東京大学
情報理工学系研究科 創造情報学専攻
修士論文

デジタルツインの相互運用性向上に向けた データ資産の包括的管理手法

Comprehensive management approach for data assets
to improve interoperability of the Digital Twin

井口 和真
Kazuma Inokuchi

指導教員　塚田 学 准教授

2023 年 1 月

概要

近年、デジタル技術を人々の生活基盤に援用することを目的として、現実世界とデジタル世界の高度な融合による情報の利活用が目指されている。特にスマートビルディングと呼ばれる分野は、エネルギーとコストの最適化や賑わい創出に向けた空間利活用などを目的とする。そのため、建物要素や什器といった施設内の主要設備に加え、制御プロセスや移動ロボットなどの情報を含めた一元的な状態管理が求められる。その要求を満たす基盤となる概念として、デジタルツインが注目を集めている。デジタルツインは、センサから取得した現実世界のあらゆる情報から、実空間全体をデジタル世界上に複製し、忠実度の高いシミュレーションをおこなう。一方で、デジタルツインはその対象の物理的・時間的・構造的複雑さを原因として、データ資産管理が課題となっている。統一的な管理手法が整っていないため、データ間の関係性が不明瞭となり、プロジェクト間の相互運用性も低下する。そこで本研究では、デジタルツインのデータ資産を包括的に管理するためのオントロジーである Digital Twin Ontology (DTO) 及びデータ編集・加工プロセスを定式化するための Data Conversion Ontology (DCO) を提案する。初めに、DTO では、現実世界のエンティティの状態を柔軟に記述可能なだけでなく、設計者視点に着目することで、エンティティが抽象化された仮想モデルとの相互連携を満たすフレームワークとなっている。次に、DCO では、ある “系” から別の “系” への変換をオントロジーとして設計することで、データの編集・加工プロセスを定式化する。提案オントロジーの活用例として、オフィスを模したビル内環境に対して 2 種類の実装をおこなった。DTO に準拠するリンクトグラフデータセット及びクエリ結果を元に環境を再現するビューアアプリケーションと、DCO に基づく座標変換システムを実装した。実装を通して、最小限の先験的知識によりデータ資産を取得可能であることを検証・考察した。また、既存手法との比較やドメインの専門家へのインタビューによる提案オントロジーの内容評価もおこない、提案手法である DTO 及び DCO の有用性を評価した。

Abstract

Recently, the application of advanced fusion of the real world and the digital world has been aimed at applying digital technology to our life foundation. In particular, in the field of smart buildings, centralized state management of not only main building facilities but also digital data such as control processes, and dynamic objects like people and robots is required to optimize energy and costs and to utilize space to promote activities. The digital twin is attracting attention as a concept that can serve as a foundation for meeting these requirements. Besideds, the digital twin replicates the entire real space in the digital world from all the information in the real world obtained from sensors and simulates it with high fidelity. On the other hand, the physical, temporal, and structural complexity of the digital twin makes data asset management challenging. The lack of a unified management approach obscures the relationships among data and reduces interoperability. In this study, we designed a Digital Twin Ontology that generically models real space and serves as a structural framework for digital twin data assets. The framework enables a flexible description of the state of entities in the real world and satisfies interoperability with virtual models in which entities are abstracted by focusing on the designers. We also design a Data Conversion Ontology as an ontology expressing conversion from one system to another, especially coordinate conversions. As an example of the application of the proposed ontology, we implemented a link graph dataset that reproduces the environment that imitates an office in a building, and a viewer based on the results of queries to this dataset. We verified and discussed the feasibility of the implementation with a minimum of a priori knowledge, and evaluated the proposed ontology by comparing it with existing methods and by interviewing domain experts.

目次

第 1 章　序論 1
1.1 はじめに 1
1.2 本研究について 2
1.3 論文構成 3

第 2 章　デジタルツインと現状の課題 4
2.1 デジタルツインの概要 4
2.2 課題: データ資産管理 6

第 3 章　関連研究 8
3.1 セマンティック Web 8
3.2 デジタルツインアーキテクチャ 12
3.3 ソフトウェア設計論 13

第 4 章　提案手法 16
4.1 要求事項 16
4.2 想定する室内環境 17
4.3 アーキテクチャ設計 17
4.4 Digital Twin Ontology 19
4.5 Data Conversion Ontology 22
4.6 Alignments 25

第 5 章　実装 27
5.1 実証フィールドの概要 27
5.2 Digital Twin Ontology に基づくリンクトグラフデータ構築 29
5.3 Data Conversion Ontology を用いたデータ変換処理 31
5.4 ビューアアプリケーションの実装と詳細 34

第 6 章　評価と考察 38
6.1 知識の網羅性と人気度に関するオントロジーの評価 38
6.2 既存オントロジーとの比較による定性評価 39

6.3 シナリオごとの評価 42
6.4 ユーザによる内容評価 45
6.5 考察 48

第 7 章 結論と今後の課題 51
7.1 まとめ 51
7.2 今後の課題 51

発表文献と研究活動 53

参考文献 54

付録 A Abbreviations 59

付録 B Digital Twin Ontology の語彙一覧 61
B.1 クラス一覧 61
B.2 オブジェクトプロパティ一覧 68
B.3 データ型プロパティ一覧 75

付録 C Data Conversion Ontology の語彙一覧 77
C.1 クラス一覧 77
C.2 オブジェクトプロパティ一覧 80
C.3 データ型プロパティ一覧 82

第 1 章

序論

1.1 はじめに

様々なデバイスの登場による IoT (Internet of Things) の普及や、計算機能力の向上と機械学習の隆興による AI (Artificial Intelligence) 技術の発展に伴い、デジタル技術を人々の生活基盤に援用し、人間中心の社会的課題の解決と産業発展を目指す取り組みが注目されている。

特に、BACS (Building Automation and Control System) [1] や Facility Management [2] 等の領域を代表とする建設業や製造業分野では、エネルギーやコストの最適化、各種プロセスの自動化、賑わいの創出などが期待されている [3]。例えば、IoT による運用コスト削減・異常検知の自動化 [4] や、AI を活用して省エネルギーと快適性の両立を目指す技術 [5] など、ウェアラブルデバイスや無線センサを導入し、ビルや工場など現場の環境情報を取得しながら設備を効率的に運用/監視する事例が登場している。

上記の背景より、近年は実空間 (物理空間) とサイバー空間 (仮想空間) の高度な融合による情報の利活用を中心とした概念である、デジタルツインが注目され始めている [6]。デジタルツインには明確な定義が存在しないが、現在では、仮想空間上に実空間を模したレプリカを構築し、現実世界からデータを取得しながら仮想空間上でシミュレーションをおこなうことでリアルタイムに現実空間と仮想空間を連携させることを指す場合が多い。例えば、東京都が主導するデジタルツイン実現プロジェクト [7] では「センサー等から取得したデータをもとに、建物や道路などのインフラ、経済活動、人の流れ等、様々な要素を、サイバー空間 (コンピュータやコンピューターネットワーク上の仮想空間) 上に『双子 (ツイン)』のように再現したもの」とされている。

デジタルツインはその特性から、対象とするデータ資産の管理に大きな課題を抱えている。デジタルツインは実空間に存在するあらゆるエンティティ (実体、実在物) の情報を対象としているため、センサー等による観測から得られる物理特性の複雑度に加え、時間変化及び過去の履歴など、ライフサイクル管理の複雑度も含まれる。さらに援用されるアプリケーション上のコンテンツデータや開発者が設計したデジタルモデルなど、エンティティに関する人間活動までその管理下におくため、デジタルツインの複雑さがデータ資産管理の大きな問題となっている。

1.2　本研究について

デジタルツインのデータ資産管理に対するアプローチとして、セマンティクスを活用した先行研究/関連技術や参照アーキテクチャが提案されている。一方、これらは (1) 実空間から取得したデータとそのデジタルデータの混在、(2) 動的 (時系列変化) 情報の記述が困難、(3) アプリケーションの実装まで見据えたアーキテクチャ及びその活用方法が不透明、といった課題を抱えており、デジタルツインの基本要件 (第 2.1 章で詳細記述) を満たしていない。

本研究では、デジタルツインが抱える多様かつ異質なデータ資産管理を、オントロジーと呼ばれるメタデータスキーマをベースとした手法により解決することを目的とする。従来の二元論的なデジタルツインを拡張し、設計者視点を導入したアーキテクチャに基づく Digital Twin Ontology (DTO) ならびに、データの編集・加工プロセスを定式化する Data Conversion Ontology (DCO) を提案した。複数センサが配置され、建物構造がデジタル化された実験場に本提案を適用し、得られたデータセットから実験場を再現するビューアアプリケーションを実装することで、データ資産の包括的管理やアプリケーションの相互連携性を検証した。

本研究の貢献を下記にまとめる。

DTO の提案

デジタルツインが抱える多様なデータ資産をセマンティクスに基づき管理可能とする DTO を提案した。DTO は、現実世界の物理状態と設計者視点での機能・振る舞いに基づき分類し、デジタルツインの構成要素を体系的かつ相互連携性を考慮した形で記述可能である。また、DTO が提供する語彙は、対象事物の属性や振る舞いを汎用的に記述することから、建築/建物基盤からセンサをはじめとする IoT デバイスまで、そのデータまたは外部オントロジーと柔軟に連携可能である。

DCO の提案

アプリケーション構築の過程で発生する、データの編集・加工プロセスを定式化する DCO を提案した。DCO は、オントロジーによるグラフ構造を有限オートマトンの状態遷移図とみなすことで、ある”系”から別の”系”への変換に必要な構成要素を構造化する。現行バージョンでは、実行関数自体を外部のオープンサービスに委譲し、最小構成として提案した。

提案アーキテクチャの実証とデジタルツイン構築

複数のセンサや動的事物が存在する実験場を用意し、提案アーキテクチャの実証実験を実施した。実験場から取得した建物情報やセンサログ等の多様かつ異質なデータ群を、提案するオントロジーに準拠するリンクトグラフデータとして構築した。また、リンクトグラフデータへ問い合わせた結果を可視化するビューアアプリケーションを実装し、提案アーキテクチャによって、多様なデータの編集性やアプリケーションの相互運用性向上が可能であることを検証した。

既存仕様の整理及び提案オントロジーの内容評価

デジタルツインの実現に向けたメタデータスキームの中で、特に建物やセンサ、IoT 分野で注目を集めている既存仕様及びアーキテクチャの提案をおこなう学術論文を整理した。また、従来手法と提案オントロジーとの比較による定性評価、さらにドメインの専門家へのインタビューを通した内容評価を実施した。提案手法を含め、デジタルツインリファレンスアーキテクチャが対面する現在及び将来の課題の検討及び考察をおこなった。

1.3 論文構成

本論文の構成は次のようになっている。第 2 章では、デジタルツインの概要を説明したのち、データ資産管理に対する課題について述べる。第 3 章で、先行研究の紹介とともに要件を整理し、第 4 章でデジタルツイン及びアプリケーション設計を見据えたオントロジーを提案する。第 5 章では実際に収集したデータを用いて提案オントロジーに基づくアプリケーションの実装について述べる。第 6 章で提案手法の評価を実施し、第 7 章で今後の課題についてまとめる。

第 2 章

デジタルツインと現状の課題

本章では、デジタルツインの歴史及び概要を述べた後、その必要性と要件、さらにデジタルツインが現在抱える課題について説明する。

2.1　デジタルツインの概要

デジタルツインが最初に登場したのは、2003 年にミシガン大学の Michael Grieves が産業界に向けた講演にて提唱されたものである [8]。Grieves は「物理システムに関するデジタル情報の構成要素で、それ自体が実体として作成され、対象の物理システムにリンクされているもの」 [9] をデジタルツインの原型として発表した。その後 2011 年に、デジタルツインが航空機の構造寿命予測に有用であることを説明した学術論文 [10] が発表されたほか、2012 年には NASA が定義を正式に発表し、航空宇宙産業における展望を描いた [11]。Tao ら [6] の調査によると、デジタルツインが 1 つの概念的アイデアから多数の実用的アプリケーションへ成長したことを反映する白書が発表された 2014 年以降、アカデミック分野でも急速に注目を集め、急成長期に移行した。

デジタルツインは、狭義では「ミクロの原子レベルからマクロの幾何学的なレベルまで、現実上の物体を完全に記述する一連の仮想情報」と提唱されており [12]、物理的な製品をいかにリアルにシミュレートするかという問題を解決する。物理的なオブジェクトに対応する仮想的な情報の集合に限定するツインの考えに基づき、理想的には、仮想的、動的、多物理的 (polyphysical)、マルチスケール、統合的、計算可能、超臨場感という特徴を持つ。一方で広義としては、システムの状態やプロセスをシミュレーションし、監視、計算、調整、制御することができる統合システムとされる。物流、情報の流れなどを体系的、リアルタイムに制御し、システム内の全ての事業活動を効果的に調整、最適化することを目的としており、仮想空間上のツインはレプリカとしてだけでなく、物理的な製品のライフサイクルを反映させ、過去及び現在のデータを元にリアルタイムに監視・制御可能なシステムとして構築される。

CPS (Cyber Physical System) は、計算機資源と物理資源の密接な結合、協調をおこなうシステム [13] であり、組み込みシステム、センサー、ネットワーク技術に基づき、物理世界とデジタル世界との相乗効果で自己認識や自己適応の能力を持つ複雑システムとして期待されてい

る [14]。CPS とデジタルツインはサイバーフィジカル統合の記述において類似しており、ともに物理的な部分とデジタルな部分から構成されているなど共通点を持つが、相違点も存在する。例えば CPS は市場戦略や特定のアプリケーションを参照しないため科学カテゴリに関連する基礎分野として考えられる一方で、デジタルツインは工学的カテゴリに関連する [15]。また、CPS はセンサーとアクチュエータを中核となる要素としており、それらのフィードバックを通じて物理プロセスの変化をデジタル世界に反映させることを主としているが、デジタルツインの中核となる要素はモデルとデータであり、現在及び過去のデータ、経験や知識によるデジタルモデルを用いて物理オブジェクトの振る舞いを解釈、予測可能なシステムである。

CPS の登場により、ライフサイクルの初期段階で完了するのが一般的であった複雑システムは、システムの全ての動作と作業プロセスを一元的に定義することが望まれるようになった。しかし物理的な異種システムや仮想的な抽象設計の複雑さから、動的な CPS では現実世界からシステムが孤立し、予測不可能になる。CPS の概念において、サイバー空間は物理オブジェクトのツインが相互接続されたネットワーク空間と定義されているように [16]、ツインは純粋にサイバー空間の中に存在し、意味を持つに過ぎない。したがって物理世界と仮想世界の融合を実現し、異種システムが織りなす複雑性に対処し、動的に増加するデータを利用できる新しいフレームワークが必要とされている。完全なデジタルツインは物理モデル、仮想モデル、それらの間の接続を含むため、物理空間とサイバー空間の相互作用による統合が期待される。

Redelinghuys ら [17] は、デジタルツインが備えるべき要件として以下の 4 つに整理している。

- 物理ツインの状態取得
  デジタルツインは現実世界における様々な種類のセンサからデータを取得し、物理ツインの状態を記述する情報セットとして参照・補強されなければならない。
- 情報リポジトリの維持
  現実世界のセンサーから得た状態情報は、大量のデータを保管し簡単に取得できるようにする必要があることから、高いスケーラビリティとアクセシビリティを備えた保存が必要である。
- 制御のシミュレーション
  物理ツインにおいて、ある初期状態から選択された一連の条件から将来の動作を予測することは、新しいプロセスや異なる生産スケジュールの評価など、想定されているデジタルツインの役割においてシミュレーションが要求される。
- 動作のエミュレーション
  センサーから取得した状態情報をもとに、現実世界の動作を模倣し、物理ツイン上において振る舞いや機能を視覚的に表現・再現する必要がある。

2.2　課題: データ資産管理

先述の通り、デジタルツインは現実世界を忠実に模した物理ツインと経験や知識によるデジタルモデルを用いて、エンティティに関する状態情報を解釈し、過去から現在そして未来に渡り予測可能なシステムとして構築されることが期待される。デジタルツインは現在の複雑システムを最適化する大きな可能性を秘めているが、モデルとデータをその中核要素とするデジタルツインにおいて、データ資産の管理に関する問題はより複雑になっている。データ管理は製品設計から資産管理・保守をおこなう既存システム上でも典型的な問題となっていることを踏まえ、本章ではデジタルツインにおけるデータ資産管理に関する現状の課題をを下記の 6 点にまとめる。

データモデルの Heterogeniety

Heterogeniety とは、異質性や不均一性などと呼ばれている。IoT の領域でも産業課題として注目されているが、現実世界に存在するあらゆるエンティティの統合管理を目指すデジタルツインにおいても同様の問題を抱えている。第一に、デジタルツインの対象となるエンティティは、建物、設備、什器から人、ロボット、デバイスなど物理的対象は多岐に渡る。さらに現在・過去・未来における対象のライフサイクルや状態管理を管理する必要があるため、時間的変化への追従も要求される。また現実世界で発生した物理現象としての振る舞いの観測だけでなく、デジタルモデルの役割としてエンティティのモデリングや継続的なミラーリングをおこなうことから、構造・機能・振る舞いといったアプリケーションデータの多様性も存在する。このように物理的・時間的・機能的異質性を抱えており、そのデータ資産の相互運用性問題の解決が必要とされている。

参照アーキテクチャの欠如

多くの論文がデジタルツインモデルの重要性を認めている一方で、それを汎用的に構築する方法について外部参照可能な形で提案されている論文は少なく、またコンセンサスも取れていない (詳細は第 3.2 章を参照)。Tao らが発表したレビュー論文においても、統一的なデジタルツインモデリングのフレームワークの必要性が唱えられている [6]。

空間記述仕様の個別発展

時空間を包括的に管理対象とするデジタルツインにおいて、既存の記述仕様もいくつか注目を集めている。その中の一つに空間記述仕様があり、KML (Keyhole Markup Language) をベースとする Google Map、CityGML [*1] をベースとする Plateau [*2] を中心とした GIS (Geographic Information System) 領域、建設業で用いられる BIM (Building Information Modeling) などが挙げられる。これら仕様間でのデータ編集性や互換性は現状議論が進んでおらず、産業領域や扱う空間・時間のスケールに応じて、空間記述仕様は個別に発展している。

[*1] https://www.ogc.org/standards/citygml
[*2] https://www.mlit.go.jp/plateau/

現実世界とサイバー空間の乖離

CPS の登場により、システムの全ての動作と作業プロセスをサイバー空間上で一元定義することが目指されてきた。しかし、物理的な異種システムや仮想的な抽象設計の複雑さを原因として、徐々にサイバー空間上のシステムが現実世界から乖離するという問題が発生する。現実世界から乖離したデジタルツインのシステムは、正確な環境状態の提示や予測が不可能となり、その意義から大きく逸脱することとなる。

静的な設計

多くのシステム記述仕様は、対象となる世界を固定し静的モデルとして取り扱う。第 3.1.1 章で述べるオントロジーや第 3.1.3 章で述べる建築情報記述仕様の BIM/IFC のデータは、その時間変化を記述することは出来ない。デジタルツインが対象とする世界のコンテキストは絶えず動的に変化しているため、このような静的モデルを前提とした記述仕様のみではその動的変化を表現することが出来ず、現実世界からの乖離を助長する。

データのサイロ化

デジタルツインアプリケーションの相互運用性向上には、物理ツインまたはそのデジタルモデルを表現する記述仕様のみならず、アプリケーションの実装工程で要求されるメタデータを適切に付与する必要がある。特に物理空間に存在するエンティティの属性情報は、センサなどの観測及びそれに基づく予測値によって決定されるため、観測値や状態値のデータ加工・編集・変換プロセスが設計者独自の記法または手続きによって構築された場合、閉じたアプリケーションとして実装されサイロ化が発生する。リソースの先験的な知識が欠如し、データ解析やアプリケーションの拡張が困難であるため、デジタルツインの相互運用性を低減する課題となり得る。

第 3 章

関連研究

本章では、デジタルツイン及びそのデータ資産の管理に向けた考察をおこなっている先行研究や既存技術について述べる。

3.1 セマンティック Web

データ資産やデータモデルの管理問題に対する手法として、セマンティック Web の活用が知られている。セマンティック Web とは、2001 年に T. Berners-Lee ら [18] によって「現在のウェブを拡張子、情報に明確な意味を持たせることで、コンピュータが人間とより協調して作業できるようにしたもの」と提唱された概念である。人間が読むためのテキスト情報中心のドキュメントベースが主流であった提唱当時の Web に対し、セマンティック Web はセマンティクス (=意味論) に基づいたメタ情報をデータに付与する。これにより、セマンティック Web はデータの Web として構築され、対人だけでなく対機械可読性を向上させる。

3.1.1 オントロジーとは

オントロジーとは、エージェントまたはアプリケーション間で合意された理解のために、データに付与するメタ情報記述に向けたドメイン知識の表現方法のことである。一般に、オントロジーはドメイン、用語の意味、およびそれらの間の関係性を記述仕様として機能し、OWL (Web Ontology Language) といった言語を用いて、記述論理を基盤とする共通語彙体系として形成される。オントロジーを構成する要素はオブジェクト指向プログラミングに準えることができ、「クラス (概念)」は現実世界に存在するエンティティのグループまたは類似の特性を持つ要素の集合を指し、「個体 (インスタンス、Individual)」はオントロジーに記述されたクラスのインスタンスを意味する。メソッドに類する概念として「プロパティ」があり、個体と個体、もしくは個体と値の関係を記述する。メタデータの記述に用いられるフォーマットとしては W3C [*1] 勧告された国際標準である RDF (Resource Description Language) がある。RDF のメタデータのモデルは、<主語、述語、目的語>の 3 つの要素でリソース (セマンティック Web にお

[*1] https://www.w3.org/

けるデータの単位) の関係性を表現し、これをトリプルと呼ぶ。例えば、RDF において「カラスの色は黒い」という情報を表現する場合、「カラス」を主語、「(〜という) 色を持つ」を述語、「黒」が目的語として記述される。3 要素の組みの集合体によって、全体として主語と目的語をノードとし述語をエッジとするグラフ構造が形成される。また、各リソースには URI (Uniform Resource Identifier) が与えられ、そのデータの所在が示される。例えば著名なオントロジーである foaf [*2]に定義される Person クラスは、`http://xmlns.com/foaf/0.1/Person` という URI を持つほか、`foaf` という名前空間を定義したうえで `foaf:Person` という表記が用いられることが多い。代表的な RDF 問い合わせ言語として SPARQL [*3]がある。このリンクトグラフデータをオープンデータとして公開したものが LOD (Linked Open Data) と呼ばれ、セマンティック Web 分野で開発・標準化されてきた技術による Web 上のデータセットである。

