【注意】 この文書は、2020年1月30日付の W3C勧告案「Web of Things (WoT) Thing Description」(原文は英語)を、WoT JP Commuityが翻訳と修正をおこなって公開しているものです。この文書の正式版は、あくまでW3Cのサイト内にある英語版であり、この文書には翻訳上の間違い、あるいは不適切な表現が含まれている可能性がありますのでご注意ください。

W3C

Web of Things (WoT) Thing Description

W3C Candidate Recommendation 16 May 2019

This version:
https://www.w3.org/TR/2019/CR-wot-thing-description-20190516/
Latest published version:
https://www.w3.org/TR/wot-thing-description/
Latest editor's draft:
https://w3c.github.io/wot-thing-description/
Implementation report:
https://w3c.github.io/wot-thing-description/testing/report.html
Previous version:
https://www.w3.org/TR/2018/WD-wot-thing-description-20181021/
Editors:
Sebastian Kaebisch (Siemens AG)
Takuki Kamiya (Fujitsu Laboratories of America, Inc.)
Michael McCool (Intel)
Victor Charpenay (Siemens AG)
Participate:
GitHub w3c/wot-thing-description
File a bug
Commit history
Pull requests
Contributors:
In the GitHub repository
Repository:
We are on GitHub
File a bug

摘要

本文書は、WoT(Web of Things) Thing Description (TD)のための形式モデルおよび共通表現を解説するものである。TDは、モノのメタデータおよびインターフェースを記述し、モノは、WoT内での対話を提供し、それに参加する物理エンティティまたは仮想エンティティのアブストラクションである。TDは、多様なデバイスの結合、および、多様なアプリの相互運用を可能にする少数のボキャブラリに基づく対話セットを提供する。TDは、デフォルトでJSON-LD 処理も可能なJSON 形式でエンコードされる。後者は、マシン理解可能な方法でモノに関する知識を表すための効果的な基盤を提供する。TDインスタンスは、モノ自体によって提供されるか、または、モノにリソース制限(例えば、制限されたメモリ空間)があるとき、あるいは、WoT互換性レガシーデバイスのTDによってレトロフィットされているときには、外部で提供される。

本文書のステータス

本項では、発行時点での本文書のステータスを説明する。このため、他文書が本文書よりも新しい場合もある。現行のW3C出版物一覧および本テクニカルレポートの最新版は、https://www.w3.org/TR/のW3Cテクニカルレポート索引に記載されている。

(本仕様書内で黄色表示されている)以下の危険状態の機能は、ＣＲ期間中に報告されるインプリメンテーション経験、及び・あるいは、受け取るコメントが不十分であるために削除されることがある。

- 第5.3.3.7項CertSecurityScheme、第5.3.3.9項PublicSecurityScheme、第5.3.3.10項PoPSecurityScheme記載のセキュリティスキームに関連する全項。
- 第5.3.3.11項OAuth25securityScheme内のimplicit, password, clientフローに関するというボキャブラリ用語とアサーション。
- 前述の 第5.4項 デフォルト値定義内の上記項に関連する全デフォルト値。
- 不完全な書き込み拒否を可能にするwriteallpropertiesのためのビヘイビアアサーション

本文書は、WoTワーキンググループによって推奨候補として発行されたものであり、W3C推奨になることを意図している。

GitHubの問題指摘は本仕様の考察の材料として歓迎される。また、それをpublic-wot-wg@w3.org(アーカイブ)で我々のメーリングリストにコメントを送ることもできる。

W3Cは、本文書が信頼できるものであると考えられていることを示し、開発者コミュニティによるインプリメンテーションを奨励するために、推奨候補を公開するもので、この推奨候補は、2019年12月4日以前に提案推奨として更新される予定である。

ワーキンググループのインプリメンテーションレポートを参照いただきたい。

推奨候補としての公開はW3Cメンバーによる推奨を意味するものではない。本文書は、常に、更新されたり、他の文書と置き換えられたり、また破棄される可能性もある草案文書であり、“作業中文書”取扱以外で本文書を引用することは不適当である。

本文書は、W3C特許ポリシーに従い運営しているグループが作成したものである。W3Cでは、ワーキンググループの成果物に関係する公的な特許公開リストを管理しており、そのページには特許開示にあたっての指示も掲載されている。必須特許請求を含むと信じる特許について実知識を持つ者は、W3C特許ポリシー第6項に従って、その情報を開示しなければならない。

本文書は、2019 年3 月1 日のW3Cプロセス文書に準拠する。

もくじ

1. **TD表現フォーマット**
    1. JSONタイプへのマッピング
    2. デフォルト値の省略
    3. 情報モデルのシリアライズ
        1. Thing Root オブジェクト
        2. 人間が読み取り可能メタデータ
        3. Version
        4. securithDefinition to security
        5. properties
        6. actions
        7. events
        8. links
        9. forms
        10. データスキーマ
    1. ID
1. **TDコンテキスト拡張子**
    1. 意味論的注釈
    2. プロトコルバインディングの追加
    3. セキュリティ方式の追加
2. **ビヘイビアアサーション**
    1. セキュリティ構成
    2. データスキーマ
    3. プロトコルバインディング
        1. HTTPに基づくプロトコルバインディング
        2. その他のプロトコルバインディング
9. **セキュリティとプライバシーに関する考慮事項**
    1. プライバシーリスクをデリフェレンスするコンテキスト
    2. 不変識別子プライバシーリスク
    3. プライバシーリスクフィンガープリンティング
    4. グローバル一意識別子プライバシーリスク
    5. TD傍受とセキュリティリスクの改竄
    6. コンテキスト傍受とセキュリティリスクの改竄
    7. 個人情報プライバシーリスクの推測
10. **IANAに関する考慮事項**
    1. application/td+json メディアタイプ登録
    2. CoAP Content-Format Registration

A.

1. CoAP プロトコルバインディングを使用したMyLampThing の例
2. MQTT プロトコルバインディングを使用したMyLightSensor の例
3. Webhookイベントの例

B. **TDインスタンス確認のためのJSONスキーマ**

C. **ＴＤテンプレート**

1. ＴＤテンプレートの例
    1. ＴＤテンプレート: ランプ
    2. ＴＤテンプレート: ブザー

D. **JSON-LDコンテキスト用法**

E. **最近の仕様変更**

1. 第一次推奨候補からの変更点
2. 第三次公開作業ドラフトからの変更点

F. **謝辞**

G. **参考文献**

1. 標準参考文献
2. 参考文献

1. はじめに

本項は標準ではない。

WoT Thing Description (TD)は、W3C Web of Things (W3CWoT)における中心的なビルディングブロックであり、モノ(ウェブサイトのindex.htmlに酷似する)のエントリポイントとみなすことができる。TDインスタンスには、4つの主要なコンポーネント、すなわち、モノ自体に関する文字メタデータ、モノはどのように使用できるかを示す対話アフォーダンス、マシンが理解できるようにするためにモノと交換されるデータスキーマ、および、ウェブ上の他のモノ、または、文書との形式的または非形式的な関係を表すためのウェブリンクで構成されている。

W3C WoTの対話モデルは、3タイプの対話アフォーダンスを定義している。すなわち、プロパティ(PropertyAffordanceクラス)は、現在の値を取得する、または、動作ステータスを設定するなどパラメータを感知・制御するために使用することができる。アクション(ActionAffordanceクラス)は、物理的な(したがって時間のかかる)プロセスの呼び出しをモデル化するが、既存のプラットフォームのRPCのような呼び出しを抽象化するために使用することもできる。イベント(EventAffordanceクラス)は、通知、個別イベント、または値のストリームが非同期で受信側に送信される通信のプッシュモデルに使用される。詳細は[WOT-ARCHITECTURE]参照。

一般に、TDは、URIスキーム [RFC3986](例えば、http、coapなど,[IANA-URI-SCHEMES])、メディアタイプに基づくコンテンツタイプ[RFC2046](例えば、application/json、application/xml、application/cbor、application/exiなど、[IANA-MEDIA-TYPES]、および、セキュリティメカニズム(TDインスタンスのシリアライズは、JSON [RFC8259]に基づいており、ここでは、本仕様書で定義するとおり、JSON名はTDボキャブラリ用語を指す。さらに、TDのJSONシリアライズは、拡張および豊富な意味処理を可能にするために、JSON–LD 1.1[JSON-LD11]のシンタックスを遵守している。

例1は、TDインスタンスであり、MyLampThingというタイトルのランプというモノを記述するプロパティ、アクション、イベントを持つ対話モデルとなっている。

例1:TDの例

```
{
    "@context": https://www.w3.org/2019/wot/td/v1,
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing",
    "securityDefinitions": {
        "basic_sc": {"scheme": "basic","in":"header"}
    },
    "security": ["basic_sc"],
    "properties": {
        "status" : {
            "type": "string",
            "forms": [{"href": "https://mylamp.example.com/status"}]
        }
    },
    "actions": {
        "toggle" : {
            "forms": [{"href": "https://mylamp.example.com/toggle"}]
        }
    },
    "events":{
        “overheating”:{
            "data": {"type": "string"},
            "forms": [{
                「href」:“https://mylamp.example.com/oh”,
```

```
                "subprotocol": "longpoll"
            }]
        }
    }
}
```

このTDの例からは、タイトルステータスを持つ1つのプロパティアフォーダンスが存在することがわかる。さらに、このプロパティが、(hrefメンバーによってforms構造内でアナウンスされる)URI　https://mylamp.example.com/statusでGETメソッドを使って（安全な形式の)HTTPプロトコルを介してアクセス可能であり、ストリングベースのステータス値を返すということを示す情報が提供されている。GETメソッドの使用は、明示的に述べられていないが、本文書で定義されるデフォルト想定の1つである。

同様に、アクションアフォーダンスが、https://mylamp.example.com/toggle資源上のPOSTメソッドを使って切り替えステータスをトグルするために指定され、ここでもPOSTは、アクションを呼び出すためのデフォルト想定である。

イベントアフォーダンスにより、モノが非同期メッセージのためのメカニズムを送信できるようになる。ここで、ランプの過熱事象が起こった場合に通知されるサブスクリプションは、https://mylamp.example.com/oh上の長いポーリングサブプロトコルを有するHTTPを使用することによって取得することができる。

本例は、また、アクセスのためにユーザ名およびパスワードを必要とするbasicセキュリティスキームを指定している。セキュリティスキームには、まず、securityDefinitionで名前が与えられ、次に、セキュリティセクションでその名前を指定することによってアクティブ化されることに留意されたい。HTTPプロトコルとの併用で、本例は、HTTP基本認証の使用を説明している。最上位レベルでの少なくとも1つのセキュリティスキームの指定は必須であり、これによってすべてのリソースに関するデフォルトのアクセス要件が与えられる。しかし、セキュリティスキームは、また、モノレベルで与えられる構成をオーバーライドするフォームレベルで与えられる構成を使ってフォームごとに指定することもでき、きめの細かいアクセス制御の指定を可能にしている。アクセス制御機構が何も使用されていないことを示すために特別なnosecセキュリティスキームを使用することも可能である。追加例が後述される。

TDは、いくつかの名前空間にコンテキスト定義を追加する可能性を提供する。このメカニズムは、正式な知識、例えば、アプリケーションの特定のドメインの論理規則が所与の名前空間で見つけることができる場合には、TDインスタンスのコンテンツに追加の意味を結合するために使用することができる。コンテキスト情報は、formsフィールドで宣言された基礎となる通信プロトコルのいくつかの構成およびビヘイビアの指定するときに役立つ。例2は、SAREF-Smart Appliance Reference Ontology（スマート家電参照オントロジー）-[SMARTM2M]を参照するものとしてプレフィックスsarefを宣言するために、@contextに第2の定義を導入して、例1のTDサンプルを拡張したものである。このＩｏＴオントロジーには、モノの意味論と対話アフォーダンスを持たせる@typeフィールドの値として設定することが出来る意味論上ラベルと解釈される用語が含まれる。下記の例では、モノはsaref:LightSwitch、 statusプロパティは saref:OnOffState、 toggleアクションはsaref:ToggleCommandとラベル付けされる。

例2: 意味論的注釈用のTDコンテキスト拡張子を持つTD

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"saref":"https://w3id.org/saref#"}
    ],
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "name": "MyLampThing",
    "@type": "saref:LightSwitch",
    "securityDefinitions": {"basic_sc": {
        “scheme”: “basic”,
        "in": "header"
    }},
    "security": ["basic_sc"],
```

