 k 超高精細映像の配信技法に関する検討. 東京都市大学環境情報学部情報メディアセンタージャーナル, Vol. 13, No. 1, pp. 33–38, 2012.

[108] 島田直希. 時系列解析—自己回帰型モデル・状態空間モデル・異常検知—. 共立出版, 2019.

[109] 塚田学, 菰原裕, 新居英明, 粕谷貴司, 高坂茂樹, 小川景子, 江崎浩. SDM360^2:音楽イベントのための自由視聴点映像音声のインタラクティブ再生. マルチメディア，分散，協調とモバイル (DICOMO2017) シンポジウム, June 2017.

[110] 塚田学, 菰原裕, 粕谷貴司, 新居英明, 高坂茂樹, 小川景子, 江崎浩. SDM360^2：インタラクティブ 3D コンテンツの自由視聴点再生". 情報処理学会論文誌デジタルコンテンツ（DCON）, Vol. 6, No. 2, pp. 10–23, aug 2018.

[111] 塚田学, 小川景子, 池田雅弘, 曽根卓朗, 丹羽健太, 齊藤翔一郎, 粕谷貴司, 砂原秀樹, 江崎浩. Software Defined Media: 視聴空間サービスのソフトウェア制御. 日本ソフトウェア科学会学会誌『コンピュータソフトウェア』「ネットワーク技術」特集, September 2017.

[112] 電気設備学会. BACnet システムインターオペラビリテイガイドライン (IEIEJ-G-0006). 電気設備学会, March 2017.

[113] 藤井竜也, 藤井哲郎, 小野定康, 白川千洋, 白井大介. ディジタルシネマ劇場へのライブ配信（ＯＤＳ）技術. 電子情報通信学会 基礎・境界ソサイエティ Fundamentals Review, Vol. 5, No. 1, pp. 80–89, 2011.

[114] 萩原淳一郎, 瓜生真也, 牧山幸史, 石田基広. 基礎からわかる時系列分析—R で実践するカルマンフィルタ・MCMC・粒子フィルター—. 技術評論社, 2018.

[115] 粕谷貴司. 最新の IoT/AI を活用したスマートビルの実施例：EQ House における AI による無人制御への挑戦 (特集 AI・IoT を活用した建築・設備における新たな価値の創出). BE 建築設備, Vol. 70, No. 12, pp. 24–30, dec 2019.

[116] 粕谷貴司. IoT・BIM を使ったデジタルツインアプリケーションの構築 (特集センシングの現状とビルの運用・保全管理への適用). BE 建築設備, Vol. 71, No. 1, pp. 68–74, jan 2020.

[117] 粕谷貴司, 岡野健二, 茂手木直也, 畠山英之, 矢内雅浩, 小松孝司, 大野翔平. ビッグデータリアルタイム解析による建物管理システム. 空気調和・衛生工学会, Vol. 89, No. 12, pp. 65–70, 2015.

[118] 粕谷貴司, 近藤正芳, 茂手木直也, 松岡康友, 矢野雅, 秋山貴紀, 境野哲, 貞田洋明, 堀越崇, 畠山英之. スマートシティのための MQTT プラットフォームの検証. 情報科学技術フォーラム講演論文集, Vol. 13, No. 4, pp. 1–6, aug 2014.

[119] 粕谷貴司, 曽根卓朗, 塚田学, 安藤亮介, 白浜妥知, 庄子琢郎, 江崎浩. インタラクティブ操作と AR 可視化を実現する立体視聴プラットフォーム. Technical Report 24, 東京大学／株式会社竹中工務店, 静岡大学／株式会社アプリックス, 東京大学, 慶應義塾大学, 株式会社アプリックス, 株式会社アプリックス, 東京大学, oct 2019.

[120] 粕谷貴司, 塚田学, 菰原裕, 高坂茂樹, 水野拓宏, 野村譲誉, 上田雄太, 江崎浩. LiVRation: VR による自由視聴点映像音声のインタラクティブ再生. マルチメディア，分散，協調とモバイル (DICOMO2018) シンポジウム,

June 2018.

[121] 豊田武二. BACS と BACnet の最新動向. 電気設備学会誌, Vol. 32, No. 2, pp. 105–109, 2012.

[122] 平田圭二. 未来の電話を考える–遠隔コミュニケーションシステム t-room (特集コミュニケーション環境の未来に向けた研究最前線). NTT 技術ジャーナル, Vol. 19, No. 6, pp. 10–12, 2007.

[123] 豊田啓介. 建築情報学が広げる可能性. 建築ジャーナル, No. 1290, pp. 14–17, May 2019.

[124] 廣瀬通孝, 小木哲朗, 石綿昌平, 山田俊郎. 多面型全天周ディスプレイ (CABIN) の開発とその特性評価. 電子情報通信学会論文誌. D-II, 情報・システム, II-情報処理, Vol. 81, No. 5, pp. 888–896, May 1998.

[125] 濱崎公男, 火山浩一郎. 22.2 マルチチャンネル音響システム. 平成 17 年電気学会電子・情報・システム部門大会講演論文集, pp. 3–7, September 2005.