また、Tim Berners-Lee は LOD 構築の視点から Five Star Open Data という 5 つ星スキームを提案した [19]。これはオープンデータのための 5 段階評価の枠組みとして知られている。本指標によると、公開データにオープンライセンスを付与したものが、オープンデータの初期段階と見なされる。公開データのデータ形式が機械可読な場合、星 2 に該当し、それが独占所有権のないオープンな形式であるとき、星 3 に該当する。RDF、特に外部へのリンクを含む RDF データが、最高評価である星 4 及び星 5 と見なされ、データが相互に繋がりあうグローバルなデータ空間の一部として活用可能な LOD として評価される。

3.1.2 オントロジーとデジタルツイン

第 2.1 章で述べた通り、デジタルツインは現実世界のエンティティの活動を解釈し関連する概念・データ構造をアプリケーションに記述するシステムである。したがってエンティティの活動内容や種類に応じたドメイン知識の処理及び理解を容易にするオントロジーの活用は、異種データベースや相互運用性に関連する問題を抱えるデジタルツインにおける共有された概念の形式化かつ明示的な仕様として期待を集めている。Boje らは、デジタルツインの実行可能性は「データとそのセマンティクスを正しく表現し、データセット全体を知識処理に利用できるようにする能力に依存する」と述べている [20]。

Singh らの研究

Singh ら [21] は、デジタルツインのデータ管理の問題に着目し、独自のデジタルツインオントロジーモデルと方法論を提案した。提案モデルの概要は図 3.1 の通りである。モデルの全体像は、実空間のエンティティを示す物理層、分析に適した情報レポジトリを示すデータ層、ライフサイクルや行動・推論モデルを示すモデル層の 3 つの情報フロー層に分割され、各層にクラスが定義されている。エンティティの物理状態を表す Assets クラス、履歴情報や論理データを表す Knowledge Base クラス、抽象概念を表す Model クラスなど、各層におけるエンティティの状態を定義し、デジタルツインのアプリケーションやシステムでの情報伝達を表現する

[*2] `http://xmlns.com/foaf/0.1/`
[*3] `https://www.w3.org/TR/sparql11-overview/`

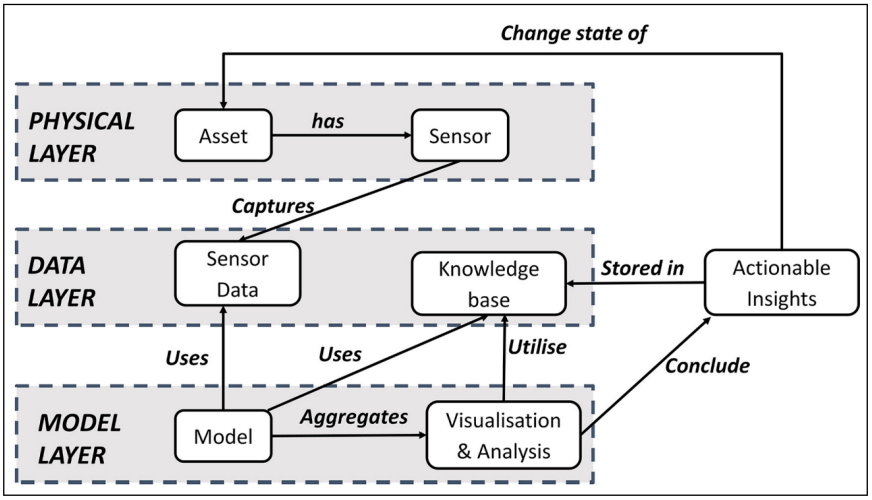


図 3.1: Singh らが提案するオントロジーの概要図 [21]

最小モデルとして構築した。

3.1.3　関連オントロジー

デジタルツインの登場以前より、データベース設計をおこなう産業界においてデータモデリングは広く取り組まれてきた。本章では、デジタルツイン以外の領域を対象としているが、デジタルツインに向けた統合が目指されている、もしくは期待されているオントロジーについて説明し、既存の空間記述仕様や現実世界の観測のデータモデリングについて整理するとともに、その課題点について述べる。

BIM/IFC, BOT

BIM (Building Information Modeling) は、建築物の物理的・機能的特性をデジタル表現するための代表的な構築手法である。建築物の 3 次元形状情報だけでなく、材料、部材の仕様や性能、さらに部屋名、面積などの属性情報を併せ持った建築情報モデルであり、BIM を使用して構築された建築士さんのライフサイクル全体に渡る情報管理は、ISO 19650 [22] として国際規格となっている。長らく 2D 設計に留まっていた建築分野において、設計・施工段階から保守、改修まで長期のライフサイクルに渡る建築エンジニアリングを推進する技術として導入が進められている [23]。また、BIM による建設データの共有やデータ交換形式として IFC (Industry Foundation Classes) があり、BIM データの構造化を担う基礎スキーマとして利用される。IFC は建築市産業界に関する標準化されたデジタル記述のオープンな国際規格となっており、buildingSMART International が策定している [24]。BIM/IFC の登場により、建設業界は設計や建設だけでなく、ファシリティマネジメント (Facility Management)、ビル運用管理、ビルエンジニアリングなど、建物のライフサイクル全体に対してデータモデリングの関心を広げるようになった。

BIM/IFC を踏襲しつつ、Web ベースのデータ交換や外部ドメインとの連携性向上のために策定されたオントロジーが、BOT (Building Topology Ontology) [25] である。主要なクラスとして、物理世界または仮想世界の 3 次元的な空間的広がりを表す bot:Zone クラス、建築物を構成する物理的な部品を表す bot:Element クラス、Zone 間や Element 間、または Zone と Element 間の関係性を定量化する概念として bot:Interface を持つ。BOT は建築物の構成要素の関係性記述を目的としていることから、IFC の純粋オントロジー変換である ifcOwl [26] よりも軽量なオントロジーとなっている。

SSN/SOSA

センサーオントロジーの代表例として、SSN/SOSA (Semantic Sensor Network / Sensor, Observation, Sample, and Actuator) オントロジーがある [27]。センサー及びアクチュエータのシステムや動作に関する記述が可能であるオントロジーであり、W3C 勧告及び OGC (Open Geospatial Consortium) 実装標準として公開されている。前身の SSNX ではデバイスのハードウェアレベルでの表現能力に留まっていたが、2017 年公開版では、センサによる観測、使用される手順、作用する対象の特性、サンプル及びサンプリングのプロセスが新たに定義されたことで、デバイスとしてのセンサの記述からソフトウェア的特性の記述可能な範囲が大幅に拡張された。一例として観測の記述として、動作を表す Observation クラスを中心に、観測の主体である Sensor クラスや結果を表す Result クラス、また被観測属性である ObservableProperty クラスとそれが属する FeatureOfInterest クラスなどが定義されている。Actuation、Sampling に関しても同様に多くのクラスが定義されており、デバイスの振る舞いを柔軟に記述することができる。

Web of Things

WoT (Web of Things) [28] は、Web 技術を用いて IoT プラットフォーム間の相互運用を実現するための標準である。物理的または仮想的な実体を表す Thing を中心概念とし、Thing が持つ相互作用を「アフォーダンス」として記述することで、デバイスの動作や振る舞いを定義可能としている。WoT は正確にはオントロジーではなく IoT のデータ交換のためのプロトコル及びインターフェイスであるが、Web のレベルにおける『モノ』の記述の標準化を目指し、Things Architecture [29] として IoT のモジュール構成を厳格に定義している。

Real Estate Core

REC (Real Estate Core) [30] は、不動産データ及びビルディングシステムの情報を分類及び標準化するオントロジーである。REC は、異なる業界標準間のギャップを埋めることを目的としており、部分的には既存の標準をベースに、建物の建築要素・制御と運用・IoT 技術・ビジネスプロセスの 4 領域の橋渡しとなるような設計が目指されている。BOT の Zone クラスと同様に 3 次元的広がりを表す Spaces 概念や、建物の構造体である Building Element 概念が定義されていることに加え、建物内部に配置されているモノを表す Asset 概念、データの生成や取り込みが可能な設備またはその能力を表す Point 概念、さらにビルディングシステム上で

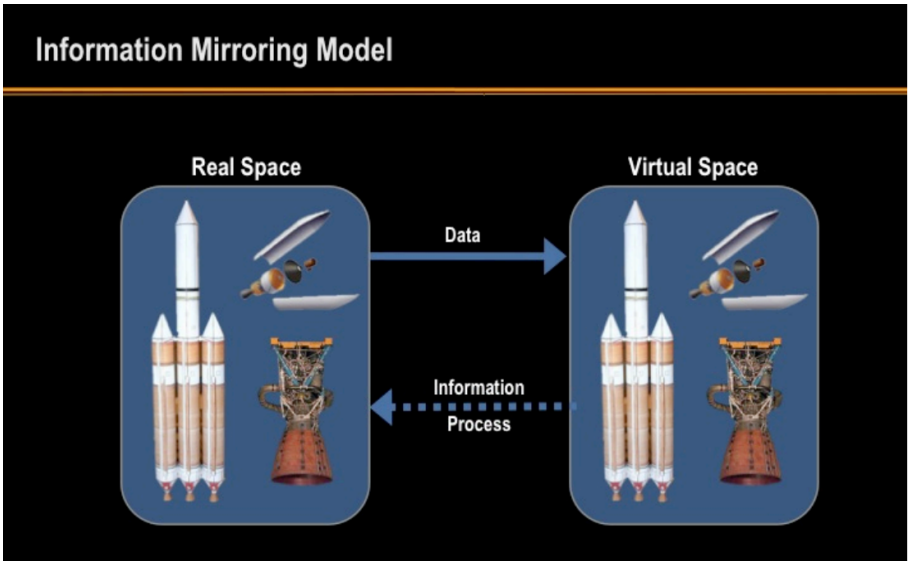


図 3.2: Grieves らが提唱するデジタルツインの 3 層モデル [8]

役割を持ち、活動をおこなう基本的なステークホルダーとして Agent 概念なども定義されている。

3.2　デジタルツインアーキテクチャ

本章では、デジタルツインを前提としたオントロジー設計に向け適切なモデルを再検討する目的で、デジタルツインアーキテクチャに関する先行研究をまとめる。

Grieves らは、製造業におけるデジタルツインの中で、基本的な 3 層モデルを提唱した [8](図 3.2)。図より物理オブジェクトが存在する実空間と仮想オブジェクトが存在する仮想空間、そしてそれらを結びつけるデータと情報の繋がりの 3 つの主要部分から構成されている。実空間の製品から測定器を用いて測定値を収集し、仮想空間のモデルに多くの動作特性を加えることで情報量を充実させる。また選択的ジオメトリ・特性・属性による軽量仮想モデルを用いることで、複雑なシステム群をその物理特性を含めてリアルタイムにシミュレーション可能だとしている。

Tao ら [6] は、2010 年以降に登場した学術論文の総合的なレビューをおこない、デジタルツイン研究の先端研究や主要な理論構成を体系化した論文の中で、デジタルツイン空間及びそのアプリケーションサービス構築のエコシステムを分析し、設計の理論構成として次の 4 段階に整理した。順に(1) 抽象化とシミュレーション、信頼性評価; (2) 前処理や最適化などを含めたデータクレンジングおよび統合; (3) 相互作用とシステム連携; (4) サービス化; としている。なかでも抽象化/モデリングの工程を重要視し、いくつかの学術論文を整理した結果から、従来の基本的な 3 層モデル (物理部、仮想部、接続部) に、データ部 (新しい知識体系を構築するための前提情報) やサービス部 (提供する課題解決能力) を加えた 5 次元モデルが完全なデジタルツインを構成すると述べている。従来と同じく物理部は仮想部構築の基礎として、仮想部は物理

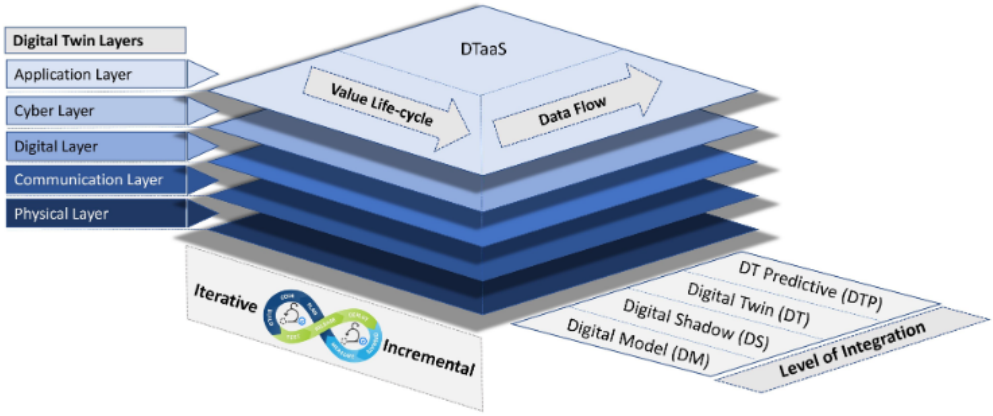


Fig. 5. A Digital Twin Reference Architecture Model in Industry 4.0.

図 3.3: Aheleroff らが提唱するデジタルツインの 5 層モデル [31]

部のシミュレーション、意思決定、制御のサポートと定めているほか、新しい知識を生み出す前提となるとしてデータ部をデジタルツインの中心に据え、デジタルツインの応用としてサービス部も主要素に置く。

Aheleroff らは、デジタルツイン対応アプリケーションの実現に向けた参照アーキテクチャとして DTaaS (Digital Twin as a Service) を提案する論文にて、5 層モデルを設計している [31](図 3.3)。図より、現実世界を表す物理層から順に、通信層、デジタル層、サイバー層、アプリケーション層としている。従来の 3 層モデルでは一括りにされていた仮想層を、現実空間のデジタルコピーとしてのデジタル層と、インターネット上でシミュレーションを利活用するためのサイバー層を分離し、一般ユーザのインターフェイスとしてアプリケーション層を追加した。

3.3 ソフトウェア設計論

デジタルツインは 2010 年以降に注目を集め始めた新しい概念であるが、センサまたはアプリケーションから集積された大量のデータをどのように扱うかという問題は、従来のデータ駆動設計が取り組んできた課題と同種である。本章では、データ駆動設計論の中から、提案手法のベースとなる FBS について述べる。

3.3.1 FBS (Function Behaviour Structure) フレームワーク

FBS フレームワークとは、1990 年に Gero らによって提唱された設計のプロセスをモデル化するための形式的表現である [32]。FBS では、設計対象のオブジェクトを「機能 (Function)」、「振る舞い (Behavior)」、「構造 (Structure)」の 3 つのオントロジーに分類して概念化している。「機能」は設計対象物の目的 (すなわちそれが何のためにあるのか) を記述し、「振る舞い」は設計対象物の構造から派生するまたはすると予想される属性 (すなわちそれが何をするか) を表し、「構造」は設計対象物の構成要素とその関係 (すなわちそれが何であるか) を表現する。ま

たこの 3 概念に基づき、設計に関する 8 つの基本プロセスを提案した。

3.3.2　situated FBS

Gero らは、先述の FBS を拡張した situated FBS を 2004 年に提唱した [33]。situated FBS は、FBS の原型で不足していた動的特性と設計者視点の概念を考慮したものであり、異なる環境間の再帰的相互関係として記述している。従来の 3 概念に加え、設計プロセスを「設計者と環境の間に存在する、作ることと見ることの相互作用」という再帰的なプロセスとして再定義することで、次の 3 世界モデルを提唱した。

- 外界 (External world)
  設計者外の表現で構成される世界。
- 解釈世界 (Interpreted world)
  設計者の内部で、感覚的な経験や知覚、概念によって構築される世界。External world との相互作用から構築され処理されたエージェントの知識が該当する。
- 期待世界 (Expected world)
  設計者の想像上の行動が生み出す世界。

またこれらの間のプロセスは以下の 3 つに分類される。

- Interpretation
  外界から解釈世界への関係性。外界で感知された変数を、解釈された世界を構成する感覚体験の解釈、知覚、概念に変換する。
- Focussing
  解釈世界から期待世界への関係性。解釈世界のある側面に焦点を当て、それを期待世界の目標かつ外界 で実行されれば目標達成となる状態を生み出すはずの行動を提案する。
- Action
  期待世界から外界への関係性。期待世界の目標に従って、外界に変化をもたらす。

設計対象の側面を表現する 3 概念をこれら 3 世界モデルで分類することにより、図 3.4 の示す通りオープンで動的な世界における設計エージェントの外界と自己の再帰的な相互作用プロセスが記述可能になった。

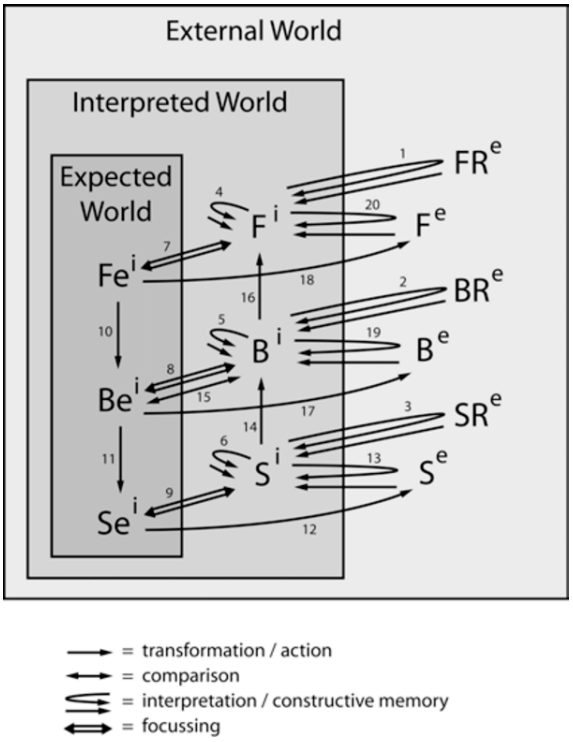


図 3.4: Gero らが提唱した situated FBS による設計プロセスの模式図 [34]

第 4 章

提案手法

本章では、デジタルツインのデータ資産を包括的に管理するためのオントロジーである Digital Twin Ontology (DTO) 及びデータ編集・加工プロセスを定式化するための Data Conversion Ontology (DCO) を提案する。

はじめに、満たすべき要件を整理し、前提とする物理環境を説明する。次に、提案手法のアプローチとなるデジタルツインアーキテクチャについて議論し、最後に各オントロジーの設計詳細を述べる。

4.1 要求事項

第 2.2 章で述べたデジタルツインにおけるデータ資産管理の課題を踏まえて、本研究において提案するデジタルツインオントロジーが満たすべき要件を以下に整理する。

セマンティックなデータ統合と外部参照性の向上

多面的な異質性を抱えたデジタルツインのデータ資産の相互運用性向上のため、複数のデータソースにまたがるデータの統合が要求される。具体的には、オントロジーを定義しデジタルツインの概念設計を形式的・明示的に表現したうえで、データ資産の知識表現や相互関係を記述する。またデータ資産の各リソースに URI を割り当てることで、外部からの検索とナビゲーションを保証する。

物理ツインと仮想モデルの接続

セマンティクスによって意味的統合されたデータ資産は、単なる現実世界の情報のコピーや、エンティティに紐づく個別のシステムにとどまってはならない。観測や予測に基づいて現実世界の属性情報を忠実に捉える物理ツインと、設計者によって期待される振る舞いを備えた仮想モデルの両者を備え、それらを高度に連携させるインターフェイスとしての情報レポジトリとしての役割を果たさなければならない。

仮想モデルの「振る舞い」の記述

仮想モデルの中心的役割は、それに対応する現実世界のエンティティがどのように存在するか、すなわち設計者視点から該当エンティティがどのような機能を提供することが

表 4.1: 課題と要求事項と提案オントロジーの対応

課題	要求事項	提案
データモデルの Heterogeniety	セマンティックなデータ統合と外部参照性の向上	DTO & DCO & 実装
参照アーキテクチャの欠如		
空間記述仕様の個別発展		
現実世界とサイバー空間の乖離	物理ツインと仮想モデルの接続	DTO
	仮想モデルの「振る舞い」の記述	
静的な設計	エンティティに関する動的情報の記述	
データのサイロ化	データ変換プロセスの記述	DCO

期待されているかを表現することである。このようにエンティティの仮想モデルに期待される機能または「振る舞い」の記述が可能であることが求められる。

エンティティに関する動的情報の記述

従来オントロジーやセマンティックデータは静的モデルを対象としており、対象のある一時点における状態もしくは時間概念が欠落した記述にとどまっていた。デジタルツインが対象とする動的なコンテキストを持つ現実世界の記述に向け、動的情報、特に時間変化が伴う情報の記述が不可欠である。

データ変換プロセスの記述

物理ツインまたは仮想モデルを表現するデータは、本研究では、デジタルツインアプリケーションの相互運用性向上に向け、アプリケーションの構築プロセスの形式化及びオープン化を目指す。

また、これらの要求事項と課題、そして DTO と DCO が満たすと考えられる対応表を表 4.1 に整理する。

4.2 想定する室内環境

本研究で対象とする環境は、屋内空間、特にビルのワンフロアや部屋のスケールとし、空間内部には、照明や空調などの設備や、什器をはじめとする静的な物理資産、人やロボットなど動的物体の存在を仮定している。また、これらのエンティティを観測するためのセンサや、インターネット経由で外部から操作・制御可能なアクチュエータ、IoT デバイスの存在を前提とする。

4.3 アーキテクチャ設計

はじめに、オントロジー、すなわち物理空間及び仮想空間の両者にまたがるエンティティの状態・振る舞いを汎用的に記述可能な形式的仕様を定義することで、データ資産やそれに付

与されるメタデータを、論理的/概念的なスキーマに準拠したリンクトデータとして管理できることが期待される。また RDF 形式で LOD として公開することで、外部からの検索とナビゲーションが保証される。したがって「セマンティックなデータ統合と外部参照性の向上」という要求が達成される。

