修士論文

ホームネットワークにおけるメタデータに関する研究

1710029　YUAN,Shuai

主指導教員　　丹　康雄　教授
審査委員主査　　丹　康雄　教授
審査委員　　リム　勇仁　准教授
篠田 陽一　教授
長谷川 忍　准教授

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

平成 31 年 2 月

概　要

メタデータは、IoT 世界において非常に重要な役割を果たしている。メタデータを利用することより、データの価値を最大化にすることが可能となり、異分野データの連携と統合に促進の一助となる。今までのデータは単一業界・単一目的で閉じた垂直方向に収集されてきたが、複数の異なる目的で収集されたデータを水平方向に統合することによってビジネス価値を創出すること等の目的で、メタデータを利用し、データ流通社会を構築するといった取り組みが期待されている。本論文は IoT の代表的な分野であるスマートホームにおいてメタデータを利活用し、ホームネットワーク向け既に存在する各プラットフォームを連携させ、プラットフォーム同士が、相互に理解できるデータカタログのあり方について提案した。具体的には、ホームネットワークにおいてメタデータの記述項目、記述方式、メタデータデータ検索システムのあり方について具体的な例を与えるものとなる。

IoT 技術は、その急速な発展に伴い、製造業や医療、農業から我々の生活に至るまで様々な分野で活用され、生成されるデータの量と種類が急速に増加している。これらの分野において収集・蓄積されたデータに対してデータマイニングやディープラーニング等の技術を活用することで、新たな価値の創出や社会課題解決が求められている。しかし現状では、業界毎にデータの用語の意味や連携するためのプロトコル等の差異が存在しており、データ利用者が欲しいデータと、データ提供者が提供可能なデータにもギャップがある。そのため、分野の枠を超えた様々な業界が共通で理解できるデータカタログを構築する必要性が生じつつある。

データの量が急速に増加を背景に、データを流通させたいと考える事業者が現れている。データ流通事業者が適切かつ安全なデータを提供でき、またデータ利用者が欲するデータを容易に判断して収集・活用できる技術的・制度的環境を整備すること等の目的で、メタデータを付与して利用する動きが高まっている。

IoT の代表的なスマートホーム分野においても機器や業界ごとにデータの形式や連携方法も差異がある。サービス提供事業者が利用するに十分な環境の整備を図るため、様々な機器メーカーやサービス提供事業者が共通で理解出来るデータカタログを構築するといった取り組みが期待されている。

また、ホームネットワークのアーキテクチャの構成も変化しつつある。従来のホームネットワークアーキテクチャを構成する主な要素には、デバイス、ゲートウェイ、IoT プラットフォーム、サービス事業者がある。四階層方式と呼ばれる、家中のデバイスを IoT ゲートウェイを経由して IoT プラットフォームに接続し、サービス事業者と通信を行う方式が一般的だった。しかし、近年 IoT デバイスの進化により、デバイスは IoT ゲートウェイを経由せず、直接インターネットを経由して IoT プラットフォームに接続し、サービス事業者と通信を行うという三階層方式と、デバイスからメーカー・ユーザクラウドを経由して、サービス事業者と通信を行う二階層方式も現れた。

ホームネットワークの現状では、多種類のホームネットワークアーキテクチャが混在しており、一つの家においても異なる技術、異なる目的で設置されたネットワークが複数存在する。しかもそれらは独立なネットワークである。また、第三者がそのネットワーク内のセンサー・機器のデータを使用することができない。

また、センサーそのものの小型化や低価格化、低消費電力化や、それを受けて家電に組み込まれたものを含め、センサーとして利用可能な機器の数は爆発的に増えている。そのためメタデータを利用し、多くのセンサーの中から利用目的に最も適したセンサーのデータを選択するしくみが求められている。

そこで本研究は ECHONET 規格や SSN 等の標準化されているデータモデルを参考し、BCADFL というメタデータモデルを提案した。そしてオントロジーを用いてデータに語彙を付与し、同じ形式にマッピングする手法を提案した。

また、OneM2M 汎用セマンティックモデルを参考し、SPARQL というクエリ言語を利用し、メタデータを検索できるシステムを提案した。サービス提供事業者がそのシステムを使い、利用目的に最も適したメタデータを検索することができると考える。

提案したメタデータの活用手法により、業界やプラットフォームの壁を越え、住宅から収集した適切なデータをスムーズにサービス提供事業者に伝送し、ユーザーにより良いサービスが実現できるようにする。家庭内様々な機器からの情報もセンシングデータとして利用可能となる。また、サービスプラットフォームを運営するデータ流通事業者にとってもこれらのデータが二次利用できるようになり、データ流通市場が活性化することも期待できる。

目次

図目次

表目次

第1章 はじめに

本章では、研究背景と研究目的、本論文の構成を示す。

1.1 研究背景

IoT 技術の急速な発展に伴い、製造業や医療、農業から我々の生活に至るまで様々な分野で活用され、生成されるデータの量と種類が急速に増加している。これらの分野において収集・蓄積されたデータに対してデータマイニングやディープラーニング等の技術を活用することで、新たな価値の創出や社会課題解決が求められている。しかし現状では、データ利用者が欲しいデータと、データ提供者が提供可能なデータに差異がある。そのため、分野の枠を超えた様々な業界が共通で理解できるデータカタログを構築する必要性が生じつつある。 [1]

データ流通量の急速な増加を背景に、近年データを流通させたいと考える事業者が多数出現している。データ流通事業者が適切かつ安全なデータを提供でき、またデータ利用者が欲するデータを容易に判断して収集・活用できる技術的・制度的環境を整備すること等を目的として、データにメタデータを付与する動きが高まっている。

IoT の代表的な分野であるスマートホーム分野においても機器や業界ごとにデータの形式や連携方法に差異がある。サービス提供事業者が利用するに十分な環境の整備を図るため、様々な機器メーカーやサービス提供事業者が共通で理解出来るデータカタログを構築するといった取り組みが期待されている。 [1] [2]

また、センサーそのものの小型化や低価格化、低消費電力化の実現により家電に組み込まれたものを含め、センサーとして利用可能な機器の数は爆発的に増えている。例えば、室内温度の場合、エアコンで測定する温度とセンサーで測定する温度が存在し、測定する位置も様々である。そのためメタデータを利用し、多くのセンサーの中から利用目的に最も適したセンサーを選択するしくみが求められている。

1.2 研究目的

本研究の目的は、ホームネットワーク向けに、既存の各ネットワークを連携させ、閉じられたネットワーク範囲を超えて、相互に理解できるデータカタログを作成することである。具体的には、ホームネットワークにおいてメタデータの記述項目、記述方式、メタデータの生成・提供方式、メタデータデータ検索システムのあり方について具体的な例を与えるものとなる。

ホームネットワーク内においてメタデータカタログを活用することにより、業界やプラットフォームの壁を越え、住宅から収集した適切なデータをスムーズにサービス提供事業者に伝送し、ユーザーにより良いサービスが実現できるようにする。家庭内の様々な機器の情

報もセンシングデータとして利用可能となる。また、サービスプラットフォームを運営するデータ流通事業者にとってもこれらのデータが二次利用できるようになり、データ流通市場が活性化も期待できる。

1.3 論文構成

本論文では、本章を含めて 7 章で構成される。
1 章では背景と目的として、メタデータの重要性について述べる。
2 章では関連研究について述べる。
3 章ではホームネットワークにおける有用性について述べる。
4 章では提案したメタデータモデル、オントロジー、検索システムについて述べる。
5 章ではシステムの実装について述べる。
6 章では考察を行う。
7 章では本論文のまとめを行う。

第2章 関連研究

本章では、本研究の提案に関連する技術について述べる。

2.1 メタデータ

メタデータは一般にデータについてのデータと定義される。あるデータのそのものではなく、データに関する情報を記述したものである。メタデータを利用することで、様々な分野で蓄積された膨大なデータに対し、効率よく管理したり収集したりすることが可能となる。

2.1.1 メタデータの必要性

ホームネットワークにおいても発生したデータはインターネット等でやり取りされる、その検索、マッチング、配信等の機能を行うため、人間が読めるだけではなく、機械が自動的に読み込んで処理できるデータ形式とする必要がある。また、複数のデータ提供者から入手したデータの仕様や所得方法等の差異が存在するため、メタデータを付与して記述する必要性が生じつつある。

メタデータの主な機能は次の通りである。

① IoT 時代を支える大量の異種データを整備・統一することができる。
② 必要な情報を迅速に見つけることができる。
③ 異なるシステム間でデータ連携することができる。

2.1.2 メタデータ記述言語 RDF

RDF は W3C により 1992 年 2 月に規格化されたメタデータの記述言語であり、Web リソースを記述するための共通のメタデータモデルを提供する。

RDF では、(主語、述語、目的語) の三つ組で、情報の意味を記述する。例えば、“この論文の著者は田中さんである”とする。トリプルで表すと

Triple(author、thesis、tanaka)

図で表すと、

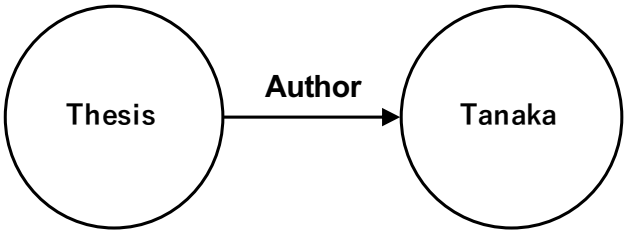


図 2-1 RDF トリプル

XML でこれらの情報を記述すると、

<document>

<author>

<uri>href="thesis"</uri>

<details> <name>tanaka</name>

</details>

</author>

</document>

<document href="http://www.w3.org/test/thesis" author="tanaka" />

[3] [4]

RDF は従来の XML に比べると、情報の属性を記述することだけでなく、他の情報との関連性にも表すことで、WEB 上の様なリソースに意味を付与して、データ処理を容易化する。

2.2 オントロジー

オントロジーとは本来哲学用語であり「存在に関する体系的な理論」を意味するものである。情報科学の分野では、「概念化の明示的な記述」と定義される。概念化は、記述者が対象世界を抽象化して記述する際に「情報記述者が対象世界に存在すると考えたものとそれらの関係」を指す。要は複数の対象や知識に共有される要素を概念化したものである。

2.2.1 オントロジーの必要性

本論文では、ホームネットワークにおける大量のデータの関連性を記述する語彙体系を構築するため、オントロジーを利用する。

オントロジーの主な機能が次の通りである。

一般性：　対象世界の全てのものを概念化にしてモデルにすることができる。

目的依存性：対象のモデルを記述する際に、全てのものを記述することができないため、明確な目的を持ち、有効なモデル構築する。

形式性：　記述した概念や則約は人間だけではなく、計算機も理解可能な形式として記述する。

共通性：　同じ対象に関する様々なエージェント間で共通できる概念や則約を与える。

2.2.2 オントロジー記述言語 OWL

OWL(Web Ontology Language)は、2004 年に W3C のワーキンググループにより開発された。オントロジー記述言語としてよく使われている。OWL は RDF の語彙拡張であり、プロパティやクラスの関係や則約の記述に対してより多くの語彙を追加したものとなる。