```
        "properties": {
            "status": {
                "type": "string",
                "forms": [{
                    "@type": "saref:GetCommand",
                    "href":"https://mylamp.example.com/status"
                }]
            }
        },
        "actions": {
            "toggle": {
                "forms": [{
                    "@type": "saref:ToggleCommand",
                    "href":"https://mylamp.example.com/toggle"
                }]
            }
        },
        "events": {
            “overheating” : {
                "data": {"type": "string"},
                "forms": [{
                    "@type": "saref:NotifyCommand",
                    "href":"https://mylamp.example.com/oh"
                }]
            }
        }
    }
```

いくつかの@context内の宣言メカニズムは、JSON-LDによって指定される。TD インスタンスは、本仕様のバージョン1.1 [json-ld11] に準拠している。TDインスタンスは、また、RDFドキュメントとして処理もできる。（意味処理の詳細は、付録D.JSON-LDコンテキスト用法と名前空間ＩＲＩ内の資料（例： https://www.w3.org/2019/wot/td）参照。）

2. 適合性

標準ではないとされる項と同様に、本仕様書におけるすべてのオーサリングガイドライン、図、例、および注釈は標準ではない。本仕様内のこのようなものはすべて標準である。

本文書内のキーワード“**場合がある**”、”**しなければならない**“、”**してはならない**”、“**推奨される**”、”**べきである**”、**及び**、”**べきではない**”は、このように太字で表示されているときにのみBCP 14 [RFC2119] [RFC8174]の記述に従い解釈するものとする。

TDインスタンスは、それがTDシリアライズに関する第５項TD情報モデル及び第６項TD表現フォーマットで　標準ステートメントに従う場合、本仕様に準拠する。

TDインスタンスを確認するためのJSONスキーマ[JSON-SCHEMA]は、付録B. TDインスタンス確認JSONスキーマで提供されている。

3. 用語

モノ、コンシューマ、TD, 対話モデル、対話アフォーダンス、プロパティ、アクション、イベント、プロトコルバインディング、サービアント、WoTインターフェース、WoTランタイム等の基本WoT用語は、WoTアーキテクチャ仕様の第３項で定義される [WOTARCHITECTURE]。

また、本仕様では、以下の定義をしている。

TDコンテキスト拡張子

追加のボキャブラリ用語を使ってTDを拡張するメカニズム。これは、プロトコルバインディング、セキュリティスキーム、データスキーマなどの中核メカニズムに対する意味論的注釈および拡張にとって基本となる。

TD情報モデル

制約が適用される事前定義のボキャブラリで構築されたクラス定義、したがって、これらのボキャブラリの意味論を定義する。クラス定義は、通常、シグネチャ(ボキャブラリ用語)及びそのシグネチャ内の関数で表現される。TD情報モデルには、また、クラスに対するグローバル関数として定義されるデフォルト値が含まれる。

TDプロセッサ

所与のフォーマットでTDの何らかの内部表現をシリアライズする、また／あるいは、そのフォーマットからそれをシリアライズ解除することができるシステム。TDプロセッサは、意味的に矛盾するTD、すなわち、モノクラスのインスタンス関係に対する制約を満たすことができないTDを検出しなければならない。そのために、TDプロセッサは、可能なすべてのデフォルト値が割り当てられているTDの標準フォームを計算することができるかもしれない。TDプロセッサは、通常、WoTランタイムのサブシステムである。ＴＤプロセッサのインプリメンテーションは、(ＴＤドキュメントをシリアライズできる)ＴＤプロデューサのみ、あるいは、（ＴＤドキュメントからシリアライズ解除することができる）ＴＤコンシューマであろう。

TDシリアライズまたはTDドキュメント

サービアント間で保存および交換できるTDのテキストまたはバイナリ表現。TDシリアライズは、ネットワーク上で交換されるときにメディアタイプによって識別される所与の表現フォーマットに従う。TDのデフォルト表現フォーマットは、本仕様が定義するようにJSONベースである。

ボキャブラリ

名前空間IRI で識別されるボキャブラリ用語集。

用語・ボキャブラリ用語

文字列。用語がボキャブラリの一部である場合、すなわち、名前空間IRIとプレフィックスがついている場合、それはボキャブラリ用語と呼ばれる。読みやすくするために、本書に記載されているボキャブラリ用語は、コンパクトな形式で書かれており、完全なIRIとはなっていない。

これらの定義は、第5.2項前付けの中でさらに詳細される。

4. 名前空間

本仕様第５項TD情報モデルで定義されている本バージョンTD情報モデルは、以下のIRIによって識別される。

https://www.w3.org/2019/wot/td/v1

URI[RFC3986]でもあるこのIRI[RFC3987]は、JSON–LDコンテキストファイル[json–ld11]を得るためにデリフェレンスすることができ、TDドキュメント内のコンパクトなストリングを完全なIRIベースのボキャブラリ用語に拡張することができる。しかしながら、この処理は、JSONベースのTDドキュメントをTDプロセッサインプリメンテーションのオプション機能であるRDFに変換するときにのみ要求される。

本仕様では、ボキャブラリ用語は常にコンパクトなフォームで表されている。それらの拡張されたフォームは、それらが属するボキャブラリの名前空間IRIでアクセスすることができる。これらの名前空間は第5.3項クラス定義の構造に従う。TD情報モデルで使用される各ボキャブラリは、以下のように、それ自体の名前空間IRIを持つ。

ボキャブラリ	名前空間IRI
コア	https://www.w3.org/2019/wot/td#
データスキーマ	https://www.w3.org/2019/wot/jsonschema#
セキュリティ	https://www.w3.org/2019/wot/security#
ハイパーメディアコントロール	https://www.w3.org/2019/wot/hypermedia#

ボキャブラリは互いに独立している。それらは、他のW3C仕様で再利用・拡張することもある。ボキャブラリの設計を大きく変更するには、必ず新しい年ベースの名前空間URIの割り当てが必要になる。TD情報モデルの総体的な一貫性を維持するために、関連するJSON-LDコンテキストファイルは、大きくはない変更、特に、新しい用語の追加を識別するために、あらゆるバージョンがそれ自体のURI (v1、v1.1、v2,)を持つようにバージョン化されることに留意されたい。

いくつかの名前空間IRIのボキャブラリでは大きくはない変更のみをすることができるので、そのコンテンツは、安全なキャッシュか、または、アプリへの組み込みが可能である。名前空間IRIの下で比較的静的なコンテンツを公開することの利点の一つは、制約されたデバイス間で交換されるメッセージのペイロードサイズの最適化である。また、プライベートネットワークから公的に利用可能なボキャブラリにアクセスするデバイスが原因となるプライバシー漏洩を回避する(第9.1項プライバシーリスクをデリフェレンスするコンテキストも参照)。

5. TD情報モデル

本項では、TD情報モデルについて紹介する。TD情報モデルは、TDおよびそのシリアライズ処理のための概念的な基盤となる。これについては、別途、第６項TD表現フォーマットで説明する。

1. 概要

TD情報モデルは、以下の独立したボキャブラリで構築される。

- プロパティ、アクション、イベントアフォーダンス、対話モデルを反映する中核TDボキャブラリ[WOT-ARCHITECTURE]
- JSON スキーマによって定義された用語(のサブセット) を含むデータ・スキーマ・ボキャブラリ[JSONSCHEMA]
- セキュリティメカニズムとその設定要件を識別するWoTセキュリティボキャブラリ
- ウェブリンクとフォームを使用してRESTful通信の主な原則をエンコードするハイパーメディア制御ボキャブラリ

これらのボキャブラリのそれぞれは、本質的に、従来のオブジェクト指向の意味でオブジェクトとして解釈される、データ構造を構築するために使用することができる用語セットである。オブジェクトは、クラスのインスタンスであり、プロパティを有する。W3C WoTのコンテキストでは、これらは、モノおよびそれらの対話アフォーダンスを示す。オブジェクトの正式定義は、第5.2項前付けに記載されている。TD情報モデルの主要要素は、第5.3項クラス定義の中で提示する。デフォルト値がある場合、TD 内で特定のオブジェクトプロパティを省略してもよい。デフォルトのリストは第５．４項デフォルト値の定義にある。

下記のUML図は、TD情報モデルの概要を示している。これは、すべてのクラスとクラス間に存在する関連付けを表として表示している。矢印が示すようにクラスThing(モノ)から始まる。読みやすくするために、図は、4つの基本ボキャブラリのおのおの一個とし、4つの部分に分割されている。

図1 TD中核ボキャブラリ

✧ ✧ ✧

図 2 データ スキーマボキャブラリ

✧ ✧ ✧

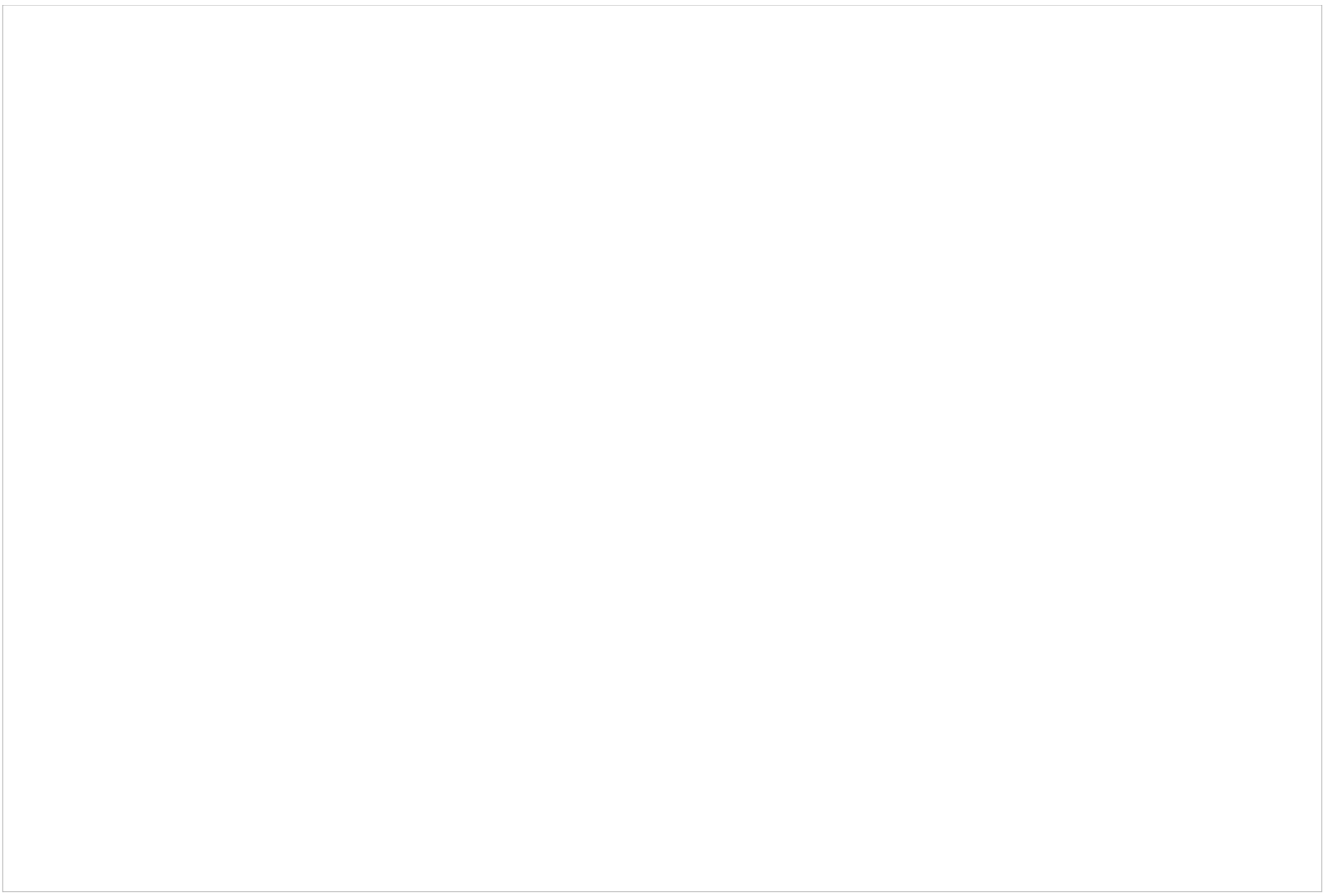

図3 WoTセキュリティボキャブラリ

✧ ✧ ✧

図4 ハイパーメディア制御ボキャブラリ

2. 前付け

ツリーベース文書(すなわち、生JSON処理)および豊富な意味論的ウェブツーリング(すなわち、JSON–LD処理)に関する簡易規則の両方による容易な処理が可能なモデルを提供するために、本文書はそれに応じてTD情報モデルを構築するための以下のような形式的な前付けを定義する。

本項における全ての定義は、セットであり、これは、直感的に、それ自体がセットであり得る要素の集合である。全ての任意に複雑なデータ構造は、セットとして定義することができる。特に、**オブジェクト**は、以下のように再帰的に定義されるデータ構造である。

- 用語は、ボキャブラリに属している場合も、また、属さない場合もあるオブジェクトである。
- 名前が用語で、値が別のオブジェクトである名前と値のペアのセットもオブジェクトである。

この定義は、オブジェクトが同じ名前で複数の名前-値ペアを含むことを妨げるものではないが、一般に、本仕様では考慮されない。要素が名前として数字のみを有するオブジェクトは、配列と呼ばれる。同様に、名前として(ボキャブラリに属さない)用語のみの要素を有するオブジェクトは、マップと呼ばれる。マップ内のいくつかの名前–値ペアに現れるすべての名前は、マップの範囲内において一意であると想定される。

さらに、オブジェクトは、いくつかのクラスのインスタンスとすることができる。ボキャブラリ用語で表されるクラスは、まず、シグネチャと呼ばれるボキャブラリ用語セットで定義される。シグネチャが空であるクラスは、シンプルタイプと呼ばれる。

クラスのシグネチャは、クラスをさらに定義する2つの関数、すなわち、**割当関数**および**型関数**を構築することができる。クラスの割当関数は、入力としてクラスのシグネチャのボキャブラリ用語をうけ、true、 または、falseのいずれかを出力として返す。直感的には、割当関数は、クラスの例を挙げるときにシグネチャの要素が必須であるか、または、任意選択であるかを示す。また、クラスの型関数も、クラスのシグニチャのボキャブラリ用語を入力として受け取り、別のクラスを出力として返す。これらの関数は部分的であり、それらのドメインは、定義されているクラスのシグニチャに限定される。

これらの2つの関数に基づいて、**インスタンスの関係**は、オブジェクトとクラスとからなるペアに対して定義することができる。この関係は満たすべき制約と定義される。すなわち、以下の2つの制約の両方ともが満たされる時にオブジェクトはクラスのインスタンスとなる。

- クラスの割り当て関数がtrueを返す全ての用語について、オブジェクトは、名前としてボキャブラリ用語を有する名前–値ペアを一個含んでいるか
- オブジェクトの何らかの名前–値ペアの中で名前として使用されるクラスのシグネチャ内の全てのボキャブラリ用語に対して、そのペアの値は、与えられたボキャブラリ用語に対するクラスの型関数が返すクラスのインスタンスであるのか

上記の定義によれば、オブジェクトは、その構造にかかわらず、あらゆる単純型のインスタンスである。しかし、インスタンス関係のもう一つの定義が単純型のために導入されている。つまり、オブジェクトが所与の語彙形式(例えば、boolean(ブール)型の場合はtrueとfalseであり、unsignedInt（アンサインドイント）型の場合1、2、3など)を持つ用語である場合、そのオブジェクトは単純タイプのインスタンスである。

さらに、**パラメータ化クラス**と呼ばれる追加のクラスを汎用マップおよび配列構造から導出することができる。オブジェクトが、その全ての名前–値ペアの値がこのクラスのインスタンスであるようなマップである場合、そのオブジェクトは、あるクラスのマップ、即ち、あるクラスでパラメータ化されたマップタイプのインスタンスである。同じことが配列にも当てはまる。

最後に、クラスは、前者のあらゆるインスタンスが後者のインスタンスでもある場合、何らかの他のクラスの**サブクラス**である。

上記のすべての定義に従い、TD情報モデルは、クラス名(ボキャブラリ用語)、シグニチャ(ボキャブラリ用語のセット)、割当関数、および型関数を含むクラス定義のセットとして理解されるべきである。これらのクラス定義は、第5.3項クラス定義で表にまとめられている。各表とも、割当列内の値「必須」(あるいは、「オプション」)は、割り当て関数が、対応するボキャブラリ用語に対してtrue（あるいは、false)を返すということを示している。

慣例により、単純型は小文字で始まる名前で表される。TD情報モデルは、XMLスキーマ [xmlschema11-2-20120405]で定義されている以下の単純型を参照する。すなわち、string, anyURI, dateTime, integer, unsignedInt, double, booleanである。これらの定義(すなわち、それらの語彙形式の仕様)は、本TD情報モデルの範囲外である。

さらに、TD情報モデルは、ボキャブラリ用語のペアに関するグローバル関数を定義する。この関数は、クラス名および別のボキャブラリ用語を入力として受け取り、オブジェクトを返す。返されたオブジェクトがnull(ヌル)でない場合、それは、入力クラスのインスタンス内の入力ボキャブラリ用語の何らかの割当の**デフォルト値**を表す。この関数で割当関数に関して上で定義された制約を緩和することが可能となる。すなわち、オブジェクトがすべての必須割当を含む場合、あるいは、デフォルト値が欠けている割当について存在する場合、そのオブジェクトはクラスのインスタンスである。デフォルト値はすべて、表第5.4項デフォルト値定義の表に記載されている。第5.3項クラス定義の各表では、TD情報モデルの クラスと ボキャブラリ用語の対応する組み合わせにデフォルト値がある場合、割当列には値「デフォルトあり」と書かれている。

ここで使用される形式化は、抽象データ構造としてのオブジェクトと、モノのような物理世界オブジェクト間で可能性のある関係を考慮していない。しかし、TD 情報モデルに含まれる全てのボキャブラリ用語をRDFリソースとして再解釈し、物理世界のより大きなモデルに結合する可能性(オントロジー)に対しては考慮された。この点については、付録D.　TDオントロジーで取り扱われる。意味処理の詳細は、付録D.JSON-LDコンテキスト用法と名前空間ＩＲＩ内の資料（例： https://www.w3.org/2019/wot/td）参照。

3. クラス定義

TDプロセッサは、TDが第５．３．１項中核ボキャブラリ定義、第５．３．２項データ・スキーマ・ボキャブラリ定義、第５．３．３項セキュリティボキャブラリ定義および第５．３．４ハイパーメディア制御ボキャブラリ定義で定義されているすべてのクラスに対するクラスインスタンス化制約条件を満足**していなければならない。**

1. 中核ボキャブラリ定義

1. Thing

メタデータおよびインターフェースがWoTTDによって記述される物理エンティティまたは仮想エンティティのアブストラクションであり、仮想エンティティは1つまたは複数のモノで構成される。

ボキャブラリ用語	説明	割当	型
@context（@コンテキスト）	JSON-LD キーワード。TDドキュメント全体で使用される用語と呼ばれる略記名を定義する。	必須	anyURIまたは配列
@type（@タイプ）	JSON-LD キーワード。オブジェクトに意味タグ(またはタイプ) をラベル付けする。	オプション	string（ストリング）または string（ストリング）の配列
id	URI[RFC3986]形式のモノの識別子(例：　安定したURI、一時的で不定なＵＲＩ，　ローカルＩＰアドレスを持つＵＲＩ、ＵＲＮ等)。	必須	anyURI
title(タイトル)	デフォルト言語で人間が読み取り可能なタイトルを提供する(たとえば、UI 表現のテキスト表示)。	必須	string(ストリング)
titles(タイトル)	複数言語の人間が読み取り可能なタイトルを提供する(例えば、異なる言語でのUI表現のためのテキスト表示)。	オプション	MultiLanguage（複数言語）
description(説明)	デフォルト言語で追加の(人間が読み取り可能な) 情報を提供する。	オプション	string(ストリング)
descriptions(説明)	さまざまな言語で(人間が読み取り可能な)情報をサポートするために使用できる。	オプション	MultiLanguage（複数言語）
version(バージョン)	バージョン情報を提供する。	オプション	VersionInfo（バージョン情報）
created(作成)	TD インスタンスが作成された時点の情報を提供する。	オプション	dateTime（日時）

modified(変更)	TD インスタンスが最後に変更された時点の情報を提供する。	オプション	dateTime（日時）
support(サポート)	TD管理者に関する情報をURIスキーム(例えば、mailto[RFC6068]、tel[RFC3966]、https)として提供する。	オプション	anyURI
base(ベース)	TDドキュメント全体にわたってすべての関連URI 参照に使用さるベースURI を定義する。TDインスタンスでは、すべての関連URIは、[RFC3986]で定義されているアルゴリズムを使用して、ベースURIに呼応して解読される。 baseは、@contextで使用されるURIや意味処理がＴＤインスタンスに適用されるときに直接的に関連するLinked Data[LINKED-DATA]グラフ内で使用されるＩＲＩに作用しない。	オプション	anyURI
properties(プロパティ)	モノのすべてのプロパティベースの対話アフォーダンス。	オプション	PropertyAffordance（プロパティアフォーダンス）のマップ
actions(アクション)	モノのすべてのアクションベースの対話アフォーダンス。	オプション	ActionAffordance（アクションフ　ォーダンス）のマップ
events(イベント)	モノのすべてのイベントベースの対話アフォーダンス。	オプション	EventAffordance（イベントフォーダンス）のマップ
links(リンク)	指定されたTDに関連する任意のリソースへのウェブリンクを提供する。	オプション	Link（リンク）の配列
forms(フォーム)	操作の実行方法を記述するフォームハイパーメディア制御のセット。フォームは、プロトコルバインディングのシリアライズである。本バージョンのTDでは、モノレベルで記述することができるすべての動作は、モノのプロパティと一度にひとまとめに対話する方法に関するものである。	オプション	Form（フォーム）の配列

security(セキュリティ)	セキュリティ定義名のセット。セキュリティ定義で定義されている名前から選択される。これら定義名は全て、リソースへアクセスするために満足されていなければならない。	必須	string（ストリング）またはstring　（ストリング）の配列
securityDefinitions (セキュリティ定義)	名前付きセキュリティ構成セット(定義のみ)。security名前ー値ペアで使用しない限り、実際には適用されない。	必須	SecurityScheme（セキュリティスキーム）のマップ

@context 名前-値ペアは、anyURI型の場合は直接的に、あるいは配列型の場合は最初の要素として、anyURI https://www.w3.org/2019/wot/td/v1を含んでいなければならない。@contextが配列である場合、anyURI https://www.w3.org/2019/wot/td/v1の後に、任意の順序でanyURI型またはマップ型の要素が続く可能性もあるが、@context 配列内のすべての名前-値ペアを持つMapは一個のみとすることが推奨される。値が、anyURI型の名前空間識別子であり、名前が、その名前空間を表す用語またはプレフィックスである場合、@context配列に含まれるマップに名前-値ペアが入っている場合もある。名前が用語@languageであり、値が[BCP47]で定義されている整形式言語タグ(例えば、en、de-AT、gsw-CH、zh-Hans、zh-Hant-HK, sl-neids)である場合、@context配列に含まれる一個のマップは、TDのデフォルト言語を定義する名前-値ペアを含んでいるべきである。

すべての人間が読み取り可能なテキストストリングの基本方向の計算は、以下の一連の規則によって定義される。

- 言語タグが与えられない場合、基本方向は、CLDR可能性サブタグ[LDML]のような最も強力なヒューリスティックまたは検出アルゴリズムによって推測されるべきである。
- MultiLanguage(複数言語)マップ以外では、基本方向はデフォルト言語の言語タグから推測できるかもしれない。
- MultiLanguage(複数言語)マップの内部では、名前-値ペアの各値の基本方向は、対応する名前で与えられた言語タグから推測されるかもしれない。
- ある言語を異なる基本方向を持つ複数のスクリプトで書くことができる場合、@languageまたはMultiLanguage(複数言語)マップで与えられる対応する言語タグは、適切な基本方向が推論できるように、スクリプトサブタグを含んでいなければならない。例としてAzeriがあり、これは、ラテン語スクリプトを使用する場合はLTR(az-Latnで指定)、アラビア語スクリプトを使用する場合はRTL(az-Arabで指定)と書かれている。

TDプロセッサは、双方向テキストを処理する際に特殊なケースに気づくべきである。TDプロセッサは、ユーザにストリングを提示するとき、特に周囲のテキストに埋め込むとき(例えば、ウェブユーザインターフェース用)、注意して双方向分離を使用するようにすべきである。混合方向テキストは、言語が適切に識別された場合でも、全言語で起こりうる。

TDプロデューサは、単純なユーザエージェントによってうまく表示できるような方法で混合方向ストリングを提供するよう試みるべきである。例えば、RTLストリングがLTRラン(ラテン語スクリプトにおける数字あるいはブランドあるい商用名など)で始まる場合、ストリングの始めにRLM文字を入れるか、または、双方向制御における逆方向ランをラップすれば、適切な表示を支援することができる。

ウェブ上のストリング:言語と方向のメタデータ[string-meta] は、何らかのガイダンスを提供し、双方向テキストを使用する場合の隠れた危険を多数例証している。

properties, actions, events(プロパティ、アクション、および、イベント)配列で明示的に提供される対話アフォーダンスに加えて、モノは、そのオプションのforms配列でFormインスタンスによって示されるメタ対話も提供することができる。モノインスタンスのforms配列がFormインスタンスを含んでいる場合、その名前opに直接または配列内で割り当てられたストリング値は、readallproperties、writeallproperties、readmultipleproperties、writemultip l epropertiesという操作タイプのうちのいずれかでなければならない。(モノインスタンスのフォーム使用例を参照。)

これらのメタ対話のそれぞれのデータスキーマは、単一のObjectSchemaインスタンス内の各PropertyAffordanceインスタンスのデータスキーマを組み合わせることによって構築され、ObjectSchemaインスタンスのpropertiesマップは、対応するPropertyAffordanceインスタンスの名前によって識別されるPropertyAffordanceの各データスキーマを含んでいる。

特に指定がない限り(例えば、TDコンテキスト拡張子を介して)、readmultipleproperties操作の要求データは、期待されるPropertyAffordanceインスタンス名を含む配列となり、これは、Formインスタンスによって指定されたコンテンツタイプにシリアライズされている。

2. InteractionAffordance

コンシューマに可能な選択肢を示し、コンシューマがそのモノとどのように対話することができるかを示唆するモノのメタデータ。潜在的なアフォーダンスには多くの型があるが、W3C WoTでは、3つの対話アフォーダンス、すなわち、プロパティ、アクション、および、イベントを定義する。

ボキャブラリ用語	説明	割当	型
@type （タイプ）	JSON-LD キーワード。オブジェクトに意味タグ(またはタイプ) をラベル付けする。	オプション	string(ストリング) またはstring（ストリングス)の配列
title(タイトル)	デフォルト言語に基づいて、人間が読み取り可能なタイトルを提供する(たとえば、UI 表現のテキストを表示)。	オプション	string（ストリング）
titles(タイトル)	複数言語の人間が読み取り可能なタイトルを提供する(例えば、異なる言語でのUI表現のためのテキスト表示)。	オプション	MultiLanguage（複数言語）
description(説明)	デフォルト言語で追加の(人間が読み取り可能な) 情報を提供する。	オプション	string（ストリング）
descriptions(説明)	さまざまな言語で(人間が読み取り可能な)情報をサポートするために使用できる。	オプション	MultiLanguage（複数言語）
forms (フォーム)	操作の実行方法を記述するフォームハイパーメディア制御のセット。フォームは、プロトコルバインディングのシリアライズである。	必須	Form(フォーム)の配列
uriVariables	URI テンプレート変数を、データスキーマ宣言に基づいてコレクションとして定義する。	オプション	DataSchema（データスキーマ）のマップ

クラス対話アフォーダンスには、以下のサブクラスがある。

- PropertyAffordance
- ActionAffordance
- EventAffordance

3. PropertyAffordance

モノの状態を公開する対話アフォーダンス。この状態は検索(読み取り)でき、随意に更新(書き込み)することができる。モノは、変更後に新しい状態をプッシュすることによって、プロパティを観察可能にすることを選択することもできる。

ボキャブラリ用語	説明	割当	型
Observable(観察可能)	モノと仲介者を提供するサービアントが、このプロパティのobserveproperty(オブザーブプロパティ) 操作をサポートするプロトコルバインディングを提供すべきかどうかを示すヒント。	オプション	boolean(ブール)

注

プロパティインスタンスは、クラスのデータスキーマのインスタンスでもある。

したがって、type、unit、readOnly、writeOnlyメンバーも含むことができる。

PropertyAffordanceは、InteractiveAffordanceクラスおよびDataSchemaクラスのサブクラスである。FormインスタンスがPropertyAffordanceインスタンス内にある場合、opに割り当てられる値は、readproperty、writeproperty、observeproperty、unobserveproperty、または、これらの組み合わせを含む配列のうちの1つで**なければならない。**

4. ActionAffordance

状態を操作する(例えば、ランプをオンまたはオフする)、または、モノ上のプロセスをトリガする(例えば、次第にランプの明るさを落とす)モノの関数の呼び出しを可能にする対話アフォーダンス。

ボキャブラリ用語	説明	割当	型
input(入力)	アクションの入力データスキーマを定義するために使用。	オプション	DataSchema（データスキーマ）
output(出力)	アクションの出力データスキーマを定義するために使用。	オプション	DataSchema（データスキーマ）
safe(安全)	アクションが安全(=true) であるかどうかシグナルする。アクションを呼び出したときに内部状態に変化がないかどうか示すために使用される。その場合、応答は例としてキャッシュすることができる。	デフォルトあり	boolean(ブール)
idemotent(冪等)	アクションが冪等（idempotent） (=true) かどうかを示す。同じ入力で、（該当する場合）同じ結果を生じるアクションの呼び出しを繰り返し行えるかどうかを通知する。	デフォルトあり	boolean(ブール)

ActionAffordanceは、InteractioniAffordanceクラスのサブクラスである。フォームインスタンスがこのActionAffordanceインスタンス内にあるとき、opに割り当てられた値は、invokeaction（アクション呼び出し）で**なければならない。**

5. EventAffordance

イベントソースを記述する対話アフォーダンスであり、非同期的にイベント・データを消費者にプッシュする(例えば、オーバーヒート警報)。

	説明	割当	型

ボキャブラリ用語			
subscription(サブスクリプション)	サブスクリプション時に渡す必要があるデータを定義する(ウェブフックを設定するためのフィルタあるいはメッセージフォ-マットなど)。	オプション	DataSchema（データスキーマ）
dara(データ)	モノによってプッシュされるイベントインスタンスメッセージのデータスキーマを定義する。	オプション	DataSchema（データスキーマ）
cancellation(キャンセル)	サブスクリプションをキャンセルするために渡す必要があるデータを定義する(ウェブフックを削除するための特定のメッセージなど)。	デフォルトあり	DataSchema（データスキーマ）

EventAffordanceは、InteractionAffordanceクラスのサブクラスである。Formインスタンスが EventAffordanceインスタンス内にあるとき、opに割り当てられた値は、配列内のsubscribeevent、unsubscribeevent、または、両方のうちのいずれかでなければならない。

6. VersionInfo

TDドキュメントのバージョン情報を提供するモノのメタデータ。必要に応じて、ファームウェアおよびハードウェアバージョン(TD名前空間外における用語定義)などの追加バージョン情報は、TDコンテキスト拡張子メカニズムを介して拡張することができる。

ボキャブラリ用語	説明	割当	型
Instance(インスタンス)	このTD インスタンスのバージョンID を提供する。	必須	string（ストリング）

VersionInfoクラス のインスタンス内の値は、意味論的バージョニングパターンに従うことが推奨される。ここで、ドットで区切られた3 つの数字のシーケンスは、メジャーバージョン、マイナーバージョン、パッチバージョンをそれぞれ示す。詳細は[SemVer]を参照。

7. MultiLanguage

[BCP47]に記述されている言語タグによって識別される異なる言語で人間が読み取り可能なテキストを提供するマップ。例えば、TDインスタンス内のコンテナの使用法に関しては、第6.3.2項人間が読み取り可能なメタデータ参照。

MultiLanguageマップの各名前は、[BCP47]で定義されているような言語タグでなければならない。