続いて、提案するオントロジーのアーキテクチャについて議論する。第 2.2 章で述べた通り、現在のシステムにおいて「現実世界とサイバー空間の乖離」が問題となっている。デジタルツインは現実世界とサイバー空間のデータをいかに融合させるかが重要とされる一方で、実用面では、物理的な製品と仮想的な製品は何らかの形で互いに分離された状態で構築、分析、更新されてきた。第 3.2 章で紹介したように、デジタルツインの参照アーキテクチャについて様々な研究がおこなわれてきたが、そのいずれもエンティティの仮想モデルは単一のレイヤー内に配置されている。例えば、Aheleroff らの 5 層モデルは、最も基本形の 3 層モデルを拡張したものであるが、現実世界に対応するレイヤーはデジタル層のみであり、サイバー層以降はシステムの開発や利用から発する要件から分割されたものである。また、デジタルツインアプリケーションへの援用が期待されているオントロジーについても、現実世界におけるエンティティの在り方と、設計者が期待する機能を有する仮想モデルが同一概念として記述されている。センサーデータ、モデルデータ、知識など複数のソースからのデータを統合することが要求されるデジタルツインにおいて、「あるデータ (またはリソース) が物理的実体を表すものなのか、仮想的なモデルの結果を表すものなのか」といった判別はコストが高く、システムの整合性と同期性の障壁となる。

そこで、本研究では、デジタルツインの設計対象に対する構造分析に、Gero らの situated FBS を応用したモデルを提案する。概要を図 4.1 に示す。設計プロセスを『設計者と環境の間に存在する、作ることと見ることの相互作用』とみなす situated FBS の立場に倣い、従来アーキテクチャでは考慮されていなかった設計者視点の概念を導入する。すなわち、本提案では物理世界と仮想世界の二元論ではなく、物理世界 (=外界) と、設計者が有する期待世界 (=期待世界) と、データ資産として表出する仮想世界 (=解釈世界) の三元論に立つ。この考えのもと提案手法のアーキテクチャでは、任意のエンティティは、その物理的な実体を高忠実度で表現する物理ツインと、設計者がエンティティに期待する状態や機能、振る舞いを有する対応物である仮想モデルの二面性を持つとして構築する。したがってデジタルツインにおけるデータモデリングは、観測者による現実世界に存在するエンティティの定式化だけでなく、設計者の期待世界で形成されるエンティティへの解釈・知覚・概念の変換プロセスも合わせて構成される。これにより現実世界と仮想世界の間に生じていた外乱や不確実性が、同一仮想世界上で重ね合わされることで、対応すべき差異が浮き彫りになり、物理世界と期待世界の間を接続するインターフェイスとしてのデジタルツインが構成できることが期待される。

以上のデジタルツインアーキテクチャをベースに、物理ツインと仮想モデルをセマンティックに統合するためのオントロジーとして Digital Twin Ontology を、各世界の連携に必要なデータ編集・加工プロセスを定式化するためのオントロジーとして、Data Conversion Ontology を提案する。

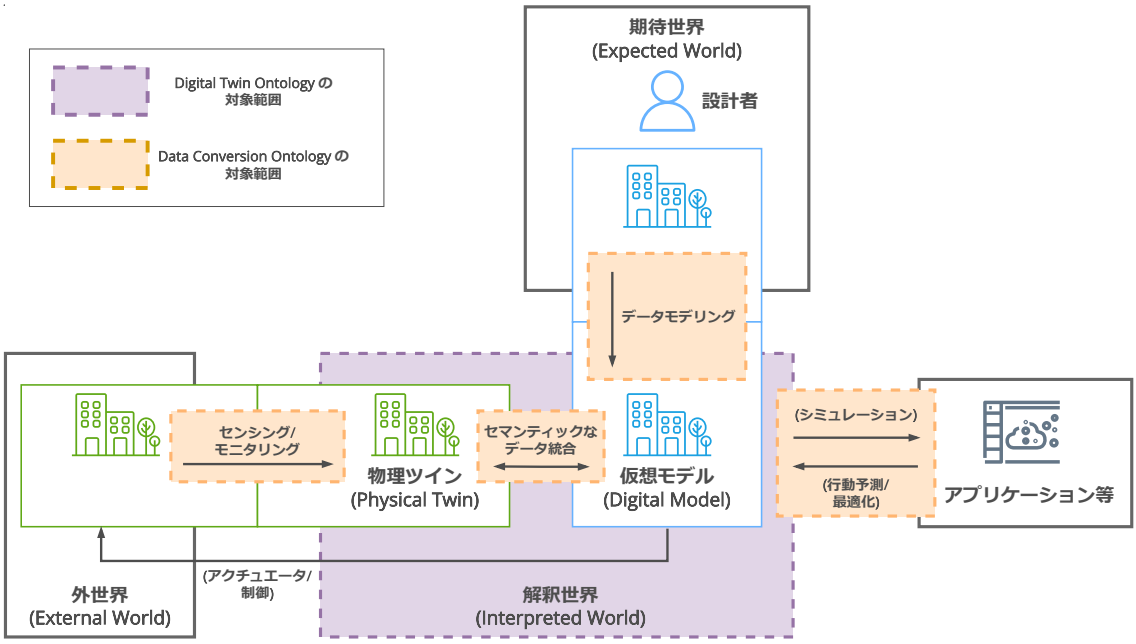


図 4.1: 提案手法におけるデジタルツインアーキテクチャ

4.4 Digital Twin Ontology

第 4.1 章で述べた要求事項をもとに、動的な物理ツインと仮想モデルを汎用的かつ包括的に記述する Digital Twin Ontology を設計した。

はじめに構造概要を述べた後、基本クラスを中心にオントロジーの構造を説明する。図 4.2 は全体像を示す。図中の各ノードはオントロジーのクラスまたはリテラルを表し、太字が名称、ノード内下部がクラスまたはリテラルの概略を記す。ノード間を結ぶ矢印の傍らに書かれているのは Data Property(データ型プロパティ) または Object Property(オブジェクトプロパティ) を示す。根本側のノード、ノードから出発する矢印、矢印が指す先のノードの 3 つの組が、RDF のトリプルを表す。各ノード/エッジにおける語彙は、原則として提案オントロジーのベース URI である `https://sdm.hongo.wide.ad.jp/~ino/dto/` からの相対 URI で示している。外部オントロジーを利用する一部のクラスのベース URI や接頭辞の情報は、RDF データファイルを参照されたい。

4.4.1 構造概要

提案するデジタルツインオントロジーの構造概要を図 2 に示す。提案オントロジーは OWL (Web Ontology Language) 形式で設計し、36 個の概念 (クラス) と 39 個の属性 (プロパティ) から構成される。第 4.3 章で述べた方針に基づき、現実世界に存在するエンティティを Entity、物理ツインを Hyle、仮想モデルを Eidos と呼ぶ。また、エンティティの属性値の時系列変化を表す Instance、設計者が期待するエンティティの振る舞いを表現する Affordance を

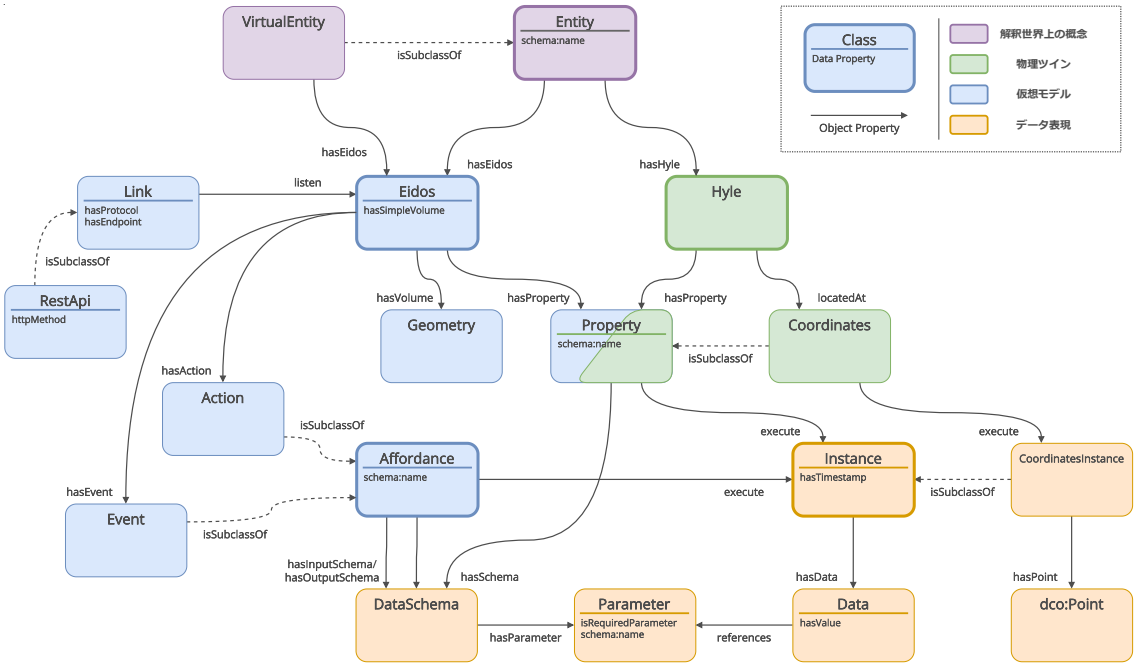


図 4.2: Digital Twin Ontology の概要図

定義した。これら 5 つのクラスを基本クラスと定め、以下の項で詳細を述べる。

4.4.2　Entity / Hyle / Eidos クラス

Entity クラスは、現実世界に存在する任意のエンティティを表現するクラスである。それ自体に意味を持たないが、個別具体のエンティティの同位概念として記述されるクラスであり、その物理的側面すなわち物理ツインとして Hyle クラス、設計者により解釈及び抽象化された仮想モデルとして Eidos クラスと紐付けられる。Entity クラスのサブクラスとして VirtualEntity クラスも定義した。これは物理的には存在しない (とみなされる) が設計者によってその存在を期待されるエンティティを表現するためのクラスであり、その個体は Hyle を持たず Eidos のみと結び付けられる。例として、ある機能的要件で定められた空間エリアや、1 つのユニットとして扱われるグループ化された集合体などが、VirtualEntity の個体として挙げられる。なお、本オントロジーでは「物理的な存在」の形式は限定しないものとし、設計者または利用者の解釈によって柔軟な変更を認めるものとする。

エンティティの物理ツインとして、Hyle クラスを定義した。Hyle クラスは FBS で Interpreted world に該当する概念として設計している。すなわち、観測者による感覚体験を通して現実世界における存在をエンティティ単位で境界づけ、デジタル世界へ忠実に投影したものとして存在する。より具体的には、Hyle の実現にデジタルツインの設計者による期待は介入せず、エンティティの物理的側面を記述することを目的として構成・運用される。

一方、エンティティの仮想モデルとして、Eidos クラスを定義した。Eidos クラスは、FBS における Expected world に該当する概念として設計している。すなわち、外界 (または解釈世

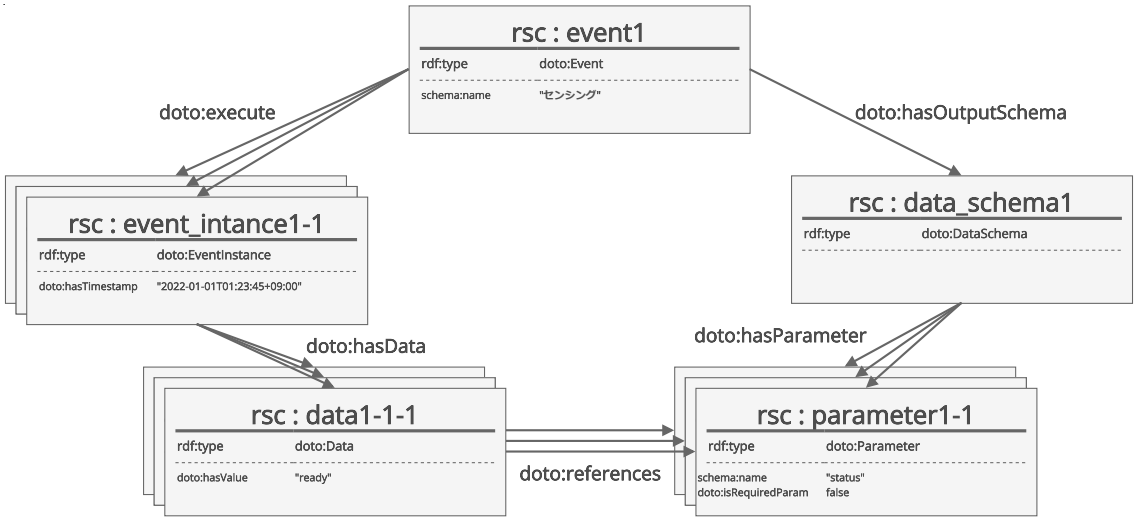


図 4.3: Instance とその関連クラスを用いてリスト 5.1 をリンクトグラフデータ化した模式図

界) に存在するエンティティに対して、設計者が期待する抽象化概念をデジタル世界上に投影したものとして存在する。より具体的には、Eidos はデジタルツインのアプリケーションやコンテンツでの利用が想定される制御可能なエンティティの仮想モデルを記述することを目的として構成・運用される。

また、Hyle と Eidos は属性情報を付与するための概念として定義された Property クラスと関連付けられる。Property は後述する Instance クラスを付随することで、時系列情報と結びついた属性値の表現をおこなう。なお、先述した Hyle と Eidos の設計意義に基づき、Hyle と関連付けられる Property は観測/測量から導き出される物理的属性情報を記述し、Eidos に関連付けられる Property は設計者が期待または意図した仮想的属性情報を記述する。

4.4.3 Instance クラス

Instance クラスは、Property の離散値や単一の Affordance を記述するための概念である。時系列情報を紐づけるため、時間情報として timestamp を Data Property として持つ。属性情報の表現のために Data クラスを定義したほか、Thing の Property の実数値や Action の入力値、センサの観測値のデータスキーマを記述するための DataSchema クラスや各パラメータ表現の Parameter クラスを定義した。例としてリスト 5.1 を提案オントロジーで表すと図 4.3 のように記述できる。また座標情報を記述するため、Instance のサブクラスとして CoordinatesInstance クラスを定義した。直交座標系、極座標系、円筒座標系など、空間位置の指定はその座標系により数多くの表現形式を取り得ることから、リテラル値を持つ Data クラスではなく Point クラスと紐づくよう設計した。

4.4.4　Affordance クラス

Affordance クラスは、設計者が意図した動作や振る舞いを表現する概念である。「アフォーダンス」理論は元来アメリカの心理学者ジェームズ・ギブソンが提唱したものであり、エージェントが環境との相互作用によって知覚できる行為から与えられる情報や特性を意味する。提案オントロジーは、Web of Things で採用されるアフォーダンス [35] を改変し、Property は基本クラスとして独立させ、Action クラスおよび Event クラスを Affordance クラスのサブクラスとして定義した。Action クラスは、Eidos が持つ機能もしくは特定の動作をエンティティに指示/実行させる操作を記述し、Event クラスは Eidos が自律的に実行する動作を記述する。なお Affordance は期待世界における動作の表出であるため、Hyle ではなく Eidos と関連付けられる。

4.5　Data Conversion Ontology

第 4.3 章のアーキテクチャを元に、データ変換を表す Data Conversion Ontology (DCO) を提案する。なお本オントロジーは DTO とは異なる名前空間 [*1] で定義している。

4.5.1　データ変換プロセスの定式化

第 4.3 章のアーキテクチャを踏まえると、提案オントロジーを用いたデータセット制作は、以下のような手順でおこなわれることが予想される。

1. 実空間のエンティティのモデリング
2. センシングによる環境・属性情報の計測
3. データセットの分析・編集・加工
4. セマンティックに基づいたデータの整理・突合・保存

前章の考察により、提案手法は手順 4 に有効であることが期待される。一方でアプリケーション構築の実運用上、前処理として建物のモデルデータや環境のセンシングデータを編集・加工し、RDF リソースとして変換する工程が必要となる。先行研究ではこの工程は考慮されていないが、第 2.2 章で述べた通り、本工程で発生するメタデータの管理方法が確立されていないことでサイロ化が助長され、物理空間の不確実性への対応が困難になる。

前提として、本研究で取り扱うデータ変換の課題を「単一のデバイスが出力した観測値/予測値は、そのまま観測対象の属性値として扱われるとは限らない」点に定める。例としてセンサオントロジーである SSN/SOSA は、「ある対象の属性を観測する『観測』概念が結果を出力する」というデータを<sosa:Observation> <sosa:observedProperty> <sosa:ObservableProperty> と<sosa:Observation>

[*1] https://sdm.hongo.wide.ad.jp/~ino/dco

<sosa:hasResult> <sosa:Result>という 2 つのトリプルで表現する。しかし sosa:ObservableProperty と sosa:Result の関係は定義されていないことに加え、複数の sosa:Observation が単一の sosa:ObservableProperty と関連付けられることも想定される。このように、あるエンティティの属性情報が複数のセンサーによって観測される場合、その物理量の予測値を表現するには単一もしくは複数の観測値を参照したデータ変換が不可欠である。以上より、要件を下記の 3 点に整理する。

- 観測値・物理量の変換・編集・加工をおこなう
- 汎用的な"系"を表現する概念を備え、標準化/カスタム双方のデータ型を表現する
- 変換処理をオープン化し、アプリケーションやデータセットの流用性を向上させる

そこで本研究では、オントロジー及びそのグラフデータを有限オートマトンの状態遷移図とみなすことで、データ変換・編集・加工プロセスの形式化を解決するアプローチを提案する。はじめに、データ変換プロセスの構成要素を整理する。本研究で対象とするデータ変換は、ある"系"から別の"系"への変換とする。例えば、「cm 単位」から「m 単位」への変換、「あるセンサから出力される生の観測値」から「0〜1 の間に正規化された物理量」への変換、さらには「予めグリッドで分割された部屋のグリッド ID」から「部屋の端点を原点とする xy 座標」への変換などが考えられる (この前後の物理量が前提としている部分空間を「系」と呼ぶ)。この時、データ変換は「系 A の元である値 a を入力とし、系 B の元である b を出力とするような関数 f を持つ変換 T」というように定義できる。続いて、オートマトンについて概略を述べる。なお、本提案ではこれらの抽象モデルを採用したものにとどまるため、計算理論に基づく厳密性を保証しないことに注意されたい。有限オートマトンとは、有限個の状態を持ち、入力とその時点の状態に応じて次の状態に遷移する機械を指し、特に有限機械の状態と入力との組によって次の状態と出力が決まる機械は順序機械と呼ばれる。ここで、有限状態集合を Q、入力の有限集合を Σ、出力の有限集合を Δ、状態遷移関数を $\delta : Q \times \Sigma \to Q$、出力関数を $\lambda : Q \times \Sigma \to \Delta$、そして初期状態を $q_0 \in Q$ と表記すると、Mealy 型順序機械 M は $M = (Q, \Sigma, \delta, \Delta, \lambda, q_0)$ と表される。これを前述のデータ変換の定義に照らし合わせると、入出力の系が Q、その元となる入出力データが Σ と Δ、変換の関数が λ となるような状態遷移関数 δ として対応付けられることが分かる。このことから、データ変換を表すオントロジーが、Mealy 型順序機械の状態遷移図を模した形式で表現できることが期待できる。

4.5.2 構造概要

DCO の概要を図 4.4 に示す。単位やデータ型といった"系"を System クラス、データ変換 (状態遷移関数) のモデルを Conversion クラス、変換プロセスで実行される個々の処理 (出力関数) を Transaction クラスと定義した。Conversion クラスは Object Property として sourceSystem と targetSystem を持ち、変換前の系と変換後の系を指定する。また executes プロパティによって、系の変換で実行される Transaction を指定する。

本提案では、出力関数をオントロジーとして表現するために、Transaction クラスに RPC (Re-

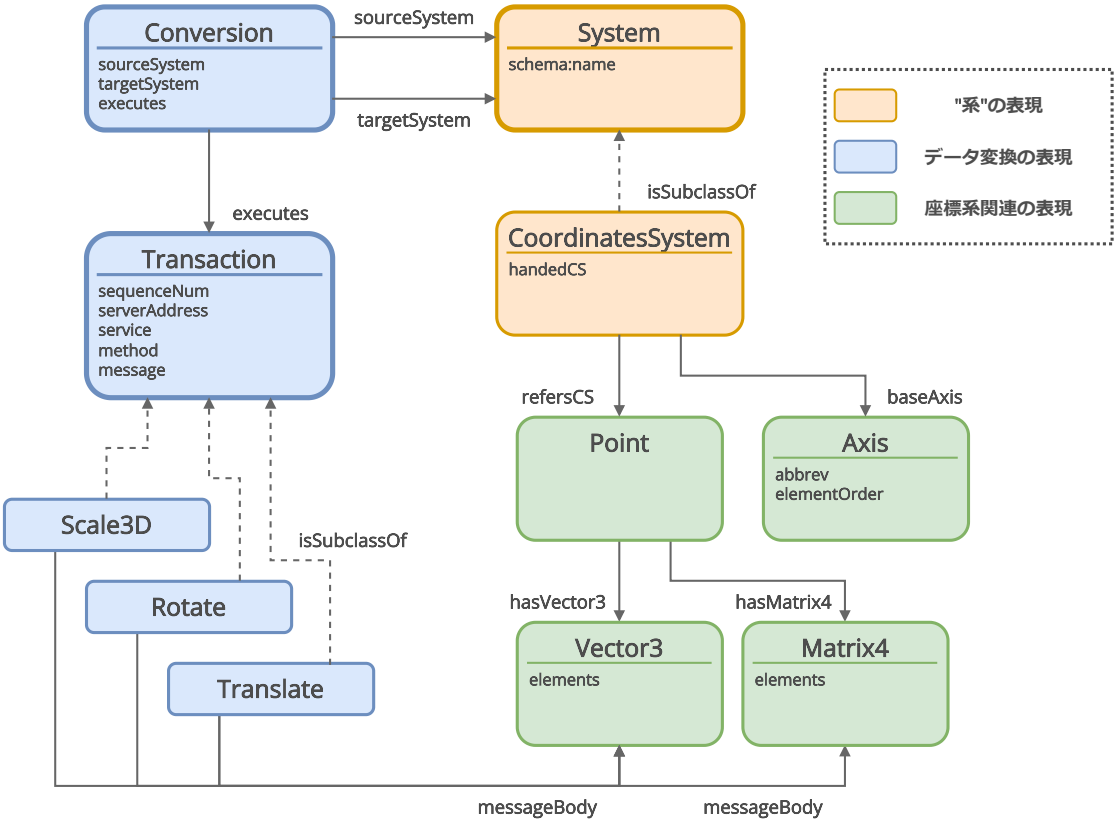


図 4.4: データ変換プロセスを定式化する Data Conversion Ontology の概要

mote Procedure Call) を活用した。RPC は遠隔手続き呼び出しとも呼ばれ、サーバ上で実装されている関数や手続きをクライアントからの呼び出しに応じて実行する技術である。RPC を前提とすることで、一般に関数そのものを記述することが困難である LOD に対し、データ変換の手続きに必要な関数を明示可能なオントロジーを設計することでこれを解決する。Transaction クラスは Data Property として、関数の適用順序を指定する sequenceNum プロパティ、RPC サーバを指定する serverAddress プロパティ、RPC のサービス及びメソッドを指定する service プロパティと method プロパティ、更にメソッドの引数として message プロパティを持つ。なお gRPC [*2] と呼ばれる Google 製 RPC を用いることを想定し、より親和性の高い語彙も定義した。