[126] 蜷川忠三, 青木佳史, 中村惇志, 森川純次, 近藤成治, 稲葉隆. ビル空調設備スマートグリッド電力抑制予測モデルの平常運転 FastADR 信号モジュレーション訓練法. 電気学会論文誌. D, 産業応用部門誌, Vol. 138, No. 3, pp. 199–205, 2018.

用語集

B-BC (BACnet Building Controller)　BACnet を用いる BACS の構成要素の 1 つ．サブシステムが持つポイントを BACnet のポイントに変換し，B-OWS と連携するためのゲートウェイ． 17, 34

B-OWS (BACnet Operator Workstation)　BACnet を用いる BACS の構成要素の 1 つ．B-BC に対して，読込・書込を行うことが可能なコンポーネントで，設備の統合的な監視・操作を行う中央監視システムなどが該当する． 17, 34

BACnet　BACS のための標準プロトコルの 1 つ．BACS の構成要素を「オブジェクト」「サービス」によってモデル化することで，システムの動作を記述する．オブジェクトとは，BACS の各機能を複数の属性の集合体として抽象化したものであり，サービスとは動作自体を抽象化したものである．1995 年に ANSI/ASHRAE Standard 135 に採用され，その後 2003 年に ISO 16484-5 として採用された． 81

BACS (Building Automation and Control System)　自動制御（インターロック含む），監視，最適化，人的操作，建築設備機器の省エネルギー的で経済的・安全な運転操作を達成するための管理を目的とするシステムと全ての製品・エンジニアリングサービスに対する呼称．従来の日本における中央監視システム (BAS:Building Automation System) や BEMS を包含する概念．ISO 16484 シリーズによって定義されている． 1, 5, 81

BEMS (Building Energy Management System)　ビルの室内環境とエネルギー性能の最適化を図るためのビル管理システム．業務用ビル等、建物内のエネルギー使用状況や設備機器の運転状況を把握し、需要予測に基づく負荷を勘案して最適な運転制御を自動で行うもので、エネルギーの供給設備と需要設備を監視・制御し、需要予測をしながら、最適な運転を行うトータルなシステム． 8, 35

BIM (Building Information Modeling)　コンピュータ上に作成した主に 3 次元の形状情報に加え，室等の名称・面積，材料・部材の仕様・性能，仕上げ等，建築物の属性情報を併せ持つ建築物情報モデルを構築するもの．BIM の導入によって，建築物に関するあらゆる情報が一元化され，情報の重複入力による手間や不整合が削減されると共に可視性が高まり，とりわけ建築生産における品質の向上，工期の短縮，費用の低減，さらには適正な維持管理の実現などに対する効果が期待されている． 1, 5, 41

DR (Demand Response)　市場価格の高騰時または系統信頼性の低下時において，電気料金価格の設定またはインセンティブの支払に応じて，需要家側が電力の使用を抑制するよう電力消費パターンを変化させること． 19, 34, 107

IFC (Industry Foundation Classes)　BIM オーサリングツール間の情報交換用フォーマット．2013 年に ISO 16739 として国際標準化されている．BIM でモデリングされる形状情報（ソリッドモデル）と，空間階層構造や設備機器や部材の属性情報の双方が記録されている． 28, 44, 49

SDM (Software Defined Media)　映像・音響システムの IP ネットワーク化を背景に，これらの設備の機能に対して抽象化・仮想化を行い，サービスとしての映像・音響を提供するための基盤的なアプローチ． 1, 5, 65

SDM **オントロジ**　SDM のアプリケーションにおけるコンテンツ，アプリケーション記述のための語彙セット．SDM ではこれらを知識ベースに格納して活用する． 10, 67

Software Defined　手作業ではなく，ソフトウェアによってサービスの再利用や移植性の向上すること，またソフトウェアによる定義と制御が可能ということ． 1

Software Defined BACS　本研究における造語．可用性の高いローカルシステムである BACS の機能を維持しつつも，その機能的限界を超えるため，ハードウェアの抽象化とデータの管理機能を持ったデータ・プラットフォームを有し，API 連携によって再利用や移植性の高いサービス構築が可能であり，それらのサービス定義（構造や手順）や制御パラメータ等の更新がソフトウェアによって可能なシステム． 1, 33

WoT (Web of Things)　IoT (Internet of Things) で利用される多くの技術やデバイスを，ウェブの技術を用いて抽象化，提供する技術仕様．具体的には，実空間上のオブジェクトを「Thing」という単位でモデリングし，そこに定義されたプロパティやアクションに対して，HTTP を介してインターネットからアクセス可能にする．2020 年 4 月に，WoT 全体のアーキテクチャを定義する WoT Architecture，Thing の記述モデルである WoT Thing Description が W3C (The World Wide Web Consortium) 勧告となった． 81, 91

コモングラウンド　京都大学の西田豊明教授の提唱した，人間社会と AI がともに依拠できる共通基盤を意味する概念．近年では，noiz の豊田啓介が，人間が認識する物理的な世界と，自律走行車やロボットのようなデジタルエージェントが認識するデジタル世界を結びつける概念として，再定義している． 2, 25