セマンティックウェブでは、OWL は一番使用されているオントロジー言語として、一番上のレイヤーに存在する。

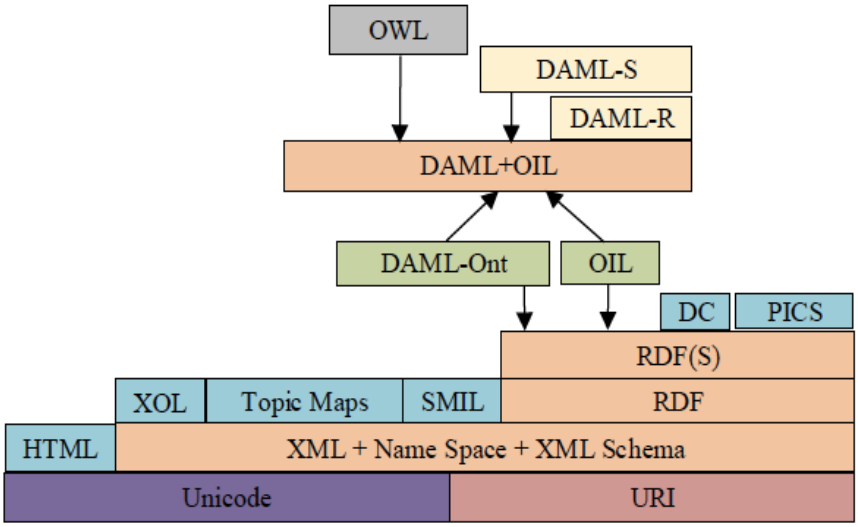


図 2-2　Semantic Web の言語レイヤー

2.3 オントロジーとメタデータの関連性

オントロジーはメタデータを使用しプロパティを定義する。そのためオントロジーを使用することでメタデータの意味的な整合性をチェックすることができる。例えば、「生産地」というメタデータのプロパティの値が「地名」クラスかどうかを問い合わせて、メタデータの整合性をチェックする。また、オントロジーはメタデータデータベースに対する問い合わせにも使用されている。オントロジーを利用することでデータを構造化されているため、決められたクラスとプロパティを利用し、より多く、複雑な問い合わせが可能となり、データベースを高速化にする。 [3]

2.4 Jena

Jena は、HP Labs が 2001 年に開発して一般公開しているセマンティック Web Application を作成するための Java フレームワークであり、以下に記載されたのは Jena メインの機能リストである。 [5]

- RDF ファイルの作成、操作、追加などの RDF Interface を提供する。
- オントロジーの作成、操作、永続化の Ontology API を提供する。
- RDF/OWL ファイルを TDB に保存する API を提供する。
- RDF/OWL ファイルを問い合わせるための SPARQL クエリエンジンも提供する。

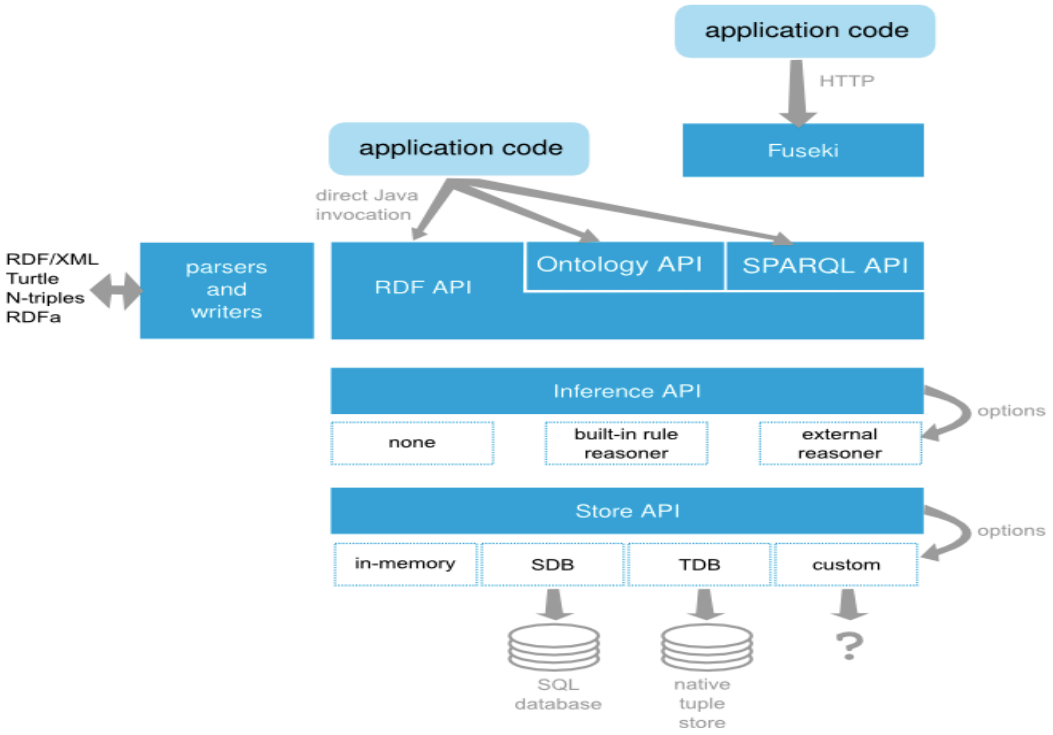


図 2-3 Jena architecture overview

2.5 OneM2M セマンティックアーキテクチャ

OneM2M [6]とは世界中 7 つの標準化団体が合意した共同プロジェクトで、M2M のためのサービスレイヤーの標準化を行う組織である。M2M 規格書「TR-0007-v1.0.0 Study of Abstraction and Semantics Enablements」では、One M2M アーキテクチャにセマンティック技術を導入する課題について詳しく紹介された。 本研究で提案した手法もこのアーキテクチャに参考したものである。

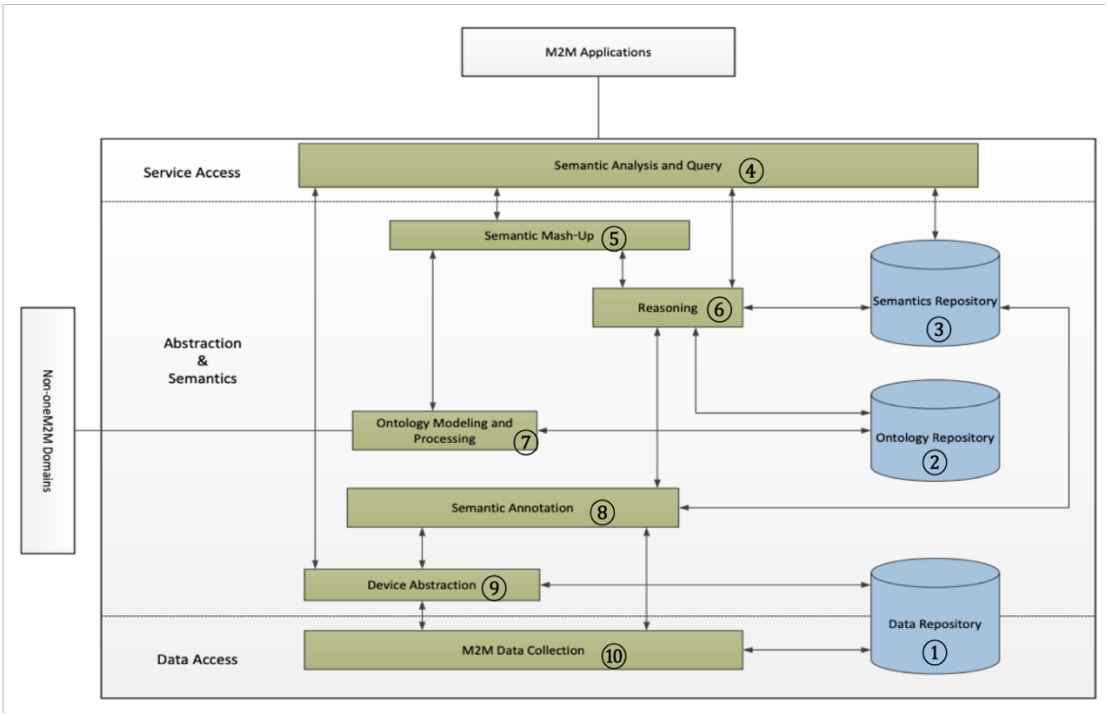


図 2-4 OneM2M 汎用セマンティックモデル

OneM2M から提唱された汎用セマンティックモデルは 3 層構造であり、デバイスやゲットウェイからデータ送信を確立する Data Access layer(データアクセス層)、データの処理を行う Abstraction & Semantic layer(データ抽象化＆語彙化層)、ユーザーAPI を提供する Service Access layer(サービスアクセス層)から構成されている。

① Data Repository ：生データの保管、検索、変更、削除等の機能を提供する。
② Ontology Repository：記述したデータを保存、検索、管理するための方法を提供する。記述言語は OWL あるいは RDF で記述し、クエリ言語は RDQL、QWL―QL、SPARQL を使用する。
③ Semantic Repository：データの意味を付与するためのオントロジーモデルを保管する。
④ Semantic Analysis and Query ：M2M アプリケーションから要求を解析し、セマンティッククエリを作成し、それをオントロジーライブラリに送信して要求された情報を M2M アプリケーションに転送する。
⑤ Semantic Mash-up：関連性があるデータを統合し、仮想データを生成する。
⑥ Reasoning：決められた概念や則約によりデータに関する潜在的な意味、関連性を推論して引き出す。
⑦ Ontology Modeling and processing：概念や則約等を保管する。
⑧ Semantic Annotation：データに意味を付与し、語彙化にする。
⑨ Device Abstraction：デバイスを抽象化するためのモデルを保管する。
⑩ Data access：デバイスやゲットウェイから生データを取得する。

第3章 ホームネットワークにおけるデータカタログの必要性

本章では、ホームネットワークアーキテクチャ構造変化の視点から、データカタログの必要性について述べる。

3.1 ホームネットワークアーキテクチャ

従来のホームネットワークアーキテクチャを構成する主な要素には、デバイス、ゲートウェイ、IoT プラットフォーム、サービス事業者がある。その中で、四階層方式と呼ばれる、家中のデバイスを IoT ゲートウェイを経由して IoT プラットフォームに接続させて、サービス事業者と通信を行う方式が一般的だった。