また、本オントロジーの利用性を向上させるため、直交直線座標系における座標変換を表現するクラスも定義した。はじめに、System のサブクラスとして座標系を表現する CoordinatesSystem クラスを定義した。CoordinatesSystem クラスは、軸を表す Axis クラスと関連付けるための baseAxis プロパティ、単位系を表す Units of measurement ontology [*3] の length_unit クラスと関連付ける unit プロパティ、右手/左手を指定する handedCS プロパティ

[*2] https://grpc.io/
[*3] https://obofoundry.org/ontology/uo.html

を持つ。続いて、CoordinatesSystem を系とする座標点として Point クラスを定義した。Point クラスは refersCS プロパティにより、その系を関連づける。座標点を表す 3 次元ベクトルまたは 4×4 サイズの変換行列は Vector3 クラスや Matrix4 クラスで定義され、それらに element プロパティで結びつくカンマ区切りの文字列を成分表示とみなす。最後に、座標変換で基本的なアフィン変換をモデル化した。拡大縮小を表す Scale3D、回転を表す Rotate、並進移動を表す Translate クラスを、それぞれ Transaction のサブクラスとして定義した。

4.6 Alignments

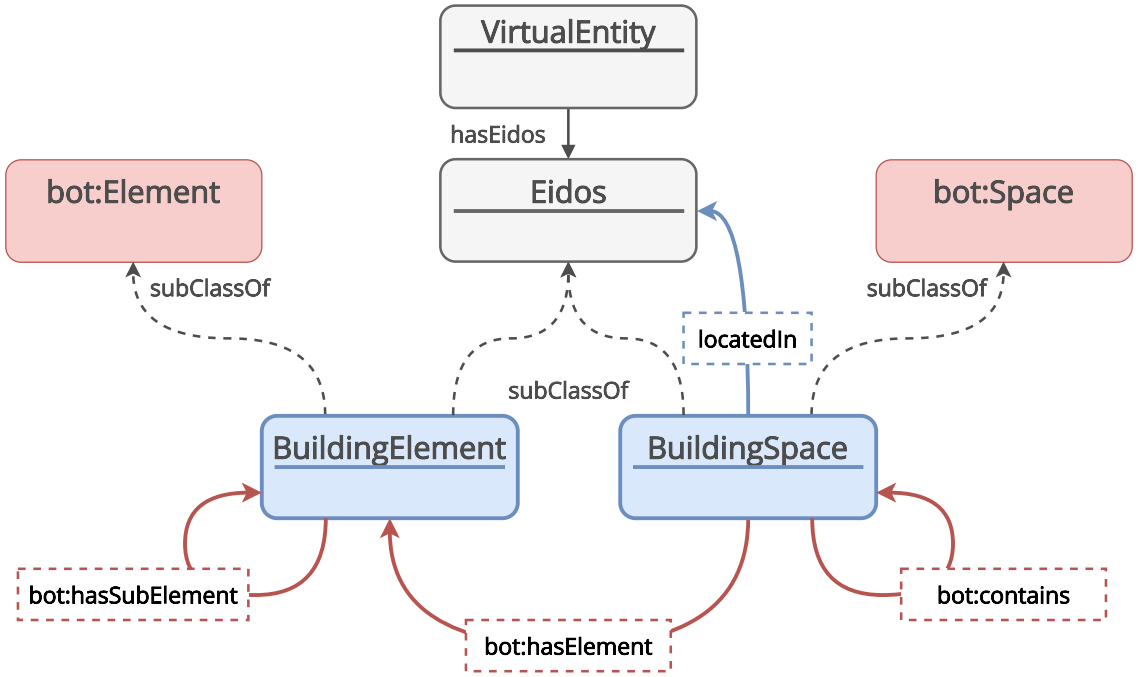
理想的な世界では、1 つの問題領域に対してそれを完全に記述し、全ての要求を満たすことが出来る 1 つのオントロジーが存在する。しかし、現実の世界では、人間の知的活動の多様化に合わせて、異なる解釈や判断に基づく複数のオントロジーが存在しており、その多くは部分的に役割が重複している。そこで、本研究では、他の主要なオントロジーに委譲または併用可能な機能を検討し、Alignments という形でまとめる。

4.6.1 BOT との連携

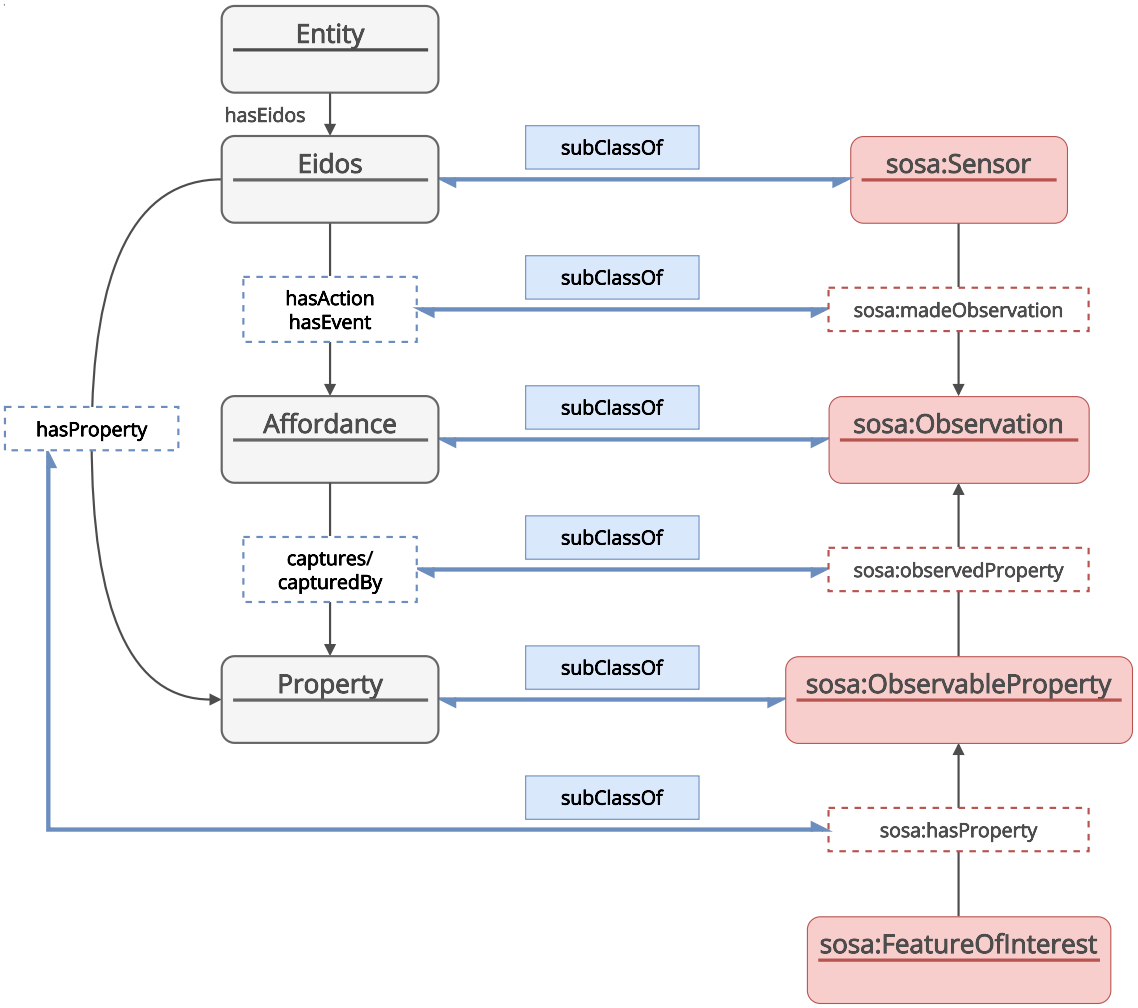
建築情報のオントロジーである BOT との連携を述べる (図 4.5a)。DTO では、建築要素を表す dto:BuildingElement、それにより境界付けられる dto:BuildingSpace を定義した。これらは dto:Eidos のサブクラスかつ bot:Element または bot:Space のサブクラスとして定義されている。したがって、これらは dto:Eidos と関連するクラスによる記述が可能であることに加え、bot:Element または bot:Space と関連するクラスを付随させることが可能である。

4.6.2 SSN/SOSA との連携

センサを始めとしたデバイスのオントロジーである SSN/SOSA との連携を述べる (図 4.5b)。SSN/SOSA では、センサを表す sosa:Sensor から、sosa:madeObservation プロパティを通して観測を表す sosa:Observation クラスに紐づく。これは DTO では dto:Eidos が dto:hasAction/hasEvent プロパティを通して dto:Affordance クラスと関連付けられる構造と等価である。さらに sosa:Observation が観測するプロパティ sosa:ObservableProperty と sosa:observedProperty も、dto:Property と dto:captures/capturedBy に対応する。両者のクラスから継承した個体を作成することにより、両者のオントロジーを踏襲しつつ、より拡張された記述が可能となる。



(a) DTO と BOT の連携



(b) DTO と SSN/SOSA の連携

第 5 章

実装

オントロジーの活用方法の検討や、アプリケーション制作者の視点から構造を考察する目的で、リンクトグラフデータの構築とビューアアプリケーションの実装をおこなった。

以降の説明では、第 2.1 章で述べた RDF の記法を用いる。すなわち提案オントロジーのクラスまたはプロパティを `dto:SomeClass`、作成したグラフデータのリソースを `rsc:SomeResource` のように表記する。なお実際の URI はインデックス等が付与されているが、以降の説明では適宜省略する。

5.1 実証フィールドの概要

本実装に先立ち、第 4.2 章で述べたビル管理システムの実証環境として、オフィスを模した実験エリアを実証フィールドに選定し、データセットの収集をおこなった。

本データは、2021 年 4 月に NTT コミュニケーションズが開設した「CROSS LAB for Smart City [*1]」(以下実験エリア) で取得したものの中から、同年 10 月 14 日から 12 月 27 日までのデータを用いる。図 5.1 に示す実験エリアは床面積約 300 m^2 であり、予め建物構造が IFC (Industory Foundation Classes) 形式でデータ化されている。照明器具や電気系統などの設備情報、椅子やキャビネットといった什器情報も取得し、形状構造に加え、寸法や材質、製品識別情報など、各々数 KB から数 MB の IFC ファイルに分割された。加えて、一部の設備 (空調、照明器具等) はネットワーク経由で動作を制御可能なデバイスが付帯されており、接続情報や API (Application Programming Interface)、通信プロトコルなどのデータも存在する。

物理エンティティの代表例として、複数センサを搭載した 自律走行ロボット (AGV、Automatic Guided Vehicle) を巡回させた。リスト 5.1 の通り、自律走行ロボットはその主要なデータとして座標、向き、バッテリー残量、滞在エリアを取得し、100B/s〜 200B/s のログを出力する。また本ロボットはネットワーク経由で移動指示可能な API が実装されており、予め定義された屋内エリアへ自律的に移動することが可能である。

[*1] `https://www.ntt.com/business/dx/smart/city/`



図 5.1: デジタルツインの検証に用いた実験エリア

リスト 5.1: AGV が出力するログの一例

```
2022-01-01T01:23:45,456 - F_STATUS - INFO - Received:
b'{
    "status": "ready",
    "command_status": "ready",
    "battery": 65,
    "x": 48.5, "y": 28.2,
    "direction": 115.3,
    "data_datetime": "2022-01-01T01:23:45+09:00"
}' @ /rb/v1/robot/frutera1/status
```

以上のデータ群を表 5.1 に整理した。これを元に、以降の章では屋内におけるデジタルツインとして次のシナリオを検討する。

- 床面積約 300 m^2 のオフィスを模した屋内空間を対象エリアとする。対象エリアを構成する建築要素がデータ化されており、その幾何形状情報や部材情報、製品情報などを保有している。
- エリア内では、自己位置推定可能なセンサを搭載した自律走行車が移動している。自律走行車は数秒毎に時刻と座標のテレメトリーログを発行する。座標情報に加え、複数の属性情報 (状態や向き、バッテリー残量など) を持つ。
- 自律走行車は、ネットワーク経由で事前に設定したポイントへ移動指示可能な「移動制御」の機能を有する。移動指示のための API が用意されている他、入力値は特定のデータフォーマットが定義されている。
- 実験中、数回の移動制御をおこなった。事前に定義された指定ポイントへ想定通りの移動制御が成功したアクションもあれば、システムトラブルにより失敗となったアクションも存在する。

項目	種別	形式
建物構造	ジオメトリ	BIM/IFC
空間定義	ジオメトリ	テキスト
什器	ジオメトリ	BIM/IFC
自律走行ロボット	物理エンティティ	JSON
空調/照明設備	物理エンティティ	テキスト

表 5.1: 実験エリアから取得可能なデータセット一覧

5.2 Digital Twin Ontology に基づくリンクトグラフデータ構築

本章では、提案オントロジーに準拠したリンクトグラフデータの構築について述べる。第 5.1 章で述べた通り、本シナリオにおけるデータセットは大きく建物情報と、動的エンティティを代表する自律走行車情報に二分される。

はじめに、建物情報について述べる。一般に、IFC ファイルは特定エリアの情報を単一ファイルにまとめており、当該実験エリアも 22.5 MB のテキストファイルとして保存されている。オフィスを模した実験エリアは、建築・設備・什器要素を含めた計 273 個の建物部位部材要素から成り、各要素には座標情報や幾何形状情報、寸法、材質、その他製品識別情報などが付与されている。ここで、IFC の問題点として情報の閲覧にかかるコストが高い点が挙げられる。独自のフォーマットで記述されているだけでなく、特定の情報取得のために複数のオブジェクトを走査・結合する必要があることから、一般に専用のビューアアプリケーションにインポートして利用される。これを回避するため、本実装では予め IfcOpenShell と呼ばれる OSS (Open Source Software) のうち、IfcOpenShell-python [*2] 並びに IfcConvert [*3] を利用し、個別建物要素の分割及び幾何情報の抽出をおこなった。図 5.2 に示す通り、ある建物 (`IfcBuilding`) の各フロア (`IfcStorey`) を順に走査し、階層構造を表すトポロジと建築要素を抽出する。各建築要素はその幾何形状情報をソリッドモデルとして持つが、本実装では簡単のため Ifcconvert を用いて OBJ ファイルに変換した。

リンクトグラフデータとして構築したものの一例が図 5.3 である。IFC から抽出したカスタムプロパティセット (psets) 及び座標情報をそれぞれ Property 個体、Coordinates 個体に変換し、各建物要素の Entity の Hyle 個体と Eidos に関連づけた。座標情報は正確には Eidos 個体に紐付けられるべきだが、建物要素の位置は不変なものであるという前提を置き、Hyle の属性とした。なお、本提案では実装を省略したが、dto:BuildingElement は dto:Eidos かつ Building Topology Ontology の element クラスのサブクラスとして定義されているため、BOT の表現力の範囲内に限られるが、IFC で定義されている各建物要素間の階層構造の記述についても拡張

[*2] `http://ifcopenshell.org/python`
[*3] `http://ifcopenshell.org/ifcconvert`

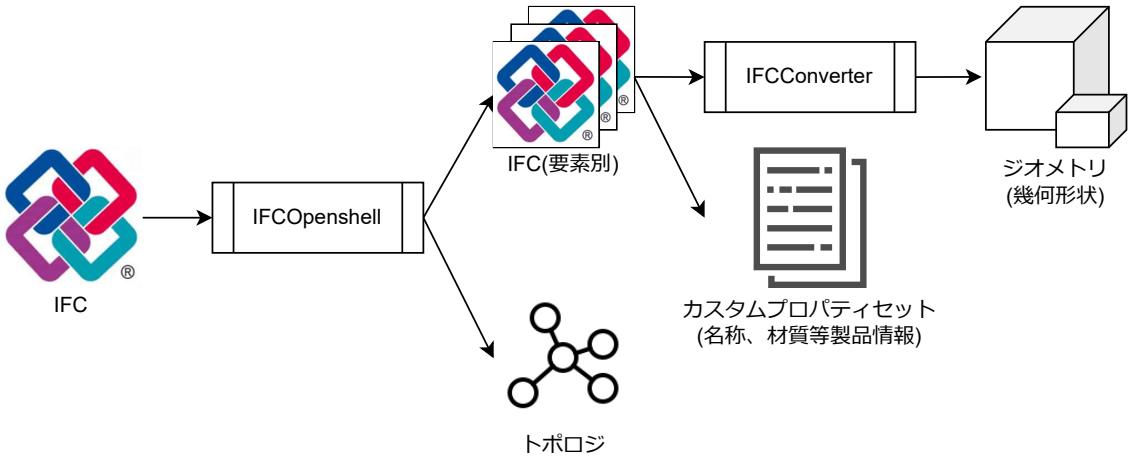


図 5.2: 単一の IFC ファイルから個別の建築要素の情報を抽出した実装フロー

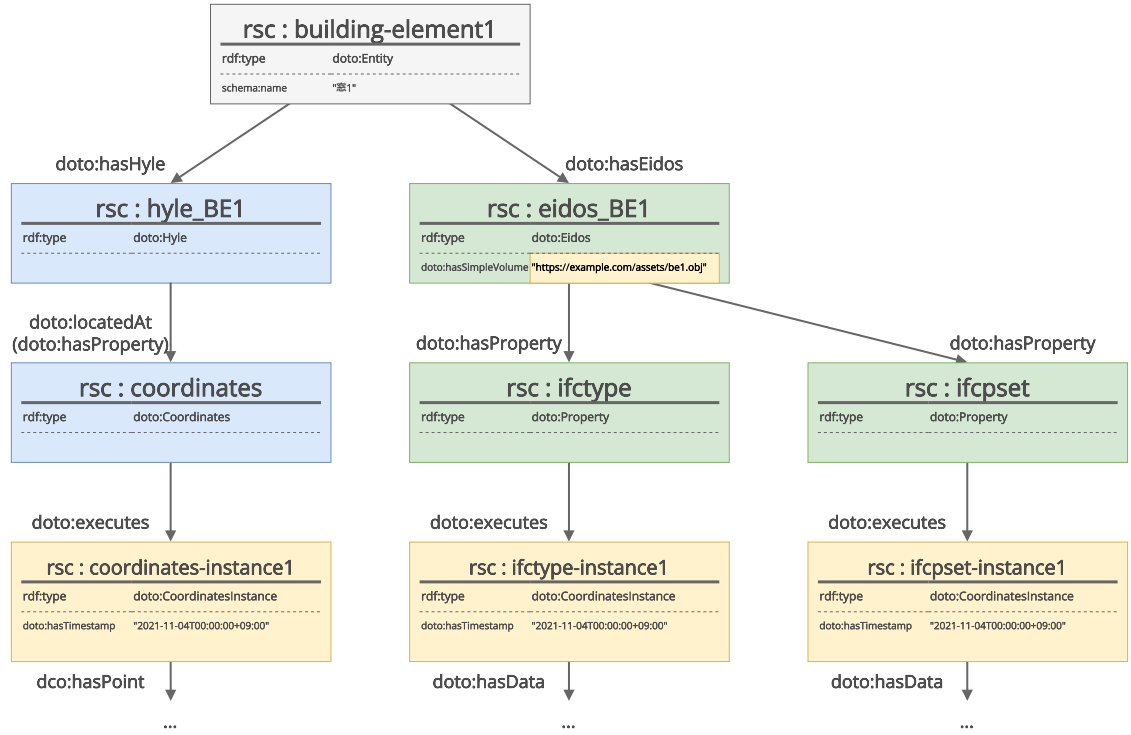


図 5.3: 建物要素を提案オントロジーに準拠したリンクトグラフとして構築した例

可能である。

続いて、自律走行車の情報について述べる。自律走行車を表現する dto:Entity 個体の rsc:agv、その物理的側面である dto:Hyle 個体の rsc:hyle_agv 及びデジタルモデルである dto:Eidos 個体の rsc:eidos_agv を基本個体として作成したリンクトグラフが図 5.4 である。自律走行車が搭載するセンサー (dto:Entity 個体の rsc:agv-sensor で表される) の観測 (dto:captures) により捕捉される属性情報として Property 個体の座標 (rsc:coordinates)、状態 (rsc:status)、方