デジタルツイン　物理空間にある現実の機器や設備の稼働状況，環境情報をなどをリアルタイムで収集する一方，仮想空間上に機器や設備を構築し，これらのデジタル情報（モデル）を用いてシミュレーションや可視化を行うシステム，ソリューション．建築分野では BIM から抽出した形状データや実空間から収集した時系列データを用いることが多い． 1, 25, 54

プラットフォーム　本研究においては異質なシステム・デバイスの通信を API (Application Programming Interface) を用いて仲介するシステム．データ・プラットフォームは，データを保存する機能も有し，サードパーティシステムからのリクエストに応じて，データを抽出・提供する． 1

ポイント　BACS で必要な物理的または通信の入出力を表す．具体的には，センサ等の計量・計測値，設備デバイスの設定値，警報・故障信号などになる．プロジェクト毎のポイント定義は，ISO16484-3 で規定されている「ポイントリスト」と「機能ブロック線図」等によってなされる． 14, 41

謝辞

本研究に対してご指導いただきました東京大学大学院 江崎浩 教授，塚田学 准教授に深甚の謝意を表します．ビル設備のエンジニアでしかなかった私に，研究テーマを追及するきっかけを与えていただきました．コンピュータサイエンスに通じた先生方のご指導とご助言がなければ，研究成果をまとめることはできませんでした．また，論文審査において，東京大学大学院 五十嵐健夫 教授，稲葉雅幸 教授，千葉滋 教授，蜂須賀恵也 准教授には多数の貴重なご教示を賜りました．ここに深謝申し上げます

本論文は，筆者が株式会社竹中工務店に勤務しながら，社会人学生として研究を行った成果をまとめたものであり，関連部門の多くの方からご助力とご指導を賜りました．竹中工務店の情報エンジニアリング本部のメンバーの応援に感謝いたします．特に，会社の義務を続けながら博士課程で学ぶことを応援してくれた，後神洋介 専門役，政井竜太 情報エンジニアリング本部長には心から感謝しております．部門のメンバーには本研究だけでなく，本業のプロジェクトにおいても多大なサポートを頂きました．講義などで本業の対応を抜けざるを得ないこともあり，多大な迷惑をかけたとも思いますが，ご理解を示していただき，ありがとうございました．

東大グリーン ICT プロジェクトの BIM 基盤 WG のメンバーには，BIM や BACS，システム・アーキテクチャについて有益なご助言を頂くとともに，ヒアリング，調査等に多大な協力を頂きました．WG メンバーでマイクロソフトの本社キャンパスに見学と意見交換にいったことは，印象深い思い出です．

Software Defined Media Consortium のメンバーには，SDM についての重要な知見をいただきました．音響機器やエンタテインメント業界が全く分からなかった当初の私にとって，皆さんのご助言や議論は非常に刺激的でした．

この成果の一部は国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の助成事業の結果得られたものです．ご協力いただいた関係者の方々，プロジェクトメンバーに感謝いたします．

最後に，長い期間にわたって研究生活を見守り支えてくれた妻 文，産まれたばかりの幼い息子 遥貴にも心から感謝いたします．家庭における十分な義務を果たせていなかったにも関わらず，また論文提出間際は，コロナ禍の最中で在宅勤務・研究が続いていましたが，それでも家の中でも暖かく見守ってくれました．同様に学業に対して理解を示してくれた母，遥貴の面倒を見てくれた妻の両親にも深く感謝しています．

皆さんの協力なしには，本論文を完成させることはできませんでした．本当にありがとうございます．

付録 A

futaba の API

本章では，futaba で提供される API について説明する．なお，これらの API は 6 章で示した参考実装においては全て実装されているものではない．詳細は実装完了時点に別途ウェブサイト等に公開予定とし，本稿においてはその概要を示すに留めた．

A.1 概要

AI 学習エンジンや外部システムなどからビッグデータ処理基盤に保存されたデータ取得のリクエストに対して，RESTful API を使用して受け渡しを行う．API を使用するには，管理ツールにて，サードパーティアプリの登録と ID/シークレットの発行，ならびに利用者認証 API にてアクセストークンを取得する必要がある．利用者認証以外の API は，取得したアクセストークンを使用して認証を行い，さらにサードパーティアプリごとに API 使用可否の判断が行われる（図 A.1，図 A.2）．なお，それぞれの API において，リクエストボディとレスポンスは全て JSON フォーマットとなっている．