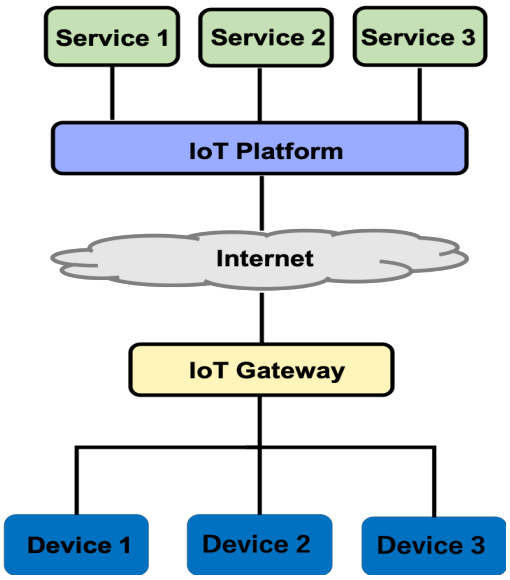


図 3-1　四階層モデル

しかし近年、IoT デバイスの進化により、WI-FI、4G などネットワークに接続できるデバイスが多数現れている。そのため、ホームネットワークのアーキテクチャ構造も変化しつつある。例えば、デバイスは IoT ゲートウェイを経由せず、インターネットを経由して IoT プラットフォームに接続し、サービス事業者と通信を行うという三階層方式と、デバイスからメーカー・ユーザクラウドに経由して、サービス事業者と通信を行う二階層方式も現れた。

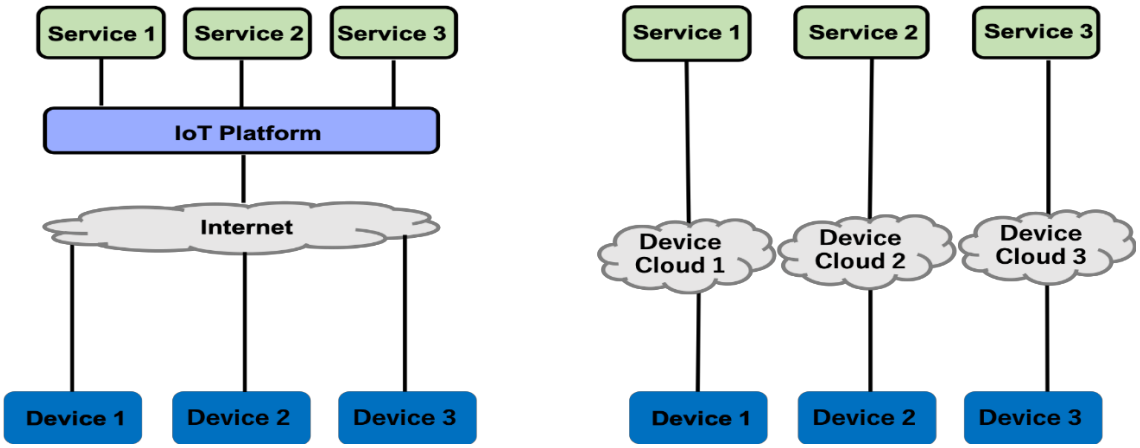


図 3-2 三階層モデルと二階層モデル

現状では、図に示した全てのホームネットワークアーキテクチャは混在しており、一つのホームネットワークにおいても異なる技術、異なる目的設置されたネットワークが複数存在する。しかもいずれも独立なネットワークである。第三者がそのネットワーク内のセンサー・機器のデータを使用することができない。

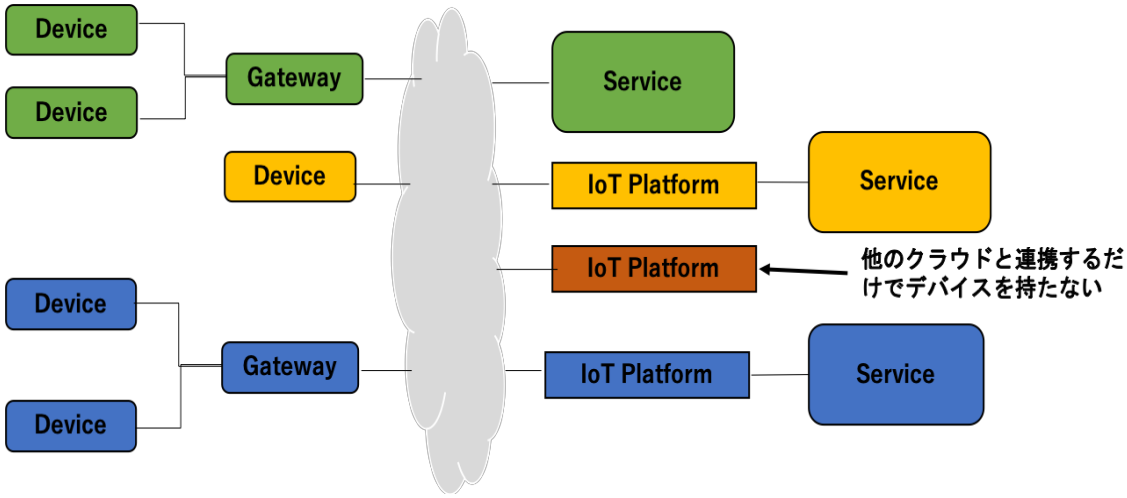


図 3-3 ホームネットワークアーキテクチャの全体図

また、ネットワーク毎にもデータの記述や詳細度に差異があり、サービス提供事業者が欲しいデータと、各ネットワークから提供されたデータにギャップが存在する。これらの不均質で膨大なデータを使用するためには、統一なデータ連携の考え方を検討する必要性も現れている。

3.2 データカタログのあり方

本来、「データカタログ」の定義は、データそのものを一覧にしたものではなく、データ

の分類、略形式等を検索するためのメタデータをデータの種類毎にまとめたもの。ホームネットワークデータカタログとはホームネットワークにおいてサービス事業者が関心のある対象(人・家・地域など)に関するデータ種類の一覧とそのデータ属性を確認するために集約したものである。

それぞれのネットワークがデータ連携プラットフォームに対し、共通のデータカタログを設計する。そこでデータを統一な形式にマッピングする。マッピングされたデータは多数有用とされるメタデータを付与されており、サービス事業者が提供された API を通じて欲しいメタデータを取得できる仕組みが期待されている。

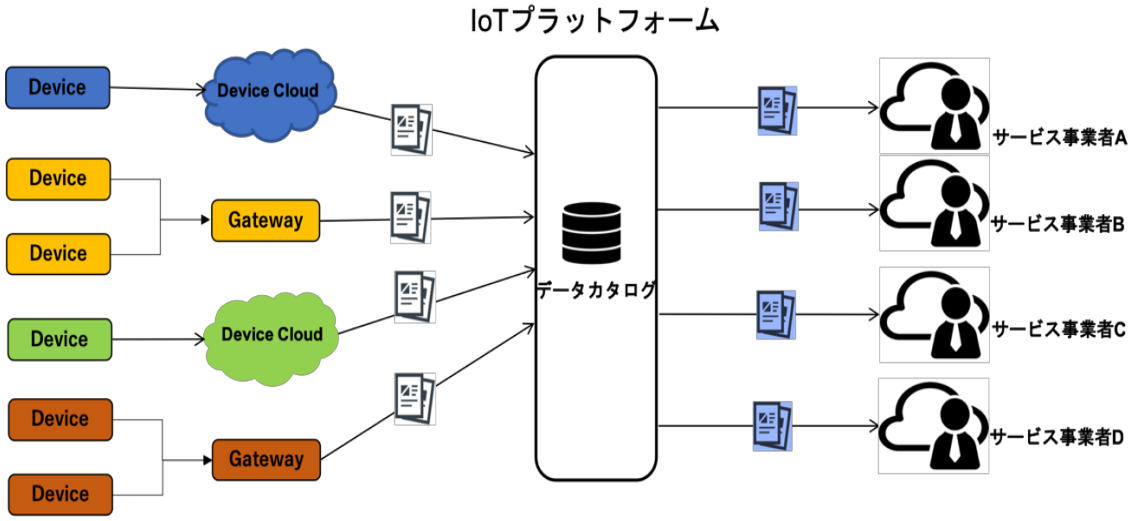


図 3-4　ホームネットワークデータ連携の姿

第4章 提案手法

本章では、2.4 章 OneM2M 汎用セマンティックモデルを踏まえ、ホームネットワークにおいてメタデータの記述項目、メタデータの付与方式、データ検索システムのあり方について提案する。

4.1 ホームネットワークにおけるデータ記述項目の定義

現在、ホームネットワークにおいて様々なネットワークが存在する。これらのネットワークにおいて収集された大量のデータを発生されており、データマイニングやディープラーニング等の技術を活用することで、新たな価値の創出や社会課題解決が求められている。しかし、それぞれのネットワークが独自の定義でデータの形式を決めており、異なるネットワーク間のデータ連携が課題となる。ホームネットワーク内全てのデータを連携するためには、データ記述項目を定義しなければならない。本研究で提案した記述項目は、全て現在標準として使用されるものから整理したものである。

各プラットフォームを連携するためには、データの提供・活用事業者等が共通で理解できるデータカタログの整備は不可欠である。データカタログを構成されるメタデータ記述項目について、ある程度標準化されているデータモデルを参考することが望ましいとされている。本論文では、標準として利用されて ECHONET Lite 規格書 [7]のオブジェクトを参考にメタデータ記述項目を決定した。また W3C が提案したセンシングデータ規格 SSN(Semantic Sensor Network) [8]、SAREF、OneM2M 等の規格に基づいて、メタデータを共通の枠組みで利用できるようなデバイスモデルを構築した。この 4 つの規格は IoT プラットフォームの運用において基本となる記述項目を提供しており、サービス提供事業者の視点からも基本なサービス提供するため必要なメタデータを充実させると考える。また項目の定義において、柔軟性が特徴である。

デバイスモデルを表現するメタデータモデルは、基本情報(BasicInformation)、通信能力(CummunicationCapability)、AoE 、データ(Data)、機能(Function)、位置情報(Location)を表現するメタデータモデルの 6 つが含まれる。

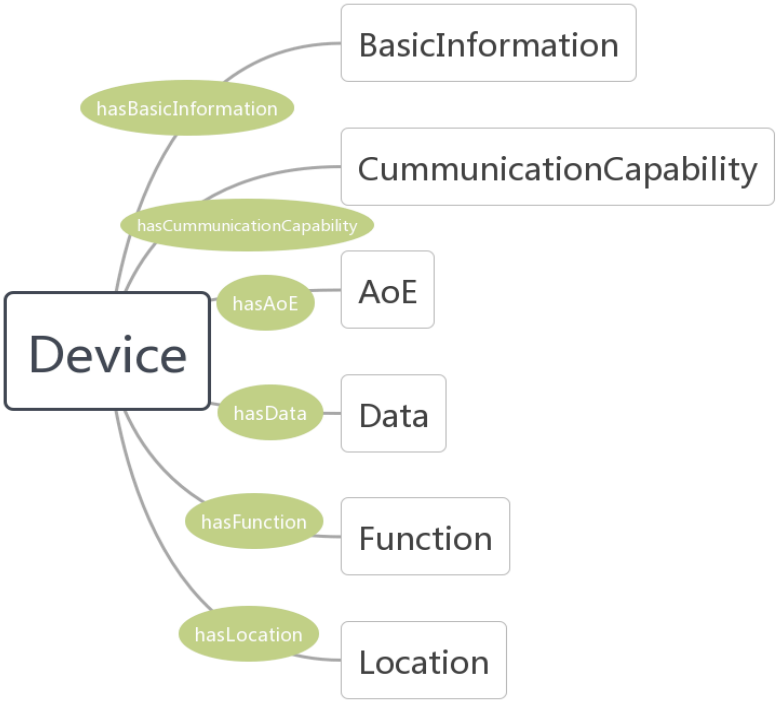


図 4-1 提案メタデータモデルの全体図

(ア) 基本情報モデル：図 4-2 に基本情報に関するメタデータモデルの構造を示す。図に示すように、デバイスの名称、タイプ、サービス対象、状態、モード、製造元、ソフトウェアとハードウェアのバージョン番号等の基本的な情報が含まれる。