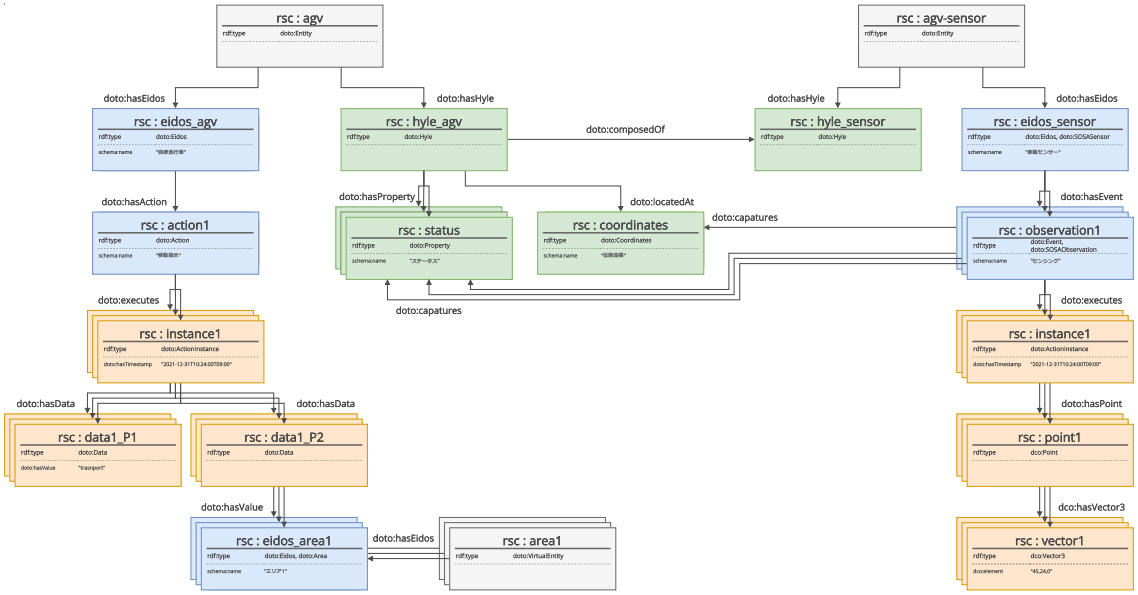


図 5.4: 自律走行車 (AGV) を提案オントロジーに準拠したリンクトグラフとして構築した例

向 (rsc:direction)、バッテリー残量 (rsc:battery) を定義した。自律走行車の属性情報のセンシング (rsc:observation) は、車載センサの Eidos が実行する dto:Affordance として定義されている。また、第 5.1 章で述べた移動指示制御は、rsc:eidos_agv に付随する doto:Action (rsc:action1) にそのモデルを、 dto:ActionInstance には複数回実行された個々の動作が記述される。各 ActionInstance の個体は実行時間を持つほか、該当 API に送信された入力データである事前に登録された実験エリア内の特定ポイント情報を表す dto:Data の個体を関連付けた。

5.3　Data Conversion Ontology を用いたデータ変換処理

デジタルツイン実装に必要なデータの編集・加工プロセスを、第 4.5 章で提案した Data Conversion Ontology に準拠する LOD として記述した。本実装では、データ編集の例として座標変換を対象とする。ビューアアプリケーションが取り扱う座標情報は、次の 2 種類が存在する。

- IFC ファイルに埋め込まれている建物要素の座標情報からビューア上 (すなわち Three.js の座標空間) への変換;
- 車載センサが取得するテレメトリーデータからビューア上への変換;

なお、本アプリケーションでは実装されていないが、座標変換含めた他のデータ編集・加工プロセスとしては以下のような例が考えられる。

- 車載センサが取得するテレメトリーデータから、部屋のある箇所を原点とする座標空間への変換;
- 離散的なテレメトリーデータを元にカルマンフィルタを用いた状態推定;

- 複数のセンサをソースとするセンサフュージョン;

本提案で実装した、ある座標系から異なる座標系への変換を表現するリンクトグラフデータを図 5.5 に示す。なお、本来リソース名はユニークであるべきであるが、本章では簡単のためユニーク ID やインデックス等の識別子は省略されていることに注意されたい。図中 rsc:CoordinatesSystem-1/2 は、dco:CoordinatesSystem の個体である。これらは各々座標系に対応しており、前者を IFC、後者を Three.js のものとする。rsc:CoordinatesSystem-1 は座標軸として rsc:axis-x/y/z_cs1 を dco:baseAxis プロパティにより関連付けられている。なお、本実装では簡単のためこれらの座標軸は正規直交基底であることを前提としている。rsc:CoordinatesSystem-1 を系とする座標点は、dco:Point の個体が dco:refersCS プロパティを介して表わされる。図中 rsc:point-1 が該当する。dco:Point の個体はそれぞれ dco:Vector3 または dco:Matrix4 の個体を有しており、具体的な成分が格納される。本オントロジーの中心であるデータ変換を表す状態遷移関数は、 Conversion 個体の rsc:conv_cs1-cs2 で表わされる。dco:sourceCS と dco:targetCS プロパティを介して rsc:coordinatessystem-1 と rsc:coordinatessystem-2 が関連付けられており、入出力の系である状態集合に対応している。すなわち、rsc:conv_cs1-cs2 は IFC の座標系から Three.js の座標系への変換として表現される。出力関数は Transaction 個体の rsc:scale3d/rotate/translate-1 として定義した。各個体は、事前に作成した gRPC サーバのアドレス、サービス、メソッドをデータプロパティとして持つ。なお、gRPC のデータフォーマットとして主流な Protocol Buffers [*4] は、リスト 5.2 で表わされるような proto ファイルによるスキーマファーストな設計を前提としている。すなわち、サーバ/クライアントともにスキーマを前提とした開発が主流であるため、本提案のように、利用する gRPC のメソッドをスキーマに従い事前に登録する実装は整合性が高いと言える。

リスト 5.2: gRPC サービスのスキーマとなる proto ファイル

```
syntax = "proto3";

package cstransform;

option go_package = "ino/GoCSTransform/pb";

service CSTransformService {
    rpc Scale3D (Matrix4Request) returns (Matrix4Reponse) {}
    rpc Translate (Matrix4Request) returns (Matrix4Reponse) {}
    rpc Rotate (Matrix4Request) returns (Matrix4Reponse) {}
}

message Matrix4Request {
    string Multiplicand= 1;
    string Multiplier = 2;
}

message Matrix4Reponse {
    string Element = 1;
}
```

[*4] https://developers.google.com/protocol-buffers

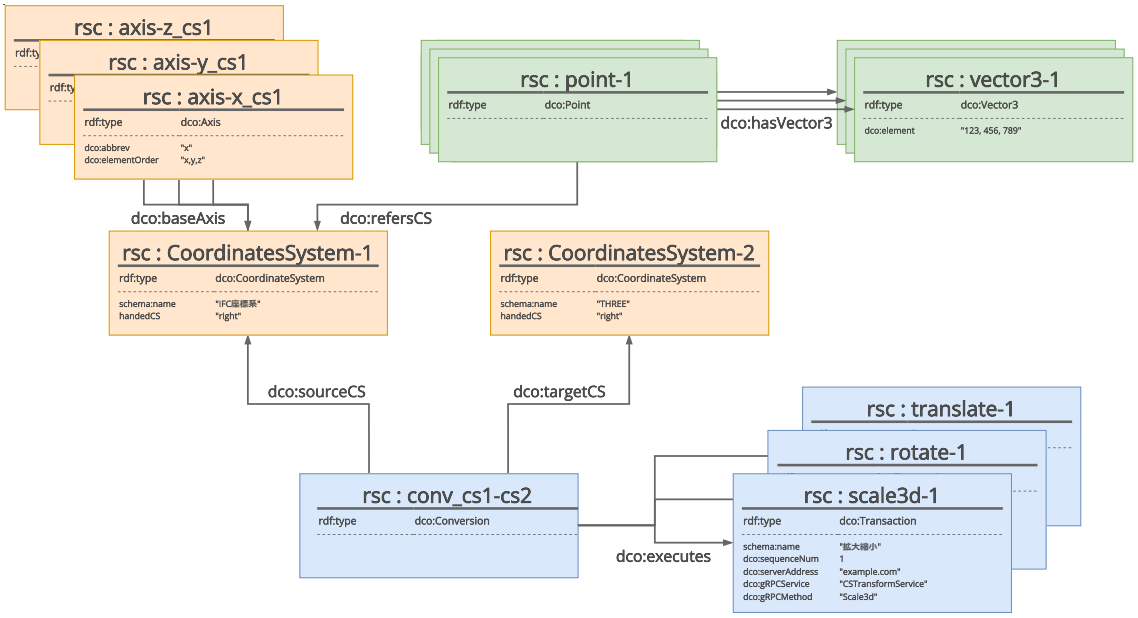


図 5.5: 基本的なアフィン変換を含む座標変換に Data Conversion Ontology を用いた例

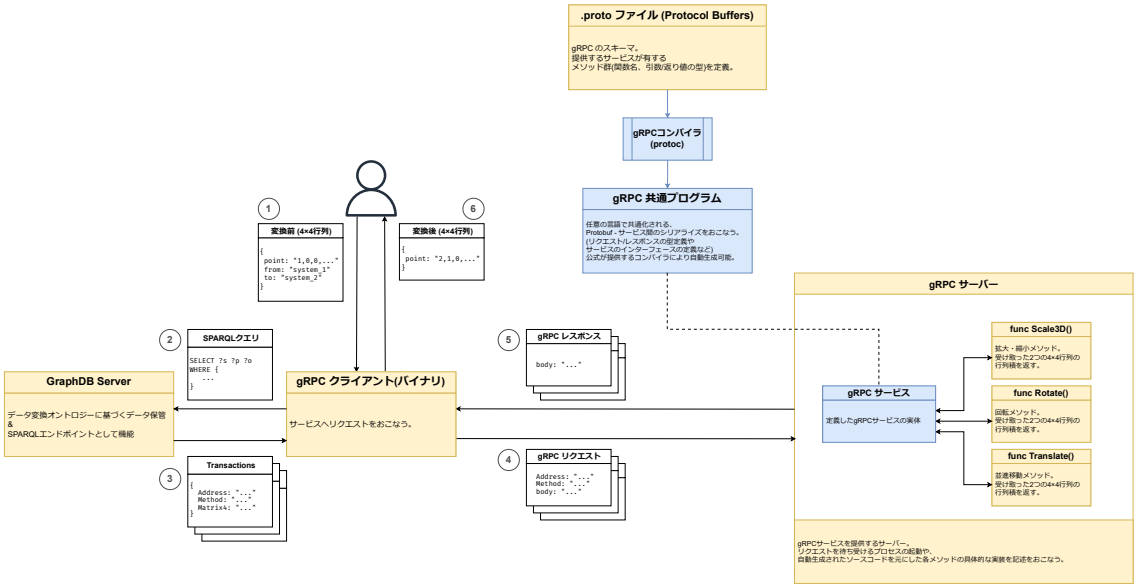


図 5.6: 提案オントロジーに基づきデータ変換をおこなう gRPC システムの概要図

このオントロジー及びリンクトデータを用いて、実際にデータ変換をおこなうシステムを実装した。実装の概要を図 5.6 に示す。なお本実装は、簡単なアフィン変換で表現可能な座標変換を対象としている。

はじめに、座標変換で用いる gRPC のサービス及びメソッドのスキーマを図 5.2 に示す proto ファイルに定義する。この proto ファイルから、protoc と呼ばれる gRPC コンパイラを通して開発言語 (本実装では Python) に則った共通プログラムを生成する。この共通プログラムは、言語問わず gRPC で要求される、リクエスト/レスポンスの型定義やサービスのインターフェ

イス定義など protobuf - サービス間のシリアライズをおこなうプログラムである。続いて、共通プログラムを用いて gRPC サーバについて述べる。本サーバーは、クライアントからのリクエストを待ち受けるプロセスの起動の他に、4 × 4 行列を引数とし、(1) 拡大縮小; (2) 回転; (3) 並進移動; の行列積を通して座標変換が達成されるような行列計算メソッドを実装した。最後にクライアントの設計について述べる。本クライアントは、変換元・先の座標系の名称と変換前の座標成分 (4 × 4 の変換行列で表わされる) を受け取る (図中 ①)。この座標系の名称を元に、リスト 5.3 のクエリを GraphDB に問い合わせ Transaction のリストを受け取る (図中 ②③)。先述の通り各 Transaction は gRPC サーバのアドレス、サービス名、メソッド名を持っており、sequenceNum プロパティで指定される実行順に引数で与えられた 4 × 4 行列の座標成分を計算していき、最終的な行列を得る (図中 ④⑤)。この行列が変換後の系の座標成分として、目的のデータ変換が達成される (図中 ⑥)。

リスト 5.3: 座標系の名称をもとに Transaction を取得する SPARQL クエリ

```
SELECT distinct ?transactions ?props ?objs ?elements
WHERE {
    ?from_cs a           dco:CoordinateSystem ;
             dco:hasName "${fromCSName}" .
    ?to_cs a           dco:CoordinateSystem ;
           dco:hasName "${toCSName}" .
    ?conv dco:sourceSystem ?to_cs ;
          dco:targetSystem ?from_cs ;
          dco:executes     ?transactions .
    ?transactions ?props ?objs ;
                  FILTER (?props != rdf:type).
    ?objs dco:element\* ?elements .
}
```

5.4　ビューアアプリケーションの実装と詳細

前章で構築したデータセットを用いて対象シナリオである屋内環境の包括的な情報が取得できることを確認するため、ビューアアプリケーションの開発をおこなった。ビューアアプリケーションは、各エンティティの属性情報や状態の時系列変化の取得、収録済みの屋内 3D コンテンツの自由視点操作及びレンダリングが可能な Web アプリケーションとして実装した。またアプリケーションへエンティティや環境に関する先験的なデータの事前埋め込みをすることなく、データセットへのクエリ呼び出しのみを用いてデータ提示が可能であることを目指す。

5.4.1　要求事項

先に述べた目的を踏まえて、ビューアの要求事項として以下を想定した。

Digital Twin Ontology の高可用性の検証

第 5.1 章で述べた通り従来のデータは独自仕様や連携性が考慮されていない形式で記述されており、サイロ化が発生していた。この課題に対し、提案オントロジーが相互運用性を向上させるかや、準拠する LOD がアプリケーションの実装コストを低減するか検

証する。

Digital Twin Ontology 準拠のデータの活用

Digital Twin Ontology の普及及び応用性の検証を目的としているため、Digital Twin Ontology 準拠のデータの活用は不可欠である。

対象シナリオの直感的理解

リンクトグラフデータはトリプルの集合かつ RDF 形式 (テキスト) で記述されているため、情報の取得に反して全体像が理解しにくい可能性が考えられる。3 次元ビューアを活用し、一連の LOD が示すシナリオや状況の直感的理解を促進する。

Web アプリケーション

専用のアプリケーションをインストールすることなく、Web ブラウザから直接利用可能な手軽さを目指す。

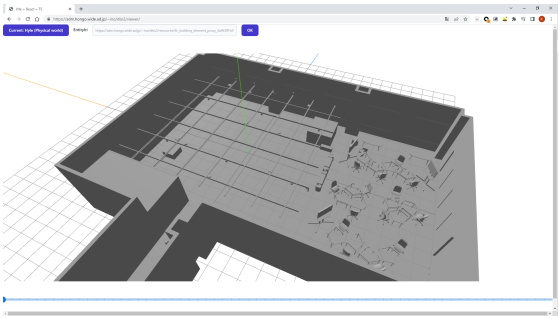
5.4.2 Web アプリケーション実装

ビューアアプリケーションのシステム設計概要を図 5.8 に示す。JavaScript のフレームワークである React.js [*5] をベースに、WebGL レンダラーを用いた汎用 3DCG ライブラリの Three.js [*6] および React Three Fiber [*7] を用いて実装した。なお、現行バージョンでは gRPC を直接ブラウザ上から実行するのは実装上困難であったため、先述の gRPC を用いたデータ変換はおこなわず、ソースコード上に座標変換を実装した。以下で各実装詳細について述べる。

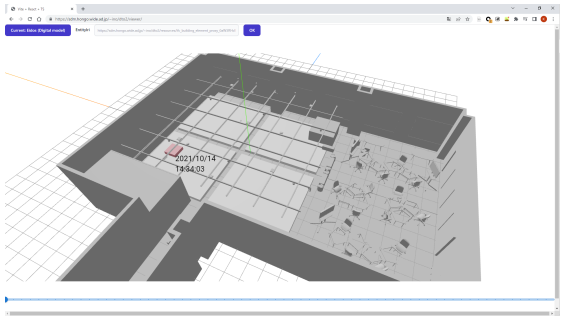
ビューア概要

ビューアのスクリーンショットを図 5.7 に示す。図 5.7a と 5.7b の通り、画面左上端のボタンで、Hyle / Eidos を切り替える。すなわち「Current: Hyle」の時は各エンティティの Hyle に付随する情報提示によって現実世界を表し、「Current: Eidos」の時は各エンティティの Eidos に付随する情報提示によって仮想世界を表す。図 5.7c と 5.7d は、画面をクリックすると該当箇所のエンティティの属性情報が表示される様子を表す。この属性情報も、現実/仮想世界の表示切替に連動して提示する内容を変える。また画面下部のスライドバーを左右に動かすことで、時間操作をおこなう。エンティティの座標情報 (dto:Coordinates) が、時間変化によってその値を変える場合、画面上でも移動することに加え、アファーダンスが実装されている場合、その振る舞いに関連するエンティティも色別表示されるよう実装した。例として図 5.7e は、自律走行車の移動制御のアフォーダンスを表しているが、自律走行車だけでなくその移動先のエリアも黄色に点灯する。

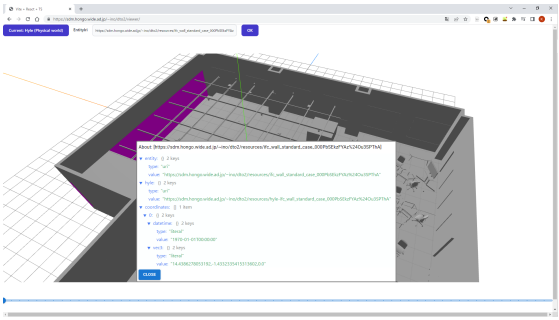
[*5] https://reactjs.org/
[*6] https://threejs.org/
[*7] https://github.com/pmndrs/react-three-fiber



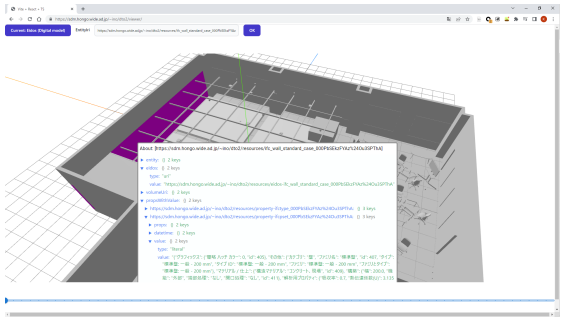
(a) Hyle を中心とした現実世界を表示する



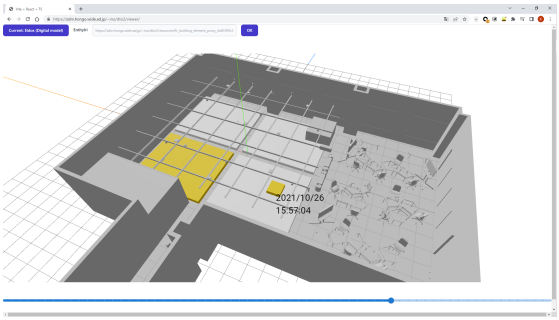
(b) Eidos を中心とした仮想世界を表示する



(c) クリックしたエンティティの Hyle を中心とした属性を表示する



(d) クリックしたエンティティの Eidos を中心とした属性を表示する



(e) 画面下部のスライドバーを左右に移動させ、時間を操作する

図 5.7: ビューアのスクリーンショット

ReactThreeFiber を用いた処理

本ビューアアプリケーションでは、エンティティ単位で SPARQL 問い合わせや属性情報の管理、座標情報の投影、UI レンダリング を制御可能とするために、コンポーネント指向で設計可能な React を採用した。SPARQL のレスポンスは大きく(1) 属性情報; (2) 座標情報; (3) 3D 幾何形状情報; の 3 種を取得でき、順に(1) テキストデータ; (2) 座標変換を経て Object3D の matrix; (3) Loader を経て Geometry/Material; として React コンポーネント、Three.js のオブジェクトへと処理が渡される。

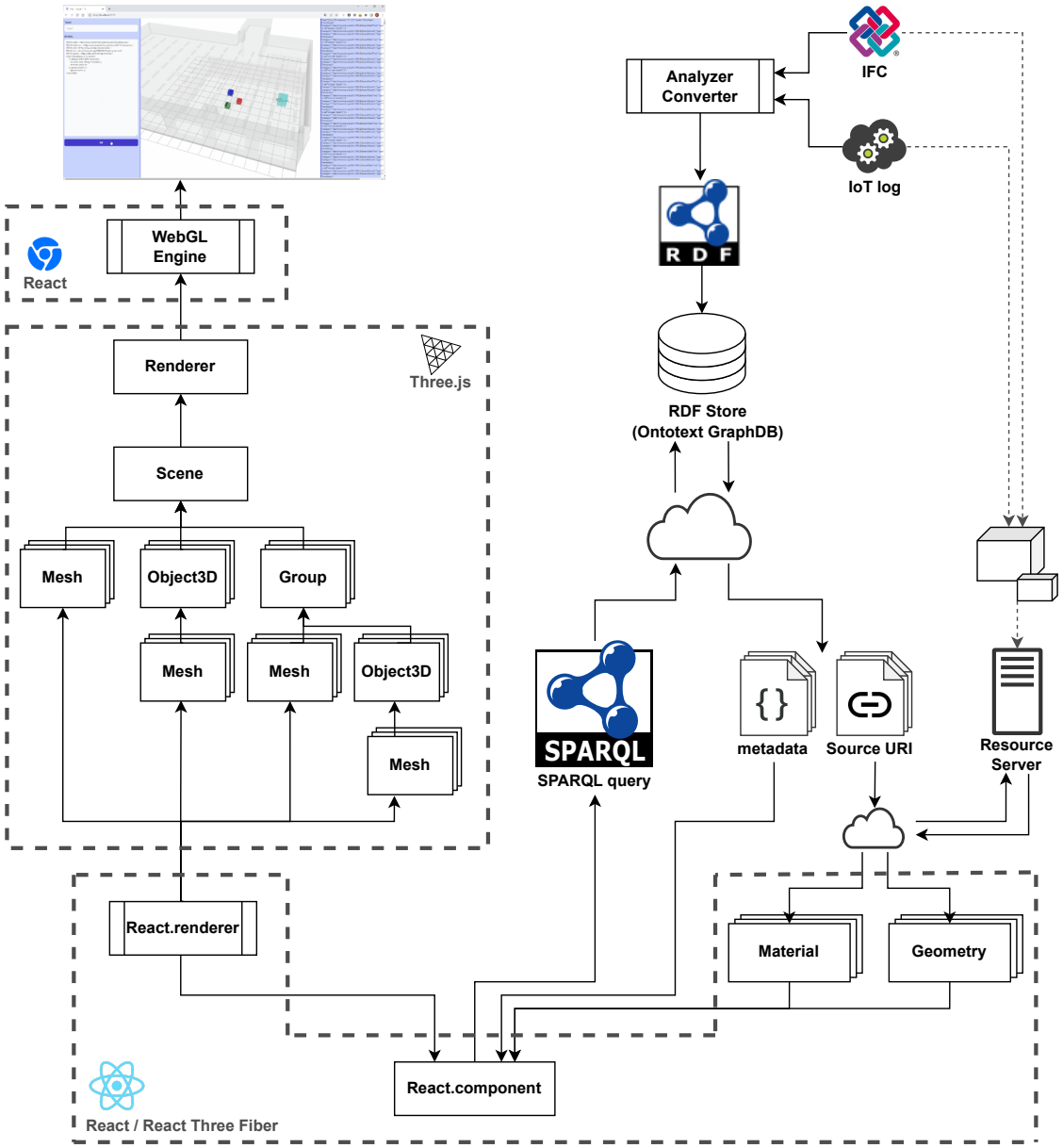


図 5.8: ビューアアプリケーションのシステム概要図

SPARQL クエリによる動的なデータ取得

実験エリアの包括的な情報が RDF データとして格納されている RDF ストアに SPARQL クエリを発行し、アプリケーションの動作に必要なデータを取得する。現状発行するクエリは(1) 環境および建物要素の座標や幾何形状などの情報取得; (2) 内部物理資産の座標や幾何形状などの情報取得; (3) アフォーダンスの取得の 3 種類ある。