図 A.1: サードパーティアプリの登録

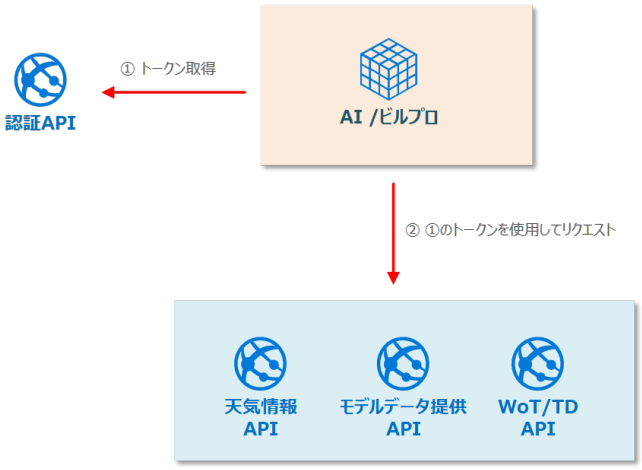


図 A.2: API 利用の流れ

■注意事項　API レスポンスボディの JSON オブジェクトの各項目に順序保証はなく，変動する可能性がある．

表 A.1. futaba の API（再掲）

No.	API 種別	説明	HTTP メソッド	管理権限
1	利用者認証 (Authentication)	アクセストークン発行	POST	○
2	利用者認証 (Authentication)	アクセストークン更新	PATCH	○
3	学習データ取得 (Data Extraction)	モデル学習データ要求タスク 作成	POST	
4	学習データ取得 (Data Extraction)	モデル学習データ要求タスク 詳細確認	GET	
5	学習データ取得 (Data Extraction)	モデル学習データ要求タスク 有効状態変更	PATCH	
6	学習データ取得 (Data Extraction)	モデル学習データ要求タスク キャンセル	DELETE	
7	学習データ取得 (Data Extraction)	モデル学習データ要求タスク WebHook 登録	POST	
8	学習データ取得 (Data Extraction)	モデル学習データ要求タスク WebHook 削除	DELETE	
9	気象情報取得 (Utility)	天気予報データ 取得	POST	
10	メタデータ読み取り (Utility)	建物メタデータ 検索（パス指定）	GET	
11	メタデータ読み取り (Utility)	建物メタデータ 検索（クエリ指定）	POST	
12	メタデータ編集 (Utility)	建物メタデータ 編集	PUT	
13	WoT 読み取り	TD 取得（パス指定）	GET	
14	WoT 読み取り	TD 取得（クエリ指定）	POST	
15	WoT 読み取り	Property 取得（TD 内全取得）	GET	
16	WoT 読み取り	Property 取得（1Property）	GET	
17	WoT 書き込み	Property 書き込み	PUT	
18	WoT 書き込み	Action 実行	POST	
19	WoT 読み取り	Action 状況取得（TD 内全取得）	GET	
20	WoT 読み取り	Action 状況取得（1Action）	GET	
21	WoT 読み取り	Action 詳細状況取得	GET	
22	WoT 読み取り	Event サブスクライブ	POST	
23	WoT 読み取り	Event サブスクライブ解除	DELETE	
24	WoT 読み取り	Event サブスクライブ状況取得	GET	
25	WoT 読み取り	Property 一括取得（パス指定）	GET	
26	WoT 読み取り	Property 一括取得（クエリ指定）	POST	
27	WoT 読み取り	Event 一括サブスクライブ	POST	
28	WoT 読み取り	Event 一括サブスクライブ解除	DELETE	
29	WoT 読み取り	Event 一括サブスクライブ状況取得	GET	

A.2 利用者認証 API

利用者認証 API は各 API を使用する際の API 利用者認証と，アクセストークンの発行および更新処理を行う．

表 A.2. 利用者認証 API 一覧

パス	操作	説明
/api/v1/token	POST	アクセストークン発行
	PATCH	アクセストークン更新

A.2.1 アクセストークン発行

管理ツールを用いて API 利用者に事前発行されたクライアント ID およびクライアントシークレットより，各 API の使用に必要なアクセストークンを発行する API．クライアント ID 発行時にリフレッシュトークンを使用する認証方法を選択し，既にリフレッシュトークンが発行されている場合は本 API で新規にアクセストークンを発行することは出来ないため，アクセストークン更新 API を使用する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-CLIENT-ID	API 利用者に発行されたクライアント ID	ヘッダー	必須	string
X-NEDO-CLIENT-SECRET	API 利用者に発行されたクライアントシークレット	ヘッダー	必須	string
X-NEDO-GRANT-TYPE	認証方法（固定値: ”client_credentials”）	ヘッダー	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
access_token	新規に生成されたアクセストークン	string
refresh_token	新規に生成されたリフレッシュトークン（発行対象クライアントのみ）	string
expires_in	アクセストークンの残り有効時間（分）	int
correlation_id	リクエスト ID（GUID）	string

A.2.2　アクセストークン更新

アクセストークン発行 API により発行されたアクセストークンを更新する API．クライアント ID 発行時にリフレッシュトークンを使用する認証方法を選択した場合は，最新のリフレッシュトークンの指定が必要．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-CLIENT-ID	発行されたクライアント ID	ヘッダー	必須	string
X-NEDO-CLIENT-SECRET	発行されたクライアントシークレット	ヘッダー	必須	string
X-NEDO-REFRESH-TOKEN	発行されたリフレッシュトークン	ヘッダー	任意	string
X-NEDO-GRANT-TYPE	認証方法（固定値: ”refresh_token”）	ヘッダー	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
access_token	新規に生成されたアクセストークン	string
refresh_token	新規に生成されたリフレッシュトークン（発行対象クライアントのみ）	string
expires_in	アクセストークンの残り有効時間（分）	int
correlation_id	リクエスト ID（GUID）	string