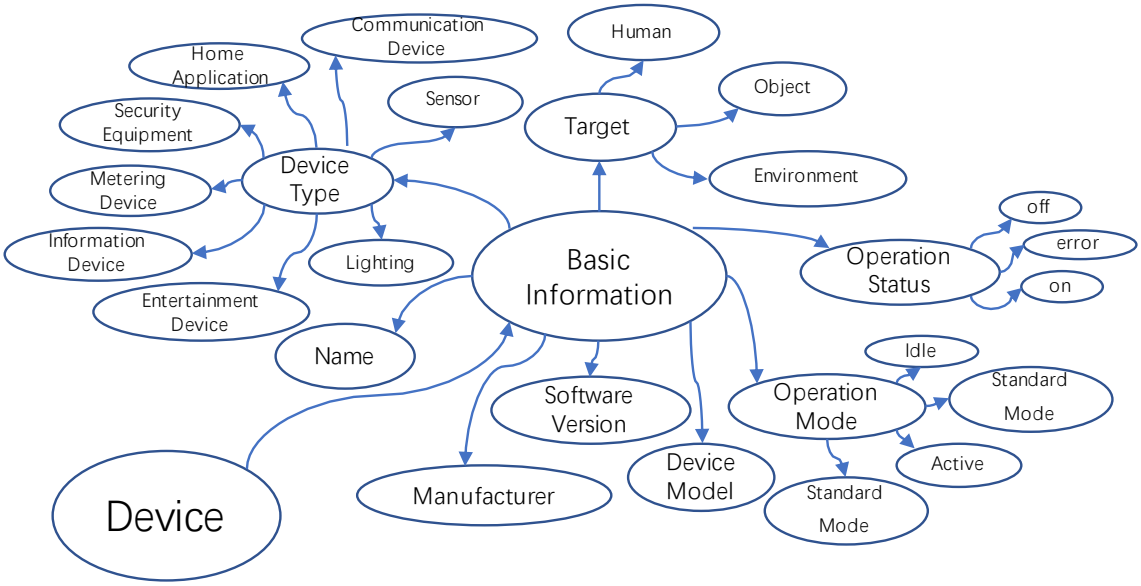


図 4-2 基本情報モデル

(イ) 通信能力モデル：図 4-3 にデバイスの通信能力を記述したメタデータモデルの構造を示す。図に示したように、デバイスが使用される通信媒体、プロトコル、通信モード、通信範囲などの通信機能情報が含まれる。

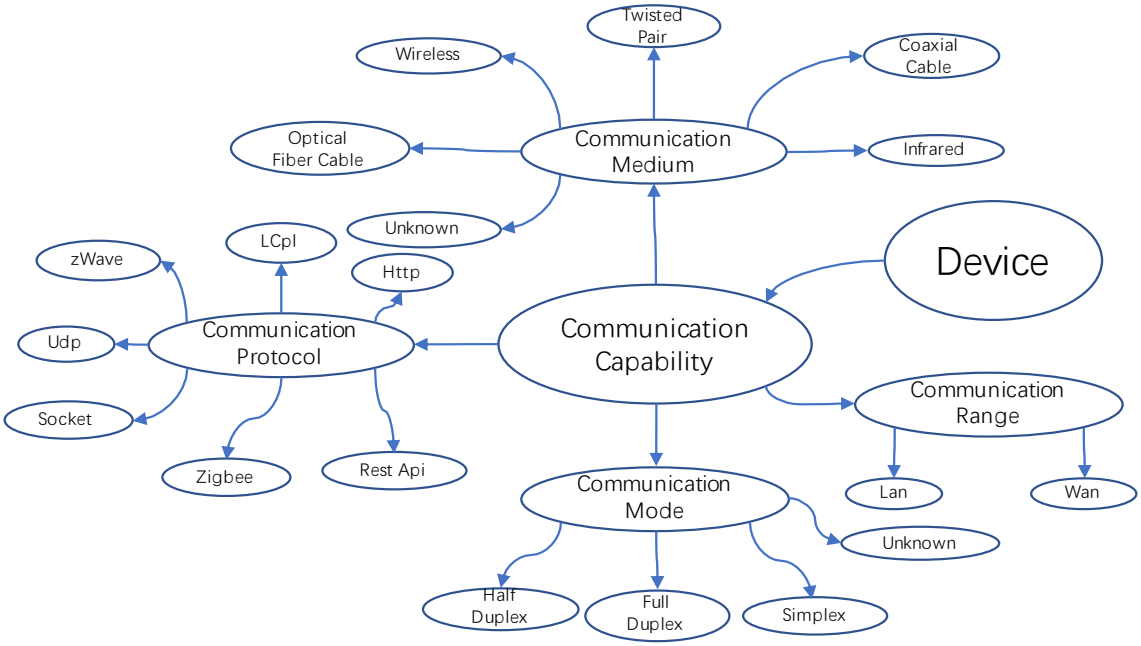


図 4-3 通信能力モデル

(イ) データモデル：：図 4-4 にデバイスに関するデータを記述したメタデータモデルの構造を示す。図に示したように、データの値、単位、値の範囲、精度、収集時間、時間更新率が含まれる。

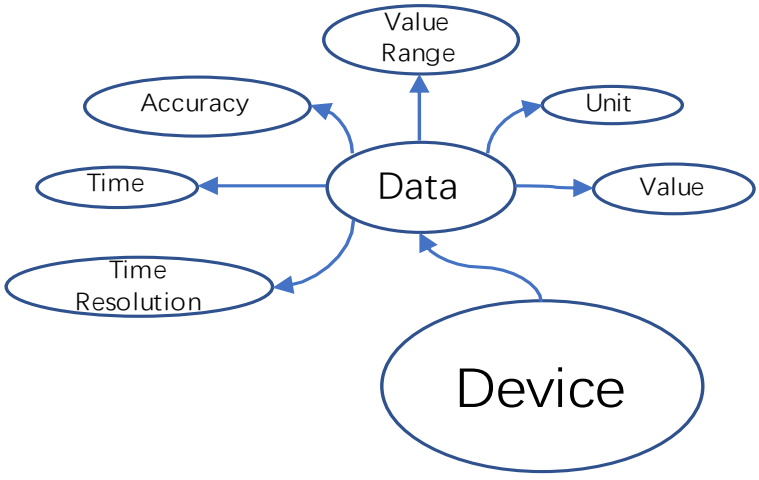


図 4-4 データモデル

(ウ) AoE：図 4-5 はデバイスに関する範囲効果、検知半径を表すメタデータが含まれる。

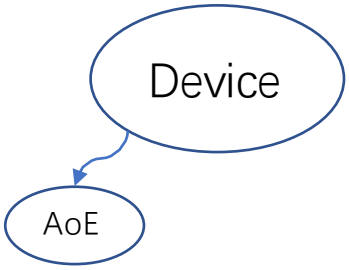


図 4-5AoE モデル

(エ) 機能モデル：図 4-6 にデバイス機能に関するメタデータモデルの構造を示す。図に示したように、動作機能、センシング機能、セキュリティ機能、表示機能が含まれる。

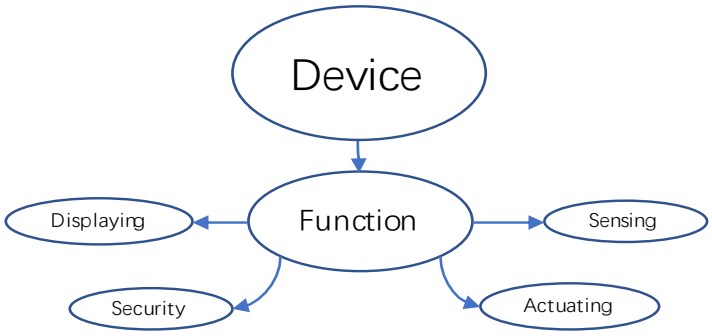


図 4-6 機能モデル

(オ) 位置情報モデル：図 4-7 にデバイス機能に関するメタデータモデルの構造を示す。図に示したように、Geography Location にデバイスが設置された地理的位置(経度,緯度,標高)の情報が記述され、General Location に家中の部屋情報が含まれる。

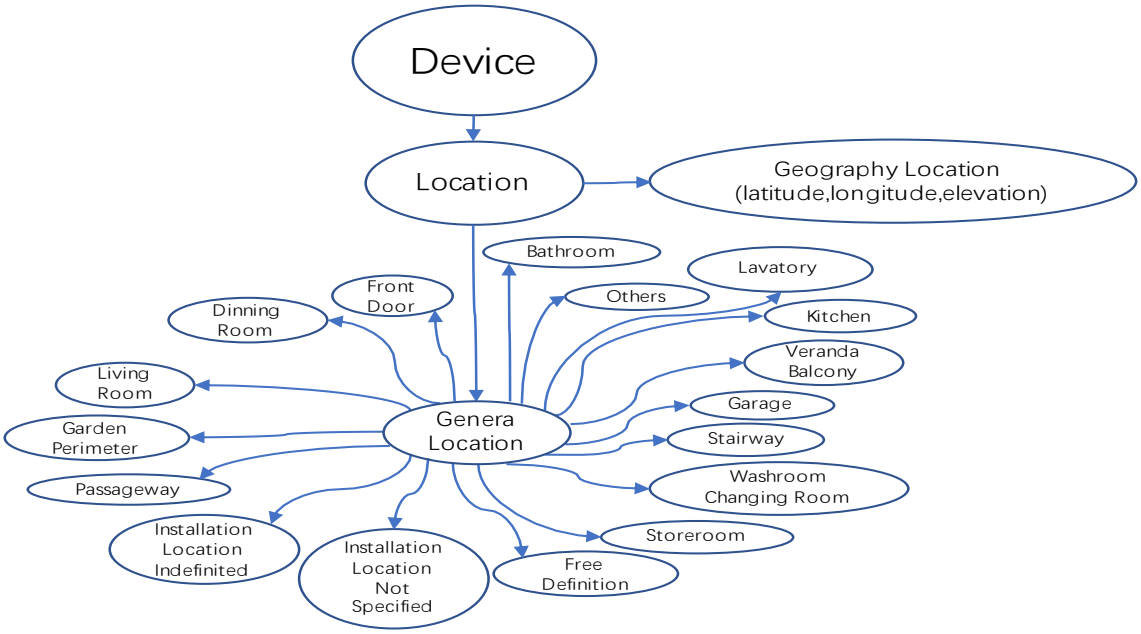


図 4-7 位置情報モデル

4.2 ホームネットワーク基本オントロジー

本論文は、ホームネットワーク向け既に存在する各プラットフォームを連携させ、共通のデータカタログを作成することを目的とするホームネットワーク基本オントロジー(BCADEF-Ontology)を構築した。基本オントロジーでは 4.2 章で提案したホームネットワークメタデータモデルの記述項目を利用して定義し、定義されたプロパティーから概念である記述項目の包含関係や階層構造を構築した。また、同じ階層の概念同士の関係性を明確にするために、記述論理を使用して概念を設計し、推論をすることが可能となる。