MultiLanguageマップの各値は、string型でなければならない。

2. データ・スキーマ・ボキャブラリ定義

データスキーマ定義は、JSON スキーマ[JSON-SCHEMA] によって定義された用語の非常に一般的なサブセットを反映している。TDインスタンス内のデータスキーマ定義は、この定義されたサブセットに限定されておらず、第7項 TDコンテキスト拡張子で解説されている追加用語のためのＴＤコンテキスト拡張子を使用しているJSONスキーマで見られる追加の用語を使用してもよいことに留意されたい。あるいは、これら用語は、ＴＤプロセッサに意味的に無視される。（意味処理の詳細は、付録D.JSON-LDコンテキスト用法と名前空間ＩＲＩ内の資料（例：https://www.w3.org/2019/wot/td）参照。）

1. DataSchema

使用されるデータフォーマットを記述するメタデータ。確認のために使用することができる。

ボキャブラリ用語	説明	割当	型
@type　（タイプ）	オブジェクトに意味論タグ(または型)をラベル付けするためのJSON-LD キーワード。	オプション	string（ストリング） またはstring（ストリング）の配列
title(タイトル)	デフォルト言語に基づいて、人間が読み取り可能なタイトルを提供する(たとえば、UI 表現のテキストを表示)。	オプション	string（ストリング）
titles(タイトル)	複数言語の人間が読み取り可能なタイトルを提供する(例えば、異なる言語でのUI表現のためのテキスト表示)。	オプション	MultiLanguage（複数言語）
description(説明)	デフォルト言語で追加の(人間が読み取り可能な) 情報を提供する。	オプション	string（ストリング）
descriptions(説明)	さまざまな言語で(人間が読み取り可能な)情報をサポートするために使用できる。	オプション	MultiLanguage（複数言語）
type(型)	操作の実行方法を記述するフォームハイパーメディア制御のセット。フォームは、プロトコルバインディングのシリアライズである。	オプション	string（ストリング） (object, array, string, number, integer, boolean, null のいずれか)
Const	URI テンプレート変数を、データスキーマ宣言に基づいてコレクションとして定義する。	オプション	任意の型
unit(単位)	国際科学、エンジニアリング、ビジネスなどで使用される単位情報を提供する。	オプション	string（ストリング）
one of（の一つ）	配列内の指定された一つのスキーマに対してデータが有効であることを保証するために使用される。	オプション	DataSchema(データスキーマ)の配列

enum	配列として提供される制限された値のセット。	オプション	任意の型の配列
readOnly（読取り専用）	プロパティ対話/値が読み取り専用(=true) かそうではないか(=false) を示すヒントとなるブール値。	デフォルトあり	boolean(ブール)
writeOnly（書き込み専用）	プロパティ対話/値が書き込み専用(=true) かそうではないか(=false)を示すヒントであるブール値。	デフォルトあり	boolean(ブール)

クラスDataSchema には、以下のサブクラスがある。

- ArraySchema
- BooleanSchema
- NumberSchema
- IntegerSchema
- ObjectSchema
- StringSchema
- NullSchema

フォーマットストリング値は、[JSON-SCHEMA] (特に第7.3項定義フォーマット) で定義されている固定値セットとそれに対応するフォーマットルールから判別できる。サービアントは、format値を使用して、それに応じて追加の確認を実行.**しても良い。**既知の値セットで見つからない値がformatに割り当てられる場合、そのような確認は成功**すべきである。**

2. ArraySchema

配列型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値のarrayによって示される。

ボキャブラリ用語	**説明**	**割当**	**型**
Items(項目)	配列の特性を定義するために使用される。	オプションの配列	DataSchemaまたはDataSchemaの配列
minItems（最小項目）	配列に含めなければならない項目の最小数を定義する。	オプション	unsignedInt
maxItems（最大項目)	配列に含める必要がある項目の最大数を定義する。	オプション	unsignedInt

3. BooleanSchema

boolean 型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値booleanによって示される。

4. NumberSchema

number型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値numberによって示される。

ボキャブラリ用語	**説明**	**割当**	**型**
minimum(最小)	最小数値を指定する。関連付けられた数値型または整数型のみに適用される。	オプション	double(ダブル)
maximum(最大)	最小数値を指定する。関連付けられた数値型または整数型のみに適用される。	オプション	double(ダブル)

5. IntegerSchema

integer型のデータを記述するメタデータ。このサブクラスは、DataSchema インスタンスでtype に割り当てられた値integer によって示される。

ボキャブラリ用語	説明	割当	型
minimum(最小)	最小数値を指定する。関連付けられた数値型または整数型のみに適用される。	オプション	integer(整数)
maximum(最大)	最小数値を指定する。関連付けられた数値型または整数型のみに適用される。	オプション	integer(整数)

6. ObjectSchema

object型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値objectによって示される。

ボキャブラリ用語	説明	割当	型
properties(プロパティ)	データスキーマのネストされた定義。	オプション	DataSchemaのマップ
required(要求される)	オブジェクト型のどのメンバーが必須であるかを定義する。	オプション	string(ストリング)の配列

7. StringSchema

string型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値stringによって示される。

8. NullSchema

null型のデータを記述するメタデータ。このサブクラスは、DataSchemaインスタンス内のtypeに割り当てられた値nullによって示される。このサブクラスは、1つの許容可能な値、すなわち、nullのみを記述する。これは、情報として使われる場合、データもnullになれるというoneOf宣言の一部として使用することができる。

3. セキュリティボキャブラリ定義

本仕様は、W3C WoTのプロトコルバインディングとして適格なプロトコルに直接組み込まれるか、または、これらのプロトコルと組み合わて広く使用されている十分に確立されたセキュリティメカニズムから選ばれたメカニズムを提供する。現在のHTTPセキュリティスキームセットは、部分的にOpenAPI 3.0.1に基づいている([OPENAPI]も参照)。しかしながら本仕様で与えられたHTTPセキュリティスキーム、ボキャブラリ、構文は、OpenAPIと多くの類似点があるが、それらは互換性を持たない。

1. SecurityScheme

セキュリティメカニズムの構成を記述するメタデータ。名前schemeに割り当てられた値は、第5項 TD情報モデルで定義される標準ボキャブラリ、あるいは、TDコンテキスト拡張子のいずれかで、TDに含まれるボキャブラリの中で定義されなければならない。

ボキャブラリ用語	説明	割当	型
@type（@型）	オブジェクトに意味タグ(またはタイプ) をラベル付けするためのJSON-LDキーワード。	オプション	string（ストリング） またはstring（ストリング）の配列
ｓchema(スキーマ)	構成されているセキュリティメカニズムの識別。	オプション	string(ストリング） （nosec, basic,cert,digest,bearer,pop,psk,public,oauth2,orapikey)
description(説明)	デフォルト言語に基づいて、追加の(人間が読み取り可能な)情報を提供する。	オプション	string（ストリング）
descriptions(説明)	さまざまな言語の(人間が読み取り可能な)情報をサポートするために使用できる。	オプション	MultiLanguage（複数言語）
proxy(プロキシ)	このセキュリティ構成がアクセスを提供するプロキシーサーバのURI。指定されていない場合、対応するセキュリティ構成はエンドポイント用である。	オプション	anyURI

クラスSecuritySchemeは、以下のサブクラスを有する。

- NoSecurityScheme
- BasicSecurityScheme
- DigestSecurityScheme
- APIKeySecurityScheme
- BearerSecurityScheme
- CertSecurityScheme
- PSKSecurityScheme
- PublicSecurityScheme
- PoPSecurityScheme
- OAuth2SecurityScheme

2. NoSecurityScheme

リソースにアクセスするために必要とされる認証または他のメカニズムがないことを示すボキャブラリ用語によって識別されるnosec (すなわち、「scheme」:「nosec」)に対応するセキュリティ構成。

3. BasicSecurityScheme

暗号化されていないユーザ名およびパスワードを使用するボキャブラリ用語basic(すなわち、「scheme」:「basic」)によって識別される基本認証[RFC7617]セキュリティ構成。本スキームは、例えば、TLSのように機密性を提供する他の何らかのセキュリティメカニズムと共に使用されるべきである。

ボキャブラリ用語	説明	割当	型
In(内)	セキュリティ認証情報の場所を指定する。	デフォルトあり	string（ストリング） (header,query,body,cookieのいずれか)
name(名前)	クエリ、ヘッダー、あるいは、クッキーのパラメータ名。	オプション	string（ストリング）

4. DigestSecurityScheme

ボキャブラリ用語digest (すなわち、「scheme」:「digest」)によって識別されるダイジェストアクセス認証[RFC7616]セキュリティ構成。本スキームは、基本認証に似ているが、中間者攻撃を回避する機能が追加されている。

ボキャブラリ用語	説明	割当	型
qop	保護の質。	デフォルトあり	string（ストリング）（authあるいは authintのいずれか）
in (内)	セキュリティ認証情報の場所を指定する。	デフォルトあり	string（ストリング） (header,query,body,cookieのいずれか)
name(名前)	クエリ、ヘッダー、あるいは、クッキーのパラメータ名。	オプション	string（ストリング）

5. APIKeySecurityScheme

ボキャブラリ用語apikey (すなわち、「scheme」:「apikey」)によって識別されるAPIキー認証セキュリティ構成。これは、アクセストークンが不透明であり、標準トークンフォーマットを使用していない場合に用いる。

ボキャブラリ用語	説明	割当	型
in (内)	セキュリティ認証情報の場所を指定する。	デフォルトあり	string（ストリング） (header,query,body,cookieのいずれか)
name(名前)	クエリ、ヘッダー、あるいは、クッキーのパラメータ名。	オプション	string（ストリング）

6. BearerSecurityScheme

ベアラトークンがOAuth2とは別個に使用される状況におけるボキャブラリ用語bearer(すなわち、「scheme」:「bearer」)によって識別されるベアラトークン[RFC6750]セキュリティ構成。本スキームは、ベアラトークンがOAuth2とは独立して使用される状況のためのものである。oauth2スキームが指定されている場合は、通常、本スキームを暗示・指定する必要はない。Formatに関しては、値jwtは[RFC7519]への適合性を示し、jwsは[RFC7797]への適合性を示し、cwtは[RFC8392]への適合性を示し、jweは[RFC7516]への適合性を示し、algの値はこれらの標準と整合して解釈される。ベアラトークンのための他のフォーマットおよびアルゴリズムは、ボキャブラリ拡張において指定**もできる。**

ボキャブラリ用語	説明	割当	型
authorization(許可)	許可サーバのURI。	オプション	anyURI
alg(アルゴリズム)	エンコーディング、暗号化、またはダイジェストアルゴリズム。	デフォルトあり	string(ストリング) (MD5,ES256,ES512-256)
format(フォーマット)	セキュリティ認証情報のフォーマットを指定する。	デフォルトあり	string(ストリング) (jwt,cwt,jwe,jws)
in（内）	セキュリティ認証情報の場所を指定する。	デフォルトあり	string(ストリング) (header,query,body,cookieのいずれか)
name(名前)	クエリ、ヘッダー、あるいいは、クッキーのパラメータ名。	オプション	string(ストリング)

7. CertSecurityScheme

本項は危険な状態にある。

ボキャブラリ用語cert (すなわち、「scheme」:「cert」)によって識別される[X509V3]に準拠する証明書ベースの非対称キーセキュリティ構成。

ボキャブラリ用語	説明	割当	型
identity（ID）	選択または確認に使用できる情報を提供する識別子。	オプション	string(ストリング)

8. PSKSecurityScheme

ボキャブラリ用語psk (すなわち、「scheme」:「psk」)によって識別される事前共有キー認証セキュリティ構成。

ボキャブラリ用語	説明	割当	型
identity（ID）	選択または確認に使用できる情報を提供する識別子。	オプション	string(ストリング)

9. PublicSecurityScheme

本項は危険な状態にある。

ボキャブラリ用語public (すなわち、「scheme」:「public」)によって識別される生の公開キー非対称鍵セキュリティ構成。

ボキャブラリ用語	説明	割当	型
identity（ID）	選択または確認に使用できる情報を提供する識別子。	オプション	string(ストリング)

10. PoPSecurityScheme

本項は危険な状態にある。

ボキャブラリ用語pop(すなわち、「scheme」:「pop」)によって識別されるPoP(Proof-of-possession)トークン認証セキュリティ構成。jwtは[RFC7519]との適合性を示し、jwsは[RFC7797]との適合性を示し、cwtは[RFC8392]との適合性を示し、jweは[RFC7516]との適合性を示し、algの値はこれらの標準と整合して解釈される。PoPトークンのための他のフォーマットおよびアルゴリズムは、ボキャブラリ拡張において指定**もできる。**

ボキャブラリ用語	説明	割当	型
authorization(許可)	許可サーバのURI。	オプション	anyURI
alg(アルゴリズム)	エンコーディング、暗号化、またはダイジェストアルゴリズム。	デフォルトあり	string(ストリング) (MD5,ES256,ES512-256)
format(フォーマット)	セキュリティ認証情報のフォーマットを指定する。	デフォルトあり	string(ストリング) (jwt,cwt,jwe,jws)
in（内）	セキュリティ認証情報の場所を指定する。	デフォルトあり	string(ストリング) (header,query,body,cookieのいずれか)
name(名前)	クエリ、ヘッダー、あるいいは、クッキーのパラメータ名。	オプション	string(ストリング)

11. OAuth2SecurityScheme

本項は危険な状態にある。

ボキャブラリ用語oauth2 (すなわち、「scheme」:「oauth2」) によって識別される [RFC6749] および [RFC8252] に準拠するシステムのOAuth2認証セキュリティ構成。implicitフローについては、authorizationが含**まれなければならない。**passwordとclientフローについて、tokenが含まれ**なければならない。**codeフローについては、authorizationとtokenの両方が含まれな**ければならない。**SecuritySchemeにscopeが定義されていない場合、それらは空であるとみなされる。

ボキャブラリ用語	説明	割当	型
authorization(許可)	許可サーバのURI。	オプション	anyURI
token(トークン)	トークンサーバのURI.	オプション	anyURI
refresh(リフレッシュ	リフレッシュサーバのURI	オプション	anyURI
scopes(範囲)	配列として提供される許可範囲 ID セット。これらは、クライアントがどのリソースにどのようにアクセスできるかを識別するために、許可サーバによって返されるフォームに関連付けられたトークンとして提供される。フォームに関連付けられている値は、そのフォーム上でアクティブな OAuth2SecurityScheme で定義される値の中から選択する必要がある。	オプション	string(ストリング)またはstring(ストリング)の配列
flow(フロー)	許可フロー	必須	string(ストリング) (implicit,password,client,codeのいずれか)

4. ハイパーメディア制御ボキャブラリ定義

本モデルは、(タイプされた) Webリンクおよびモノが公開されるWebフォームの表現を提供する。Linkクラス定義は、Web Linking [RFC8288]で定義されている非常に一般的な用語のサブセットを反映している。定義された用語は、例えば、Switchというモノによって制御されるLampというものなどの別のモノとの関係を記述するために使用することができる。Formクラスは、モノ(及び他のウェブリソース)の状態を操作するために新たに導入されたハイパーメディア制御に対応する。

1. Link(リンク)

リンクは、「リンクコンテキストは、リンクターゲットにリレーション型リソースを有する」という形式のステートメントとみなすことができ、ここでオプションのターゲット属性によって、リソースをさらに記述することができる。

ボキャブラリ用語	説明	割当	型
href	リンクのターゲットIRI、または、フォームの送信ターゲット。	必須	anyURI
type(型)	リンクをデリフェレンスした結果のメディアタイプ[RFC2046]が何であるべきかを示すヒントを提供するターゲット属性。	オプション	string(ストリング)
rel	リンクリレーション型はリンクの意味を識別する。	オプション	string(ストリング)
anchor(アンカー)	所与のＵＲＩあるいはＩＲＩを使って、リンクコンテキスト(デフォールトで、それ自体がそのidにより識別されるモノ)を無効にする。	オプション	anyURI

2. Form

フォームは、「フォームコンテキスト上でオペレーションタイプ操作を実行するために、送信ターゲットへ要求メソッドを要求する」というステートメントとみなすことができ、オプションのフォームフィールドは、必要な要求をさらに記述することができる。TDでは、フォームコンテキストは、Properties、Actions、およびEventsなどの周囲のオブジェクト、または、メタ対話のためのモノ自体となる。

ボキャブラリ用語	説明	割当	型
op	フォームで記述された操作を実行する意味・意図を示す。例えば、Property対話は、getおよびset動作を可能にする。プロトコルバインディングは、ゲットオペレーションのためのフォームと、セットオペレーションのための別のフォームを含むことができる。op属性は、どのフォームがどのフォームのためのものであるかを示し、クライアントが必要とされる動作のための正しいフォームを選択することができるようにする。opには、操作の意味意図をそれぞれが表す1つまたは複数の対話動詞を割り当てることができる。	デフォルトあり	string（ストリング）またはstring（ストリング）の配列 (readproperty、writeproperty、observeproperty、unobserveproperty、invokeaction、subscribeevent、unsubscribeevent、readallproperties、writeallproperties、readmultipleproperties、writemultiplepropertiesのいずれか)
href	リンクのターゲットIRI、または必須フォームの送信ターゲット。		anyURI
contentType（コンテンツタイプ）	メディアタイプ (例: 'text/plain') およびメディアタイプ [RFC2046]の潜在パラメータ (例: 'charset=utf-8') に基づいて、コンテンツタイプを割り当てる。	デフォルトあり	string(ストリング)
contentCoding（コンテンツコーディング）	コンテンツコーディング値は、 表現に使用されている、あるいは、適用可能な符号化変換を示す。コンテンツコーディングは、主に、そのメディアタイプのアイデンティティを失うことなく、かつ、情報を失うことなく、表現が圧縮、または、そうでなければ有用に変換を可能にするために使用される。コンテンツコーディングの例は、「gzip」、「deflate」など。	オプション	string(ストリング)

subptotocol(サブプロトコル)	複数のオプションがある場合に、特定プロトコルに対して対話が実行されるそのメカニズムを示す。例えば、HTTPおよびイベントの場合、これは、ロングポーリング(longpoll)、ウェブサブ[websub](websub),サーバが送信したイベント[eventsource](sse)などいくつか利用可能なメカニズムのうちのどれを非同期通知に使用すべきかを示す。サブプロトコルの選択に制限はなく、他のメカニズムもこのサブプロトコル用語によってアナウンスできることに留意されたい。	オプション	string（ストリング） (例: longpoll, websub、または sse)
security(セキュリティ)	securityDefinitionsで定義されている。定義から選択されたセキュリティ定義名セット。リソースへのアクセスのためにはこれら全てが満たされなければならない。	オプション	string（ストリング） またはstring （ストリング）の配列
scopes(範囲)	配列として提供される許可範囲 ID セット。これらは、クライアントがどのリソースにどのようにアクセスできるかを識別するために、許可サーバによって返される、フォームに関連付けられたトークンとして提供される。フォームに関連付けられている値は、そのフォーム上でアクティブな OAuth2SecurityScheme で定義される値の中から選択する必要がある。	オプション	string（ストリング） またはstring （ストリング）の配列

response(応答)	本オプション用語は、たとえば、以下の場合に使用することができる。出力通信メタデータが入力メタデータとは異なる(例えば、出力コンテンツ型が入力コンテンツ型とは異なる)。応答名には、応答メッセージにのみ有効なメタデータが含まれる。	オプション	ExpectedResponse（期待される応答）

フォームの可能な操作型リストは確定されたものである。本仕様書の本バージョンでは、[WOT- ARCHITECTURE]に記述されたWoT対話モデルをインプリメンテーションするために必要な周知の型のみを含んでいる。本標準の将来バージョンはこのリストを拡張したものになるかもしれない。が、操作型は、サービアントが任意に設定する**べきではない。**

オプションのresponse名前-値ペアは、期待される応答メッセージのメタデータを提供するために使用することができる。中核ボキャブラリでは、そのペアはコンテンツタイプ情報を持つのみであるが、TDコンテキスト拡張子を適用することができる。応答名-値ペアが与えられていない場合、応答のコンテンツタイプは、フォームインスタンスに割り当てられたコンテンツタイプと同じであると仮定しなければならない。ExpectedResponseクラス内のcontentTypeは、デフォルト値を持っていないということに留意されたい。例えば、フォームのコンテンツタイプの値がapplication/xmlである場合、応答のコンテンツタイプの仮定値もapplication/xmlとなる。

いくつかの使用事例では、入出力データは、異なる形式、例えば、JSONを受け入れるが画像を返すアクションで表される事もある。そのような場合、オプションのresponse名前-値ペアは、期待される応答のコンテンツタイプを記述することができる。期待される応答のコンテンツタイプがフォームのコンテンツタイプと異なる場合、フォームインスタンスは、名前responseを持つ名前-値ペアを含んでい**なければならない。**例えば、ActionAffordanceは、その入力データとしてapplication/jsonのみを受け入れることができ、一方、その出力データとしてimage/jpegコンテンツタイプで応答する。その場合、コンテンツタイプは異なり、response名前-値ペアが、応答コンテンツタイプ(image/jpeg)情報をコンシューマに提供するために使用されなければならない。

contentCodingプロパティとして可能な値は、例えば、IANA HTTPコンテンツコーディングレジストリで見つけることができる。

3. ExpectedResponse

期待される応答メッセージを記述する通信メタデータ。

ボキャブラリ用語	**説明**	**割当**	**型**
contentType（コンテンツタイプ）	メディアタイプ(例：'text/plain') およびメディアタイプ [RFC2046]の潜在パラメータ (例: 'charset=utf-8') に基づいて、コンテンツタイプを割り当てる。	必須	string(ストリング)

5.4 デフォルト値定義

TD内の割当が欠落している場合、TDプロセッサは、第5.4項デフォルト値定義内の表に示されているデフォルト値割当に従わなければならない。

下表は、TD情報モデルで定義されているデフォルト値のすべてである。

クラス	ボキャブラリ用語	デフォルト値	注釈
Form	readOnly	false	
DataSchema	writeOnly	false	
DataSchema	safe	false	
ActionAffordance	idempotent	false	
Form	op	readpropertyおよびwriteproperty要素を持つstring(ストリング)の配列。	PropertyAffordanceインスタンス内で定義されている場合
Form	op	invokeAction	ActionAffordanceインスタンス内で定義されている場合
Form	op	subscribeevent	EventAffordanceインスタンス内で定義されている場合
BasicSecurityScheme	in	header	
DigestSecurityScheme	in	header	
BearerSecurityScheme	in	header	
PoPSecurityScheme 本機能は危険な状態である。	in	header	
APIKeySecurityScheme	in	query	
DigestSecurityScheme	qop	auth	
BearerSecurityScheme	alg	ES256	
PoPSecurityScheme 本機能は危険な状態である。	alg	ES256	
BearerSecurityScheme	format	jwt	
PoPSecurityScheme 本機能は危険な状態である。	format	jwt	

6. TD表現形式

WoTTDは、モノを表し、第5項TD情報モデルに基づいてモデル化され、構成されている。本項では、TD 情報モデルが定義するクラスThing のインスタンスのシリアライズであるモノのJSONベースの表現形式を定義している。

TDプロセッサは、第6.1項 JSONタイプへのマッピングと第6.3項情報モデルシリアライズに記載されている規則に従って、JSONフォーマット[RFC8259]にTDをシリアルライズする、また・あるいは、そのフォーマットからTDをシリアライズ解除することができ**なければならない。**

TD 情報モデルのJSONシリアライズは、意味評価を簡素化するために、JSON-LD 1.1[jsonld11]の構文と整合化されている。従って、TD表現フォーマットは、生のJSONとして、または、第D項RDFへの変換でさらに詳述するように、JSON–LD 1.1プロセッサを用いて処理することができる。意味処理の詳細は、付録D.JSON-LDコンテキスト用法と名前空間ＩＲＩ内の資料（例： https://www.w3.org/2019/wot/td）参照。

相互運用可能な国際化をサポートするために、TDは、オープンエコシステムのためのRFC8259[RFC8259]の第８項で定義されている要件に従ってシリアライズされ**なければならない。**要約すると、以下が要求される：

- TDはUTF-8[RFC3629]を使って符号化され**なければならない。**
- インプリメンテーションは、TDドキュメントの先頭にBOM（byte order mark）(U+FEFF)を追加し**てはならない。**
- TDプロセッサは、BOMをエラーとして扱うのではなく、その存在を無視**してもよい。**

1. JSON タイプへのマッピング

TD情報モデルは、モデルオブジェクトとJSONタイプ間で容易にマッピングできるように構築される。すべてのクラスはJSONオブジェクトへのマップを例に挙げ、クラスインスタンスの各名前–値ペアはJSONオブジェクトのメンバーである。

第5.3項クラス定義(すなわち、string、anyURI、dateTime、integer、unsignedInt、double、boolean)で言及されているシンプルタイプはすべて、下記の規則に従って、基本JSONタイプ(string、number、boolean)へマップする。これらの規則、名前-値ペアに適用される。

- string型またはanyURI型の値は、JSONストリングとしてシリアライズされ**なければならない。**
- dateTime型の値は、[RFC3339]で指定された"日-時"形式でJSONストリングとしてシリライズされなければならない。例には、2019-05-24T13:12:45Z および2015-0711T09:32:26+08:00などがある。dateTime型の値は、オフセットの代わりにUTCタイムゾーンを表すリテラルzを使用**すべきである。**
- integer型またはunsignedInt型の値は、小数部または指数部のないJSON数としてシリアライズし**なければならない。**
- double型の値は、JSON番号としてシリアライズし**なければならない。**
- boolean型の値は、JSONブールとしてシリアライズし**なければならない。**

TD情報モデルの複合型(すなわち、配列、マップ、およびクラスインスタンス)はすべて、以下の規則に従って、構成されたJSON型(配列およびオブジェクト)にマッピングされる。

- 型Arrayの値は、JSON配列としてシリアライズされ**なければならない。**名前-値ペアの各値は、そのペアの数値名で順序付けられたJSON配列の要素とする。
- 型Mapの値は、JSONオブジェクトとしてシリアライズされなければならず、名前-値ペアの各値は、JSONオブジェクトのメンバーとする。
- クラスインスタンスはJSONオブジェクトとしてシリアライズされ**なければならない。**これは、個々に第6.3項 情報モデルシリアライズで与えられている詳細な規則に従って行われなければならない。

2. デフォルト値の省略

TDシリアライズでは、第5.4項デフォルト値の定義の表のようにデフォルト値が定義されているボキャブラリ用語を省略してもよい。

以下の例は、デフォルト値(=チェックがついたチェックボックス) を持つメンバーも含めるためのチェックボックスを持つ例1のTD インスタンスである。これらのメンバーは、TDシリアライズを単純化するために省略することができる(=チェックがついていないチェックボックス)。TDプロセッサは、あたかも所与のデフォルト値によって明らかに存在するかのように、全く同じようにこれらの省略されたメンバー

を解釈することに留意されたい。

例3

デフォルト値で