A.3　モデル学習データ取得 API

モデル学習データ取得 API は，AI システム等のサードパーティがモデル再学習のため建物の日次のデータを取得するリクエストに対し，ビッグデータ処理基盤からのデータ取得，受け渡しを行う API である．

表 A.3. モデル学習データ取得 API 一覧

パス	操作	説明
/api/v1/task	POST	モデル学習データ要求タスク 作成（即時，スケジュール）
	GET	モデル学習データ要求タスク 詳細確認
	PATCH	モデル学習データ要求タスク 有効状態変更
	DELETE	モデル学習データ要求タスク キャンセル
/api/v1/webhook	POST	モデル学習データ要求タスク WebHook 登録
	DELETE	モデル学習データ要求タスク WebHook 削除

表 A.4. WebHook 一覧

操作	説明
POST	モデル学習データ要求タスク 処理完了通知

A.3.1　モデル学習データ要求タスク 作成 (即時)

ビッグデータ処理基盤からのモデル学習データ生成を要求する API．モデル学習データ生成タスクは，単発かつ即時に実行される．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
exec_type	実行タイプ（固定値: 1）	必須	int
building	土地建物指定	必須	string
start_date	開始日　例: ”2020-01-01”	必須	string
end_date	終了日　例: ”2020-01-02”	必須	string
sample_interval_min	データサンプル間隔（分）　[1, 5, 10, 15, 30]	任意	int
utc_offset	UTC との差　例: ”+9:00”	必須	string
points	ポイント指定配列	必須	object[]
file_type	出力ファイルタイプ [”parquet”, ”orc”]	必須	string

レスポンス

名前	説明	データタイプ
task_id	モデル学習データ要求タスク ID	string
correlation_id	リクエスト ID (GUID)	string

A.3.2　モデル学習データ要求タスク 作成（スケジュール）

ビッグデータ処理基盤からのモデル学習データ生成を要求する API．モデル学習データ生成タスクは，単発かつスケジュール実行される．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
exec_type	実行タイプ（固定値: 2）	必須	int
exec_datetime	スケジュール実行日時（ISO8601 UTC）	必須	string
building	土地建物指定	必須	string
start_date	開始日　例: ”2020-01-01”	必須	string
end_date	終了日　例: ”2020-01-02”	必須	string
sample_interval_min	データサンプル間隔（分）　[1, 5, 10, 15, 30]	任意	int
utc_offset	UTC との差　例: ”+09:00”	必須	string
points	ポイント指定配列	必須	object[]
file_type	出力ファイルタイプ [”parquet”, ”orc”]	必須	string

レスポンス

名前	説明	データタイプ
task_id	モデル学習データ要求タスク ID	string
correlation_id	リクエスト ID (GUID)	string

A.3.3 モデル学習データ要求タスク 作成 (定期スケジュール)

ビッグデータ処理基盤からのモデル学習データ生成を要求する API．モデル学習データ生成タスクは，定期スケジュール実行される．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
exec_type	実行タイプ（固定値:3）	必須	int
exec_datetime	スケジュール実行日時（ISO8601 UTC）	必須	string
building	土地建物指定	必須	string
sample_interval_min	データサンプル間隔（分） [1, 5, 10, 15, 30]	任意	int
utc_offset	UTC との差 例: ”+09:00”	必須	string
schedule	スケジュールパターン文字列	必須	string
points	ポイント指定配列	必須	object[]
file_type	出力ファイルタイプ [”parquet”, ”orc”]	必須	string

レスポンス

名前	説明	データタイプ
task_id	モデル学習データ要求タスク ID	string
correlation_id	リクエスト ID (GUID)	string

A.3.4 モデル学習データ要求タスク 詳細確認

クライアントシステムが登録したモデル学習データ要求タスクの一覧と，定期スケジュールタスクの実行履歴を取得する API．リクエストパラメータでモデル学習データ要求タスク ID を指定した場合は，指定したタスクの詳細情報と実行履歴（最新 1 年分）を返却する．モデル学習データ要求タスク ID を指定しない場合はクライアントシステムが登録した全ての削除済みでないタスクの詳細情報を返却する．このとき，各タスクの実行履歴は最新 10 件を返却する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
task_id	モデル学習データ要求タスク ID	クエリパラメータ	任意	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
tasks	タスク詳細配列	object[]
correlation_id	リクエスト ID (GUID)	string

A.3.5 モデル学習データ要求タスク 有効状態変更

モデル学習データ要求タスク（単発スケジュール/定期スケジュール）のスケジュール実行の有効状態を変更する API．現在の有効状態を再度指定して実行してもエラーとしない．（冪等性あり）

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
task_id	モデル学習データ要求タスク ID	必須	string
enabled	有効状態 (false で一時停止) [”true”, ”false”]	必須	string

レスポンス

名前	説明	データタイプ
correlation_id	リクエスト ID (GUID)	string

A.3.6 モデル学習データ要求タスク キャンセル

モデル学習データ要求タスクの次回以降のスケジューリングを全て解除し，タスク状態を「削除済」に変更する API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
task_id	モデル学習データ要求タスク ID	クエリパラメータ	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
correlation_id	リクエスト ID (GUID)	string