4.2.1 オントロジーマッピング

SemanticWeb では、異種データを統合ためオントロジーを利用し、マッピングする手法を提案した。本論文もその提案に基づき、ホームネットワーク基本オントロジーを利用することで、記述項目で表現された概念間の関係に基づいたデータマッピングが可能になると考えられる。データマッピングとは、二つの異なるデータを関連つけることで、あるいはソースとなるデータを目的のデータに到達できるように基礎を策定することである。データマッピングにおいては以下のような手順が必要とされる。

- ２つのデータ間での変形や仲介
- 分析によって２つのデータの関連性を調べる
- データの中から隠れた特性を調べる
- 複雑なデータベースをシンプルなデータベースに移行させる
- マッピングにおいて必要の無いデータを削除したり無視したりする

ホームネットワークのデータを連携・統合するため、全てのデータを同じ形式に変換しなければならない。そこで本研究で提案したホームネットワーク基本オントロジーを利用してデータマッピングを行い、データを同じ形式に変換させる。図 4-8 は可視化したホームネットワーク基本オントロジーである。

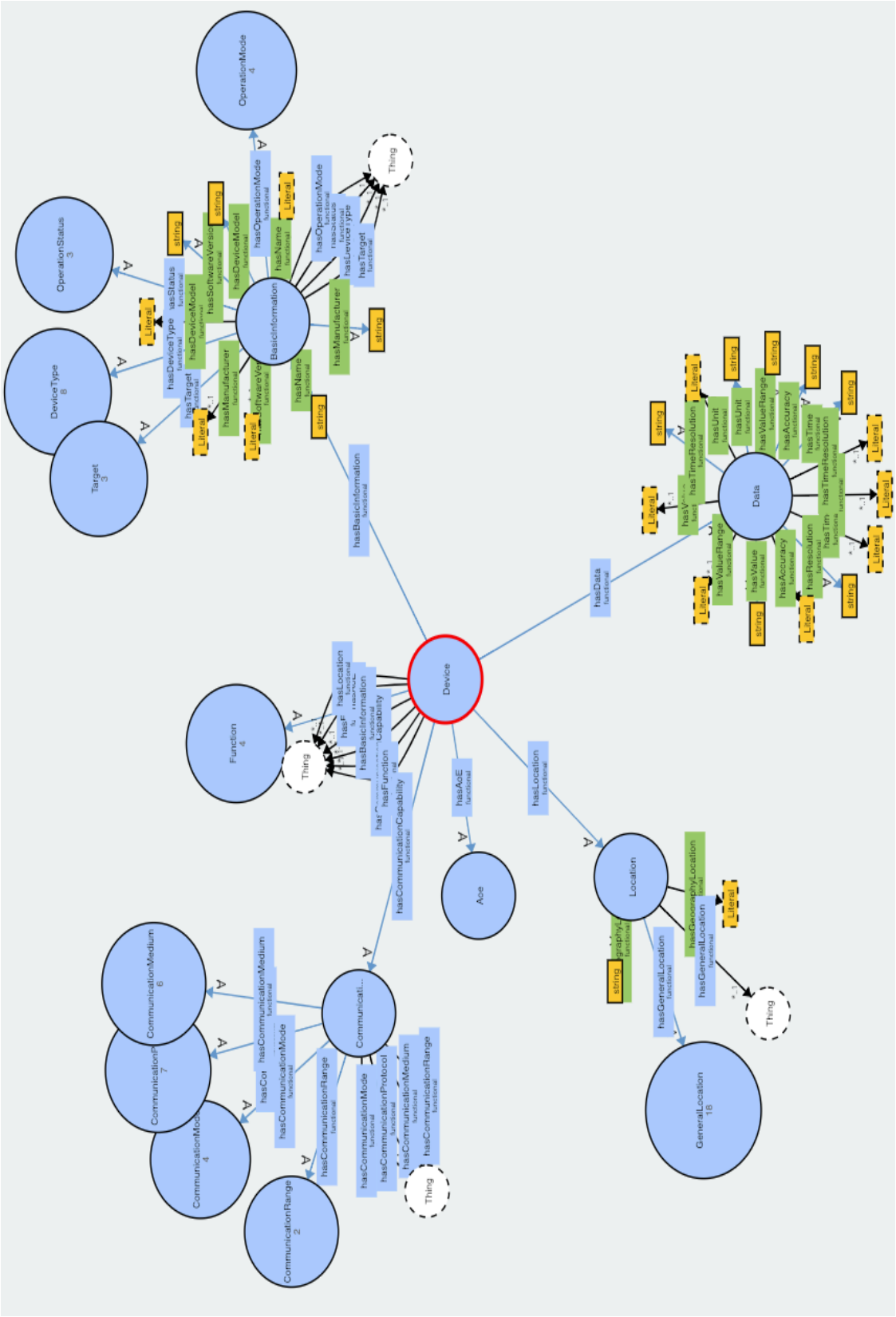


図 4-8 ホームネットワーク基本オントロジー

4.3 提案システムの構成

システムの構成は図 4.1 のように 4 階層形式とし、データストレージ層、オントロジー層、セマンティック層、アプリケーションサービス層に分割する。 [9]

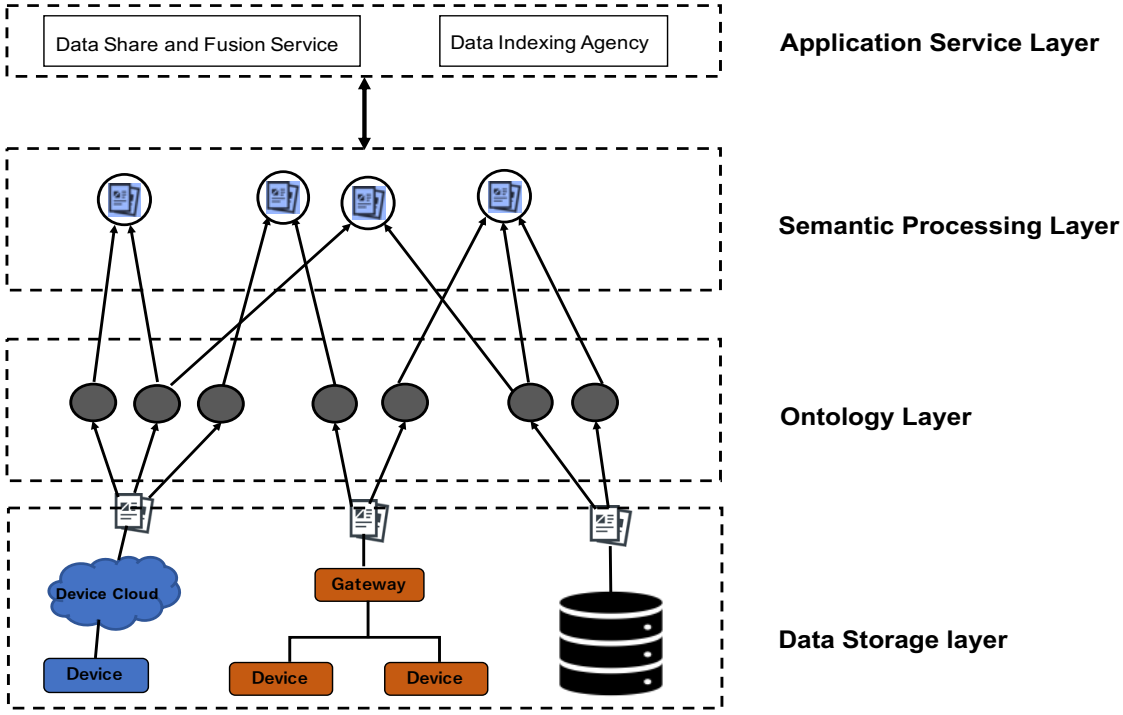


図 4-9 提案システムの構成

(1) データストレージ層(Data storage layer)：ホームネットワークにおいて大量な異種データが混在している複数多種類のデータを抽象化した層である。例えばデータベースに蓄積される構造化データ、センサーデバイスに蓄積される非構造化データ、XML ファイルのような半構造化データ等。

(2) オントロジー層(Ontology Layer)：異なった由来のデータを統合する前に、オントロジーライブラリをオントロジー層にインポートし、そして解析してオントロジーの概念情報、インスタンス情報、及びオブジェクト関係情報を取得する。データがシステムに接続すると、データマッピングを行い、新たなデータを生成する。

(3) セマンティック層(Semantic Processing Layer)：マッピングされたデータを保存し、さらに関連性があるデータを統合し、仮想データを生成する。例えば、同じ部屋中に全て温度センサーの集合などが相当する。

(4) アプリケーションサービス層(Application Service Layer)：データ利用者がデータを取得するための API を提供する。

4.3.1 実用的 IoT におけるデータの種類

実際の IoT 実装のデータストレージ層においては、データベースから収集された構造

化データ、一度処理された XML、JSON で保存された半構造化データ、直接センサーから収集された非構造化データの 3 種類のデータが混在する。

そのためデータを統合する際、異なる形式のデータに対し異なる技術を使用する。構造化データの場合、RDB から RDF へのマッピング技術(RDB2RDF)を利用する。半構造化及び非構造化データの場合、オントロジーマッピング用いてデータを RDF /OWL ファイルに統合する。

本研究では、主にホームネットワークにおいて発生されたデータを統合する対象としているため、システムはセンサーから収集された非構造化データのみが対応する。そこで 4.2 章で構築したホームネットワーク基本オントロジーを利用してデータマッピングを行い、データを RDF/OWL ファイルに変換する。

ホームネットワーク基本オントロジーを通じて、実デバイスから収集したデータを RDF/OWL ファイルに抽象化して、データを同じ形式に統合する。またデータに意味情報を加えて記述することにより、関連性があるデータを連携することも実現する。