```
{
    "@context": https://www.w3.org/2019/wot/td/v1,
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing”,
    "securityDefinitions": {
        "basic_sc": {
            “Scheme”:”basic”,、
            "in": "header"
        }
    },
    “security”: [
        "basic_sc"
    ],
    "properties": {"status": {
            "type": "string", "readOnly" : false, "writeOnly" : false, "forms": [{
                "op": [
                    "readproperty"、"writeproperty"
                ],
                “href”:”https://mylamp.example.com/status」,
                "contentType": "application/json"
            }]
        }
    },
    "actions": {
        "toggle": {
            "safe": false,
            "idempotent": false,
            "forms": [{
                "op": "invokeaction",
                「href」:”https://mylamp.example.com/toggle2,
                "contentType": "application/json"
            }]
        }
    },
    "events": {
        "overheating": {"data": {
                "type": "string",
                "readOnly" : false, "writeOnly" : false
            },
```

```
            "forms": [{
                "op": "subscribeevent",
                “href”:https://mylamp.example.com/oh,,
                "contentType": "application/json",
                "subprotocol": "longpoll"
            }]
        }
    }
}
```

使用されるプロトコルバインディングに応じて、追加のプロトコル固有のボキャブラリ用語が適用されることがあるということに留意されたい。これらは、関連するデフォルト値を持っており、したがって、本サブ項で説明されているように省略することもできる。さらなる情報は、第8.3項プロトコルバインディングに記載されている。

3. 情報モデルのシリアライズ

1. モノのルートオブジェクト

TDは、型Thingのオブジェクトをルートとするデータ構造である。その代り、TDのJSONシリアライズは、TD情報モデルから構築された構文ツリーのルートであるJSONオブジェクトである。

TDシリアライズのルート要素は、@contextという名前のメンバーと、https://www.w3.org/2019/wot/td/v1と等しいか、それを含むストリングまたは配列型の値を含むJSONオブジェクトで**なければならない。**

一般に、このURIは、本仕様が定義するTD表現フォーマットバージョンを識別するために使用される。JSON-LD 処理[json-ld11] の場合、このURI はTDコンテキストファイルを指定する。配列型の@contextは、TDコンテキスト拡張子を示す(詳細は第7項ＴＤコンテキスト拡張子を参照)。

例4

```
{
    "@context": https://www.w3.org/2019/wot/td/v1,
    ...
}
```

名前がThingのシグニチャ内のボキャブラリ用語である場合、Thingのインスタンスの名前-値ペアは、ルートオブジェクトのJSONメンバーとしてシリアライズされな**ければならない。**すべての必須およびオプションメンバーを含むシリアライズされたルートオブジェクトのTDスニペットを以下のとおり：

例5: モノシリアライズの例

```
{
    "@context": https://www.w3.org/2019/wot/td/v1,
    "@type": "Thing",
    "id": "urn:dev:ops:32473-Thing-1234",
    "title": "MyThing",
    "titles": {...},
    "description": "Human readable information.",
    "descriptions": {...},
    "support": mailto:support@example.com,
    "version" : {...},
    "created" : "2018-11-14T19:10:23.824Z",
    "modified" : "2019-06-01T09:12:43.124Z"、"securityDefinitions": {...}、"security": ...,
    "base":https://servient.example.com/,
    "properties": {...},
    "actions": {...},
    “events”: {...},
    “Link”: {...},
    "forms": {...}
}
```

クラス Thingのインスタンス内のversion、securityDefinitions、properties、actions、eventsに割当てられたすべての値は、JSONオブジェクトとしてシリアライズされ**なければならない。**

クラスThingのインスタンス内のlinksとformsに割り当てられた値はすべて、第6.3.8項linksと第6.3.9項formsでの定義のようにJSONオブジェクトを含むJSON配列としてシリアライズされなければならない。

クラスThingのインスタンス内でsecurityに割り当てられた値は、JSONストリングとして、または、要素がJSONストリングのJSON配列としてシリアライズされ**なければならない。**

2. 人間が読み取り可能なメタデータ

titleおよびdescriptionという名前のJSONメンバーは、人間が読み取り可能なメタデータを表示するTDドキュメント内で使用される。これらは、TDドキュメントを確認する開発者のためのコメントとして、または、ユーザインターフェースのための表示テキストとして使用することができる。

第5.3.1.1項Thingで定義されているように、人間が読み取り可能なメタデータを表示するために使用される基本テキストの方向は、最初の強力な規則などのヒューリスティックスを使用して推定するか、言語情報から推論することができる。TDドキュメントでは、デフォルト言語は、@context内の@languageに割り当てられた値によって定義され、これは、必要に応じてスクリプトサブタグと共に、ベーステキスト方向を決定するために使用することができる。しかし、人間が読み取り可能なテキストを解釈するとき、各人間が読み取り可能なストリング値は、独立して処理され**なければならない。**言い換えれば、TDプロセッサは、1つのストリングから別のストリングへの方向を変化させること、または、TD内の他の場所から別のストリングの方向を推論することはできない。

注

ウェブ上のストリング[string-meta]は、基本テキストの方向を決定する手段として、強力な最初の推論、また、言語ベースの推論の両方を示唆している。TDフォーマットがJSON–LD 1.1[json–ld11]に基づいており、これは、現在、明示的な方向メタデータを欠いているので、これらのアプローチは、現在、本公開時点で適切であると考えられている。しかしながら、JSON-LD 1.1が[string-meta]が推奨するような明示的な基本方向メタデータのサポートを採用する場合、TDフォーマットは、その機能を利用するために更新されるべきである。

titleおよびdescriptionを使用するTDスニペットを以下のとおりである。デフォルト言語は、@context 配列内のJSON オブジェクト内の @language メンバーの定義によってen に設定される。

例6

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"@language" : "en"}
    ],
    "title": "MyThing",
    “description”:”human readable information.”,
    ...
    "properties": {
        "on": {
            “title” : “On/Off”,
            "type": "boolean",
            "forms": [...]
        },
        "status": {
            “title” : “Status”,
            "type": "object",
            ...
            "forms": [...]
        }
    },
    ...
}
```

titlesおよびdescriptionsという名前のJSONメンバーは、一個のTDドキュメント内において複数言語で人間が読み取り可能なメタデータを提供するためにTDドキュメント内で使用される。MultiLanguageマップのすべての名前-値ペアは、JSONオブジェクトのメンバーとしてシリアライズされ**なければならない。**ここで、名前は[BCP47]によって定義された周知の言語タグであり、値は、そのタグによって示された言語の人間が読み取り可能なストリングである。詳細は第5.3.1.7項MultiLanguage参照。TDドキュメント内のすべてのMultiLanguageオブジェクトには、同じ言語メンバーセットが含まれる**べきである。**

異なるレベルでtitlesおよびdescriptions使用するTDスニペットは以下のとおりである。

例7

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1"、"title": "MyThing",
    「タイトル」:{
        "en":"MyThing",
        "de": "MeinDing",
        "ja" : "私のモノ",
        "zh-Hans" : "我的东西",
        "zh-Hant" : "我的東西"
    },
    "descriptions": {
        「en」:「人間が読める情報」、"de": "Menschenlesbare Informationen."、"ja" : "人間が読むことができる情報",
        "zh-Hans" : "人们可阅读的信息",
        "zh-Hant" : "人們可閱讀的資訊"
    },
    ...
    "properties": {"on": {
            「タイトル」:{
                「en」:「On/Off」、
                「de」:「An/Aus」、
                "ja": "オンオフ",
                "zh-Hans": "开关",
                "zh-Hant": "開關"},
            "type": "boolean",
            "forms": [...]
        },
        "status": {
            「タイトル」:{
                「en」:「Status」、
                「de」:「Zustand」、
                "ja": "状態",
                "zh-Hans": "状态",
                "zh-Hant": "狀態"},
            "type": "object",
            ...
            "forms": [...]
        }
    },
    ...
}
```

TDインスタンスは、titleおよびdescriptionの使用をtitlesおよびdescriptionsと組み合わせることもできる。titleとtitles、あるいは、descriptionとdescriptionsが同じJSONオブジェクト内に存在する場合、titleおよびdescriptionの値はデフォルトテキストとして見ることができる。titleとtitles、あるいは、descriptionとdescriptionsがTDドキュメントに存在する場合、各titleおよびdescriptionメンバーは、それぞれ対応するtitleおよびdescriptionメンバーを持っているべきである。デフォルトテキストの言語は、デフォルト言語で示され、これは、通常、TDンスタンスの作成者によって設定される。

例8


```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"@language" : "de"}
    ],
    "title": "MyThing"、 "titles": {
        "en":"MyThing",
        "de": "MeinDing",
        "ja" : "私のモノ",
        "zh-Hans" : "我的东西",
        "zh-Hant" : "我的東西"
    },
    "description": "Menschenlesbare Informationen."、 "descriptions": {
        「en」:「人間が読める情報」、 "de": "Menschenlesbare Informationen."、 "ja" : "人間が読むことができる情報",
        "zh-Hans" : "人们可阅读的信息",
        "zh-Hant" : "人們可閱讀的資訊"
    },
    ...
    "properties": {"on": {
            「タイトル」:「An/Aus」、 「タイトル」:{
                「en」:「On/Off」、
                「de」:「An/Aus」、
                "ja": "オンオフ",
                "zh-Hans": "开关",
                "zh-Hant": "開關"},
            "type": "boolean",
            "forms": [...]
        },
        "status": {
            「タイトル」:「Zustand」、 「タイトル」:{
                「en」:「Status」、
                「de」:「Zustand」、
                "ja": "状態",
                "zh-Hans": "状态",
                "zh-Hant": "狀態"},
            "type": "object",
            ...
            "forms": [...]
        }
    },
    ...
```

}

もう一のデフォルト言語の設定用法は、HTTPのAccept–Languageヘッダーフィールドなどの言語ネゴシエーションメカニズムを介することである。デフォルト言語がネゴシエートされていた場合、ネゴシエーションの結果と返されたコンテンツのデフォルト言語を示すために、@languageメンバーが存在しな**ければならない。**デフォルト言語のネゴシエーションが成功した場合、TDドキュメントは、titlesおよびdescriptionsメンバー内のMultiLanguageオブジェクトよりも優先して、適切で整合するtitleおよびdescriptionメンバー値を持っている**べきである。**しかしながら、モノは、そのような動的に生成されたTDをサポートしない、また、(例えば、リソース制約のために)言語ネゴシエーションをサポートしないと選択して**もよいこと**に留意されたい。

3. version

名前がVersionInfoのシグニチャに含まれるボキャブラリ用語である場合、VersionInfoのインスタンスの名前-値ペアは、すべて、名前としてボキャブラリ用語を持つJSONメンバーとしてシリアライズされ**なければならない。**

バージョン情報オブジェクトのTDスニペットは以下のとおりである。

例9

```
{
    ...