A.3.7 モデル学習データ要求タスク WebHook 登録

モデル学習データ要求タスク終了時の WebHook 送信先 URL を設定する API. WebHook URL はクライアントシステムあたり 1 件まで登録可能であり，既に登録されている場合はエラーとする.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
webhook_url	WebHook URL	必須	string

レスポンス

名前	説明	データタイプ
webhook_id	WebHook ID	string

A.3.8 モデル学習データ要求タスク WebHook 削除

モデル学習データ要求タスク終了時の WebHook 送信先 URL 設定を削除する API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
webhook_id	WebHook ID	クエリパラメータ	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
correlation_id	リクエスト ID (GUID)	string

A.3.9 モデル学習データ要求タスク 処理終了通知

モデル学習データの生成処理が終了したことをクライアントシステムに通知する WebHook

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	データタイプ
exec_status	タスク処理結果 [”complete”, ”error”, ”canceled”]	string
download_url	タスク実行結果ファイル保存先 URL	string
error_code	エラーコード文字列 ※「”error”」時のみ	string
error_description	エラーメッセージ列 ※「”error”」時のみ	string
error_datetime	エラー発生時間 ※「”error”」時のみ	string
correlation_id	リクエスト ID（GUID）	string

A.4　天気情報データ取得 API

天気予報データ取得 API は外部の気象情報サービスベンダが提供する天気予報情報を取得し，サードパーティに取得した情報の受け渡しを行う API である．

表 A.5. 天気情報データ取得 API 一覧

パス	操作	説明
/api/v1/weather	POST	天気予報データ 取得
	DELETE	モデル学習データ要求タスク WebHook 削除

A.4.1　天気予報データ

データ・プラットフォームに蓄積済みの天気予報データを取得する API．リクエストボディの取得対象日を無指定とした場合は最新の 1 件を返却し，指定した場合は指定日の天気予報データを全て返却する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
date	取得対象日 (”yyyy-MM-dd”)	任意	string

レスポンス

名前	説明	データタイプ
（無名）	天気予報データ JSON オブジェクト配列	object[]

A.5 建物メタデータ API

建物メタデータ API はサードパーティのシステムより，建物メタデータの検索および編集を行う API である．

表 A.6. API 一覧

パス	操作	説明
/api/v1/metadata	GET	建物メタデータ 検索（パス指定）
	POST	建物メタデータ 検索（クエリ指定）
	PUT	建物メタデータ 編集

A.5.1 建物メタデータ 検索（パス指定）

BOT パス指定により，建物メタデータの検索と閲覧を行う API

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
path	BOT パス指定	クエリパラメータ	必須	string

■BOT パス指定例

例１）Light 以下の要素を全て指定

```
hongo.wide.ad.jp/Sample_House/-/Entrance/Light/*
```

例 2）Light を指定

```
hongo.wide.ad.jp/Sample_House/-/Entrance/Light
```

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
result	メタデータ検索結果 JSON 配列	object[]
correlation_id	リクエスト ID(GUID)	string

A.5.2 建物メタデータ 検索（クエリ指定）

例）クエリ指定により，建物メタデータの検索と閲覧を行う API

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
query_type	クエリタイプ　[”odata”/”sparql”]	必須	string
query	クエリ文字列	必須	string

■クエリ例：OData フィルタ (odata)

例）Entrance にある EcoWine の建物メタデータを指定

```
$filter=space eq 'Entrance' and type eq 'Eco-wine'
```

■クエリ例：SPARQL クエリ (query)

例）機器 AHU-1 に紐づく要素の建物メタデータを指定

```
select * where { inst_device:AHU-1 brick:hasPoint ?point . }
```

レスポンス

名前	説明	データタイプ
result	メタデータ検索結果 JSON 配列	object[]
correlation_id	リクエスト ID(GUID)	string

A.5.3 建物メタデータ編集

建物メタデータ上の項目を追加，更新する API．リクエストボディの id で指定された建物メタデータ内の項目を data の内容で置換する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
id	対象建物メタデータの ID	必須	string
data	置換する項目とデータの組	必須	object

レスポンス

名前	説明	データタイプ
result	メタデータ検索結果 JSON 配列	object[]
correlation_id	リクエスト ID(GUID)	string

A.6 WoT API

WoT API はサードパーティのシステムが WoT に沿って TD (Thing Description) の情報取得や値のセット (遠隔制御などの目的) を行う際に利用する API である．WoT API は TD API，Property API，Action API，Event API に大別される．

表 A.7. WoT API 種別

API	説明
TD API	WoT で操作を行う際の基点となる，存在するデータモデルの各要素の TD を検索・取得する API
Property API	TD で規定される Property エンドポイントを通じて，ポイントの現在値取得や，建物遠隔制御のためポイントへの値のセットを行う API
Action API	TD で規定される Action エンドポイントを通じて，事前に建物側 GW に定義されたメソッドの実行要求や履歴の確認を行う API
Event API	各ポイントからアップロードされる値を外部のシステムが非同期で受け取るため，TD で規定される Event エンドポイントを通じて購読の登録解除を行う API