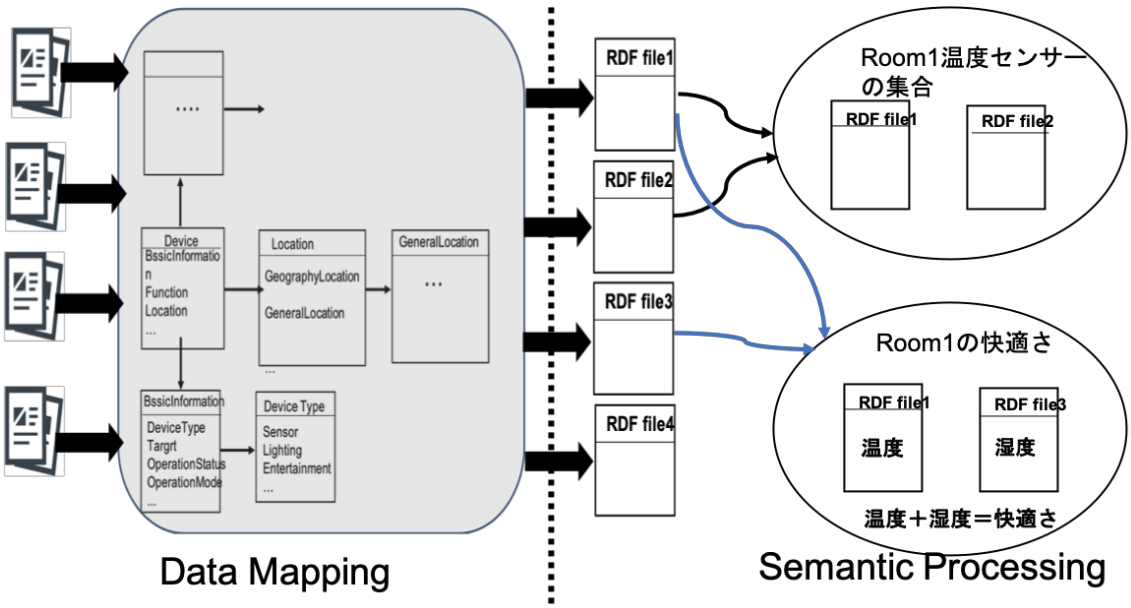


図 4-10 データマッピングの流れ

4.3.2 オントロジーの更新

特にホームネットワーク内のデータはリアルタイムに更新することが多いため、データをマッピングする際、オントロジーライブラリ内のインスタンスを常に更新する必要がある。本研究では、オントロジーモデル内の「Data」クラス内のメンバがリアルタイマに更新することを想定する。オントロジーライブラリ内の「Data」に関するインスタンスの数が所定の閾値をオーバーするか指定された時間閾値を超えると、インスタンスのサイズを制御・削除し、インスタンスの更新率を向上させる。アルゴリズムを表 1 に示す。 [10]

このアルゴリズムは Jena を利用し、デバイスオントロジーを解析してオントロジー内のクラス、則約及びインスタンスを抽出し、それぞれを事前に定義されたコンテナーに格納する。その中で、classList にデバイスオントロジーのクラスを格納し、relationList に則約を格納し、Individuallist にインスタンスを格納する。

また「Data」以外で更新する必要な場合、類似度関数 conceptMatch を利用し、インスタンスに問い合わせる。更新されたインスタンスは、事前に定義された updateInstanceList コンテナに格納する。

最後に、instanceContain 関数を利用し、更新する必要があるインスタンスが存在するかどうかを判断し、存在しない場合はインスタンスを更新し、そうではない場合は更新しない。

Algorithm

```
1:    Begin
2:        ontoModel ⇐ontoParse (DeviceOntolog y)
3:        classList ⇐ontoModel.getClass()
4:        individualist ⇐ontoModel.getIndividual()
5:        While DeviceDataFile! = null
6:            attributeList ⇐getDeviceAttribute()
7:            if  conceptMatch(class,attributeList)
8:                updateInstanceList ⇐getUpdateInstance()
9:                if  !instanceContain(individualist,updateInstanceList)
10:                   classAssertList ⇐classAssert(updateInstanceList)
11:                   if relationEqual(relationList,updateInstanceList)
12:                       relationAssertList ⇐relationAssert(relationList,updateInstanceList)
13:                   end if
14:               end if
15:           end if
16:       end While
17:   end
```

表 1　オントロジー更新アルゴリズム

4.3.3 オントロジーの保存

現在一般的に使用されているリレーショナルデータベースは、RDF を保存する際に大規模な保存、更新、修正及びクエリ効率等の問題を抱えている。TDB とは Jena が提供する RDF 永続化及び照会のためのコンポーネントである。本システムでは RDF ファイルを格納するには TDB 利用する。

URL を通じてオントロジーファイルの保存先のアドレスを取得し、次に Java が提供

している TDBPortal を利用し、構成ファイルを通じてオントロジーインスタンスを永続的に TDB に保存する。TDB 内のデータは、TDBPortal を通じて追加、削除、及び変更することができ、さらにオントロジー照会や推論等も可能である。

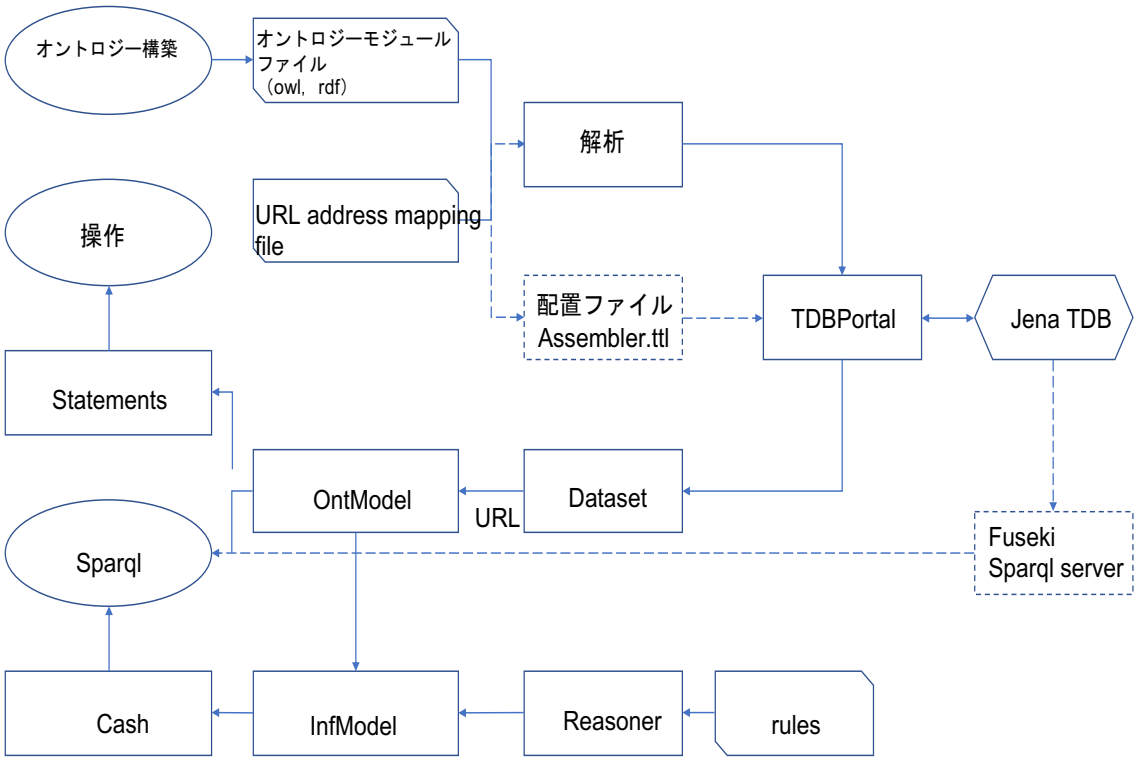


図 4-11 TDB のアーキテクチャ

4.3.4 検索機能

SPARQL [5]は OWL/RDF データを検索するため標準クエリ言語である。SQL のクエリ構文に似たようなものだが、SPARQL の対象はグラフモデル、SQL の対象はリレーショナルモデルである。本研究では、SPARQL クエリ機能を実現するため、Java 環境に ARQ クエリエンジンをインストールする。また、Jena は Fuseki という SPARQL サーバーを利用し、外部に検索できるような API を作成する。

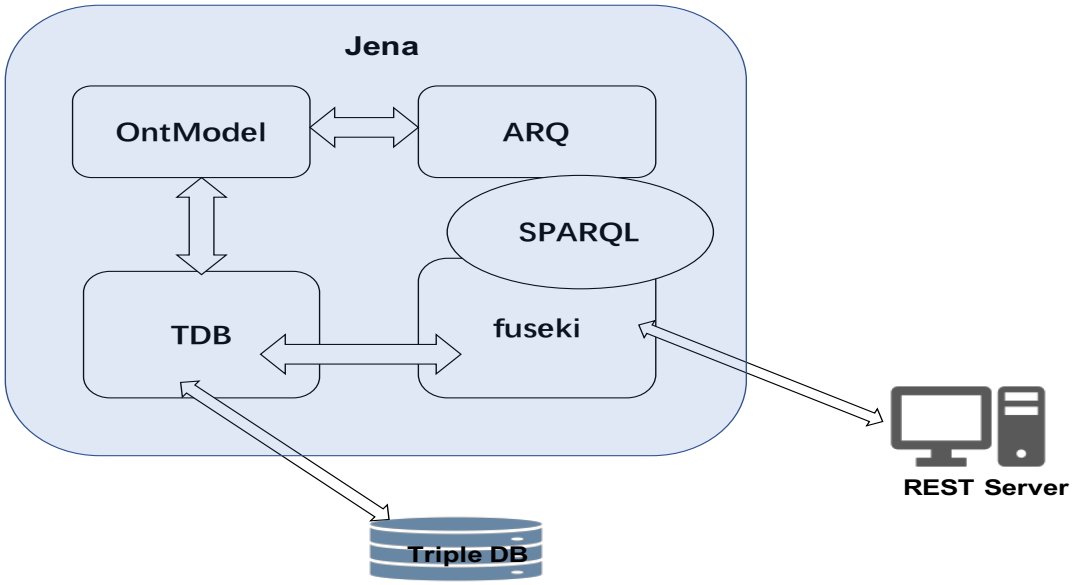


図 4-12　SPARQL query structure

第5章 実装

4 章で提案したシステムの構成とアーキテクチャの設計について詳述したが、本章ではシステムの実装について述べる。

5.1 開発環境

OS：Ubuntu 16.04 64bit
ハート: INTEL NUC PC
統合開発環境：Eclipse,Jena(Java api)
言語：Java

5.2 システム試作

iHouse 内すべての温度センサーのデータを抽出し、提案した BCADFL オントロジーモデルに基づいて、RDF/OWL というデータ形式にマッピングする。また、マッピングされたデータを TDB に保存する。外部からデータ検索できるような API を開発する。ここでは、Fuseki という SPARQL サーバを利用し、SPARQL クエリ言語で検索できるような環境を構築する。

5.2.1 オントロジーを構築

オントロジー編集ツール Protégé を利用し、ホームネットワーク基本オントロジーを作成する。図 5. 1 は Protégé で作ったオントロジーの一部構文である。

```
<rdf:RDF xmlns="http://jaist.ac.jp/Device.owl#"
     xml:base="http://jaist.ac.jp/Device.owl"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:pvn="http://ontology.universAAL.org/uAAL.owl#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Ontology rdf:about="http://jaist.ac.jp/Device.owl">
        <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Device</rdfs:
label>
        <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">The
collection of Device ontology</rdfs:comment>
    </owl:Ontology>

    <!-- http://jaist.ac.jp/Device.owl#hasAoE -->

    <owl:ObjectProperty
rdf:about="http://jaist.ac.jp/Device.owl#hasAoE">
        <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
        <rdfs:domain rdf:resource="http://jaist.ac.jp/Device.owl#Device"/>
    </owl:ObjectProperty>
```