    ...
}
```

Versionメンバーは、TDコンテキスト拡張子に基づく追加のアプリケーションおよび/またはデバイス固有のバージョン情報のコンテナである。詳細については、第7.1項意味論的注釈を参照。

4. securityDefinitionsとsecurity

Thingインスタンスでは、securityDefinitionsに割り当てられる値は、SecuritySchemeのインスタンスのマップである。SecuritySchemeインスタンスのマップのすべての名前-値ペアは、マップをシリアライズした結果のJSONオブジェクトのメンバーとしてシリアライズされ**なければならない。**ペアの名前はJSONストリングとして、SecuritySchemeのインスタンスであるペアの値はJSONオブジェクトとしてシリアライズされ**なければならない。**

SecuritySchemeの中の一つのサブクラスのインスタンスの名前-値ペアすべては、その名前がそのサブクラスのシグニチャまたはSecuritySchemeのシグニチャに含まれるボキャブラリ用語である場合、SecuritySchemeサブクラスのインスタンスを名前としてボキャブラリ用語でシリアライズした結果であるJSONオブジェクトのメンバーとしてシリアライズされ**なければならない。**

以下のTD スニペットは、基本的なユーザ名/パスワード認証をヘッダーで指定する単純なセキュリティ構成である。inに与えられる値は、実際には、デフォルト値(header)であり、省略することもできる。名前付きセキュリティ構成は、securityDefinitions マップで指定されなければならない。その定義は、securityメンバーにそのJSON 名を組み込むことによってアクティブ化されなければならない。そのJSON 名は1 つの定義だけがアクティブ化された時にストリング型となることができる。

例10

```
...
"securityDefinitions": {"basic_sc": {
        「スキーム」:「基本」、
        "in": "header"
    }
},

...
```

これは、より複雑な例であり、モノのベアラトークン認証と組み合わされたプロキシのダイジェスト認証を示すTDスニペットである。digestスキームでは、inのデフォルト値(すなわち、header)は省略されるが、依然として適用される。ユーザ名/パスワードおよびトークンなどの対応するプライベートセキュリティ構成は、正常に対話するためにコンシューマ内で構成されなければならないことに留意されたい。複数のセキュリティ定義をアクティブ化すると、securityメンバーは配列となる。

例11

```
...
"securityDefinitions": {"proxy_sc": {
        「スキーム」:「ダイジェスト」、
        "proxy": "https://portal.example.com/"
    },
    "bearer_sc": {"in":"header",
        「スキーム」:「ベアラ」、
        "format": "jwt",
        alg:"ES256"、
        "authorization": "https://servient.example.com:8443/"
    }
},

...
```

TD でのセキュリティ構成は必須である。セキュリティ定義は少なくとも1つ、モノレベルで(すなわち、TDルートオブジェクトで)security配列を通してアクティブ化されな**ければならない**。この構成は、モノと対話するために必要なデフォルトのセキュリティメカニズムと見なすことができる。また、セキュリティ定義は、モノレベルでアクティブ化されたすべての定義を無効にする(すなわち、完全に置き換える)フォームオブジェクトにsecurityメンバーを含めることによって、フォームレベルでのアクティブ化も**できる**。

nosecセキュリティスキームは、セキュリティが必要とされない場合に提供される。モノの最小セキュリティ構成は、以下の例に示すように、モノレベルでのnosecセキュリティスキームのアクティブ化である。

例12

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1"、"id": "urn:dev:ops:32473-Thing-1234",
    "title": "MyThing",
    「description」:「人間が読める情報」、「対応」:「https://servient.example.com/contact」、
    "securityDefinitions": {"nosec_sc": {"scheme": "nosec"}}、"security": "nosec_sc",
    "properties": {...},
    "actions": {...},
    「events」: {...}、
    「リンク」: [...]
}
```

より複雑な例として、我々が、すべての対話アフォーダンスが、認証が必要とされないものを除いて、基本認証を要求するモノを持っていると仮定してみよう。statusプロパティおよびtoggleアクションについては、basic認証が要求され、モノレベルで定義される。しかしながら、overheatingイベントに関しては、認証は必要なく、従って、セキュリティ構成は、フォームレベルでオーバーライドされる。

例13

```
{
    ...
    "securityDefinitions": {"basic_sc": {"scheme": "basic"},
        "nosec_sc": {"scheme": "nosec"}
    },
    "security": ["basic_sc"],
    ...
    "properties": {"status": {
            ...
            "forms": [{
                "href":"https://mylamp.example.com/status"
            }]
        }
    },
    "actions": {
                                        "toggle": {
            ...
            "forms": [{
                "href":"https://mylamp.example.com/toggle"
            }]
        }
    },
    "events": {
        「過熱」:{
            ...
            "forms": [{
                「href」:「https://mylamp.example.com/oh」、"security": ["nosec_sc"]
            }]
        }
    }
}
```

セキュリティ構成は、同じ対話アフォーダンス内の異なるフォームに対して指定することもできる。これは、例えば、HTTP及びCoAP [RFC7252]のような異なるセキュリティメカニズムをサポートする複数のプロトコルをサポートするデバイスに対して要求される。また、代替の認証メカニズムが許可される場合にも有用である。ここで、プロパティアフォーダンスをアクティブ化する3 つの方法を示すTD スニペットを紹介する。基本認証を使用したHTTPS、ダイジェスト認証を使用したHTTPS、ベアラトークン認証を使用したＨＴＴＰである。言い換えれば、複数のフォーム内で異なるセキュリティ構成を使用すれば、「OR」方式でセキュリティメカニズムを組み合わせられる。対照的に、複数のセキュリティ構成を同じsecurityメンバーに構成すると、それらを「AND」方式で組み合わせるということになるが、その場合、それらはすべて、対話アフォーダンスのアクティブ化を可能にするために満足される必要があるためである。モノレベルで (デフォルト)構成を1つアクティブ化することは、依然として、必須であることに留意されたい。

例14

```
{
    ...
    "securityDefinitions": {
        "basic_sc": {"scheme": "basic"},
        "digest_sc": {"scheme": "digest"、 "qop": "auth"、 "in": "header"}、 "psk_sc": {"scheme": "psk"}
    },
    "security": ["basic_sc"],
    ...
    "properties": {"status": {
            ...
            "forms": [{
                "href":"https://mylamp.example.com/status"
            }, {
                「href」:「https://mylamp.example.com/status」、 "security": ["digest_sc"]
            }, {
                "href": "coaps://mylamp.example.com:5684/status","security": ["psk_sc"]
            }]
        }
    },
    ...
}
```

もう一つより複雑な例として、OAuth2はスコープを利用する。トークン内に現れる可能性があり、そのリソース(またはW3C WoTの場合は対話アフォーダンス)へのアクセスを可能にするために、リソース内の対応する識別子と一致しなければならない識別子である。例えば、以下の例では、statusプロパティは、スコープlimitedを含むベアラトークンを使用するコンシューマが読み取ることができるが、configureアクションは、specialスコープを含むトークンを用いることによって呼び出すことができるのみである。スコープは、ロールと同一ではないが、しばしばロールに関連付けられ、例えば、おそらく、管理ロール内のスコープのみが、「特別な」対話を実行することを許可される。トークンは複数のスコープを持つことができる。本例では、管理者には、おそらく、limitedでsepcialなスコープ両方を持つトークンが発行され、一方、通常のユーザにはlimitedスコープを持つトークンのみが発行される。

例15

```
{
    ...
    "securityDefinitions": {"oauth2_sc": {
            "scheme": "oauth2",
            ...
            "flow": "implicit",
            "authorization": "https://example.com/authorization","scopes": ["limited"、"special"]
        }
    },
    "security": ["oauth2_sc"],
    ...
    "properties": {"status": {
            ...
            "forms": [{
                「href」:「https://scopes.example.com/status」、"scopes": ["limited"]
            }]
        }
    },
    "action": {
        "configure": {
            ...
            "forms": [{
                「href」:「https://scopes.example.com/configure」、"scopes": ["special"]
            }]
        }
    },
    ...
}
```

5. properties

Thingインスタンス内のpropertiesに割り当てられる値は、PropertyAffordanceのインスタンスのマップである。PropertyAffordanceインスタンスのマップの名前-値ペアは、すべて、マップをシリアライズした結果のJSONオブジェクトのメンバーとしてシリアライズされなければならない。ペアの名前はJSONストリングとして、PropertyAffordanceインスタンスであるペアの値はJSONオブジェクトとしてシリアライズされなければならない。

PropertyAffordanceインスタンスの名前-値ペアは、すべて、その名前がPropertyAffordance、InteractionAffordance、あるいは、DataSchemaのシグニチャ(の1つ)に含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたPropertyAffordanceインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされなければならない。DataSchemaインスタンスのシリアライズの詳細については、第6.3.10項参照データスキーマ参照。

PropertyAffordanceインスタンス内のformsに割り当てられた値は、第6.3.9項formsで定義されているように1つ以上のJSONオブジェクトシリアライズを含むJSON配列としてシリアライズされなければならない。

2つのプロパティアフォーダンスのスニペットを以下に示す。

例16: プロパティのシリアライズ例

```
...
"properties": {"on": {
        "type": "boolean",
        "forms": [...]
    },
    "status": {
        "type": "object"、 "properties": {

            "明るさ": {"type": "number"、 "minimum":0.0,
            「最大」:100.0
            },
            "rgb": {"type": "array"、 "items" : {
                "type" : "number"、 "minimum": 0,
                「最大」:255
            },
            "minItems": 3、
            "maxItems": 3
        }
    },
                                                                                    }
...
"required": ["brightness"、 "rgb"],
"forms": [...]
```

6. actions

Thingインスタンスでは、actionsに割り当てられる値は、ActionAffordanceのインスタンスのマップである。ActionAffordanceインスタンスのマップの名前-値ペアは、すべて、マップをシリアライズした結果のJSONオブジェクトのメンバーとしてシリアライズされ**なければならない。**ペアの名前はJSONストリングとして、ActionAffordance インスタンスであるペアの値はJSONオブジェクトとしてシリアライズされ**なければならない。**

ActionAffordance インスタンスの名前-値ペアはすべて、その名前がActionAffordance、あるいは、InteractionAffordanceのシグニチャ(の1つ)に含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたActionAffordanceインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされな**ければならない。**

ActionAffordanceインスタンスでoutputとinputに割り当てられる値は、JSONオブジェクトとしてシリアライズされ**なければならない。**これらは、クラスdataschemaに依存し、そのシリアライズは、第6.3.10項データスキーマで定義される。

ActionAffordanceのインスタンスでformsに割り当てられる値は、第6.3.9項formsで定義されているように1つ以上のJSONオブジェクトシリアライズ含むJSON配列としてシリアライズされな**ければならない。**

アクションアフォーダンスのTDスニペットを以下に示す。

例17: アクションのシリアライズ例

```
...
"actions": {
    "fade" : {
        「タイトル」:「フェードイン/フェードアウト」、
        "description": "スムーズなフェードインとフェードアウトアニメーション"、"input": {
            "type": "object"、"properties": {
                "from": {
                    "type": "integer"、"minimum": 0,
                    「最大」:100
                },
                "to": {
                    "type": "integer"、"minimum": 0,
                    「最大」:100
                },
                "duration": {"type": "number"}
            },
            "required": ["to","duration"],
        },
                                                                                    }
    },
    ...
"output": {"type": "string"},
"forms": [...]
```

7. events

Thingインスタンスでは、eventsに割り当てられる値は、EventAffordanceのインスタンスのマップである。EventAffordanceインスタンスのマップの名前-値ペアは、すべて、マップをシリアライズした結果のJSONオブジェクトのメンバーとしてシリアライズされなければならない。ペアの名前はJSONストリングとして、EventAffordanceインスタンスであるペアの値はJSONオブジェクトとしてシリアライズされ**なければならない。**

EventAffordanceインスタンスの名前-値ペアはすべて、その名前がEventAffordance、あるいは、InteractionAffordanceのシグニチャ(の1つ)に含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたEventAffordanceインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされな**ければならない。**

EventAffordanceインスタンスでsubscription、data、および、cancellationに割り当てられる値は、JSONオブジェクトとしてシリアライズされ**なければならない。**これらはクラスDataSchemaに依存し、そのシリアライズは第6.3.10項データスキーマで定義される。

EventAffordanceのインスタンスでformsに割り当てられる値は、第6.3.9項formsで定義されているように1つ以上のJSONオブジェクトシリアライズを含むJSON配列としてシリアライズされ**なければならない。**

イベントオブジェクトのTDスニペットを以下に示す。

例18: イベントのシリアライズ例

```
...
"events": {
    "overheated": {"data" : {
                                        "type": "string"
        },
        "forms": [...]
    }
},
...
```

イベントアフォーダンスは、既存の(例えばWebSub[websub])または顧客向けイベントメカニズム(例えばWebhoks)を採用するために、柔軟に定義されている。このため、所望のメカニズムに従って、subscriptionおよびcancellationを定義することができる。詳細は[WoT-BindingTemplates]参照。例A.3 Webhook イベント例 は、Webhook を説明するためにイベントがどのようにsubscriptionおよびcancellationを使用できるかを例示している。

8. links

linkインスタンスの名前-値ペアはすべて、その名前がlinkのシグニチャに含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたlinkインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされな**ければならない。**

links配列内のリンクオブジェクトのTDスニペットを以下に示す。

例19: リンクのシリアライズ例

```
...
"links": [{
    "rel": "controlledBy",
     「href」:「https://servient.example.com/things/lampController」、 "type": "application/td+json"
}]
...
```

9. forms

formインスタンスの名前-値ペアはすべて、その名前がFormのシグニチャに含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたFormインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされなければならない

必要に応じて、フォームオブジェクトは、プレフィックスで識別されるプロトコル固有のボキャブラリ用語で補足され**てもよい。**第8.3項プロト

コルバインディングも参照。

forms配列内のフォームオブジェクトのTDスニペットを以下に示す。

例20: Formのシリアライズ例

```
...
"forms": [{
    "op": "writeproperty",
    「href」:「http://mytemp.example.com:5683/temp」、"contentType": "application/json"、
    "htv:methodName": "POST"
}]
...
```

hrefには、http://192.168.1.25/left?p=2 & d=1のpやdなどのダイナミック変数を含むURIを入れることもできる。その際、URIは、[RFC6570]http://192.168.1.25/left{?p,d}で定義されているようにテンプレートとして定義することができる。

そのような場合、URIテンプレート変数は、JSON名として関連付けられた(一意の)変数名を持つJSONオブジェクトベースのuriVariablesメンバーに集められな**ければならない。**

Formインスタンス中のuriVariablesに割り当てられるマップ中の各値のシリアライズは、クラスDataSchemaに依存しな**ければならない。**そのシリアライズは、第6.3.10項データスキーマで定義される。

URI テンプレートとuriVariables を使用したTD スニペットを以下に示す。

例21

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"eg": "http://www.example.org/iot"}
    ],
    ...
    "actions": {
        "LeftDown": {
            ...
            "uriVariables": {
                "p" : {"type": "integer"、"minimum": 0、"maximum": 16、"@type": "eg:SomeK
                "d" : {"type": "integer"、"minimum": 0、"maximum": 1、"@type": "eg:Direct
```

},

...

}

},

...

},

"forms": [{

"href" :"http://192.168.1.25/left{?p,d}","htv:methodName": "GET"

}]

contentTypeメンバーは、「；文字」で区切られた属性-値ペアとしてメディアタイプパラメータを含むメディアタイプ[RFC2046]を割り当てるために使用される。例:

例22

```
...
    「contentType」:「text/plain; charset=utf-8」
...
```

いくつかの使用事例では、対話アフォーダンスのフォームメタデータは、要求を記述するだけでなく、期待される応答のためのメタデータも提供する。例えば、アクションtakePhotoは、要求ペイロードのJSON(すなわち、”contentType”:”application/json”)を使用してカメラのパラメータ設定(アパーチャ優先順位、タイマなど)を送るためのinputスキーマを定義する。このアクションの出力は撮影された写真であり、これは、例えば、JPEGフォーマットで可能となる。そのような場合、responseメンバーは、応答ペイロードの表現フォーマット(例えば、”contentType”:”image/jpeg”)を示すために使用される。ここでは、コンテンツタイプが表現フォーマットを完全に指定するのでoutputスキーマは必要とされない。

Formインスタンスでresponseに割り当てられる値は、それが存在する場合、JSONオブジェクトでなければならない。応答オブジェクトは、それが存在する場合、ExpectedResponseのクラス定義の中で定義されているcontentTypeメンバーを含んでいなければならない。

上記のアクションtakePhotoに基づいて、responseメンバーを持つformスニペットを以下に示す。

例23

```
{
    ...
    "actions": {
        "takePhoto": {
            ...
            "forms": [{
                "op": "invokeaction",
                 「href」:「http://camera.example.com/api/snapshot」、"contentType": "application/json",
                "response": {
                    "contentType": "image/jpeg"
                }
            }]
        }
    },
    ...
}
```

formsがトップレベルに存在する場合、それは、モノが提供するメタ対話を記述するために使用することができる。例えば、操作タイプ「readallproperties」および「writeallproperties」は、コンシューマがすべてのプロパティを一度に読み取りと書き込みをすることができるモノとのメタ対話のためのものである。以下の事例では、formsメンバーがTDルートオブジェクトに含まれ、コンシューマが一つのプロトコルトランザクションでモノのすべてのプロパティ(すなわち、on, brightness, timer)を読み取るか、または、書き込むために、送信対象https://mylamp.example.com/allpropertiesを使用することができる。

例24

```
{
    ...
    "properties": {"on": {
            "type": "boolean",
            "forms": [...]
        },
        「明るさ」:{「タイプ」:「番号」、
            "forms": [...]
        },
        "timer": {
            "type": "integer",
            "forms": [...]
        }
    },
    ...
    "forms": [{
        "op": "readallproperties",
        「href」:「https://mylamp.example.com/allproperties」、"contentType": "application/json"、
        "htv:methodName": "GET"
    }, {
        "op": "writeallproperties",
        「href」:「https://mylamp.example.com/allproperties」、"contentType": "application/json"、
        "htv:methodName": "PUT"
    }]
}
```

オペレーションタイプwriteallpropertiesの場合、コンシューマは書き込み可能なプロパティ**すべて**と（新しく）割り当てられた値（例：ペイロード内）を提供することが求められる。さもなければ、モノは不整合を避けるためにこの呼び出しを拒否してよい。

10. データスキーマ

DataSchemaクラスで定義されたWoTTDのデータスキーマは、JSON スキーマ用語のサブセット[JSON-SCHEMA] のに基づいている。したがって、モノとやり取りされるデータを検証するためにTDデータスキーマのシリアライズをJSONスキーマバリデーターのインプリメンテーションに直接与えることができる。

データスキーマのシリアライズは、PropertyAffordanceインスタンス、ActionAffordanceインスタンスでinputとoutputに

割り当てられる値、EventAffordanceインスタンスでsubscription, data, cancellationに割り当てられる値、および、(フォームオブジェクトがURIテンプレートを使用する場合)InteractionAffordanceのサブクラスのインスタンスでuriVariablesに割り当てられる値に適用される。

DataSchemaの一つのサブクラスインスタンスの名前-値ペアはすべて、その名前がそのサブクラスのシグニチャ、あるいは、DataSchemaのシグニチャに含まれるボキャブラリ用語である場合、名前としてボキャブラリ用語のついたDataSchemaのサブクラスインスタンスをシリアライズした結果得られるJSONオブジェクトのメンバーとしてシリアライズされ**なければならない。**

ObjectSchemaインスタンス内のpropertiesに割り当てられる値は、JSONオブジェクトとしてシリアライズされなけ**ればならない。**

DataSchemaインスタンスのenum、required、および、oneOfに割り当てられる値は、JSON配列としてシリアライズされな**ければならない。**

ArraySchemaのインスタンス内のitemsに割り当てられる値は、JSONオブジェクトまたはJSONオブジェクトを含むJSON配列としてシリアライズされなければならない。

TDスニペットデータスキーマメンバーを以下に示す。周囲のオブジェクトは、データスキーマオブジェクト(例えば、input、 output用)又は付加的なメンバーを含むプロパティオブジェクトであってもよいことに留意されたい。

例25: DataSchema のシリアライズ例

```
...
"type": "object"、"properties": {
    "status": {
        「タイトル」:「ステータス」、
        "type": "string",
        "enum": ["On"、"Off"、"Error"]
    },
    「明るさ」: {

        "title": "Brightness value"、"type": "number"、"minimum": 0.0,
        「最大」:100.0
    },
    "rgb": {
        "title": "RGB color value"、"type": "array",
        「アイテム」:{
            "type" : "number"、"minimum": 0,
            「最大」:255
        },
        "minItems": 3、
        "maxItems": 3
    }
},
...
```

readOnlyおよびwriteOnlyという用語は、読み取り対話(すなわち、プロパティを読み取る時)においてどのデータ項目をやり取りするか、および書き込み対話(すなわち、プロパティを書き込む時)においてどのデータ項目をやりとりするかを知らせるために使用することができる。これは、従来型でないモノのプロパティが、読み取りおよび書き込みのために異なるデータを示すときの回避策として使用することができ、これは、TDを用いて既存のデバイスまたはサービスを増補するときに発生する可能性がある。

readOnly およびwriteOnly を使用したTD スニペットを以下に示す。

例26

```
...
"properties": {"status": {
        "description": "Read or write On/Off status."、 "type": "object",
        "properties": {"latestStatus": {
                "type": "string"、 "enum": ["On"、 "Off"]、 "readOnly": true
            },
            "newStatusValue": {"type": "string"、 "enum": ["On"、 "Off"]、 "writeOnly": true
            }
        },
        形式: [...]
    }
}
...
```

statusプロパティが読み込まれると、ペイロード内のlatestStatusメンバーを使用してステータスデータが返される。statusプロパティを更新するには、ペイロード内のnewStatusValueメンバーを介して新しい値が提供されなければならない。

追加機能として、TDインスタンスでは、データスキーマ内のunitメンバーが使用できる。これによって、測定単位をデータアイテムに関連付けることができる。そのストリング値は、自由に選択することができる。しかしながら、周知のボキャブラリで定義されている単位を選択することが推奨される。例については、第７項TDコンテキスト拡張子参照。

6.4 Identification

TDのJSONベースのシリアライズは、メディアタイプapplication/td+json、または、CoAPコンテンツフォーマットID T.B.D.によって識別される(第10項IANA考慮事項を参照)。

注: CoAPコンテンツフォーマット

CoAPベースのWoTインプリメンテーションでは、正式コンテンツフォーマットIDが割り当てられるまで、試験的なコンテンツフォーマット65100を使用することができる。

7. TDコンテキスト拡張子

本項は標準ではない。

第５項TD情報モデルの標準ボキャブラリ定義に加えて、WoTTDは、追加の名前空間からコンテキスト知識を追加する機能を提供している。本メカニズムは、TDインスタンスを追加の(例えば、ドメイン固有の)意味論で強化するために使用することができる。また、将来、追加のプロトコルバインディングや新しいセキュリティスキームをインポートするために使用することもできる。

そのようなTDコンテキスト拡張子について、TDは、JSON–LD [json–ld11]で周知の@contextメカニズムを使用する。TDコンテキスト拡張子を使用する場合、クラスThing の@context値は、JSON-LD コンテキストファイルを識別するanyURI 型の追加要素を持つ配列、あるいは、第5.3.1.1項Thingで定義されている通り、名前空間IRI を含むマップである。.

第6.1項 JSON型へのマッピングにある複合型のシリアライズ規則は、拡張@context名前-値ペアのシリアライズを定義している。TDコンテキスト拡張子を持つスニペットを以下に示す。

例27

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {
            「iot」:「http://example.org/iot」、
            "cov": "http://www.example.org/coap-binding#"
        },
        "https://schema.org/"
    ],
    ...
}
```