表 A.8. WoT API 一覧

パス	操作	説明
/api/v1/things	GET POST	TD 取得（パス指定） TD 取得（クエリ指定）
/api/v1/things/{TD-ID}/properties	GET	Property 取得（TD 内全取得）
/api/v1/things/{TD-ID}/properties/{PointID}	GET PUT	Property 取得（1Property） Property 書き込み
/api/v1/things/{TD-ID}/actions	GET	Action 状況取得（TD 内全取得）
/api/v1/things/{TD-ID}/actions/{ActionName}	GET POST	Action 状況取得（1Action） Action 実行
/api/v1/things/{TD-ID}/actions/{ActionName}/{TaskID}	GET	Action 詳細状況取得
/api/v1/things/{TD-ID}/events/{EventName}	POST	Event サブスクライブ
/api/v1/things/{TD-ID}/events/{EventName}/{SubscriptionID}	DELETE	Event サブスクライブ解除
/api/v1/things/{TD-ID}/events	GET	Event サブスクライブ状況取得

A.6.1 TD 取得（パス指定）

BOT パス指定により，TD を検索，取得する API．BOT パスの末尾が ”*” の場合は指定したパス配下の全ての TD を取得し，”*” 以外の場合は指定されたパスの TD1 件のみを取得する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
path	BOT パス指定	クエリパラメータ	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
things	TD 本文 JSON 配列	object[]
correlation_id	リクエスト ID(GUID)	string

A.6.2 TD 取得（クエリ指定）

クエリ指定により TD を検索，取得する API．クエリにより検索された全ての対象の TD を返却する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
query_type	クエリタイプ　[”odata”/”sparql”]	必須	string
query	クエリ文字列	必須	string

レスポンス

名前	説明	データタイプ
things	TD 本文 JSON 配列	object[]
correlation_id	リクエスト ID(GUID)	string

A.6.3 Property 取得（TD 内全取得）

TD-ID で指定された特定の TD に紐づく，全ての Property（ポイントおよびモード値）の現在値を取得する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
properties	TD に紐づく全 Property の現在値連想配列	{[key: string]: object;}
correlation_id	リクエスト ID(GUID)	string

A.6.4 Property 取得（1Property）

TD-ID で指定された TD に紐づく，特定の Property（ポイントおよびモード値）の現在値を取得する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{PointID}	ポイント ID	URL パス	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
query_type	クエリタイプ　[”odata”/”sparql”]	必須	string
query	クエリ文字列	必須	string

レスポンス

名前	説明	データタイプ
property	Property の現在値連想配列	{[key: string]: object;}
correlation_id	リクエスト ID(GUID)	string

A.6.5 Property 書き込み

TD-ID で指定された TD に紐づく，特定の Property（ポイントおよびモード値）に値を書き込む WoT 準拠 API. Property の拡張属性である 「isVirtualProperty」 属性が true の場合，遠隔制御ではなく，現在値データベースの値更新動作となる.（モード値の更新に使用）

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{PointID}	ポイント ID	URL パス	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
values.value	書き込む Property の値列	必須	object

レスポンス

名前	説明	データタイプ
values/value	書き込みを行った Property の値	object
correlation_id	リクエスト ID(GUID)	string

A.6.6 Action 実行

TD-ID で指定された TD に紐づく，特定の Action を実行する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{ActionName}	Action 名	URL パス	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
values	Action を実行するために TD で規定されている入力事項	必須	object

レスポンス

名前	説明	データタイプ
values	指定された input 値	object
href	Action の実行状況を取得するための URL	string
time_requested	Action 実行要求時刻 (ISO8601 UTC)	string
status	Action の状況 (例：”executing”)	string
correlation_id	リクエスト ID(GUID)	string

A.6.7 Action 状況取得（TD 内全取得）

TD-ID で指定された特定の TD に紐づく，全ての Action の実行要求履歴を取得する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
actions	TD に紐づく全 Action の実行要求履歴.	object
correlation_id	リクエスト ID(GUID)	string

A.6.8 Action 状況取得（1Action）

TD-ID で指定された TD に紐づく．特定の Action の実行要求履歴を取得する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{ActionName}	Action 名	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
actions	指定した特定の Action タスクの実行要求履歴	object
correlation_id	リクエスト ID(GUID)	string

A.6.9 Action 詳細状況取得

TD-ID で指定された TD に紐づく．特定の Action タスクの実行要求履歴を取得する WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{ActionName}	Action 名	URL パス	必須	string
{TaskID}	Action タスク ID	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
actions	指定した特定の Action タスクの実行要求履歴	object
correlation_id	リクエスト ID(GUID)	string

A.6.10 Event サブスクライブ

TD-ID で指定された TD に紐づく．特定の Event のサブスクライブ（購読登録）を行う WoT 準拠 API．リクエストボディに必要な項目やレスポンス項目は TD に規定され．Event によって変化する．また．Event サブスクリプション失効日時を経過すると．自動的にサブスクライブが解除される．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{EventName}	Event 名	URL パス	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
(例：callback_url)	(例: WebHook 送信先 URL)．Request Body のパラメータは TD で規定される	TD 依存	object