図 5-1 オントロジー構文

5.2.2 データの抽出

Jena を利用し、iHouse 内の温度センサーを抽出して、オントロジーにマッピングする。図 5.2 は iHouse 内のデータ情報、図 5 .3 は一部マッピングされたデータである。

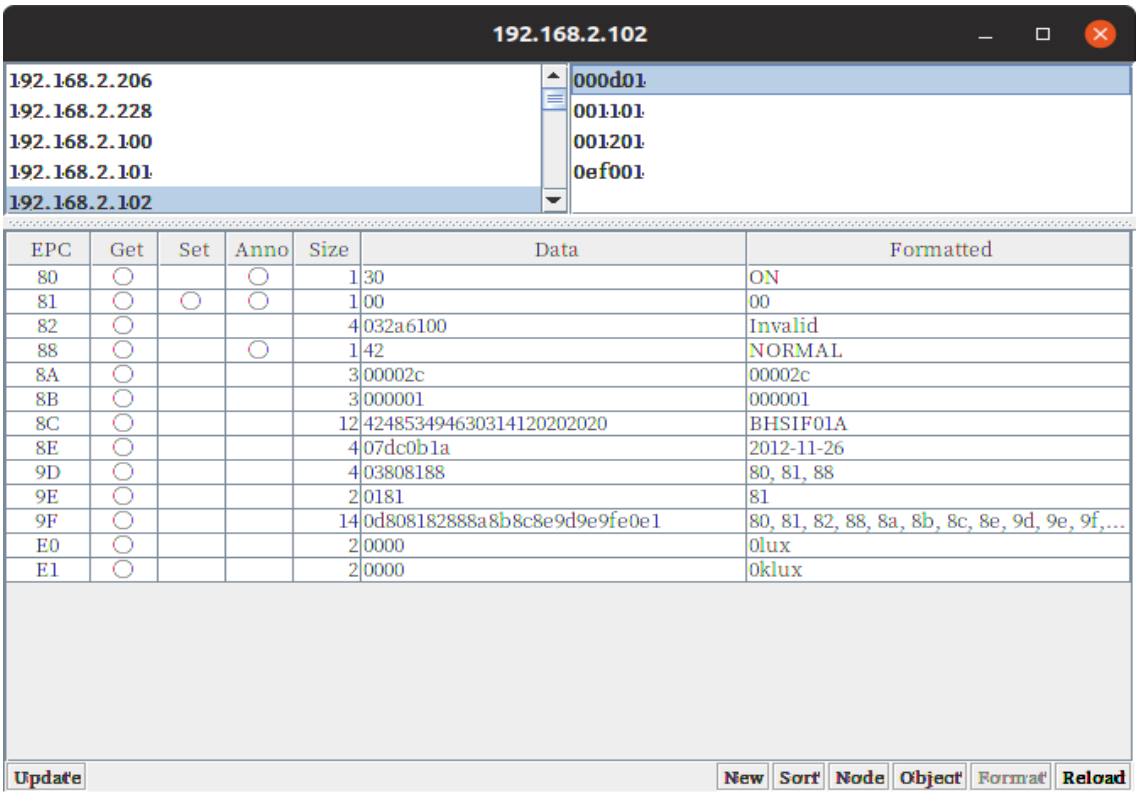


図 5-2　iHouse 内のデータ

```
<Device
rdf:about="http://jaist.ac.jp/Device.owl#TemperatureSensor192.168.2.194
_7">
    <hasData>
      <Data
rdf:about="http://jaist.ac.jp/Device.owl#DataTemperatureSensor192.168.2
.194_7">
        <hasTime>2019-01-16 14:47:16.153</hasTime>
        <hasValueRange>From -273.2 To 3276.6</hasValueRange>
        <hasResolution
rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
        >0.1</hasResolution>
        <hasUnit>Celcius Degree</hasUnit>
        <hasValue
rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
        >11.9</hasValue>
        <hasTimeResolution>5000 ms</hasTimeResolution>
      </Data>
    </hasData>
    <hasCommunicationCapability>
      <CommunicationCapability
rdf:about="http://jaist.ac.jp/Device.owl#CommunicationCapabilityTemper
atureSensor192.168.2.194_7">
        <hasCommunicationMedium
rdf:resource="http://jaist.ac.jp/Device.owl#unknown"/>
        <hasCommunicationRange
rdf:resource="http://jaist.ac.jp/Device.owl#lan"/>
        <hasCommunicationMode
rdf:resource="http://jaist.ac.jp/Device.owl#unknown"/>
        <hasCommunicationProtocol
rdf:resource="http://jaist.ac.jp/Device.owl#udp"/>
      </CommunicationCapability>
    </hasCommunicationCapability>
    <hasAoF>
```

図 5-3 マッピングされた一部のデータ

5.2.3 TDB の構築・データの保存

マッピングされたデータ(OWL 構文)を TDB に保存する。図 5 .4 に一部 TDB を作成するソースコードを示す。

```
public class TDBPersistence {
    public static final Log LOG = LogFactory.getLog(FaqDemo.class);
    public Dataset dataset = null
    public TDBPersistence(String tdbName) {
        dataset = TDBFactory.createDataset(tdbName);
    }
    public void loadModel(String modelName, String rdfFilePath, Boolean
isOverride) {
        int result;
        Model model = null;
        dataset.begin(ReadWrite.WRITE);
        try {

            if (dataset.containsNamedModel(modelName) &&
(!isOverride)) {
                result = 1;
            }
            else {
                if (dataset.containsNamedModel(modelName))
                    result = 2;
                else
                    result = 3;
                dataset.removeNamedModel(modelName);
                        model =
dataset.getNamedModel(modelName);
                model.begin();
                FileManager.get().readModel(model, rdfFilePath);
                model.commit();
                dataset.commit();
            }
```

図 5-4　TDB を作成する一部のソースコード

5.2.4 検索用 API の作成

Fuseki サーバーに TDB ファイルに添付し、ブラウザからデータを検索できるような API を作成し、Web で公開する。本論文が提案した BCADEF モデル内の記述項目が付与したデータを手に入れる事が可能となる。図 5.5 は SPARQL のクエリ構文、図 5.6 は Fuseki を利用し、温度センサーの情報を検索した結果である。

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
prefix device:<http://jaist.ac.jp/Device.owl#>