1. 意味論的注釈

TDコンテキスト拡張子は、TDインスタンスへのボキャブラリ用語追加を可能にする。含まれる名前空間が、RDFスキーマまたはOWLによって提供される定義などのクラス定義に基づく場合、それらを使用して、インスタンスをそのような外部クラス定義に関連付けることによって、TDのクラスインスタンスに意味論的に注釈を付けることができる。これは、@type名前-値ペアにクラス名を割り当てるか、または、複数の関連付け/注釈の配列値にクラス名を入れて行う。第6.1項JSON型へのマッピング内のシリアライズ規則に従って、@type は、JSON ストリングまたはJSON 配列としてシリアライズされる。@type は、ノード型を設定するために使用されるJSON-LD キーワード[json-ld11] である。

TDコンテキスト拡張子は、TDの任意のクラスインスタンス内に追加の名前–値ペアおよび明確に定義された値を含めることも可能にする。これらのペアおよび値は、含まれるボキャブラリ用語で定義され、それぞれ、対応するJSONオブジェクト内の追加メンバーまたは既存メンバーの値としてシリアライズされる。例として、モノの追加バージョンメタデータ、または、データアイテムの測定単位がある。

一例として、以下に示すTDスニペットは、モノのハードウェアおよびファームウェアのバージョン番号を追加することによってバージョン情報コンテナを拡張し、モノや例２とOM（測定単位オントロジー[RIJGERSBERG]）でも使用されているデータスキーマユニット： SAREF用に外部のボキャブラリの値を使用する。これらのボキャブラリは、例として使用されており、家庭自動化領域では特にその他のボキャブラリが存在するかもしれない。

例28

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {
            「v」:「http://www.example.org/versioningTerms#」、"saref": "https://w3id.org/saref#",
            「om」:「http://www.wurvoc.org/vocabularies/om-1.8/」
        }
    ],
    "@type": "Thing"、"version": {
        「インスタンス」:「1.2.1」、
        「v:firmware」:「0.9.1」、
        "v:hardware": "1.0"
    },
    ...
    "properties": {"temperature": {

            "@type": "saref:Temperature"、"description": "Temperature value of the weather station"、"type": "number"、"minimum": -32.5,
            「最大」:55.2、
            "unit": "om:degree_Celsius"、"forms": [...]
        },
        ...
    },
    ...
}
```

多くの場合、TDコンテキスト拡張子は、対話中に（レスポンスのペイロード内で）データ交換により表示される物理世界のオブジェクトの状態情報の意味論的処理を可能にするために使用されるかもしれない。たとえば、RDF内のこの状態情報の意味論的説明は、TDドキュメントに埋め込むことができ、データスキーマの項目は、物理世界のオブジェクトのRFDモデル状態の特定部分の参照を促す注釈を個々につけることができる。

下記のTDスニペットは、ランプの上他ウィを説明するためにSAREFを使用している。SSN（Semantic Sensor Network Ontology意味論的センサーネットワークオントロジー）[Vocab-SSN]から取り入れた外部ボキャブラリ用語ssn:forPropertyは、statusプロパティのデータスキーマを物理世界オブジェクトの実際のオン／オフ状態にリンクするために使用されている。

例29


```
{
"@context": [
"https://www.w3.org/2019/wot/td/v1",
{
"saref": "https://w3id.org/saref#",
"ssn": "http://www.w3.org/ns/ssn/"
}
],
"id": "urn:dev:ops:32473-WoTLamp-1234",
"@type": "saref:LightSwitch",
"saref:hasState": {
"@id": "urn:dev:ops:32473-WoTLamp-1234/state",
"@type": "saref:OnOffState"
},
...
"properties": {
"status": {
"ssn:forProperty": "urn:dev:ops:32473-WoTLamp-1234/state",
"type": "string",
"forms": [{"href": "https://mylamp.example.com/status"}]
},
"fullStatus": {
"ssn:forProperty": "urn:dev:ops:32473-WoTLamp-1234/state",
"type": "object",
"properties": {
"statusString": { "type": "string" },
"statusCode": { "type": "number" },
"statusDescription": { "type": "string" }
},
"forms": [{"href": "https://mylamp.example.com/status?full=true"}]
},
...
},
...
  }
```


例２では、モノの状態は、それ自体のstatusアフォーダンスによって与えられ、起こりうる状態変化は、toggleアフォーダンスで与えられる。つまり、物理世界オブジェクトの状態は、直接、モノの対話アフォーダンスを提供するということである。この設計はシンプルなケースにおいては十分である。より複雑なケースでは、しかし、複数のアフォーダンスが同じ物理的状態に使用できることもある。上記の例では、fullStatusプロパティが、ランプの状態をより多くの言葉で表現する他の方法を提供している。

2. プロトコルバインディングの追加

本項は標準ではない。

TDのTDコンテキスト拡張子を使って通信メタデータを補足することができ、または、Formインスタンスを表すJSONオブジェクトにシリアライズされた追加のボキャブラリ用語で新しいプロトコルバインディングを追加することができる。(第8.3項プロトコルバインディングも参照)。

以下のTDの例では、仮想CoAPプロトコルバインディングを使用する。というのは、このようなプロトコルバインディングが本仕様書執筆時点では存在しないためである。このTDコンテキスト拡張子は、名前空間例http://www.example.org/coap-binding#を介してアクセス可能なRDF1.0[HTTP-in-RDF10]内のHTTPボキャブラリに類似したCoAP RDFボキャブラリがあると仮定している。補足されたcov:methodNameメンバーは、どのCoAPメソッドが適用されなければならないかをコンシューマに指示する(例えば、CoAPメソッドコード0.01の場合はGET、CoAPメソッドコード0.02の場合はPOST、またはCoAPメソッドコード0.07の場合はiPATCH)。

例30: TDコンテキスト拡張によるフォームの特殊化

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        {"cov": "http://www.example.org/coap-binding#"}
    ],
    ...
    "properties": {"brightness": {

            "description": "The current brightness setting"、 "type": "integer"、 "minimum": -64,
            「最大」:64、
            "forms": [{
                "op": "readproperty",
                "href": "coap://example.org:61616/api/brightness","cov:methodName": "GET"
            }, {
                "op": "writeproperty",
                "href": "coap://example.org:61616/api/brightness","cov:methodName": "POST"
            }]
        },
        ...
    },
    ...
}
```

3. セキュリティスキームの追加

第5.3.3項セキュリティボキャブラリ定義に含まれていない新しいセキュリティスキームは、TDコンテキスト拡張子メカニズムを使用してインポートすることができる。本例では、http://www.example.org/ace-security#で名前空間が本例のために定義する[ACE]に基づく仮想ACE セキュリティ方式を使用している。このような追加のセキュリティスキームは、クラスセキュリティスキームのサブクラスでなければならないことに留意されたい。

例31

```
{
    @context: [
        "https://www.w3.org/2019/wot/td/v1",
        {
                                                                                                                    }
    ],
    ...
「cov」:「http://www.example.org/coap-binding#」、"ace": "http://www.example.org/ace-security#"
    "securityDefinitions": {"ace_sc": {
            "scheme": "ace:ACESecurityScheme",
            ...
            "ace:as": "coaps://as.example.com/token","ace:audience":
            "coaps://rs.example.com","ace:scopes": ["limited"、"special"]、"ace:cnonce": true
        }
    },
    "security": ["ace_sc"]、"properties": {
                                        "status": {
            ...
            "forms": [{
                "op": "readproperty",
                "href": "coaps://rs.example.com/status","contentType": "application/cbor"、"cov:methodName":
                "GET",
                "ace:scopes": ["limited"]
            }]
        }
    },
    "action": {
        "configure": {
            ...
            "forms": [{
                "op": "invokeaction",
                "href": "coaps://rs.example.com/configure","contentType": "application/cbor"、
                "cov:methodName": "POST",
                "ace:scopes": ["special"]
            }]
        }
    },
    ...
}
```

第5.3.3項セキュリティボキャブラリ定義の中で定義されているセキュリティスキーム全てが既にTDコンテキストの一部であり、TDコンテキスト拡張子を使って含める必要はないということに留意された。

8. ビヘイビアのアサーション

以下のアサーションは、TDの表現または情報モデルとは対照的に、WoTシステムのコンポーネントのビヘイビアに関するものである。しかしながら、TDは記述的であり、特に、前から存在するネットワークインターフェースを記述するために使用されることもあるということに留意されたい。この場合、そのようなすでに存在するインターフェースのビヘイビアを制約するアサーションを行うことはできない。代わりに、アサーションは、そのようなインターフェースを正確に表すために、TDに対する制約となると解釈されなければならない。

1. セキュリティ構成

安全な相互運用を可能にするために、セキュリティ構成はモノの要件を正確に反映しなければならない。

- モノが対話のために特定のアクセスメカニズムを要求する場合、そのメカニズムはTDのセキュリティ構成の中で指定さ**れなければならない。**
- モノが対話のために特定のアクセスメカニズムを要求しない場合、そのメカニズムはTDのセキュリティ構成の中で指定さ**れてはならない。**

2. データスキーマ

TDで提供されるデータスキーマは、TDで指定された対話の中で記述されたモノが返し、また、受け入れるデータペイロードを正確に表すべきである。一般に、コンシューマは、WoTTDに与えられていないものを生成せず、厳密にデータスキーマに従うべきである。が、WoTTDに明示的に与えられていないモノからの追加データを受け入れるべきである。一般的に、モノは、WoTTDによって記述されるが、コンシューマは、モノと対話するときにWoTTDに従わざるを得ない。

- WoTTDに記述されている別のターゲットのモノと対話するときに、コンシューマとして動作するモノは、その対話で与えられたデータスキーマに従って編成されたデータを生成しな**ければならない。**
- WoTTDは、各対話によって返され、また、受け入れられたデータを正確に記述しな**ければならない。**
- モノは、そのWoTTDで与えられたデータスキーマにそのようなデータが記述されていない場合でも、対話から追加データを返して**もよい。**これは、返されるデータ内に追加のプロパティあるいはアイテムがある可能性がある場合に、ObjectSchemaとArraySchema（itemsがDataSchemaの列にある場合）に適用される。これは、「JASON-SCHEMA」内に定義されるとおりに、”additioalProperties”: true あるいは “additionalItems”: trueがであるかのように挙動する。
- 別のモノと対話するときにコンシューマとして動作するモノは、ターゲットのモノのTDで与えられたデータスキーマに記述されていない追加データを確実に受け取らな**ければならない。**これは、返されるデータ内に追加のプロパティあるいはアイテムがある可能性がある場合に、ObjectSchemaとArraySchema（itemsがDataSchemaの列にある場合）に適用される。これは、「JASON-SCHEMA」内に定義されるとおりに、”additioalProperties”: true あるいは “additionalItems”: trueがであるかのように挙動する。
- 別のモノと対話するときにコンシューマとして動作するモノは、そのモノのTDで与えられたデータスキーマに記述されていないデータを生成し**てはならない。**
- 別のモノと対話するときにコンシューマとして動作するモノは、URIテンプレート、ベースURI、および、ターゲットのモノのTDで与えられるhrefパラメータに従ってURIを生成し**なければならない。**
- WoTTD内のURIテンプレート、ベースURI、およびhrefメンバーは、モノのWoTインターフェースを正確に記述し**なければならない。**

3. プロトコルバインディング

プロトコルバインディングは、対話アフォーダンスから、HTTP [RFC7231]、CoAP [RFC7252]、MQTT [MQTT]などの特定のプロトコルの具体的メッセージへのマッピングである。対話アフォーダンスのプロトコルバインディングは、第6.3.9項 formsで定義されているような形式でシリアライズされる。

WoTTD内のすべてのフォームには、hrefメンバーが与える送信ターゲットが入っていなければならない。この送信ターゲットのURIスキームは、モノがどのプロトコルバインディングを実装しているか[WoTArchitecture]を示す。例えば、ターゲットがhttpまたはhttpsで始まる場合、コンシューマは、モノがHTTPベースのプロトコルバインディングを実装していることを推測することができ、フォームインスタンス内のHTTP固有の用語を期待すべきである(第8.3.1項 HTTPベースのプロトコルバインディングを参照)。

- WoTTD内のすべてのフォームは、そのhrefメンバーのURIスキームが示すプロトコルバインディングの要件に従わな**ければならない。**
- WoT TD内のすべてのフォームは、対話内でモノが受け入れる要求(要求ヘッダーなどがあれば)を正確に記述しな**ければならない。**

1. HTTPベースのプロトコルバインディング

デフォルトに従い、TDは、RDF1.0 [HTTP-in-RDF10]内のHTTPボキャブラリのHTTPRDFボキャブラリ定義を入れることによって、HTTPベースのプロトコルバインディングをサポートする。このボキャブラリは、http://www.w3.org/2011/http#を指すプレフィックスhtvを使ってTDインス

タンス内で直接的に使用することができる。さらに、HTTPベースのプロトコルバインディングの詳細に関しては、[WOT-BINDING-TEMPLATE]を参照。

HTTPベースのプロトコルバインディングを実装するモノと対話するために、コンシューマは、フォームを送信するときにどのHTTPメソッドを使用するかを知っている必要がある。一般的なケースでは、TDは、メソッドを示す用語、すなわちhtv:methodNameを明示的に含むことができる。簡潔にするために、HTTPベースのプロトコルバインディングは、各操作タイプのデフォルト値を定義し、これは、また、モノが期待するメソッド(例えば、読み取りのためのGET、書き込みのためのPUT)の収束を目的とする。HTTPベースのプロトコルバインディングを表すフォームの中でメソッドが示されていない場合、デフォルト値は下表のように仮定されなければならない。

ボキャブラリ用語	デフォルト値	コンテキスト
htv:methodName	GET	操作タイプreadpropertyを持つのフォーム
htv:methodName	PUT	操作タイプwriteproperty を持つフォーム
htv:methodName	ポスト	操作タイプinvokeaction を持つフォーム

例えば、第１項はじめにの例1では、この形式のオペレーションタイプとHTTP方法は入っていない。以下のデフォルト値は、例１の形式のためであると考えるべきである。

例32

HTTPベースのプロトコルバインディングデフォルト値を使用

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1"、 "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing"、 "securityDefinitions": {
        "basic_sc": {
            「スキーム」:「基本」、
            "in": "header"
        }
    },
    「セキュリティ」: [
        "basic_sc"
    ],
    "properties": {"status": {
            "type": "string"、 "forms": [
                {
                    "op": "readproperty",
                    「href」:「https://mylamp.example.com/status」、 "htv:methodName": "GET"
                },
                {
                    "op": "writeproperty",
                    「href」:「https://mylamp.example.com/status」、 "htv:methodName": "PUT"
                }
            ]
        }
    },
    "actions": {
        "toggle": {
                                        "forms": [
                                            {
                    "op": "invokeaction",
                    「href」:「https://mylamp.example.com/toggle」、 "htv:methodName": "POST"
                }
            ]
        }
    },
    "events": {
        "overheating": {"data": {
                "type": "string"
            },
            "forms": [
                                            {
```

「href」:「https://mylamp.example.com/oh」、"subprotocol": "longpoll"
        }
    ]
}