第 6 章

評価と考察

第 4 章〜5 章で述べた提案オントロジー及びビューアアプリケーションに対し、4 種類の評価をおこない、得られた結果について考察する。

6.1　知識の網羅性と人気度に関するオントロジーの評価

Turchet ら [36] が用いた指標を適用し Digital Twin Ontology の定量評価をおこない、結果を表 6.1 に示した。これは Fernández ら [37] が定義した、知識の網羅性と人気度に関するオントロジーの評価測定のための 12 の指標を、Turchet らが一部を修正しつつ自身らの Internet of Musical Things Ontology[36] に適用した指標である。ただし、いずれの指標についてもオントロジーを評価するための有意な統計的意味をもっているとの報告はされておらず、あくまで参考としての値であることに注意されたい。

指標	値
クラス数	36
プロパティ数	39
- オブジェクトプロパティ	30
- データ型プロパティ	9
個体数	0
外部オントロジーに利用されている数	0
外部オントロジーを利用している数：	
- 外部オントロジーの利用数	3
- 外部クラスの割合	8.3%
- 外部プロパティの割合	12.8%

表 6.1: Turchet ら [36] が適用した指標を用いた、提案オントロジーの構造評価

6.2 既存オントロジーとの比較による定性評価

既存オントロジーと提案オントロジーの比較を行い、定性評価を行った。比較対象として、デジタルツインに向けた新規オントロジーを提案した Singh らの研究 [21] に加え、オントロジーや関連技術を活用したデジタルツイン研究で注目される BIM/IFC、BOT、REC (Real Estate Core)、SSN、WoT を挙げる。第 3 章の通り、BIM/IFC、BOT、REC は主に建築分野、SSN、WoT は主にセンサや IoT 分野のセマンティクスである。なお、REC は 2022 年 8 月末に大幅なアップデートを含む ver4 が公開された [*1]が、プレビュー版であるため本稿では前身の ver3.3 を対象とする。本研究で取り組む課題および既存オントロジーの設計思想を考慮し、本定性評価の指標として次の 6 項目を適用した。

記述可能な対象領域
: 提案オントロジー及び関連する既存オントロジーが対象とする記述領域を整理する。下位項目としてエンティティ、環境、その他の 3 項目を挙げる。

物理ツインと仮想モデルの分離
: 現実世界の忠実な反映と設計者が意図する期待世界とを切り分けた上で、連携がおこなわれているかを評価する。

物理ツインの記述
: 現実世界のエンティティの属性や状態を汎用的に記述可能かを評価する。

仮想モデルの記述
: 各クラスが表現するエンティティの属性情報等の記述が可能かを評価する。下位項目としてアプリケーションやデジタルコンテンツに用いる設計者定義の状態や属性 (3D モデルの URI、接続プロトコルなど)、アフォーダンス (設計者が意図した動作/振る舞いなど) の 2 項目を挙げる。

時系列/動的データの記述
: 静的データのみならず、時系列情報や動的情報を含めたデータの記述が可能かを評価する。

データ変換プロセスの記述
: アプリケーションの実装まで見据えたデータ変換のプロセスを記述可能かを評価する。

上記の指標に基づき、提案手法を含めた 7 個のオントロジーを、元となる論文やドキュメント、関連するレビュー論文を参考に独自評価した結果を表 6.2 に示す。なお、表中で用いている 3 種類の記号の意味/役割はそれぞれ次の通りである。

✓: 十分に記述可能、またはそれに準ずる語彙等が定義されている。
△: 評価不能、または該当する語彙が存在するが表現可能な範囲が十分ではない。
×: 表現不可能、またはそれに準ずる語彙の定義が無い。

[*1] https://github.com/RealEstateCore/rec/releases/tag/v4.0-preview

	記述対象領域			物理ツインと仮想モデルの分離
	エンティティ	環境	その他	
提案オントロジー	任意の事物	空間および建物要素	時系列変化、抽象モデル	✓
Singh, et al. [21]	任意の事物	-	抽象モデル	✓
BIM/IFC [24]	一部設備や什器	建物要素と階層	-	×
BOT [25]	一部設備	建物要素と階層	-	×
REC v3.3 [30]	関連システム	空間および建物要素	事物の動作	×
SSN [27]	任意のセンサデバイス	-	プロセス、動作など	×
WoT [28]	△	-	動作、作用、プロトコルなど	×

	物理ツインの記述	仮想モデルの記述		時系列情報の記述	データ変換プロセスの記述
	現実世界の状態や属性	設計者定義の状態や属性	機能、振る舞いの記述		
提案オントロジー	✓	✓	✓	✓	✓
Singh, et al. [21]	△	△	△	✓	×
BIM/IFC [24]	×	✓	△	×	×
BOT [25]	×	△	×	×	×
REC v3.3 [30]	×	✓	✓	×	×
SSN [27]	×	×	✓	✓	×
WoT [28]	×	✓	✓	×	×

表 6.2: 6 項目の評価指標に基づくオントロジーの定性評価

以下、各項目ごとの評価詳細および考察を述べる。

記述可能な対象領域

Singh らのオントロジーは、事物とその抽象モデルの関係性記述に特化している。BIM/IFC および BOT は、建築物の構造および空間を区切る要素を表す IfcBuildingElement 下、床や壁、窓やドアに渡り建物要素および関連設備を正確に記述可能であるが、要素間の相互作用や動作等の記述は想定されていない。また SSN や WoT は環境情報の記述は定義されていないものの、センサを始めとする IoT デバイスに関して柔軟な記述が可能である。しかし、デバイスが持つ機能の記述として確立されていることから、環境情報や他オブジェクトとの相互作用の語彙は定義されておらず、汎用的なデジタルツインオントロジーとしては不足点が多い。REC は、提案オントロジーと同様に環境から内在する事物の情報までを包括的に記述することを目指していると言える。

物理ツインと仮想モデルの分離

Digital Twin Ontology 及び Singh らの手法は、実空間の属性情報を正確に記述するクラスとオブジェクトの抽象モデルの定義を分割することで、アプリケーションやコンテンツ制作に必要な情報をエンティティへ柔軟に関連付けることを可能としている。Digital Twin Ontology と記述対象領域が共通する REC の大きな相違点として本項が挙げられる。

物理ツインの記述

上記の通り、物理ツインと仮想モデルの分離をおこなうオントロジーは提案手法と Singh らの研究のみであり、その他のオントロジーでエンティティの物理状態や属性を記述可能としているものは見られなかった。一方、Singh らが物理ツインとして定義する Asset クラスは、最小要件しか定義されておらず、その状態値や属性値の記述に関して論文中で述べられていないため、本評価では △ とした。

仮想モデルの記述

Digital Twin Ontology 及び WoT は、Property や Action、Event 概念によって仮想モデルの属性や状態情報、機能、振る舞いを柔軟に記述することが可能である。REC は属性情報を記述するための概念こそ持たないが、汎用的にエンティティを記述する Asset 概念に具体的なサブクラスを多数定義することで、必要な記述量を減らしていると考えられる。Singh らのオントロジーは、状態や機能の表現に該当すると考えられる概念を定義しているが、前述の通り具体的な記述方法が述べられていないため △ とした。BIM/IFC は IfcProperty [*2]によりカスタムプロパティを柔軟に記述可能である。一方、汎用的な「振る舞い」を記述するための概念は定義されていない。建築資産の記述に特化しているため、機能や振る舞いはエンティティに割り当てられたクラスの暗黙的な共通理解となっていると考えられる。BOT は BIM/IFC の軽量オントロジーとして、一部表現手法が削減されているため BIM/IFC から評価を一段階ずつ落としている。SSN/SOSA はセ

[*2] https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/lexical/IfcPropertySet.htm

ンサデバイスの範囲内でその機能や振る舞いを記述するための概念が多数定義されている。一方で、センサ自体や観測対象として定義される FeatureOfInterest などの属性記述に関してはドキュメント内で述べられていない。

時系列/動的データの記述

Digital Twin Ontology はある時点における属性情報や動作の記述を目的として、Instance クラスを導入し動的データの表現も可能とした。Singh らの手法や SSN も同様に、センサから取得したデータに時間情報を記述することで時系列情報の付与をおこなっている。BIM/IFC や BOT、また REC、WoT にはこのような時間情報の定義はなく、静的情報のみ取り扱う。

データ変換プロセスの記述

観測データをはじめとするテレメトリデータや各種属性値に関する処理、特に編集や加工プロセスに関する記述は、提案オントロジーを除いて見られなかった。

6.3　シナリオごとの評価

本研究で提案したオントロジー及び LOD の利用可能性を評価する。具体的な方針として、まず、第 5.1 章で設定した実証フィールドにおいて想定される利用シナリオを挙げ、その後、それぞれの目的を満たすデータの取得が可能であることを示す。

シナリオ 1: ある建物を構成する建築要素一覧の取得

フロアや部屋など、ある空間を起点としてデジタルツインを構築する時、その空間を構成するエンティティを取得したいという要求が想定される。なお、本シナリオでは、簡単のため予め設計者によって空間とエンティティが紐付けられたものに限定する。

DTO の LOD に対してリスト 6.1 の SPARQL クエリを投げることで、題意のエンティティを取得することができる。対象の空間をその URI (リスト中 `rsc:$buildingUri`) で検索し、そのエンティティが有する Eidos から dto:locatedIn プロパティで結びつくエンティティがレスポンスとして返る。

リスト 6.1: ある建物を構成する建築要素一覧の取得

```
PREFIX dto: <https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#>
PREFIX rsc: <https://sdm.hongo.wide.ad.jp/~ino/dto/resources/>

SELECT DISTINCT ?entity WHERE
{
    ?entity dto:hasEidos ?eidos .
    ?eidos a              dto:BuildingElement ;
           dto:locatedIn ?building_eidos .
    ?building dto:hasEidos ?building_eidos .
              FILTER(?building = rsc:${buildingIri})
}
```

シナリオ 2: あるエンティティが有する属性情報や 3D 幾何形状情報、位置情報等の取得

シナリオ 1 等で取得したエンティティを起点として、関連する属性情報を全て取得したいという要求が想定される。

DTO の LOD に対してリスト 6.2 の SPARQL クエリを投げることで、題意のエンティティを取得することができる。両クエリ共に $name を元にエンティティを検索し、それに紐づく Hyle と Eidos を出発点としている。前者は Hyle に関するクエリである。dto:locatedAt プロパティで関連付けられる座標情報から、時系列表現として記述される dto:CoordinatesInstance 個体を取得し、DCO で定義されたプロパティを利用して 3 次元ベクトルで表わされる座標成分を取得する。後者は Eidos に関するクエリである。3D 幾何形状情報を dto:hasSimpleVolume プロパティ経由で取得するほか、dto:hasProperty で付随する属性情報、その時系列表現の Instance 個体を取得することで、題意の情報がレスポンスとして得られる。

リスト 6.2: あるエンティティが有する属性情報や 3D 幾何形状情報、位置情報等の取得

```
PREFIX dto: <https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#>
PREFIX dco: <https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#>
PREFIX rsc: <https://sdm.hongo.wide.ad.jp/~ino/dto/resources/>

SELECT DISTINCT ?entity ?hyle ?datetime ?vec3_element WHERE
{
    ?entity a            dto:Entity ;
            dto:hasEidos ?eidos ;
            dto:hasHyle  ?hyle .
            FILTER(?entity = rsc:${entityIri})
    ?eidos a dto:BuildingElement .
    ?hyle dto:locatedAt ?coordinates .
    ?coordinates dto:execute ?coordinstances .
    ?coordinstances dto:hasTimestamp ?datetime ;
                    dto:hasPoint     ?point .
    ?point dco:hasVector3 ?vec3 .
    ?vec3 dco:element ?vec3_element .
}

SELECT DISTINCT ?entity ?eidos ?volumeUri ?props ?datetime ?value WHERE
{
    ?entity a             dto:Entity ;
            dto:hasEidos ?eidos ;
            dto:hasHyle ?hyle .
            FILTER(?entity = rsc:${entityIri})
    ?eidos a                    dto:BuildingElement ;
           dto:hasSimpleVolume ?volumeUri ;
           dto:hasProperty     ?props .
    ?props dto:execute ?instances .
    ?instances dto:hasTimestamp ?datetime ;
               dto:hasData ?data .
    ?data dto:hasValue ?value .
}
```

シナリオ 3: ある名前を持つエンティティに付帯する座標情報及び振る舞いと、それを観測するセンサ情報の取得

本シナリオでは、対象エンティティをその名前で検索し、それに紐づく属性情報の取得を目的とする。属性情報を観測するセンサの情報も併せて検索することで、センサが発行するテレメトリデータを取得する。

DTO の LOD に対してリスト 6.1 の SPARQL クエリを投げることで、題意のエンティティを取得することができる。両クエリ共に `$name` を元にエンティティを検索し、それに紐づく Hyle と Eidos を出発点としている。前者は Hyle に関するクエリである。dto:locatedAt プロパティで関連付けられる座標情報の根拠である観測を dto:captures プロパティから探索する。さらに、その観測の実行主体であるセンサを dto:sosaMadeObservation から探索するほか、3 次元ベクトルで表わされる座標成分も、DCO で定義されたプロパティを利用して取得する。後者は Eidos に関するクエリである。なお、属性情報はシナリオ 2 で説明したため省略する。エンティティがその仮想モデルとして機能を持つ場合、eidos から `hasAction` プロパティを介して取得可能である。各 Action は時系列表現として dto:ActionInstance 個体を持つため、dto:hasTimestamp で時間情報、dto:hasData に紐づく dto:Data 個体から物理量や観測量を表すリテラル値が導ける。さらに、dto:Data 個体は、dto:Parameter を介して dto:DataSchema とも関連付けられており、参照するデータスキーマ情報も取得できる。

リスト 6.3: ある名前を持つエンティティに付帯する属性情報及びそれを観測するセンサ情報の取得

```
SELECT DISTINCT ?entity ?hyle ?datetime ?vec3_element ?sensor WHERE
{
    ?entity a            dto:Entity ;
            dto:hasName  "${name}" ;
            dto:hasHyle  ?hyle ;
            dto:hasEidos ?eidos .
    ?hyle dto:locatedAt ?coordinates .
    ?observation dto:captures ?coordinates ;
                 dto:executes ?coordinstances .
    ?eidos_sensor dto:sosaMadeObservation ?observation .
    ?sensor dto:hasEidos ?eidos_sensor .
    ?coordinstances dto:hasTimestamp ?datetime;
                    dto:hasPoint     ?point .
    ?point dco:hasVector3 ?vec .
    ?vec dco:element ?vec3_element .
}

SELECT DISTINCT ?entity ?eidos ?actionName ?datetime ?paramName ?value WHERE
{
    ?entity a           dto:Entity ;
            dto:hasName "${name}" ;
            dto:hasEidos ?eidos .
    ?eidos dto:hasAction ?action .
    ?action dto:hasName ?actionName ;
            dto:execute ?actionInstances .
    ?actionInstances dto:hasTimestamp ?datetime ;
                     dto:hasData      ?data .
    ?data dto:hasValue   ?value ;
          dto:references ?ref .
    ?ref dto:hasName ?paramName .
}
```

以上のように、DTO を用いて LOD 化したデータは、対象空間及び内在するエンティティの情報を網羅的にデータ提供可能である。独自仕様や連携性が考慮されていないオントロジーに基づく従来手法では、目的とするデータ資産の偏在や、取得後のデータ統合などの問題が発生した一方で、セマンティックに統合された本提案の LOD の利用により、対象エンティティのセマンティック検索や関連情報の柔軟な取得が可能であることが示された。

6.4 ユーザによる内容評価

オントロジーの質を保証する上で、オントロジーが対象世界を適切に記述可能か評価し、その結果を元に洗練をおこなうフェーズは重要である [38]。本研究では、オントロジーの洗練のフェーズに向けた取り組みとして、提案オントロジーのユーザを想定した被験者を対象に、表現の妥当性や拡張性などの内容評価をおこなった。

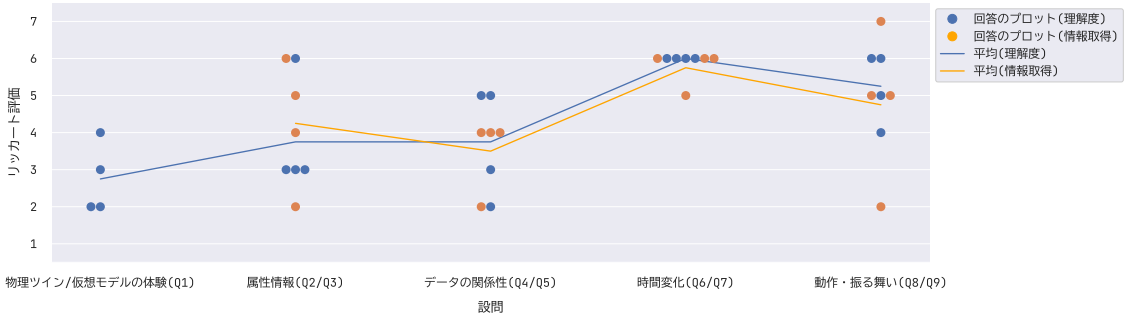
6.4.1 調査概要

本インタビューは 2 部構成でおこない、第 5 章で述べたビューアアプリケーションとリンクトグラフデータそれぞれについて調査を実施した。本提案の利用者や開発者を想定し、オントロジー / デジタルツイン / 建築分野に関して業務での利用経験を持ち、一定のドメイン知識を持つと考えられる専門家 4 名を被験者とした。はじめに共通するシナリオについて説明した後、各実装の概要と操作方法について動画及びデモンストレーションを通して説明した。その後被験者に自由に操作してもらいながら、9 つの設問によるアンケートと自由記述に回答する形でインタビューを進めた。

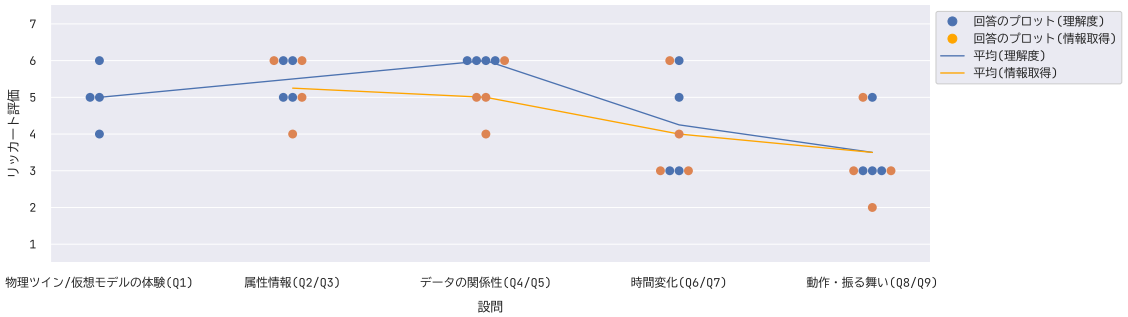
6.4.2 質問項目

アンケートは、以下に示す Q1 から Q9 までの 9 項目から成る。それぞれ 1 を最低評価、7 を最高評価とした 7 段階のリッカート尺度を用いて回答してもらった。Q1 は本提案のアーキテクチャのベースとなる物理ツインと仮想モデルの概念を直感的に理解できたかを問う設問である。Q2 以降の設問は、特定の項目に対してその理解度とデータの獲得度に観点から評価してもらった。Q2 及び Q3 はエンティティの属性情報について、Q4 及び Q5 はエンティティ間やデータ間の関係性について、Q6 及び Q7 はエンティティの時間変化について、そして Q8 及び Q9 はエンティティの動作・振る舞いについて問うものとして作成した。

Q1 実空間（Hyle）と仮想空間（Eidos）を体験できたか (概念及びその差異が直感的に理解できたか)
Q2 (クリック先など) 対象の物体が持つ属性情報を理解できたか
Q3 (クリック先など) 対象の物体が持つ属性情報が十分に得られたか
Q4 (クリック先など) 対象の物体や各データの関係性が理解できたか
Q5 (クリック先など) 対象の物体や各データの関係性の情報が十分に得られたか



(a) ビューアアプリケーションに対する回答



(b) グラフデータに対する回答

Q6　(クリック先など) 対象の物体の時間変化に関する情報が理解できたか

Q7　(クリック先など) 対象の物体の時間変化に関する情報が十分に得られたか

Q8　画面中央付近で時間変化に伴い移動する物体について、その物体がおこなう動作について理解できたか

Q9　画面中央付近で時間変化に伴い移動する物体について、その物体がおこなう動作について十分な情報が得られたか

上記の設問を、ビューアとリンクトグラフデータに対してそれぞれ回答してもらった。

6.4.3　評価結果

はじめに、アンケートの結果を図 6.1a 及び図 6.1b に示す。縦軸は前項で述べた Q1〜Q9 までの質問項目およびその有効回答数を，横軸は 1〜7 までの 7 段階評価の回答比率を積み上げ棒グラフで表し，中間評価を表す 4 の回答比率の中間を原点に置いた．評価 5，6，7 が多いほど横軸正方向に偏り，評価 1，2，3 は負方向に偏る．