レスポンス

名前	説明	データタイプ
subscription_id	Event サブスクリプション ID	object
expire_at	Event サブスクリプション失効日時 (ISO8601 UTC)	string
(例：eventhub_name)	(例：EventHub 名)．TD で規定されている Response 項目	object
correlation_id	リクエスト ID(GUID)	string

A.6.11 Event サブスクライブ解除

TD-ID で指定された TD に紐づく．特定の Event のサブスクライブ解除（購読解除）を行う WoT 準拠 API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string
{EventName}	Event 名	URL パス	必須	string
{SubscriptionID}	Event サブスクリプション ID	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
correlation_id	リクエスト ID(GUID)	string

A.6.12 Event サブスクライブ状況取得

TD-ID で指定された TD に紐づく．全ての有効な Event サブスクライブ状況を取得する WoT 準拠 API．他のクライアントシステムのよるサブスクライブ状況は含まれず．自身のサブスクライブ状況のみを返却する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{TD-ID}	TD の固有 ID	URL パス	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
events	全ての Event サブスクライブ状況	object[]
correlation_id	リクエスト ID(GUID)	string

A.7 WoT 拡張 API

A.7.1 概要

WoT 拡張 API は WoT API と同様に外部のシステムからのやりとりを行う API である．WoT では規定されていないが，建物管理では必要となる API として設計した．具体的には TD を跨る Property の一括取得や Event 一括サブスクライブなど，複数のポイントを 1 回のリクエストで処理できるようにしたものである．

表 A.9. WoT 拡張 API 一覧

パス	操作	説明
/api/v1/things/properties	GET	TD 取得（パス指定）
	POST	TD 取得（クエリ指定）
/api/v1/things/events/{EventName}	GET	Property 取得（TD 内全取得）
/api/v1/things/events/{EventName}/{SubscriptionID}	GET	Property 取得（1Property）
/api/v1/things/events	PUT	Property 書き込み

A.7.2 Property 一括取得（パス指定）

BOT パス指定により TD を検索し．検索された TD に紐づく全ての Property の現在値を一括取得する API．BOT パスの末尾が ”*” の場合は指定したパス配下の全ての TD を取得対象とし．”*” 以外の場合は指定されたパスの TD1 件のみを取得対象とする．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
path	BOT パス指定（URL エンコーディング適用）	クエリパラメータ	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
building	土地建物指定	string
things	TD-ID を Key．PropertyValuesResponse を Value とするオブジェクトの配列	object[]
query	クエリ文字列	string

A.7.3 Property 一括取得（クエリ指定）

クエリ指定により TD を検索し．検索された TD に紐づく全ての Property の現在値を一括取得する API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
query_type	クエリタイプ [”odata”/”sparql”]	必須	string
query	クエリ文字列	必須	string

レスポンス

名前	説明	データタイプ
building	土地建物指定	string
things	TD-ID を Key．PropertyValuesResponse を Value とするオブジェクトの配列	object[]
query	クエリ文字列	string

A.7.4 Event 一括サブスクライブ (ポイント ID 指定)

建物内の複数のポイントに対して．システムで定義済みの一括設定 Event のサブスクライブ（購読登録）を行う API．リクエストボディに必要な項目やレスポンス項目は一括設定 Event ごとにそれぞれ規定される．また．Event サブスクリプション失効日時を経過すると．自動的にサブスクライブが解除される．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{EventName}	一括設定 Event 名	URL パス	必須	string

リクエストボディ

名前	説明	必須指定	データタイプ
building	土地建物指定	必須	string
points	ポイント ID 配列	必須	string[]
(例：callback_url)	(例：WebHook 送信先 URL) 一括設定 Event ごとにそれぞれ規定	Event 定義による	object

レスポンス

名前	説明	データタイプ
subscription_id	Event サブスクリプション ID	string
expire_at	Event サブスクリプション失効日時 (ISO8601 UTC)	string
(例：eventhub_name)	(例：EventHub 名). 一括設定 Event ごとに規定されている Response 項目	object
correlation_id	リクエスト ID(GUID)	string

A.7.5 Event 一括サブスクライブ解除

サブスクライブ済みの一括設定 Event のサブスクライブ解除（購読解除）を行う API.

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string
{EventName}	一括設定 Event 名	URL パス	必須	string
{SubscriptionID}	Event サブスクリプション ID	URL パス	string	

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
correlation_id	リクエスト ID(GUID)	string

A.7.6 Event 一括サブスクライブ状況取得

全ての有効な一括設定 Event のサブスクライブ状況を取得する API．他のクライアントシステムのよるサブスクライブ状況は含まれず，自身のサブスクライブ状況のみを返却する．

リクエストパラメータ

名前	説明	タイプ	必須指定	データタイプ
X-NEDO-ACCESS-TOKEN	有効なアクセストークン	ヘッダー	必須	string

リクエストボディ

なし

レスポンス

名前	説明	データタイプ
body	全ての一括設定 Event サブスクライブ状況．Event 名を Key．EventSubscribeInfoResponse オブジェクトを Value とした連想配列オブジェクトの配列	object[]
correlation_id	リクエスト ID(GUID)	string