```
        }
    }
```

2. その他のプロトコルバインディング

モノが実装できるプロトコルバインディングの数は制限されていない。他のプロトコルバインディング(例えば、CoAP、MQTT、または、OPC UAのための)は、RDF 1.0[HTTP–in–RDF10]のHTTPボキャブラリと同様のプロトコルボキャブラリ、または、デフォルト値定義を含む仕様など別個の文書で標準化される事になっている。このようなプロトコルは、TDコンテキスト拡張メカニズムの使用によってTDに単純に結合することができる(第7項TDコンテキスト拡張子を参照)。

IoTプラットフォームとエコシステムの説明方法に関する情報に関しては、[WOT-BINDING-TEMPLATE]を参照。

9. セキュリティとプライバシーに関する考慮事項

本項は標準ではない。

一般に、WoTシステムを保護するために取られるセキュリティ対策は、システムが直面する可能性がある脅威および攻撃者、ならびに、保護する必要がある資産の価値に依存する。さらに、プライバシーリスクは、識別可能な人とモノの関連性、および、直接的な情報とそのような関連性から入手できる推測情報に依存することになる。様々な状況に適応させることができる脅威モデルを含め、WoTに関するセキュリティおよびプライバシーの考慮事項の詳細な考察は、参考文献[WOT-SECURITY-CONSIDERATIONS]に記載されている。本項では、セキュリティとプライバシーリスクと、WoTTDに直接関連する実行可能な軽減対策についてのみ説明する。

WoTTDは、安全なネットワークインターフェースと安全でないネットワークインターフェースの両方を記述することができる。TDが既存のネットワークインターフェースに組み込まれる場合、そのネットワークインターフェースのセキュリティ状態に変化は期待できない。

WoTTDの使用は、以下の項で挙げられるセキュリティおよびプライバシーリスクを紹介している。各リスク説明の後、いくつかの実行可能な軽減対策を提案する。

1. プライバシーリスクをデリフェレンスするコンテキスト

JSON-LD [json-ld11] ドキュメントの@context メンバーで指定されたボキャブラリファイルのデリフェレンスはプライバシーリスクになりうる。WoTの場合、攻撃者は、そのようなデリフェレンスによって生成されたネットワークトラフィックを観察することができ、特にドメイン固有のボキャブラリが使用される場合、デバイスに関する情報を推論するために、宛先IPアドレスなどのデリフェレンスのメタデータを使用することができる。これは、たとえ接続が暗号化されていてもリスクとなり、DNSプライバシーリークにつながる。

軽減対策:

ボキャブラリファイルの実際のデリフェレンスは避ける。ボキャブラリファイルは、可能な限りキャッシュされるべきである。理想的には、(既知の)ボキャブラリの識別子としてのみ機能する@contextメンバー内のURIで、それを変更不能にし、解釈デバイスに組み込み、全くデリフェレンスができないようにする。これには、既存のURIが変更不能データを参照できることを保証するために更新には新しいURIの使用が必須であるため、厳密なバージョン制御が必要となる。コンテキストファイルがTD内のメタデータを解釈するシステムにローカルで利用可能になる見込みを高めることが可能な場合には必ず、周知の標準ボキャブラリファイルを使用する。

2. 変更不能識別子のプライバシーリスク

識別子（id）を含んでいるTDは、識別可能な人と関連付けられているモノを説明することができる。このような識別子は、トラッキングなどのさまざまなリスクを呈する。しかし、その識別子も変異することができない場合、デバイスが他の人に譲渡あるいは販売され、既知のIDがその人をトラッキングするために使用されるため、トラッキングリスクは増幅する。

軽減対策:

すべての識別子は可変でなければならず、モノのidを更新するメカニズムでなければならない。具体的には、モノのidは、ハードウェアに固定されるべきではない。しかしながら、これは、識別子が固定されたURIであるというLinked　Data（リンクドデータ）の理想と矛盾する。多くの状況では、モノが再初期化される場合、識別子への更新が許容される。この場合、ソフトウェアエンティティとして、古いモノが存在しなくなり、新しいモノが作成される。これは、例えば、デバイスが新しい所有者に売られると、十分にトラッキングチェーンを遮断することができる。あるいは、デバイスの動作状態中により頻繁な変更が望まれる場合、変更が行われたときに識別子の変更を許可されたユーザのみに通知するメカニズムを導入することができる。しかしながら、いくつかのクラスのデバイス、例えば、医療デバイスは、

いくつかの管轄区域において法律によって不変のIDを必要とすることがあるということに留意されたい。この場合、そのような不変の識別子を含むTDなどのファイルへのアクセスを保証するために特別な注意が払われるべきである。また、できる限り、 そのような場合、“真に”変異不可能な識別子をTD内で共有しないことが望ましい。

3. 指紋プライバシーリスク

上述したように、TD内のidメンバーは、プライバシーリスクを引き起こしうる。しかしながら、その追跡リスクを軽減するために説明の通りにidが更新されたとしても、指紋を介して、TDを特殊な物理デバイスに関連付け、そこから、指紋を使って識別可能な人にたどり着くことが依然として可能でる。

特定のデバイスインスタンスが指紋で識別できない場合、一連の対話などTD内の情報からデバイスタイプを推測し、医療状態など、識別可能な人に関する個人情報を推測するために使用することができる。

軽減対策:

許可されたユーザのみが、モノのTDへのアクセスを提供されるべきである。また、許可レベルに必要な情報量と使用例のみが提供されるべきである。TDが、たとえば、認証を要求するディレクトリサービスを通してなど安全で機密性のある経路で許可されたユーザのみに配信できるならば、外部の許可のない人たちは指紋のためのTDへのアクセス権を持たない。 このリスクをさらに軽減するために、TTDの特定の使用には不必要な情報は、可能な限り、省くべきである。たとえば、コンシューマがモノの状態を保存しないデバイスへの特別な接続に関しては、idは省くことができる。コンシューマがその使用のためにある対話が必要でない場合、それを省くことができる。コンシューマが、ある対話を使用する許可を得ていない場合、それも省くことができる。コインシュー間が、人間が読み取り可能な情報、たとえば、タイトルや説明を表示する能力を持たない場合、省くことができる。あるいは、ゼロ長ストリングに変えることができる。

4. グローバル一意識別子プライバシーリスク

グローバル一意識別子は、第三者が識別子を知っていることになるので、一元的権限者がこれらを生成／配布する必要がある場合プライバシーリスクを呈する。

軽減対策:

TD内のidフィールドは、意識してグローバル一意にする必要はない。一元的登録を必要としない配布方法で適切なIDを生成するために利用できる暗号メカニズムが複数存在する。これらが、通常、同一識別子を生成する可能性は非常に低い。また、これがシステム設計で考慮する必要がある。たとえば、必要に応じ、同一のIDを検知し、IDの再生成を行う。IDの範囲も、また、グローバルである必要はない。あるコンテキストでモノを判別する識別子の使用が好ましい。例としては、家庭あるいは工場内。

5. TD傍受と改竄セキュリティリスク

コンテキストファイルの傍受および改竄は、ボキャブラリの解釈を変更して攻撃を容易にするために使用され得る。

軽減対策:

理想的には、コンテキストファイルは、認証されたチャネルを介してのみ取得されるということであるが、デリフェレンスされた場合、傍受および変更に対して脆弱であるHTTP URLを使って多くのコンテキストが表示されるということは注目に値する(かつ、残念なことである)。しかし、コンテキストファイルが変更不能でキャッシュされ、可能な限りデリフェレンスが回避されれば、このリスクは低減できる。

6. 個人情報プライバシーリスクの推測

たくさんの場所で、ユーザのプライバシーを保護するために、個人情報、すなわち、特定の個人に関連付けることができる情報を処理するための法的要件がある。このような情報は、もちろん、IoTデバイスが直接生成することができる。しかしながら、IoTデバイスの存在及びメタデータ(TDに格納されたデータの種類)は、個人情報を持っているか、又は、推論するために使用することもできる。この情報は、特定の個人がある種のデバイスを所有しているという事実と同じくらい単純であり得る。そして、その個人に関する追加の推論につながり得る。

軽減対策:

個人デバイスに関連付けられたTDは、それが個人情報を含んでいるかのように扱う。本原則の適用例として、ユーザ同意の取得方法を考えてみてください。モノが生成する個人データの使用に対する同意は、モノが、データを消費するシステムと組み合わされるときに取得されることが多く、これは、TDがデバイスにアクセスするためにローカルディレクトリまたはTDを消費するシステムに登録されるときにも頻繁に発生する。この場合、モノから送信されるデータの使用同意は、モノのTDにアクセスするための同意と組み合わせることができる。第2の例として、TDが個人情報を含むと考える場合、個人情報は無期限に保持されるべきではなく、同意が与えられた目的以外に使用されるべきでない。

10. IANAの考慮事項

1. application/td+jsonメディアタイプ登録

型名：

アプリケーション

サブ型名:

td+json

必須パラメータ:

なし

オプションパラメータ:

なし

エンコーディングに関する考慮事項:

RFC 6839、第3.1項を参照。

セキュリティ考慮事項:

RFC 8259を参照。

WoT TDは、モノのメタデータの純粋なデータ交換フォーマットのためのものであるため、シリアライズは、構文解析されるJavaScriptのeval()関数などのコード実行メカニズムをすり抜けて**はならない。**(無効な)文書は、実行されると、システムのセキュリティを危うくする予想外の副次的影響をもたらす可能性があるコードを含むこともある。

WoTTDはJSON-LD 1.1 プロセッサで評価することができる。JSON-LD 1.1 プロセッサは、通常、自動的にリモートコンテキスト(TDコンテキスト拡張子、第7項TDコンテキスト拡張子を参照)へのリンクに従い、各コンシューマからの明示的なリクエストを得ずに、ファイルが転送される。リモートコンテキストが第３者から提供されると、第３者が、プライバシーの懸念につながる使用パターンまたは同様の情報を収集することができるようになる。リソースが制約されたデバイス上のインプリメンテーションには、(JSON-LD処理とは対照的に)未加工のJSON処理実行が期待されるが、一般的には、インプリメンテーションは、サポートされているコンテキスト拡張子の精査済みバージョンを静的にキャッシュする**べきであり、**リモートコンテキストへのリンクに従うべ**きではない。**サポートされているコンテキスト拡張子は、その代わり、安全なソフトウェア更新メカニズムで管理することができる。

HTTPなどの安全でない接続でウェブからロードされるコンテキスト拡張子(第7項TDコンテキスト拡張子を参照)には、セキュリティを危険にさらしうる方法でTD情報モデルを変更するように攻撃者が変更するリスクがある。このため、コンシューマは、システムがリモートコンテキストを使用できるようにする前に、再度、リモートコンテキストを精査し、キャッシュするべきである。

JSON-LD処理には、通常、長いIRI[RFC3987]を短い用語で置き換えるということを鑑み、WoTTDは、JSON-LD 1.1プロセッサを使用して処理されるときにかなり拡張することがあり、最悪の場合には、結果として得られるデータは、受信者のリソース全てを消費することもある。コンシューマは、何らかのTDメタデータを相当の疑いを持って扱う**べきである。**

相互運用性の考慮事項:

RFC 8259を参照。

適合コンテンツと非適合コンテンツの両方を処理するための規則は、本仕様で定義される。

公開されている仕様:

https://w3c.github.io/wot-thing-description

本メディアタイプを使用するアプリケーション:

W3C WoT内の全参加エンティティ、すなわち、Web of Things (WoT)アーキテクチャで定義されているモノ、コンシューマ、および仲介者。

フラグメント識別子の考慮事項:

RFC 6839、第3.1項を参照。

追加情報

マジックナンバー:

適用外

ファイル拡張子:

.jsontd

Macintosh ファイルタイプコード:

テキスト

詳細情報に関する連絡先とメールアドレス:

Matthias Kovatsch <w3c@kovatsch.net>

意図されたアプリケーション:

共通

使用上の制約事項:

なし

著者:

WoTTD仕様は、Web of Things Working Groupの成果物である。

変更管理者:

W3C

2. CoAPコンテンツフォーマット登録

IANAは、Constrained RESTful Environments (CoRE)パラメータ登録 [RFC7252]内のCoAPコンテンツフォーマットサブレジストリのメディアタイプに対し簡潔なCoAPコンテンツフォーマットIDを割り当てている。WoTTDのコンテンツフォーマットIDは、256から9999までの(t.b.d.)である(IETF レビューまたはIESG承認)。

メディアタイプ:

application/td+json

エンコーディング:

-

ID:

T.B.D.

参考:

[「Web of Things (WoT)Thing Description」、2019年5月]

A. TDインスタンスの例

本項は標準ではない。

A.1 CoAP プロトコルバインディングを使用したMyLampThing の例

モノの特徴リスト:

- タイトル: MyLampThing
- コンテンツ拡張子: なし
- 提供されるアフォーダンス:プロパティ一つ、アクション一つ、 イベント一つ
- セキュリティ: PSKSecurityScheme（PSKセキュリティスキーム）
- プロトコルバインディング: TLSのCoAP [RFC7252]
- コメント: 第7.2項 プロトコルバインディングの追加を参照

例33: CoAP プロトコルバインディングを使用したMyLampThing


```
{
    "@context": ["https://www.w3.org/2019/wot/td/v1",
      {
      "cov": "http://www.example.org/coap-binding#"
      }
    ],
    "id": "urn:dev:ops:32473-WoTLamp-1234"、"title": "MyLampThing",
    "description" : "MyLampThing uses JSON serialization"、"securityDefinitions": {"psk_sc":{"scheme":
    "psk"}}、"security": ["psk_sc"],
    "properties": {"status": {
            「description」:「ランプの現在のステータスを表示します」、「type」:「string」、
          "forms": [{
              "href": "coaps://mylamp.example.com/status","cov:methodName" : "GET"
          }]
        }
    },
    "actions": {
        "toggle": {
          "description" : "ランプのオン/オフを切り替える"、"forms": [{
              "href": "coaps://mylamp.example.com/toggle","cov:methodName" : "POST"
          }]
        }
    },
    "events": {
        「過熱」:{
          「description」:「ランプが臨界温度(過熱)」、「データ」:{「type」:「string」}、
          "forms": [{
              "href": "coaps://mylamp.example.com/oh","cov:methodName" : "GET",
              "subprotocol" : "cov:observe"
          }]
        }
    }
}
```


A.2 MQTT プロトコルバインディングを使用したMyLightSensor の例

モノの特徴リスト:

- タイトル: MyLampSensor
- コンテンツ拡張子: なし
- 提供されるアフォーダンス: イベント一つ
- セキュリティ: なし

- プロトコルバインディング: MQTT [MQTT]
- コメント: MQTTクライアントは、アドレス192.168.1.187:1883 の背後で実行されているMQTT ブローカーによって、トピック/lightSensor に光センサデータ(数字はテキスト形式でシリアライズされている) を頻繁に発行する。

例34: MQTT プロトコルバインディングを使用したMyLightSensor

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1"、"title": "MyLightSensor",
    "id": "urn:dev:ops:32473-WoTLightSensor-1234"、"securityDefinitions": {"nosec_sc": {"scheme": "nosec"}}、
    "security": ["nosec_sc"],
    "events": {
        "lightSensor": {"data":{"type": "integer"}、"forms": [
                {
                    "href": "mqtt://192.168.1.187:1883/lightSensor"、"contentType" : "text/plain"
                }
            ]
        }
    }
}
```

A. 3 Webhookイベントの例

モノの特徴リスト:

- タイトル: WebhookThing
- コンテキスト拡張子: HTTPプロトコルバインディング 補足(TD コンテキストにすでに含まれているhtv プレフィックス)を使用する。
- 提供されるアフォーダンス: 1 イベント
- セキュリティ: なし
- プロトコルバインド: HTTP
- コメント: WebhookThingは、Webhookメカニズムを使用して、最新の温度値を定期的にコンシューマにプッシュするイベントアフォーダンスtemperatureを提供し、モノは、コンシューマが提供するコールバックURIにPOST要求を送信する。これを説明するために、subscriptionメンバーは、subscribeeventフォームを使って送信されなければならない書き込み専用パラメータcallbackURLを定義する。読み取り専用パラメータsubscriptionID は、そのサブスクリプションが返信する。WebhookThingは、dataによって定義されたペイロードを有するこのコールバックURIに定期的に**ポスト**する。サブスクライブを解除するには、コンシューマは、URIテンプレートを利用するunsubscribeeventフォームを送信しなければならない。uriVariablesメンバーは、subscriptionIDストリングを入れるようにコンシューマに通知する。これは、適切な意味注釈を含めるために、TDコンテキスト拡張子を使ってさらに自動化することができる。あるいは、subscriptionと同様にcancellationメンバーを使ったサブスクライブ解除を想定し、これを、サブスクライブ解除するためのペイロードで**ポスト**要求を記述するunsubscribeeventフォームと組み合わせることができる。

例35:　サブスクリプションおよびキャンセルを伴う温度イベント


```
{
    "@context": "https://www.w3.org/2019/wot/td/v1"、 "id": "urn:dev:ops:32473-Thing-1234",
    「タイトル」:「WebhookThing」、
    "description": "Webhook-based Event with subscription and unsubscribe form.","securityDefinitions": {"nosec_sc": {"scheme":
    "nosec"}},
    "security": ["nosec_sc"]、 "events": {
        "temperature": {
            "description": "周期的な温度値の更新を提供します。""subscription": {
                "type": "object"、 "properties": {
                    "callbackURL": {"type": "string",
                        "format": "uri",
                        "description": "Webhook no "writeOnly" のためにサブスクライバーによって提供されたコールバックURL: true
                    },
                    "subscriptionID": {"type": "string",
                        「description」:「readOnly」で提供されるキャンセル用の一意のサブスクリプションID: true
                    }
                }
            },
            "data": {

                "type": "number"、 "description": "コールバックURLに送信される最新温度値。
            },
            "cancellation": {"type": "object"、 "properties": {
                    "subscriptionID": {"type": "integer",
                        「description」:「サブスクリプションを取り消すために必要なサブスクリプションID」、 "writeOnly": true
                    }
                }
            },
            "uriVariables": {
                "subscriptionID": {"type": "string"}
            },
            "forms": [
                                                {
                    "op": "subscribeevent",
                    "href": "http://192.168.0.124:8080/events/temp/subscribe"、 "contentType": "application/json",
                    "htv:methodName": "POST"
                },
                {
                    "op": "unsubscribeevent",
                    "href": "http://192.168.0.124:8080/events/temp/{subscriptionID}"、 "htv:methodName": "DELETE"
```

```
                    }
                ]
            }
        }
    }
```

B. TDインスタンス確認のためのJSONスキーマ

本項は標準ではない。

以下は、JSONベースのフォーマットでシリアライズされたTDインスタンスを構文的に確認するためのJSONスキーマ[JSON-SCHEMA]文書である。

注

本文書によって定義されたTDは、JSON-LD [json-ld11]からわかる@contextメカニズムを使用することによって外部ボキャブラリを追加することができる。また、この外部ボキャブラリ中の用語は、第5項TD情報モデルで定義されている用語に加えて使用することができる。このため、下記JSON スキーマは意図的にその点に関して厳密には記載されていない。外部ボキャブラリが使用されていない場合により厳密な確認を実行するために、異なるスコープ/レベルでadditionalPropertiesスキーマプロパティtrue の値をfalseに置き換えることができる。

注

一部のJSON スキーマ確認ツールでは、iriストリングフォーマットをサポートしていないということに注意する。

TDインスタンスを確認するための以下のJSONスキーマは、デフォルト値を持つ用語が存在することを要求しない。したがって、デフォルト値を持つ用語は任意選択である。(第5.4項 デフォルト値定義も参照)

```
 {
"title": "WoT TD Schema - 16 October 2019",
"description": "JSON Schema for validating TD instances against the TD model. TD instances can be with or without terms that have default values",
"$schema ": "http://json-schema.org/draft-07/schema#",
"definitions": {
"thing-context-w3c-uri": {
"type": "string",
"enum": [
"https://www.w3.org/2019/wot/td/v1"
]
},
"thing-context": {
"oneOf": [{
```

```
"type": "array",
"items": {
"anyOf": [{
"$ref": "#/definitions/anyUri"
},
{
"type": "object"
}
]
},
"contains": {
"$ref": "#/definitions/thing-context-w3c-uri"
}
},
{
"$ref": "#/definitions/thing-context-w3c-uri"
}
]
},
"type_declaration": {
"oneOf": [{
"type": "string"
},
{
"type": "array",
"items": {
"type": "string"
}
}
]
},
"property_element": {
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"title": {
```

```
"$ref": "#/definitions/title"
},
"titles": {
"$ref": "#/definitions/titles"
},
"uriVariables": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/dataSchema"
}
},
"forms": {
"type": "array",
"minItems": 1,
"items": {
"$ref": "#/definitions/form_element_property"
}
},
"observable": {
"type": "boolean"
},
"writeOnly": {
"type": "boolean"
},
"readOnly": {
"type": "boolean"
},
"oneOf": {
"type": "array",
"items": {
"$ref": "#/definitions/dataSchema"
}
},
"unit": {
"type": "string"
},
"enum": {
"type": "array",
"minItems": 1,
"uniqueItems": true
},
"format": {
"type": "string"
},
```

```
"const": {},
"type": {
"type": "string",
"enum": [
"boolean",
"integer",
"number",
"string",
"object",
"array",
"null"
]
},
"items": {
"oneOf": [{
"$ref": "#/definitions/dataSchema"
},
{
"type": "array",
"items": {
"$ref": "#/definitions/dataSchema"
}
}
]
},
"maxItems": {
"type": "integer",
"minimum": 0
},
"minItems": {
"type": "integer",
"minimum": 0
},
"minimum": {
"type": "number"
},
"maximum": {
"type": "number"
},
"properties": {
"additionalProperties": {
"$ref": "#/definitions/dataSchema"
}
},
```

```
"required": {
"type": "array",
"items": {
"type": "string"
}
}
},
"required": [
"forms"
],
"additionalProperties": true
},
"action_element": {
"type": "object",
"properties": {
"description": {
"type": "string"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"title": {
"$ref": "#/definitions/title"
},
"titles": {
"$ref": "#/definitions/titles"
},
"uriVariables": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/dataSchema"
}
},
"@type": {
"$ref": "#/definitions/type_declaration"
},
"forms": {
"type": "array",
"minItems": 1,
"items": {
"$ref": "#/definitions/form_element_action"
}
},
"input": {
```

```
"$ref": "#/definitions/dataSchema"
},
"output": {
"$ref": "#/definitions/dataSchema"
},
"safe": {
"type": "boolean"
},
"idempotent": {
"type": "boolean"
}
},
"required": [
"forms"
],
"additionalProperties": true
},
"event_element": {
"type": "object",
"properties": {
"description": {
"type": "string"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"title": {
"$ref": "#/definitions/title"
},
"titles": {
"$ref": "#/definitions/titles"
},
"uriVariables": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/dataSchema"
}
},
"@type": {
"$ref": "#/definitions/type_declaration"
},
"forms": {
"type": "array",
"minItems": 1,
```

```
"items": {
"$ref": "#/definitions/form_element_event"
}
},
"subscription": {
"$ref": "#/definitions/dataSchema"
},
"data": {
"$ref": "#/definitions/dataSchema"
},
"cancellation": {
"$ref": "#/definitions/dataSchema"
},
"type": {
"not": {}
},
"enum": {
"not": {}
},
"const": {
"not": {}
}
},
"required": [
"forms"
],
"additionalProperties": true
},
"form_element_property": {
"type": "object",
"properties": {
"href": {
"$ref": "#/definitions/anyUri"
},
"op": {
"oneOf": [{
"type": "string",
"enum": [
"readproperty",
"writeproperty",
"observeproperty",
"unobserveproperty"
]
},
```

```
{
"type": "array",
"items": {
"type": "string",
"enum": [
"readproperty",
"writeproperty",
"observeproperty",
"unobserveproperty"
]
}
}
]
},
"contentType": {
"type": "string"
},
"security": {
"type": "array",
"items": {
"type": "string"
}
},
"scopes": {
"type": "array",
"items": {
"type": "string"
}
},
"subProtocol": {
"type": "string",
"enum": [
"longpoll",
"websub",
"sse"
]
},
"response": {
"type": "object",
"properties": {
"contentType": {
"type": "string"
}
}
```

```
}
},
"required": [
"href"
],
"additionalProperties": true
},
"form_element_action": {
"type": "object",
"properties": {
"href": {
"$ref": "#/definitions/anyUri"
},
"op": {
"oneOf": [{
"type": "string",
"enum": [
"invokeaction"
]
},
{
"type": "array",
"items": {
"type": "string",
"enum": [
"invokeaction"
]
}
}
]
},
"contentType": {
"type": "string"
},
"security": {
"type": "array",
"items": {
"type": "string"
}
},
"scopes": {
"type": "array",
"items": {
"type": "string"
```

```
}
},
"subProtocol": {
"type": "string",
"enum": [
"longpoll",
"websub",
"sse"
]
},
"response": {
"type": "object",
"properties": {
"contentType": {
"type": "string"
}
}
}
},
"required": [
"href"
],
"additionalProperties": true
},
"form_element_event": {
"type": "object",
"properties": {
"href": {
"$ref": "#/definitions/anyUri"
},
"op": {
"oneOf": [{
"type": "string",
"enum": [
"subscribeevent",
"unsubscribeevent"
]
},
{
"type": "array",
"items": {
"type": "string",
"enum": [
"subscribeevent",
```

```
"unsubscribeevent"
]
}
}
]
},
"contentType": {
"type": "string"
},
"security": {
"type": "array",
"items": {
"type": "string"
}
},
"scopes": {
"type": "array",
"items": {
"type": "string"
}
},
"subProtocol": {
"type": "string",
"enum": [
"longpoll",
"websub",
"sse"
]
},
"response": {
"type": "object",
"properties": {
"contentType": {
"type": "string"
}
}
}
},
"required": [
"href"
],
"additionalProperties": true
},
"form_element_root": {
```

```
"type": "object",
"properties": {
"href": {
"$ref": "#/definitions/anyUri"
},
"op": {
"oneOf": [{
"type": "string",
"enum": [
"readallproperties",
"writeallproperties",
"readmultipleproperties",
"writemultipleproperties"
]
},
{
"type": "array",
"items": {
"type": "string",
"enum": [
"readallproperties",
"writeallproperties",
"readmultipleproperties",
"writemultipleproperties"
]
}
}
]
},
"contentType": {
"type": "string"
},
"security": {
"type": "array",
"items": {
"type": "string"
}
},
"scopes": {
"type": "array",
"items": {
"type": "string"
}
},
```

```
"subProtocol": {
"type": "string",
"enum": [
"longpoll",
"websub",
"sse"
]
},
"response": {
"type": "object",
"properties": {
"contentType": {
"type": "string"
}
}
}
},
"required": [
"href"
],
"additionalProperties": true
},
"description": {
"type": "string"
},
"title": {
"type": "string"
},
"descriptions": {
"type": "object"
},
"titles": {
"type": "object"
},
"dataSchema": {
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"title": {
```

```
"$ref": "#/definitions/title"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"titles": {
"$ref": "#/definitions/titles"
},
"writeOnly": {
"type": "boolean"
},
"readOnly": {
"type": "boolean"
},
"oneOf": {
"type": "array",
"items": {
"$ref": "#/definitions/dataSchema"
}
},
"unit": {
"type": "string"
},
"enum": {
"type": "array",
"minItems": 1,
"uniqueItems": true
},
"format": {
"type": "string"
},
"const": {},
"type": {
"type": "string",
"enum": [
"boolean",
"integer",
"number",
"string",
"object",
"array",
"null"
]
},
```

```
"items": {
"oneOf": [{
"$ref": "#/definitions/dataSchema"
},
{
"type": "array",
"items": {
"$ref": "#/definitions/dataSchema"
}
}
]
},
"maxItems": {
"type": "integer",
"minimum": 0
},
"minItems": {
"type": "integer",
"minimum": 0
},
"minimum": {
"type": "number"
},
"maximum": {
"type": "number"
},
"properties": {
"additionalProperties": {
"$ref": "#/definitions/dataSchema"
}
},
"required": {
"type": "array",
"items": {
"type": "string"
}
}
}
},
"link_element": {
"type": "object",
"properties": {
"anchor": {
"$ref": "#/definitions/anyUri"
```

```
},
"href": {
"$ref": "#/definitions/anyUri"
},
"rel": {
"type": "string"
},
"type": {
"type": "string"
}
},
"required": [
"href"
],
"additionalProperties": true
},
"securityScheme": {
"oneOf": [{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"nosec"
]
}
},
"required": [
"scheme"
]
},
{
```

```
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"basic"
]
},
"in": {
"type": "string",
"enum": [
"header",
"query",
"body",
"cookie"
]
},
"name": {
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
```

```
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"cert"
]
},
"identity": {
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"digest"
]
},
"qop": {
```

```
"type": "string",
"enum": [
"auth",
"auth-int"
]
},
"in": {
"type": "string",
"enum": [
"header",
"query",
"body",
"cookie"
]
},
"name": {
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"bearer"
]
},
```

```
"authorization": {
"$ref": "#/definitions/anyUri"
},
"alg": {
"type": "string",
"enum": [
"MD5",
"ES256",
"ES512-256"
]
},
"format": {
"type": "string",
"enum": [
"jwt",
"jwe",
"jws"
]
},
"in": {
"type": "string",
"enum": [
"header",
"query",
"body",
"cookie"
]
},
"name": {
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
```

```
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"psk"
]
},
"identity": {
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"public"
]
},
"identity": {
"type": "string"
```

```
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"oauth2"
]
},
"authorization": {
"$ref": "#/definitions/anyUri"
},
"token": {
"$ref": "#/definitions/anyUri"
},
"refresh": {
"$ref": "#/definitions/anyUri"
},
"scopes": {
"type": "array",
"items": {
"type": "string"
}
},
"flow": {
"type": "string",
```

```
"enum": [
"implicit",
"password",
"client",
"code"
]
}
},
"required": [
"scheme",
"flow"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"apikey"
]
},
"in": {
"type": "string",
"enum": [
"header",
"query",
"body",
"cookie"
]
},
"name": {
```

```
"type": "string"
}
},
"required": [
"scheme"
]
},
{
"type": "object",
"properties": {
"@type": {
"$ref": "#/definitions/type_declaration"
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"proxy": {
"$ref": "#/definitions/anyUri"
},
"scheme": {
"type": "string",
"enum": [
"pop"
]
},
"authorization": {
"$ref": "#/definitions/anyUri"
},
"format": {
"type": "string",
"enum": [
"jwt",
"jwe",
"jws"
]
},
"alg": {
"type": "string",
"enum": [
"MD5",
"ES256",
```

```
"ES512-256"
]
},
"in": {
"type": "string",
"enum": [
"header",
"query",
"body",
"cookie"
]
},
"name": {
"type": "string"
}
},
"required": [
"scheme"
]
}
]
},
"anyUri": {
"type": "string",
"format": "iri-reference"
}
},
"type": "object",
"properties": {
"id": {
"type": "string",
"format": "uri"
},
"title": {
"$ref": "#/definitions/title"
},
"titles": {
"$ref": "#/definitions/titles"
},
"properties": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/property_element"
}
```

```
},
"actions": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/action_element"
}
},
"events": {
"type": "object",
"additionalProperties": {
"$ref": "#/definitions/event_element"
}
},
"description": {
"$ref": "#/definitions/description"
},
"descriptions": {
"$ref": "#/definitions/descriptions"
},
"version": {
"type": "object",
"properties": {
"instance": {
"type": "string"
}
},
"required": [
"instance"
]
},
"links": {
"type": "array",
"items": {
"$ref": "#/definitions/link_element"
}
},
"forms": {
"type": "array",
"minItems": 1,
"items": {
"$ref": "#/definitions/form_element_root"
}
},
"base": {
```

```
"$ref": "#/definitions/anyUri"
},
"securityDefinitions": {
"type": "object",
"minProperties": 1,
"additionalProperties": {
"$ref": "#/definitions/securityScheme"
}
},
"support": {
"$ref": "#/definitions/anyUri"
},
"created": {
"type": "string"
},
"modified": {
"type": "string"
},
"security": {
"type": "array",
"minItems": 1,
"items": {
"type": "string"
}
},
"@type": {
"$ref": "#/definitions/type_declaration"
},
"@context": {
"$ref": "#/definitions/thing-context"
}
},
"required": [
"title",
"security",
"securityDefinitions",
"@context"
],
"additionalProperties": true
}
```