続いて、口頭での議論及び自由記述からの回答について述べる。回答内容を分析し、大きく、(1) オントロジーの構成; (2) 時系列データの扱い; (3) ビューアアプリケーションの UI (User Interface); (4) 物理ツインと仮想モデル; (5) オントロジーのアンケート評価; の 5 つに分類した。各分類ごとに得られた評価を下記にまとめる。

オントロジーの構成

- オントロジーやその概念については簡易に整理がされている。
- 関係性は分かり易い。
- 建築関係用途であれば Value 内の値が見たいので階層が深いのが気になる。
- オントロジーのヒューマンリーダブルとマシンリーダブルのバランスが考慮されている。
- インシデント発生時に関する記述が成されていない。「実空間システムは異常があった場合に安全側に動く」という原義に則って、例えば仮想モデルが持つアフォーダンスの拡張として、正常系と異常系の汎用モデルが記述できると良い。

時系列データの扱い

- Instance クラスを見ると理解しやすい。
- 時系列データはわかりにくい。
- 操作履歴やセンサなどの時系列データは SPARQL サーバにインスタンスとして入っていることは少ない（別途 IoT プラットフォームなど、時系列データベースに保存することが多い）。したがってデータを JOIN する必要があるが、その際に無理に RDF で表現する必然性はないように思う。
- 時系列データをそのまま RDF データとして表現する意図がよいのか悪いのか判断が難しい。
- なぜオントロジーに埋め込んだのか気になった。

ビューアアプリケーションの UI

- タイムラインは分かり易い。
- デジタルツインは、オントロジーやセマンティクスだけでなく、デジタル表現とバンドルされたもの。ユーザ体験として視覚情報は重要である。
- ビューアは属性表示の UI が少し見にくい。スクロールができないことも問題。
- 3D のビューワーについては、データのわかりやすさとは別に UI としてのわかりにくさという問題があります。特に、テキストボックスの内部がスクロールできないため、データの詳細が見えない。
- 情報提示として RDF の URI をそのまま載せているが、人にとっては分かりにくい。特定のスキーマに従って変換するなどの留意は必要に思う。
- 表現についてはヒューマンリーダブルな状態ではない。

物理ツインと仮想モデル

- Hyle と Eidos の用語の語源が気になった。

- 実空間と仮想空間で表示する情報を変えることで、差分の理解が深まる。
- 現実空間とデジタルのモデルの差異をどう見せたいのかがよくわからない
- 差異を見たいというユースケースはあると思う
- 表現としての違いは分かるが、目的が分かりづらい

オントロジーのアンケート評価

- 「オントロジーの専門家」ではないので、それを正しく評価できるか難しい。
- オントロジーには明確な指標がないので難しい。
- 「具体的なシナリオを設定して、ロールプレイ形式で被験者のドメイン理解を測定する」といった評価方法も考えられる。

6.5　考察

本節では、提案手法に関する 4 種類の評価を踏まえ、第 2.2 章で述べた課題に対する提案手法の有効性及び今後の普及に向けた課題について議論する。

6.5.1　提案オントロジーの有用性について

Digital Twin Ontology は、デジタルツインに関するデータ資産を包括的に管理することを目的として、第 2.2 章の課題をもとに第 4.1 章で整理した要求事項を満たす手法として提案した。

はじめに、オープンなオントロジーとそれに準拠する LOD を実装することで、セマンティックに基づく属性及びデータ間の関係性を表すメタデータを個別のデータ資産に付与することが可能となった。データ資産は従来、独自仕様のシステムによって閉じた管理がおこなわれ、相互運用性の低下が課題となっていたが、本提案により全てのデータが相互に連携する巨大なネットワークとして機械処理が可能となったほか、各データは意味論的に接続されるためデータ資産のユーザ理解も促進することが期待される。

また、提案オントロジーは個別の Entity 概念に対して、デジタルツインの原義である実世界の忠実なデジタル複製としての Hyle 概念と、アプリケーションやシステムの設計者が解釈・抽象化したエンティティのデジタルモデルとしての Eidos 概念を設計した。これにより、同一エンティティの概念の下に、現実世界を観測・再現することを目的としたデータ資産と、アプリケーション内で期待される特定の機能や振る舞いを実装することを目的としたデータ資産とを分離することが可能となる。一方、本概念は物理ツインと仮想モデルの乖離を抑止する手法として提案したが、その確からしさに関するユーザ評価は未検証である。物理ツインと仮想モデルの乖離度について厳密な指標を定義した後に、実際の運用現場において長期的な追跡実験が必要と考えており、今後の課題となっている。

最後に、Data Conversion Ontology にてアプリケーション実装で必要なデータ変換プロセスの形式化をおこなった。具体的な実行プログラム自体は外部のオープンサービス (本提案で

の gRPC に該当する) に委譲するが、オントロジーに基づき構築されるリンクトグラフデータを有限オートマトンの状態遷移図としてみなすことで、データ変換に必要な手続きの呼び出しを定式化することが可能となった。本研究では簡単な座標変換のみを対象としたが、Data Conversion Ontology に基づく LOD を実装したことでその応用可能性を示した。

6.5.2 今後の普及に向けた課題

オントロジーの質は、ドメインの専門家による表現の妥当性や拡張性の内容評価を繰り返すことにより保証されることから、世間への周知や導入コストの低下などオントロジーの普及活動は非常に重要だと言える。したがって本節では、オントロジーの普及の観点から提案手法の課題や改善点を考察する。

はじめに、時系列情報の提示について述べる。Digital Twin Ontology では、時系列情報を記述するために Instance クラスを定義した。単一かつ離散的なエンティティの動作を表現しながら他の概念と意味的統合をおこなうために、オントロジーのクラスとして定義し、個体を RDF データ形式で作成した。一方で、ユーザ評価でも指摘された通り、この手法は、(1) 多くの先行オントロジーが採用する手法と異なる; (2) データ肥大化の誘因となる; という 2 つの課題を抱えている。先行オントロジーの多くは時系列データを RDF 形式では持たず、それを保存するデータベースなどの場所を指定するためのプロパティを持つことで対象のエンティティと関連付けている。例として、Real Estate Core と同様に建築資産の記述を対象とする Brick [39] は、`hasExternalReference` プロパティを介して特定の建築資産とその時系列データが保存されている場所を関連付けている。また RDF 形式は一般に時系列データとして最適化されたものよりもサイズが大きくなる傾向にあると言われている。本提案で実装した LOD では、1 インスタンスあたり数百バイトにのぼる。しかし時系列データを外部に委譲する先行手法は、データベースインスタンスに適切なスキーマやプロパティを設計する必要があるという欠点も存在するため、オントロジーにおける時系列データ表現は、スキーマファーストな設計と技術的最適化のトレードオフにあると言える。

次に、ビューア設計及び情報提示について述べる。本研究で実装したビューアアプリケーションは、提案オントロジーの応用可能性を検証するとともに、リンクトグラフデータが表す情報の理解を促進させることを目的として設計した。Q2 〜 Q9 の設問は、Q2 〜 Q5 が静的な情報、Q6 〜 Q9 が動的な情報に関する理解度・情報獲得度を問うものとして分類される。図 6.1a と図 6.1b に示す設問 Q2 〜 Q9 の回答から、ビューアとリンクとグラフデータはこの静的情報・動的情報の分類に関して逆の評価となる傾向が見られる。ユーザ評価で指摘のあった通り、ビューアで提示する属性情報は RDF データ形式そのままであり、対人可読性に欠けることやスクロールしづらいという実装上の課題を起因として、理解度・情報獲得度が低くなったと考えられる。一方、動的情報に関してはビューアの方が理解度・情報獲得度が高い傾向が見られる。したがって、動的情報の提示には、RDF データやリンクトグラフデータだけでなくビューアを併用することで、ユーザの理解度・情報獲得度促進を向上させると結論付けられる可能性がある。

最後に、概念の理解のしやすさについて述べる。DTO では、基本的なデジタルツインアーキテクチャである物理世界と仮想世界の二元論ではなく、外界と物理ツインと仮想モデルから成る三元論による分類を基礎として設計した。またそれらに対応するクラスとして、Hyle クラスと Eidos クラスを定義した。インタビューの冒頭では、複数人の被験者からこれらの語彙の語源または意味に関する質問が上がった。Hyle 及び Eidos はアリストテレス哲学をまとめた「形而上学」で登場する概念である [40] が、意味が十分に知れ渡った概念とは言い難く、DTO に際してもクラス名だけでその概念を理解するのは難しい。また物理ツインと仮想モデルの理解促進も課題となった。ビューアアプリケーションは提案する三元論のデジタルツインアーキテクチャの直感的理解を促進させる目的で設計したが、アンケート Q1 の結果では平均評価の 4 より高い評価が得られなかった。これはユーザ評価で回答を得た通り、アプリケーション内で確認可能な差分が少なかったことから、それぞれの概念の目的や意義が把握しづらかったことが原因と考えられる。これは、各概念に紐づくクラスやデータが確認できる LOD に対する同設問にて、平均評価の 4 未満の回答が無かった点とも合致している。提案したアーキテクチャに対する否定的な意見が無かったことも踏まえ、正常系と異常系の汎用イベントを追加するなどユースケースや応用例を充実させることで、本提案設計の意義や目的の理解を促進し、普及に繋げていくことが重要だと考えられる。

第 7 章

結論と今後の課題

7.1　まとめ

本研究では、デジタルツインが抱える多様かつ異質なデータ資産を、オントロジーにより包括的管理を可能とする手法を提案した。建築情報から IoT デバイスに至るまで、既存の記述仕様や参照アーキテクチャを調査し、モデルベースでデータ駆動設計をおこなう現状の取り組み及び課題、要求事項を整理した。提案手法として、屋内エリアの時空間及び物理資産を表現可能なデジタル環境記述仕様である DTO を設計した。従来の二元論的なデジタルツインを拡張し設計者視点を導入することで、Hyle 及び Eidos と定義した概念により、現実世界での物理状態と設計者視点での機能/振る舞いを体系的に構造化できる。また、センサから取得したテレメトリデータや独自に設計者が指定したスキーマを相互に連携し自由に変換するため、DCO を提案した。DCO はベースとなる”系”の概念と、変換及び具体的な関数の概念から構成される最小要件を満たしたものであるが、基本的なアフィン変換で実現される座標変換を表現可能であることを示した。これらの提案オントロジーを、オフィス環境を模した実験場にて収集したデータに適用し、LOD として構築した。さらに、プロトタイプとしてビューアアプリケーションの実装をおこない、デジタルツインの実装に至るプロセスを前提としたデータ資産の包括的管理やアプリケーションの相互運用性向上が可能であることを示した。加えて、提案オントロジーの表現の妥当性を検証するために、ドメインの専門家に対してアンケートを通した内容評価を実施した。オントロジーによって構成されるリンクトグラフデータは、エンティティに付随する属性情報や関係性の理解は十分におこなえるが、時系列変化を伴う情報や動作・振る舞いの表現は直感的ではないという傾向が見られた。一方、ビューアアプリケーションは動的表現を得意とするなど、リンクトグラフデータと逆の評価が得られたが、UI として対人可読性を考慮した適切な情報提示の工夫が要求されるといった課題が発見された。

7.2　今後の課題

今後の課題として、下記が挙げられる。

- 対象シナリオの拡張と検証

- 提案手法のユースケースの多様化
- オントロジー評価の検討

はじめに、対象シナリオの拡張と検証について述べる。本研究では、オフィスを模した実験環境を対象として LOD 及びビューアの実装をおこなった。個別の建物要素を含む建築基盤データから動的な IoT デバイスとしてセンサを搭載する自律走行車に至るまで、多様なエンティティの表現をおこなったが、その振る舞いとしては移動制御指示のみを対象とした。BACS をはじめとするビル管理を例にとっても、照明空調の制御や人流データを起点とした各設備の統合制御など、様々なシナリオが考えられる。これらのシナリオを本提案に準拠する形で実装し、データを正確に記述出来るか検証し、必要に応じてオントロジーの構造を見直していく。

続いて、提案手法のユースケースの多様化について述べる。本提案では、FBS を基礎とした三元論的アーキテクチャを提案した。既存手法と異なり、現実世界の物理状態と設計者視点での機能を分割できるため、デジタルツインの原義に基づく実空間と仮想空間の親和性を向上させる連携基盤として機能することが期待される。一方、これらは本研究の実装の範囲内にとどまり、ユーザによる中長期的な運用などの追跡調査はおこなっていない。また、第 6.4 章のユーザ評価において物理ツインと仮想モデルの有用性に関するより多くのユースケースが求められた通り、提案手法の理解促進や普及に向けた取り組みとして、適切なユースケースの検証と提示をおこなう必要があると考えられる。

最後に、オントロジー評価の検討について述べる。第 6.4 章でも述べた通り、現在のオントロジー評価手法において、一般に認めらた統計的または定量的指標は存在しない。したがって、本研究で取り入れたドメインの専門家を交えた内容評価など、主観評価が中心的である。オントロジーの質を向上させるための洗練フェーズとしてこの取り組みは重要だと考えられるが、適切な手法を取らなければ有効的な評価は得られない。本研究で取り入れたリッカート指標に基づくアンケート形式だけでなく、あるシナリオを対象としたロールプレイ形式のユーザ理解度の検証など、オントロジーの評価及び改善に向けた調査指標の検証も必須である。

発表文献と研究活動

(1) **井口和真**, 塚田 学, 江崎 浩, デジタルツインの相互運用性向上に向けた動的事物を記述するオントロジーの設計, 人工知能学会第二種研究会資料, 2022, 2022 巻, SWO-058 号, p. 06-, Online ISSN 2436-5556, https://doi.org/10.11517/jsaisigtwo.2022.SWO-058_06, https://www.jstage.jst.go.jp/article/jsaisigtwo/2022/SWO-058/2022_06/_article/-char/ja

参考文献

[1] Iso - iso 16484-6:2020 - building automation and control systems (bacs) — part 6: Data communication conformance testing. https://www.iso.org/standard/79630.html. (Accessed on 01/16/2023).

[2] Iso - iso 41001:2018 - facility management — management systems — requirements with guidance for use. https://www.iso.org/standard/68021.html. (Accessed on 01/16/2023).

[3] 第 1 回 スマートビル将来ビジョン検討会 事務局提出資料. https://www.ipa.go.jp/dadc/architecture/pdf/pj_report_smartbuilding_doc-appendix_202209_1.pdf. (Accessed on 01/16/2023).

[4] 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.

[5] 岡本英之, 米澤仁, 柴田克彦, 片山健一郎. 人工知能を用いた熱源システムの運用最適化技術の開発　（第 2 報）実証試験の概要と結果. 空気調和・衛生工学会大会　学術講演論文集, Vol. 2017.2, pp. 149–152, 2017.

[6] Fei Tao, He Zhang, Ang Liu, and A. Y. C. Nee. Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 4, pp. 2405–2415, 2019.

[7]【本編】デジタルツインの社会実装に向けたロードマップ. https://info.tokyo-digitaltwin.metro.tokyo.lg.jp/docs/roadmap/roadmap_docs.pdf. (Accessed on 01/20/2022).

[8] Michael Grieves. Digital twin: Manufacturing excellence through virtual factory replication. 03 2015.

[9] Michael Grieves and John Vickers. *Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems*, pp. 85–113. Springer International Publishing, Cham, 2017.

[10] Eric J. Tuegel, Anthony R. Ingraffea, Thomas G. Eason, and S. Michael Spottswood. Reengineering aircraft structural life prediction using a digital twin. *International Journal of Aerospace Engineering*, Vol. 2011, p. 154798, Oct 2011.

[11] Edward Glaessgen and David Stargel. The digital twin paradigm for future nasa and u.s. air force vehicles. 04 2012.

[12] Yu Zheng, Sen Yang, and Huanchong Cheng. An application framework of digital twin and its case study. *J. Ambient Intell. Humaniz. Comput.*, Vol. 10, No. 3, pp. 1141–1153, March 2019.

[13] Cyber-physical systems (cps) (nsf11516). https://www.nsf.gov/pubs/2011/nsf11516/nsf11516.htm. (Accessed on 12/26/2022).

[14] Duansen Shangguan, Liping Chen, and Jianwan Ding. A hierarchical digital twin model framework for dynamic cyber-physical system design. In *Proceedings of the 5th International Conference on Mechatronics and Robotics Engineering*, ICMRE'19, p. 123–129, New York, NY, USA, 2019. Association for Computing Machinery.

[15] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda. Cyber-physical systems in manufacturing. *CIRP Annals*, Vol. 65, No. 2, pp. 621–641, 2016.

[16] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, Vol. 3, pp. 18–23, 2015.

[17] A J H Redelinghuys, A H Basson, and K Kruger. A six-layer architecture for the digital twin: a manufacturing case study implementation. *J. Intell. Manuf.*, Vol. 31, No. 6, pp. 1383–1402, August 2020.

[18] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific american*, Vol. 284, No. 5, pp. 34–43, 2001.

[19] 5-star open data. https://5stardata.info/en/. (Accessed on 12/28/2022).

[20] Calin Boje, Annie Guerriero, Sylvain Kubicki, and Yacine Rezgui. Towards a semantic construction digital twin: Directions for future research. *Autom. Constr.*, Vol. 114, p. 103179, June 2020.

[21] Sumit Singh, Essam Shehab, Nigel Higgins, Kevin Fowler, Dylan Reynolds, John A Erkoyuncu, and Peter Gadd. Data management for developing digital twin ontology model. *Proc. Inst. Mech. Eng. Pt. B: J. Eng. Manuf.*, p. 0954405420978117, December 2020.

[22] Iso - iso 19650-1:2018 - organization and digitization of information about buildings and civil engineering works, including building information modelling (bim) — information management using building information modelling — part 1: Concepts and principles. https://www.iso.org/standard/68078.html. (Accessed on 12/28/2022).

[23] Azhar Salman. Building information modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry. *Leadership and Management in Engineering*, Vol. 11, No. 3, pp. 241–252, July 2011.

[24] Industry foundation classes (ifc) - buildingsmart international. https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/. (Accessed on 12/28/2022).

[25] Mads Holten Rasmussen, Maxime Lefrançois, Georg Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 11

2020.

[26] 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, 2016.

[27] Armin Haller, Krzysztof Janowicz, Simon J. D. Cox, Maxime Lefrançois, Kerry L. Taylor, Danh Le-Phuoc, Joshua Lieberman, Raúl García-Castro, Robert Atkinson, and Claus Stadler. The sosa / ssn ontology : A joint w 3 c and ogc standard specifying the semantics of sensors , observations , actuation , and sampling. 2018.

[28] Home - web of things (wot). https://www.w3.org/WoT/. (Accessed on 01/10/2023).

[29] Web of things (wot) architecture. https://www.w3.org/TR/wot-architecture/. (Accessed on 12/28/2022).

[30] Homepage - realestatecore. https://www.realestatecore.io/. (Accessed on 01/10/2023).

[31] Shohin Aheleroff, Xun Xu, Ray Y Zhong, and Yuqian Lu. Digital twin as a service (DTaaS) in industry 4.0: An architecture reference model. *Advanced Engineering Informatics*, Vol. 47, p. 101225, January 2021.

[32] John S. Gero. Design prototypes: A knowledge representation schema for design. *AI Magazine*, Vol. 11, No. 4, p. 26, Dec. 1990.

[33] John S. Gero and Udo Kannengiesser. The situated function–behaviour–structure framework. *Design Studies*, Vol. 25, No. 4, pp. 373–391, 2004.

[34] Udo Kannengiesser and John Gero. Can pahl and beitz' systematic approach be a predictive model of designing? Vol. 3, , 12 2017.

[35] Dominic W. Massaro. *The American Journal of Psychology*, Vol. 103, No. 1, pp. 141–143, 1990.

[36] Luca Turchet, Francesco Antoniazzi, Fabio Viola, Fausto Giunchiglia, and György Fazekas. The internet of musical things ontology. *Journal of Web Semantics*, Vol. 60, p. 100548, 2020.

[37] Miriam Fernández, Chwhynny Overbeeke, Marta Sabou, and Enrico Motta. What makes a good ontology? a case-study in fine-grained knowledge reuse. In Asunción Gómez-Pérez, Yong Yu, and Ying Ding, editors, *The Semantic Web*, pp. 61–75, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[38] 太田衛, 古崎晃司, 溝口理一郎. 概念マップによるオントロジー構築・洗練支援に関する基礎的考察. 人工知能学会全国大会論文集, Vol. JSAI2010, pp. 2B11–2B11, 2010.

[39] Home - brickschema. https://brickschema.org/. (Accessed on 01/15/2023).

[40] Thomas Ainsworth. Form vs. Matter. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2020 edition, 2020.

謝辞

本論文の執筆にあたり、多くの方に御指導、御協力を頂きましたことを、ここに心より感謝を申し上げます.

指導教員である塚田学准教授には、研究の立案から内容に至る細部まで、研究への的確な御指導と御鞭撻をいただきました。また、修士 2 年間のみならず、学部時代や休学の際から親身に相談に乗っていただきました。深く感謝申し上げます。江崎浩教授には、研究内容や発表に関して常に大局的な観点から御指導いただいたほか、的確な助言や数多くのサポートを賜り、深く感謝いたします。また、研究室の発表練習や合宿の際に、様々なアドバイスやご指摘をくださった落合秀也准教授、修論の執筆に向けた進捗管理や細やかな添削まで、数多くの助言をくださった中里仁氏に深く感謝いたします。

Software Defined Media コンソーシアムの皆様、曽根卓朗氏には、研究会を通して様々な視点から御指摘や御助言をいただきました。東京大学グリーン ICT プロジェクト BIM 基盤 WG の皆様、エヌ・ティ・ティ・コミュニケーションズ株式会社の皆様には、本研究へご協力いただいたほか、実証実験を行う際に手順などを取りまとめてくださり、大変お世話になりました。学術的視点に加え、商業的観点から御意見・御指摘いただけたことは、研究生活において良い刺激となりました。

日々の研究室生活において、様々なことをご教示してくださった先輩方、研究について意見を交換し合い、刺激を与えてくれた同期一同、後輩一同に感謝いたします。また、学部から互いに励まし合い、支えてくれた研究室及び学科のかつての同期一同に感謝申し上げます。諸事務を通じて研究室生活を支えてくださった高橋富美秘書、岩井愛映子秘書に感謝いたします。

最後に、これまでの私の生活を支えてくださった家族をはじめとする全ての皆様に、心から感謝申し上げます。

付録 A

Abbreviations

本論文に記載した略語をまとめる。

DTO
: Digital Twin Ontology.