SELECT ?data ?value ?unit ?time ?Resolution ?ValueRange
WHERE {
 ?dev device:hasData ?data.
 ?data device:hasValue ?value.
 ?data device:hasUnit ?unit.
 ?data device:hasTime ?time.
 ?data device:hasResolution ?Resolution.
 ?data device:hasValueRange ?ValueRange
}
LIMIT 100
```

図 5-5 SPARQL のクエリ構文

	data	value	unit	time	Resolution	ValueRange
1	device:DataTemperatureSensor192.168.2.202_1	"37.4"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:13.434"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
2	device:DataTemperatureSensor192.168.2.194_7	"11.9"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:16.153"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
3	device:DataTemperatureSensor192.168.2.145_1	"3.0"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.054"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
4	device:DataTemperatureSensor192.168.2.107_1	"13.4"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.533"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
5	device:DataTemperatureSensor192.168.2.191_1	"15.5"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:13.845"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
6	device:DataTemperatureSensor192.168.2.194_8	"11.4"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:16.381"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
7	device:DataTemperatureSensor192.168.2.193_5	"12.3"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.293"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
8	device:DataTemperatureSensor192.168.2.194_2	"11.1"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.667"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
9	device:DataTemperatureSensor192.168.2.191_6	"14.8"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.489"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
10	device:DataTemperatureSensor192.168.2.193_6	"11.3"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.623"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
11	device:DataTemperatureSensor192.168.2.194_3	"12.2"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.990"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
12	device:DataTemperatureSensor192.168.2.191_7	"14.0"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.817"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
13	device:DataTemperatureSensor192.168.2.193_1	"12.0"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.128"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
14	device:DataTemperatureSensor192.168.2.191_2	"15.1"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:14.172"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
15	device:DataTemperatureSensor192.168.2.193_7	"11.8"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.901"	"0.1"^^xsd:double	"From -273.2 To 3276.6"
16	device:DataTemperatureSensor192.168.2.194_4	"11.4"^^xsd:float	"Celcius Degree"	"2019-01-16 14:47:15.268"	"0.1"^^xsd:double	"From -273.2 To 3276.6"

図 5-6　fuseki による「温度センサー」の情報を検索した結果

5.3 試作システム

サービス提供事業者から熱中症予防サービスを提供すると想定し、判断材料(データ)として部屋の室温が必要となる。一方、部屋中に室温を測定できるデバイス三つが存在するため、サービス事業者がメタデータから三つのデバイスから適切な情報を選択する。

デバイス: ECHONET Lite 温度センサ 2 個　OneM2M 温度センサ 1 個

判断メタデータ: このサービスには、メタデータ TimeResolution が判断材料だと定義する

手順 1 ：データマッピング

三つのデバイスを提案した BCADFL モデルにマッピングし、データを RDF 形式に統合し、TDB に保存する。

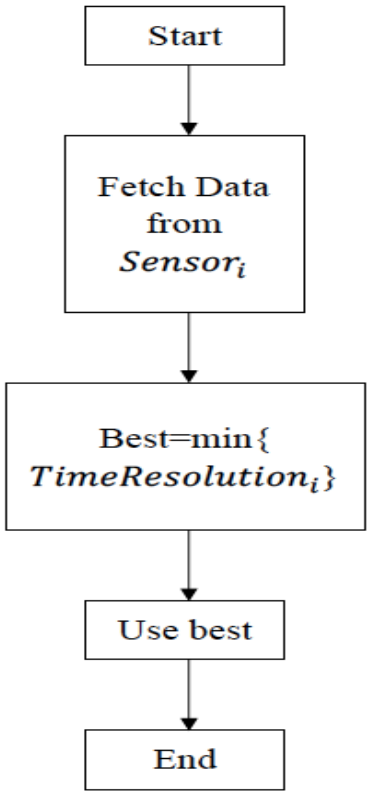
手順 2 ：要求データの検索

サービスに必要な“室温”を SPARQL で検索し、“LivingRoom”と“温度”に関するデータを確認する。

Name	OneM2M_TemperatureSensor	DataTemperatureSensor192.168.2.194_6	DataTemperatureSensor192.168.2.193_7
Time	2019/1/18 19:16:37	2019/1/18 19:16:36	2019/1/18 19:16:37
Value	11.6	11.3	11.8
Unit	Celcius Degree	Celcius Degree	Celcius Degree
TimeResolution	10000ms	5000ms	10000ms
Resolution	0.1	0.1	0.1
GeneralLocation	livingRoom	livingRoom	livingRoom
Function	sensing	sensing	sensing

手順３：室温情報の選択

このサービスには TimeResolution の方が重要だと定義したため、メタデータ Time Resolution の値が小さいデバイスを選択する。(DataTemperatureSensor192.168.2.194_6)



第6章 評価

6.1 評価 1

評価内容：
既存のモデルである SAREF と OneM2M と比較する。

結果：

	BCADFL	SAREF	OneM2M base Ontology
class 数	少ない	多い	中
layer 数	少ない	中	多い
individual 数	多い	少ない	中
通用性	低い	やや高い	高い

結論：
BCADFL モデルは Class 数と Layer 数が少なく、Individual 数が多いことがわかった。そのためデータマッピング処理や検索の速度が速くなると、モデルの汎用性が高くなると考えられる。

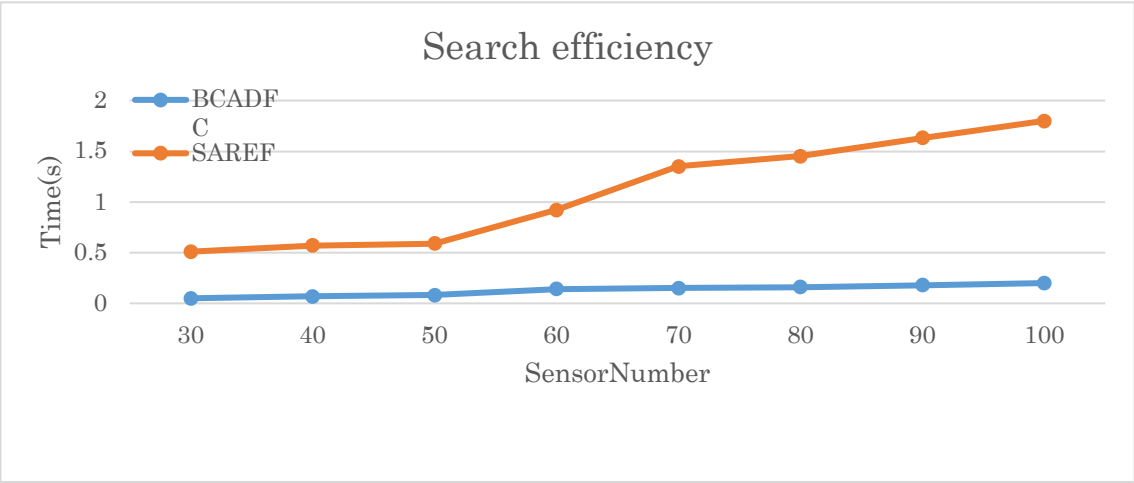
6.2 評価 2

評価内容：
BCADFL モデルと SAREF モデルのマッピング処理の比較

実験環境:
OS：Ubuntu 16.04 64bit
ハート: INTEL NUC PC
IDE：Eclipse

評価手順：
iHouse から 100 個 Echonet Lite センサを抽出し、データマッピングを行った

結果：
BCADFC モデル



SensorNumber	30	60	100
BCADFC	0.05	0.14	0.2
SAREF	0.46	0.78	1.6

結論：

BCADFC モデルのマッピン処理速度は SAREF モデルより約 80%速くなる。

6.3 評価 3

評価内容：

BCADFL モデルと SAREF モデルの検索時間の比較。検索項目三つを決まって、評価を行う。

実験環境：

Server：

INTEL NUC PC

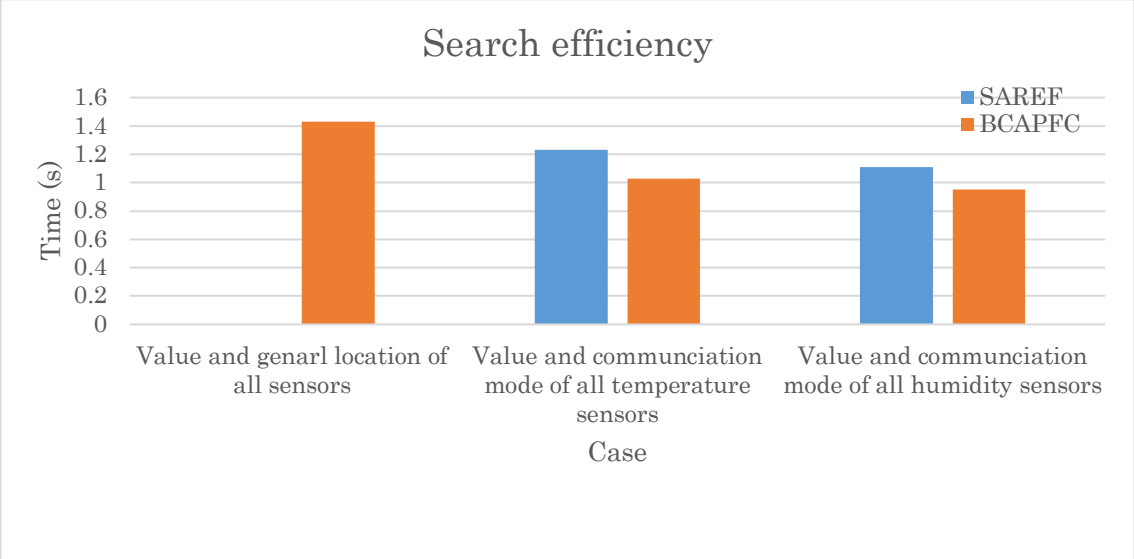
OS: Ubuntu 16.04

IDE:Eclipse

Client：

MACBook Air

結果：



Value and genarl location of all sensors												
Time(SAREF)	0	0	0	0	0	0	0	0	0	0		0
Time(BCAPFC)	1.43	1.48	1.44	1.42	1.45	1.44	1.42	1.48	1.47	1.41		1.444
Value and communciation mode of all temperature sensors												
Time(SAREF)	1.23	1.29	1.21	1.24	1.28	1.26	1.21	1.22	1.21	1.29		1.244
Time(BCAPFC)	1.03	1.05	1.01	1	1.06	1.08	1.08	1.07	1.06	1.05		1.049
Value and communciation mode of all humidity sensors												
Time(SAREF)	1.11	1.15	1.17	1.11	1.1	1.11	1.12	1.15	1.16	1.14		1.132
Time(BCAPFC)	0.95	0.98	1.01	0.92	0.91	0.93	0.9	0.97	0.96	0.95		0.948

結論：

同じ内容を検索する場合、 BCADFL モデルは SAREF モデルより検索時間約 15%速くなる。

第7章 考察

7.1 メタデータモデルの有用性

本研究は、サービス事業者に向けホームネットワークのデータを連携することが一つの目的である。提案手法としては、メタデータモデルを構築し、オントロジーを用いてデータを同じ形式にする。本メタデータモデルは ECHONET 規格や SSN 等標準化されているデータモデルを参考して構築したが、メタデータがふさわしいかどうかの判断は長い時間をかける検証する必要があると考える。

7.2 データマッピング手法の適用性

本研究は、オントロジー技術と Jena を利用し、データマッピングすることができた。しかしマッピングは自動化ではない。マッピングする際に毎回システムに登録する必要がある。そのため、手を煩わずに自動的にデータマッピングをできるようにする必要がある。

7.3 検索方法の適用性

本研究は、SPARQL というクエリ言語を利用し、検索機能を実現した。しかし SPARQL を使うには、時間かけて学ぶ必要があるため、自然言語で検索できる仕組みが必要となる。

第8章 おわり

本研究は、データ流通量の急速な増加を背景に IoT 代表的なスマートホーム分野においてメタデータを利活用し、ホームネットワーク向け既に存在する各プラットフォームを連携させ、相互に理解できるデータカタログのあり方について提案した。

具体的には、まずホームネットワーク内のデータを対象として、メタデータモデルを設計する。メタデータモデルの構築は，情報の権威性を持たせることが非常に重要である。そこで、ECHONET 規格や SSN 等標準化されているデータモデルを参考し、BCADFL というメタデータモデルを提案した。そのため、データモデルの正確性や厳密性など品質の維持や保証することができた。そしてオントロジーを用いてデータに語彙を付与し、同じ形式にマッピングする手法を提案した。 [11]評価実験結果により、BCADFL モデルは SAREF モデルによりマッピング処理が速いことがわかった

次に、OneM2M 汎用セマンティックモデルをベースとして、SPARQL というクエリ言語を利用し、メタデータを検索できるシステムを提案した。

評価実験により、 BCADFL モデルは SAREF モデルにより検索時間が速いことがわかった。

提案したメタデータの活用手法により、業界やプラットフォームの壁を越え、住宅から収集した適切なデータをスムーズにサービス提供事業者に伝送し、ユーザーにより良いサービスが実現できると考えられる。また、第三者にとってもこれらのデータが二次利用できるようになり、データ流通市場が活性化することも期待できると考える。

謝辞

本研究を行うにあたり、終始ご指導ご鞭撻を賜りました丹康雄教授に深く感謝致します。また審査員をお引き受け頂いた本学リム勇仁准教授、篠田陽一教授、長谷川忍准教授には、本論文を執筆するにあたり多大なご助言を頂きました、深く感謝致します。

副テーマにおいてご指導ご鞭撻を賜りました本学知念賢一教授に深く感謝致します。

本研究室の博士後期課程の PHAM Van Cu 氏、博士前期課程北川 貴博氏には、研究に関して活発な議論、ご指導を賜りました。心から感謝致します。

本論文をまとめるにあたりご協力頂いた丹研究室、リム研究室の諸兄に厚く御礼申し上げます。

最後に、私の研究に対し理解を示して頂き、支えて頂いた家族に感謝を致します。

参考文献 [7]

[1] 総務省・経済産業省, "データ流通プラットフォーム間の連携を実現するための," *総務省 経済産業省,* 4 平成 29.

[2] "「スマートホームの実現に向けた機器接続・データ利活用等の検討事項」報告書," 平成 28 年度 IoT 推進のための新産業モデル創出基盤整備事業, 平成 29 年 3 月.

[3] 溝口理一郎, オントロジー光学, 人工知能学会, 平成 17 年 1 月 20 日.

[4] 来村　徳信, オントロジーの普及と応用, 人工知能学会, 平成 24 年 4 月 20 日.

[5] "Apache Jena Fuseki," . [Online]. Available: https://jena.apache.org/documentation/fuseki2/index.html.

[6] ONEM2M, "ONEM2M　TECHNICAL PEPORT," 2014. [Online].

[7] Release K, "APPENDIX ECHONET 機器オブジェクト詳細規定," 25 Oct 2018. [Online]. Available: https://echonet.jp/wp/wp-content/uploads/pdf/General/Standard/Release/Release_K_jp/Appendix_Release_K.pdf.

[8] W3C, "Semantic Sensor Network Ontology," 19 October 2017. [Online]. Available: https://www.w3.org/TR/vocab-ssn/.

[9] J. University(China), "Research and Implementation of the Smart Home System Based on Semantic Fusion". China Patent CN201310603413.4, 19 2 2014.

[10] W. Shun, “The Research and Application of Based on The Research and Application of Based on Fusion Method,” *Nanjing University of Aeronautics and Astronautics(China),* 1 2018.

[11] 正. 長井, 雅. 小野 and 亮. 柴崎, "地球観測データのためのリモートセンシングオントロジーの構築," 1 2009.

[12] オムロン（株）技術・知財本部　ＳＤＴＭ推進室, "センシングデータ流通市場におけるメタデータの定義・生成・活用の一方式," 2018.