DCO
: Data Conversion Ontology.

IoT
: Internet of Things.

AI
: Artificial Intelligence.

BACS
: Building Automation and Control System.

CPS
: Cyber Physical System.

KML
: Keyhole Markup Language.

GIS
: Geographic Information System.

BIM
: Building Information Modeling.

IFC
: Industry Foundation Classes.

OWL
: Web Ontology Language.

RDF
: Resource Description Language.

W3C

World Wide Web Consortium.

URI

Uniform Resource Identifier.

LOD

Linked Open Data.

BOT

Building Topology Ontology.

SSN/SOSA

Semantic Sensor Network / Sensor, Observation, Sample, and Actuator.

OGC

Open Geospatial Consortium.

WoT

Web of Things.

REC

Real Estate Core.

FBS

Function Behavior Structure.

RPC

Remote Procedure Call.

API

Application Programming Interface.

AGV

Automatic Guided Vehicle.

OSS

Open Source Software.

付録 B

Digital Twin Ontology の語彙一覧

本論文執筆時点で最新版の Digital Twin Ontology の語彙を一覧としてまとめる。なお、DTO は、Web ドキュメント[*1]として公開されている。簡単のために 以下の表は接頭辞を用いずに記述していることに注意されたい。

B.1 クラス一覧

Action	Class
URI	`https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Action`
Comment	Eidos が有する振る舞いのうち、外部からの指示や制御を入力とするものを表現するためのクラス。Affordance のサブクラス。
Super Classes	Affordance
Sub Classes	

ActionInstance	Class
URI	`https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#ActionInstance`
Comment	ある時刻に発生した単一の Action を表すためのクラス。Instance のサブクラス。
Super Classes	Instance
Sub Classes	

[*1] `https://sdm.hongo.wide.ad.jp/~ino/dto/widoco/index-en.html`

Affordance　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Affordance
Comment	Eidos が有する、設計者が意図した動作や振る舞いを表現するためのクラス。心理学における「アフォーダンス」理論を拡張し、エンティティが環境との相互作用によって知覚できる行為から与えられる情報を意味する。
Super Classes	
Sub Classes	Action, Event, SOSAObservation

Area　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Area
Comment	物理世界または仮想世界の中で、理論的に境界付けられた 3 次元的広がりを表現するためのクラス。
Super Classes	Eidos, Space
Sub Classes	

BuildingElement　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#BuildingElement
Comment	[Alignment] see also https://w3id.org/bot#Element
Super Classes	Eidos, Element
Sub Classes	

BuildingSpace　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#BuildingSpace
Comment	[Alignment] see also https://w3id.org/bot#Space
Super Classes	Eidos, Space
Sub Classes	

Coordinates　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Coordinates
Comment	座標情報を表現するためのクラス。Property のサブクラス。
Super Classes	Property
Sub Classes	

CoordinatesInstance

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#CoordinatesInstance
Comment	ある時刻における座標情報を表現するためのクラス。Instance のサブクラスであるが、実値として付加されるのは Data クラスではなく、Data Conversion Ontology の Point クラスである。
Super Classes	Instance
Sub Classes	

CoverageZone

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#CoverageZone
Comment	ある機能を持つエンティティが、その機能により影響を与える範囲、またはカバーする空間を表現するためのクラス。例: カメラが映す範囲、HVAC ユニットがカバーするアパートとか
Super Classes	Eidos, Space
Sub Classes	

Data

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Data
Comment	単一のリテラル値で表現されるデータを表すためのクラス。参照元のパラメータを表現する Parameter クラスと関連付けられる。
Super Classes	
Sub Classes	

DataSchema

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#DataSchema
Comment	データまたはその集合を表現するためのスキーマ。エンティティの属性情報やステータス情報、または特定の動作による入出力データの照会などにおけるインターフェイスとして機能するために利用され、パラメータの集合体として表現される。
Super Classes	
Sub Classes	

Eidos　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Eidos
Comment	エンティティの仮想 (デジタル、ソフトウェア) 的モデルを表現するためのクラス。物理的構造から導かれる振る舞いや動作ではなく、設計者や観測者による解釈の上で成り立つ属性や振る舞いを有する。形而上学的用語を用いるならば、『形相』に該当する。
Super Classes	
Sub Classes	BuildingSpace, BuildingElement, Area, CoverageZone, SOSASensor, SOSAFeatureOfInterest

Entity　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Entity
Comment	実世界に物理的に存在する任意のエンティティ (実体、「モノ」など) を表現するためのクラス。それ自体に意味は持たず、物理的側面を表す Hyle とデジタルモデルを表す Eidos と関連付けられることで、個別具体的なエンティティの上位概念として機能する。形而上学的用語を用いるならば、『結合体』に該当する。なお、本オントロジーによって「物理的な存在」の形式は限定されないものとし、作成者または利用者の解釈によって変わり得るものとする。
Super Classes	
Sub Classes	VirtualEntity

Event　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Event
Comment	Eidos が有する振る舞いのうち、エンティティが自律的に実行するものを表現するためのクラス。Affordance のサブクラス。
Super Classes	Affordance
Sub Classes	

EventInstance　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#EventInstance
Comment	ある時刻に発生した単一の Event を表すためのクラス。Instance のサブクラス。
Super Classes	Instance
Sub Classes	

Geometry

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Geometry
Comment	エンティティが有する 3 次元立体情報を表現するためのクラス。なお、本オントロジーではクラスの定義にとどめ、ジオメトリやマテリアルを含む 3D モデルのデータ形式は既存のものと連携することを前提とする。
Super Classes	
Sub Classes	

Hyle

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Hyle
Comment	エンティティの物理的側面を表現するためのクラス。実世界におけるエンティティの物理的状態や属性を忠実に模する概念として機能し、センシングや測量の結果から予測・推定される物理量の集合体として記述される。形而上学的用語を用いるならば、『質料』に該当する。
Super Classes	
Sub Classes	

Instance

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Instance
Comment	ある時刻に発生した単一イベントのデータを表すためのクラス。動的 (時系列) 情報を表現するために利用され、属性情報や Entity の動作と日時情報を紐づける。例: ある時刻におけるエンティティの属性の物理量、ある時刻におこなわれたセンサーの観測の作動など
Super Classes	
Sub Classes	CoordinatesInstance, ActionInstance, EventInstance

Link

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Link
Comment	Eidos が有する API を表現するためのクラス。例: Affordance の入出力データを送受信するための Web API。
Super Classes	
Sub Classes	RestApi

Parameter

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Parameter
Comment	DataSchema の構成に必要なパラメータ。
Super Classes	
Sub Classes	

Point　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Point
Comment	
Super Classes	
Sub Classes	

Property　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#Property
Comment	属性を表現するためのクラス。原則として、Hyle に紐付けられる場合は物理的属性情報を、Eidos に紐付けられる場合は仮想的 (ソフトウェア上で用いられる) 属性情報として定義される。類似概念として、Semantic Sensor Network オントロジーの Property クラスや、Web of Things における Property 概念など。
Super Classes	
Sub Classes	Coordinates, SOSAObservableProperety

RestApi　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#RestApi
Comment	Eidos が有する API のうち、REST 形式の Web API を表現するためのクラス。Link のサブクラス。
Super Classes	Link
Sub Classes	

SOSAFeatureOfInterest　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#SOSAFeatureOfInterest
Comment	[Alignment] see also http://www.w3.org/ns/sosa/FeatureOfInterest
Super Classes	bot:FeatureOfInterest, Eidos
Sub Classes	

SOSAObservableProperety　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#SOSAObservableProperety
Comment	[Alignment] see also http://www.w3.org/ns/sosa/ObservableProperty
Super Classes	bot:ObservableProperty, Property
Sub Classes	

SOSAObservation Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#SOSAObservation
Comment	[Alignment] see also http://www.w3.org/ns/sosa/Observation
Super Classes	bot:Observation, Affordance
Sub Classes	

SOSASensor Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#SOSASensor
Comment	[Alignment] see also http://www.w3.org/ns/sosa/Sensor
Super Classes	bot:Sensor, Eidos
Sub Classes	

VirtualEntity Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#VirtualEntity
Comment	物理的に存在しないが、設計者によってその存在を仮定されるエンティティを表現するためのクラス。Entity のサブクラス。個別具体的なエンティティの上位概念として機能するが、物理的側面である Hyle は持たない。例: ある機能的要件で定められた空間エリアや、1 つのユニットとして扱われるグループ化された集合体などなお、本オントロジーによって「物理的な存在」の形式は限定されないものとし、作成者または利用者の解釈によって変わり得るものとする。
Super Classes	Entity
Sub Classes	

B.2　オブジェクトプロパティ一覧

botContainsElement　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#botContainsElement
Comment	[Alignment] see also https://w3id.org/bot#containsElement
Domain	BuildingSpace
Range	BuildingElement
Inverse	

botHasSpace　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#botHasSpace
Comment	[Alignment] see also https://w3id.org/bot#hasSpace
Domain	BuildingSpace
Range	Area
Inverse	

capturedBy　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#capturedBy
Comment	Property を観測する Affordance を表現するためのプロパティ。リンク先の Affordance からは captures で逆方向リンクを表現できる。
Domain	
Range	
Inverse	ontology.captures

captures　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#captures
Comment	Affordance が観測する Property を表現するためのプロパティ。リンク先の Property からは capturedBy で逆方向リンクを表現できる。
Domain	Affordance
Range	Property
Inverse	ontology.capturedBy

composedOf — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#composedOf
Comment	Hyle に包含されている Hyle を表現するためのプロパティ。
Domain	Hyle
Range	Hyle
Inverse	ontology.constitutes

constitutes — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#constitutes
Comment	自身を包含する Hyle を表現するためのプロパティ。
Domain	
Range	
Inverse	ontology.composedOf

eidosOf — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#eidosOf
Comment	Eidos が属する Entity を表現するためのプロパティ。リンク先の Entity からは hasEidos で逆方向リンクを表現できる。
Domain	
Range	
Inverse	ontology.hasEidos

execute — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#execute
Comment	自身が実行する Instance を表現するためのプロパティ。
Domain	
Range	Instance
Inverse	ontology.executedBy

executedBy — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#executedBy
Comment	Instance の実行主体を表現するためのプロパティ。
Domain	
Range	
Inverse	ontology.execute

hasAction　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasAction
Comment	Eidos による実行を設計された Action を表現するためのプロパティ。
Domain	Eidos
Range	Action
Inverse	

hasData　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasData
Comment	Instance に付随する Data を表現するためのプロパティ。
Domain	Instance
Range	Data
Inverse	

hasEidos　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasEidos
Comment	Entity に付随する Eidos を表現するためのプロパティ。リンク先の Eidos からは eidosOf で逆方向リンクを表現できる。
Domain	Entity — VirtualEntity
Range	Eidos
Inverse	ontology.eidosOf

hasEvent　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasEvent
Comment	Eidos による実行を設計された Event を表現するためのプロパティ。
Domain	Eidos
Range	Event
Inverse	

hasHyle　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasHyle
Comment	Entity に付随する Hyle を表現するためのプロパティ。リンク先の Hyle からは hyleOf で逆方向リンクを表現できる。
Domain	Entity
Range	Hyle
Inverse	ontology.hyleOf

hasInputData

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasInputData
Comment	Affordance の実行に要求される入力の DataSchema を表現するためのプロパティ。
Domain	Affordance
Range	DataSchema
Inverse	

hasOutputData

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasOutputData
Comment	Affordance の実行により出力される DataSchema を表現するためのプロパティ。
Domain	Affordance
Range	DataSchema
Inverse	

hasParameter

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasParameter
Comment	DataSchema に付随する Parameter を表現するためのプロパティ。
Domain	DataSchema
Range	Parameter
Inverse	

hasPoint

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasPoint
Comment	CoordinatesInstance が有する座標情報 (dco:Point) を表現するためのプロパティ。
Domain	CoordinatesInstance
Range	Point
Inverse	

hasProperty　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasProperty
Comment	Hyle や Eidos が有する Property を表現するためのプロパティ。原則として、Hyle に紐付けられる場合は (観測により導出された予測値として) 物理的属性情報を、Eidos に紐付けられる場合は仮想的 (ソフトウェア上で用いられる) 属性情報として定義される。
Domain	Eidos — Hyle
Range	Property
Inverse	

hasSchema　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasSchema
Comment	Property が準拠する Dataschema を表現するためのプロパティ。
Domain	Property
Range	DataSchema
Inverse	

hasVolume　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasVolume
Comment	Eidos に付随する Geometry を表現するためのプロパティ。
Domain	Eidos
Range	Geometry
Inverse	

hyleOf　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hyleOf
Comment	Hyle が属する Entity を表現するためのプロパティ。リンク先の Entity からは hasHyle で逆方向リンクを表現できる。
Domain	
Range	
Inverse	hasHyle

listen　Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#listen
Comment	自身に付随する Link を表現するためのプロパティ。
Domain	Link
Range	
Inverse	

locatedAt — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#locatedAt
Comment	Hyle が位置する Coordinates を表現するためのプロパティ。hasProperty のサブプロパティ。
Domain	Hyle
Range	Coordinates
Inverse	

locatedIn — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#locatedIn
Comment	Eidos が位置している空間を表現するためのプロパティ。
Domain	Eidos
Range	bot:Space
Inverse	

references — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#references
Comment	Data が表す Parameter を表現するためのプロパティ。
Domain	Data
Range	Parameter
Inverse	

serves — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#serves
Comment	ある機能を持つ Hyle がカバーする CoverageZone を表現するためのプロパティ。
Domain	Hyle
Range	CoverageZone
Inverse	

sosaMadeObservation — Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#sosaMadeObservation
Comment	[Alignment] see also http://www.w3.org/ns/sosa/madeObservation
Domain	SOSASensor
Range	SOSAObservation
Inverse	

sosaObservedProperty	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#sosaObservedProperty
Comment	[Alignment] see also http://www.w3.org/ns/sosa/observedProperty
Domain	SOSAObservation
Range	SOSAObservableProperety
Inverse	

ssnHasProperty	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#ssnHasProperty
Comment	[Alignment] see also http://www.w3.org/ns/ssn/hasProperty
Domain	SOSAFeatureOfInterest
Range	SOSAObservableProperety
Inverse	

B.3 データ型プロパティ一覧

hasDescription — Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasDescription
Comment	自身が有する説明文を表現するためのプロパティ
Domain	
Range	str

hasEndpoint — Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasEndpoint
Comment	Link に付随する API エンドポイントを表現するためのプロパティ。
Domain	Link
Range	str

hasName — Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasName
Comment	自身が有する名前を表現するためのプロパティ。
Domain	
Range	str

hasProtocol — Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasProtocol
Comment	Link にアクセスするための通信プロトコルを表現するためのプロパティ。 例: ‘http’、‘https’、‘mqtt’
Domain	Link
Range	str

hasSimpleVolume　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasSimpleVolume
Comment	Eidos に付随する Geometry を表現するためのプロパティ。hasVolume プロパティと異なり、3D データへの URI を直接示す。
Domain	Eidos
Range	str

hasTimestamp　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasTimestamp
Comment	Instance に付随する時間情報を表現するためのプロパティ。
Domain	Instance
Range	datetime

hasValue　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#hasValue
Comment	Data に付随するリテラルな値を表現するためのプロパティ。
Domain	Data
Range	

httpMethod　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#httpMethod
Comment	RestApi にリクエストするための HTTP リクエストプロトコルを表現するためのプロパティ。例: ‘GET’、‘POST’
Domain	RestApi
Range	str

isRequiredParameter　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dto/ontology.rdf#isRequiredParameter
Comment	主となる Parameter が、DataSchema で必須となるパラメータか否かを表現するためのプロパティ。
Domain	Parameter
Range	bool

付録 C

Data Conversion Ontology の語彙一覧

本論文執筆時点で最新版の Data Conversion Ontology の語彙を一覧としてまとめる。なお、DCO は、Web ドキュメント[*1]として公開されている。簡単のために 以下の表は接頭辞を用いずに記述していることに注意されたい。

C.1 クラス一覧

Axis	Class
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Axis
Comment	座標軸を表現するためのクラス。
Super Classes	
Sub Classes	

Conversion	Class
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Conversion
Comment	System 間の変換を表現するためのクラス。
Super Classes	
Sub Classes	CoordTransform

[*1] https://sdm.hongo.wide.ad.jp/~ino/dco/widoco/index-en.html

CoordTransform　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#CoordTransform
Comment	座標変換を表現するためのクラス。
Super Classes	Conversion
Sub Classes	

CoordinatesSystem　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#CoordinatesSystem
Comment	座標系を表現するためのクラス。現行バージョンでは、直交直線座標系に限定される。System のサブクラス。
Super Classes	System
Sub Classes	

Matrix4　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Matrix4
Comment	4 × 4 行列を表現するためのクラス。
Super Classes	
Sub Classes	

Point　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Point
Comment	座標点を表現するためのクラス。
Super Classes	
Sub Classes	

Rotate　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Rotate
Comment	回転を表現するためのクラス。
Super Classes	gRPCTransaction
Sub Classes	

Scale3D　Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Scale3D
Comment	拡大縮小を表現するためのクラス。
Super Classes	gRPCTransaction
Sub Classes	

System

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#System
Comment	'系' を表現するためのクラス。例: 単位やデータ型など
Super Classes	
Sub Classes	CoordinatesSystem

Transaction

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Transaction
Comment	変換プロセスで実行される個別の処理を表現するためのクラス。現行バージョンでは、出力関数の記述は外部に委譲している。
Super Classes	
Sub Classes	gRPCTransaction

Translate

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Translate
Comment	並進移動を表現するためのクラス。
Super Classes	gRPCTransaction
Sub Classes	

Vector3

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#Vector3
Comment	3 次元ベクトルを表現するためのクラス。
Super Classes	
Sub Classes	

gRPCTransaction

Class

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#gRPCTransaction
Comment	gRPC に特化した表現をおこなうための Transaaction のサブクラス。
Super Classes	Transaction
Sub Classes	Rotate, Scale3D, Translate

C.2　オブジェクトプロパティ一覧

baseAxis	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#baseAxis
Comment	CoordinatesSystem が有する Axis を表現するためのプロパティ。
Domain	CoordinatesSystem
Range	Axis
Inverse	

executes	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#executes
Comment	Conversion がシーケンスとして有する Transaction を表現するためのプロパティ。
Domain	Conversion
Range	Transaction
Inverse	

gRPCMessageBody	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#gRPCMessageBody
Comment	gRPC 用のメッセージボディを表現するためのプロパティ。
Domain	gRPCTransaction
Range	Matrix4
Inverse	

hasVector3	Object Property
URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#hasVector3
Comment	Vector3 の付随を表現するためのプロパティ。
Domain	Axis — Point
Range	Vector3
Inverse	

refersCS

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#refersCS
Comment	Point が参照する CoordinatesSystem を表現するためのプロパティ。
Domain	Point
Range	CoordinatesSystem
Inverse	

sourceSystem

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#sourceSystem
Comment	変換元となる系を表現するためのプロパティ。
Domain	Conversion
Range	System
Inverse	

targetSystem

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#targetSystem
Comment	変換先となる系を表現するためのプロパティ。
Domain	Conversion
Range	System
Inverse	

unit

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#unit
Comment	自身が有する単位系を表現するためのクラス。
Domain	CoordinatesSystem
Range	obo:UO_0000001
Inverse	

upAxis

Object Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#upAxis
Comment	自身が持つ Axis の内、高さ方向と一致する軸を表現するためのプロパティ。なお、本オントロジーは屋内空間を対象とするため、「高さ」の形式は限定されないものとし、作成者または利用者の解釈によって変わり得るものとする。
Domain	CoordinatesSystem
Range	Axis
Inverse	

C.3　データ型プロパティ一覧

abbrev　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#abbrev
Comment	Axis の略称を指定するためのプロパティ。
Domain	Axis
Range	str

element　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#element
Comment	Vector3 の成分表示を表現するためのプロパティ。現行バージョンでは csv 形式を想定している。
Domain	Matrix4 — Vector3
Range	str

elementOrder　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#elementOrder
Comment	各成分における Axis の順序を指定するためのプロパティ。
Domain	Axis — Point
Range	str

gRPCMethod　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#gRPCMethod
Comment	gRPC サーバの実行メソッドを指定するためのプロパティ。
Domain	gRPCTransaction
Range	str

gRPCServerAddress

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#gRPCServerAddress
Comment	gRPC サーバの実行アドレスを指定するためのプロパティ。
Domain	gRPCTransaction
Range	str

gRPCService

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#gRPCService
Comment	gRPC サーバの実行サービスを指定するためのプロパティ。
Domain	gRPCTransaction
Range	str

handedCS

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#handedCS
Comment	右手系/左手系を指定するためのプロパティ。
Domain	CoordinatesSystem
Range	str

hasName

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#hasName
Comment	自身が有する名前を表現するためのプロパティ。
Domain	
Range	str

messageBody

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#messageBody
Comment	出力関数を記述する外部のメッセージを指定するためのプロパティ。
Domain	Transaction
Range	str

method

Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#method
Comment	出力関数を記述する外部のメソッド名を指定するためのプロパティ。
Domain	Transaction
Range	str

sequenceNum　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#sequenceNum
Comment	Transaction の実行順序を指定するためのプロパティ。なお、本オントロジーでは実行順序の形式は限定しない。
Domain	Transaction
Range	str

serverAddress　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#serverAddress
Comment	出力関数を記述する外部を指定するためのプロパティ。
Domain	Transaction
Range	str

service　Data Property

URI	https://sdm.hongo.wide.ad.jp/~ino/dco/ontology.rdf#service
Comment	出力関数を記述する外部のサービス名を指定するためのプロパティ。
Domain	Transaction
Range	str