C. TDテンプレート

本項は標準ではない。

TDテンプレートは、モノのクラスの記述であり、クラウドサーバによる何千ものデバイスの共通処理を可能にするために、モノのグループ全体で共有されるプロパティ、アクション、イベント、および共通メタデータを記述しており、モノそれぞれで利用するものではない。TDテンプレートは、第5項TD情報モデルからの同じ中核ボキャブラリおよび情報モデルを使用する。

TDテンプレートを使って次のことが可能になる。

- クラウドサービスによる複数のモノの管理
- まだ開発されていないデバイス/モノのシミュレーション
- 共通のモノモデルを共有する異なるメーカーのデバイス間で共通のアプリ
- 複数のモデルをモノと組み合わせる

TDテンプレートは、デバイスとのインターフェース及び可能な対話(プロパティ、アクション、イベント)の論理的記述であるが、シリアル番号、GPS位置、セキュリティ情報、具体的なプロトコルエンドポイントなどデバイス固有の情報を含まない。

TDテンプレートは、特定のエンドポイントへのプロトコルバインディングを含まず、特定のセキュリティメカニズムを定義していないので、フォームおよびセキュリティ定義とセキュリティキーは存在してはならない。

同じTDテンプレートは、複数のベンダの複数のモノによって実装することができる。モノは複数のTDテンプレートを実装し、追加のメタデータ(ベンダ、位置、セキュリティ)を定義し、具体的なプロトコルへのバインディングを定義することができる。共通のモノに組み合わされている異なるTDテンプレートからのプロパティ、アクション、イベントの間の衝突を回避するために、これらの識別子はすべて、モノの中で唯一無二でなければならない。

あるクラスのデバイスの共通のTDテンプレートは、ベンダを超えてアプリを書くことを可能にし、アプリ開発者にとってより魅力的な市場を生み出す。具体的なTDは、複数のTDテンプレートを実装することができ、したがって、機能ブロックを結合されたデバイスに集約することができる。

クラウドベンダーのビジネスモデルは、通常、何千もの同一のデバイス管理することで構築される。同一TDテンプレートのデバイスすべては、クラウドアプリにより同じ方法で管理することができる。インターフェースとインスタンスが別々に扱われる場合、複数のシミュレートされたデバイスを作成することは容易である。

しかし、TDテンプレートは、いくつかの任意のボキャブラリ用語と必須のボキャブラリ用語が存在しないTDのサブセットであるので、TDと同じ方法・同じフォーマットでシリアライズすることができる。TDテンプレートインスタンスは、いくつかの必須用語が欠落しているため、TDインスタンスと同じ方法で確認することができないことに留意されたい。

C. 1 TDテンプレートの例

本項では、ランプのTDテンプレートとブザーのTDテンプレートを紹介する。

C.1.1 TDテンプレート: ランプ

例36: JSON でシリアライズされたMyLampTDテンプレート

```
{
    "@context": ["https://www.w3.org/2019/wot/td/v1"]、"@type" : "ThingTemplate",
```

```
        "title": "Lamp Thing Template"、"description" : "Lamp Thing Template"、"properties": {
            "status": {
                「description」:「ランプの現在のステータス(on|off)」、 「type」:「string」、
               "readOnly": true
            }
        },
        "actions": {
            "toggle": {
                「説明」:「ランプのオン/オフを切り替える」
            }
        },
        "events": {
             「過熱」:{
                "description" :"ランプが臨界温度に達する(過熱)"、"data": {"type": "string"}
            }
        }
    }
```

C.1.2 TDテンプレート: ブザー

例37: JSON でシリアライズMyBuzzerTDテンプレート

```
    {
"@context": ["https://www.w3.org/2019/wot/td/v1"],
"@type" : "ThingTemplate",
"title": "Buzzer Thing Description Template",
"description" : "Thing Description Template of a buzzer that makes noise for 10 seconds",
"actions": {
"buzz": {
"description" : "buzz for 10 seconds"
}
}
    }
```

C. JSON-LD　コンテキスト用法

本項は標準ではない。

現行の仕様書は、TD情報モデルを異なるボキャブラリに関する一連の制約、すなわち、ボキャブラリ用語として取り入れている。本項では、これらの制約の機会読み取り可能定義が、TDドキュメントの必須@contextを使用して以下にクライアントアプリに結合されたのかを簡単に説明する。

TDドキュメントからTD情報モデルへのアクセスは、２段階で行われる。まず、クライアントは、IRIへのJSONストリングからのマッピングを検索する。気尾のマッピングは、後述のとおりJSON-LDコインテキストを定義されている。そこで、クライアントは、それらをデレファランスしてIRI上で定義されている制約にアクセスすることができる。制約はクライアントプログラムが簡単に解読できるRDF形式で論理原理として定義されている。

第５項TD情報モデル内で参照されているボキャブラリ用語はすべて、TDドキュメント内で（コンパクトな）JSONストリングとしてシリアライズされている。しかしながら、これら用語のはそれぞれ、第一のLINKED Data原理[LINKED-DATA]どおり、完全なIRIで明確に識別できる。JSONキーからIRIへのマッピングは、TDの@context値が示すものである。たとえば、https：//www.w3.org/2019/wot/td/v1のファイルは、以下のマッピングを含んでいる（さらに存在する中で）；

properties -> https://www.w3.org/2019/wot/td#hasPropertyAffordance

object -> https://www.w3.org/2019/wot/json-schema#ObjectSchema

basic -> https://www.w3.org/2019/wot/security#BasicSecurityScheme__

href -> https://www.w3.org/2019/wot/hypermedia#hasTarget__

このJSONファイryはJSON-LD 1.1シンテックス[JSON-LD11]に続くものである。多数のJSON-LDライブラリが自動的にTDの@contextを処理し、その中のJSONストリングすべてを拡張することができる。

TDのボキャブラリ用語すべてがIRIに拡張されると、第二ステップは、そのボキャブラリ用語を参照するTD情報も出るのフラグメントを取得するためにこのIRIをデレファランスする。たとえば、

https://www.w3.org/2019/wot/json-schema#ObjectSchemaデレファランスすると、ObjectSchemaという用語はクラスであり、もっと正確にするとDataSchemaのサブクラスであるというRDFドキュメントとなる。このような理論原理は、さまざまな複雑な形式を使ったRDFで表現される。サブクラス関係は、RDF Schema原理「RDF-SCHEMA」として表現される。さらに、これら原理は、さま座万形式でシリアライズされることがある。ここでは、Turtle　形式[TURTL]でシリアライズされている。

<https://www.w3.org/2019/wot/json-schema#ObjectSchema>

a rdfs:Class .

<https://www.w3.org/2019/wot/json-schema#ObjectSchema>

rdfs:subClassOf <https://www.w3.org/2019/wot/json-schema#DataSchema> .

デフォルトで、ユーザエージェントが内容交渉をしない場合、人間が読み取り可能なHTML資料がRDFドキュメントの代わりに返される。内容を交渉するためには、クライアントは、自身のリクエスト内にHTTPヘッダーAccept: text/turtleを入れなければならない。

E.. 最近の仕様変更

E.1 第１次推奨候補からの変更点

- 一般

  ・TDプロセッサとの明確化された定義と、すべてのインプリメンテーションが正確に想定するように、インプリメンテーションはTDプロデューサあるいはTDコンシューマであるという関連テキスト（インプリメンテーションに影響なし）

  ・モノidの明確にされた一意性と可能な値。特に、ローカルIPアドレスを持つURIあるいは一時的で変異するURIのようなプライバシー保護する値

  ・新されたレファランス

  ・味論注釈の見直しテキスト

  ・バグ修正のために更新された例２

  nameは、第5.3.1.1Thingクラス定義への前回変更を反映するために titleへ改名された。

  @typeは、formsから改名された。

  ・バグ修正のために更新された例33

  opは、デフォルト値を無効にするためにstatusプロパティのformに追加された。

  [“readproperty”, “writeproperty”]のデフォルト値は　両方がGET法にしてしまったであろう。

  ここは意図的ではなかった。

  ・小さな編集上の見直しと修正

- 用語法

  ・第３項用語法　葉基準ではない。

- TD情報モデル

  ・第5.3.1.1Thingで、モノのボキャブラリ用語idは、オプションとなった。また、idの一意性の表現は、

  必要がないため削除された。

  ・図１のラベル位置調整

  ・アサーション・強調部分でない大文字MAYを修正。

  （データスキーマのTDコンテキスト拡張子；インプリメンテーションに影響なし）

  ・図２“JSONスキーマボキャブラリ”のタイトルは、“データスキーマボキャブラリ”に変更。

  ・OAuth2認証キュリティ構成のボキャブラリ用語は、現在、デフォルト値はない。

  よって、第5.4項デフォルト値定義から削除された。

- TD表形式

  ・第6.3.9項　formsで、オペレーションタイプwriteallproperties取り扱い時のコンシューマとモノに対する期待事項が明確化された。

- TDコンテキスト拡張子

  ・見直しされた例２８と例２９

  モノを拡張クラスで意味論的に注釈するためにsaref:TemperatureSensorを使用する。

  例中で整合性を取るためにプロパティを注釈するためにSAREFを使用する。

  より新しいontology-of-units-of measure(om)を使用する。

  ・例２との設計上の違いを明確にするために例２９を更新した。

- ビヘイビアアサーション

  ・第8.2項Data Schemasはデータスキーマに説明されていないにもかかわらず、

  モノが追加データを返し、コンシューマが追加データを受け取らなければならないときの

詳細な状況を説明している。

・第8.3.1項HTTPベースのプロトコルバインディングは、

“HTTPプロトコルバインディング”から改名された。

- セキュリティとプライバシーに関する考慮事項

・一意でなくオプションのモノのボキャブラリ用語idと整合させるために

第9項セキュリティとプライバシーに関する考慮事項を更新。

使用例に必要のないときにIDをフィルタリングするといった軽減対策を追加。

デバイスタイプを特徴で識別するといったリスクを追加。

グローバル一意識別子リスクを追加し、これが不要であり、配布された暗号メカニズムあるいはローカルな範囲内のidを使用することができるということを指摘している。

- 付録

・付録C　TDテンプレートは、“モノテンプレート”から改名。

・付録D JDON-LDコンテキスト用法は、RDF内のTD情報検索のための

２段階ステップを説明している。

・WoT Architecture仕様[WOT-ARCHITECTURE]書参照は、有益である。

E. 2 第3次公開作業ドラフトからの変更点

第3次公開作業ドラフトからの変更点は、Candidate Recommendation推奨候補内で説明されている。

E. 謝辞

編集者は、御寄稿、ガイダンスおよび専門知識を御提供頂いたMichael Koster, Michael Lagally, Kazuyuki Ashimura, Ege Korkan, Daniel Peintner, Toru Kawaguchi, María Poveda, Dave Raggett, Kunihiko Toumura, Takeshi Yamada, Ben Francis, Manu Sporny, Klaus Hartke, Addison Phillips, Jose M. Cantera, Tomoaki Mizushima, Soumya Kanti Datta and Benjamin Klotzに謝意を表します。

また、本文書改良を可能にしたサポート、技術的入力、および提案を頂いたW3Cスタッフ及びW3C WoTインタレストグループ (WoT IG)およびワーキンググループ (WoT WG)の他のすべての現役および前参加者に対し謝意を表します。

G. References

G.1 Normative references

[BCP47]
Tags for Identifying Languages. A. Phillips; M. Davis. IETF. September 2009. IETF Best Current Practice. URL: https://tools.ietf.org/html/bcp47
[HTTP-in-RDF10]
HTTP Vocabulary in RDF 1.0. Johannes Koch; Carlos A. Velasco; Philip Ackermann. W3C. 2 February 2017. W3C Note. URL: https://www.w3.org/TR/HTTP-in-RDF10/
[IANA-MEDIA-TYPES]
Media Types. IANA. URL: https://www.iana.org/assignments/media-types/
[MQTT]
MQTT Version 5.0. Andrew Banks; Ed Briggs; Ken Borgendale; Rahul Gupta. OASIS. 31 October 2018. Candidate OASIS Standard 01. URL: http://docs.oasis-open.org/mqtt/mqtt/v5.0/cos01/mqtt-v5.0-cos01.html
[RFC2119]
Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997. Best Current Practice. URL: https://tools.ietf.org/html/rfc2119
[RFC3339]
Date and Time on the Internet: Timestamps. G. Klyne; C. Newman. IETF. July 2002. Proposed Standard. URL: https://tools.ietf.org/html/rfc3339
[RFC3629]
UTF-8, a transformation format of ISO 10646. F. Yergeau. IETF. November 2003. Internet Standard. URL: https://tools.ietf.org/html/rfc3629
[RFC3966]
The tel URI for Telephone Numbers. H. Schulzrinne. IETF. December 2004. Proposed Standard. URL: https://tools.ietf.org/html/rfc3966
[RFC3986]
Uniform Resource Identifier (URI): Generic Syntax. T. Berners-Lee; R. Fielding; L. Masinter. IETF. January 2005. Internet Standard. URL: https://tools.ietf.org/html/rfc3986
[RFC6068]
The 'mailto' URI Scheme. M. Duerst; L. Masinter; J. Zawinski. IETF. October 2010. Proposed Standard. URL: https://tools.ietf.org/html/rfc6068
[RFC6570]
URI Template. J. Gregorio; R. Fielding; M. Hadley; M. Nottingham; D. Orchard. IETF. March 2012. Proposed Standard. URL: https://tools.ietf.org/html/rfc6570

[RFC6749]
The OAuth 2.0 Authorization Framework. D. Hardt, Ed.. IETF. October 2012. Proposed Standard. URL: https://tools.ietf.org/html/rfc6749
[RFC7231]
Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. R. Fielding, Ed.; J. Reschke, Ed.. IETF. June 2014. Proposed Standard. URL: https://tools.ietf.org/html/rfc7231
[RFC7252]
The Constrained Application Protocol (CoAP). Z. Shelby; K. Hartke; C. Bormann. IETF. June 2014. Proposed Standard. URL: https://tools.ietf.org/html/rfc7252
[RFC7516]
JSON Web Encryption (JWE). M. Jones; J. Hildebrand. IETF. May 2015. Proposed Standard. URL: https://tools.ietf.org/html/rfc7516
[RFC7519]
JSON Web Token (JWT). M. Jones; J. Bradley; N. Sakimura. IETF. May 2015. Proposed Standard. URL: https://tools.ietf.org/html/rfc7519
[RFC7797]
JSON Web Signature (JWS) Unencoded Payload Option. M. Jones. IETF. February 2016. Proposed Standard. URL: https://tools.ietf.org/html/rfc7797
[RFC8174]
Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017. Best Current Practice. URL: https://tools.ietf.org/html/rfc8174
[RFC8252]
OAuth 2.0 for Native Apps. W. Denniss; J. Bradley. IETF. October 2017. Best Current Practice. URL: https://tools.ietf.org/html/rfc8252
[RFC8259]
The JavaScript Object Notation (JSON) Data Interchange Format. T. Bray, Ed.. IETF. December 2017. Internet Standard. URL: https://tools.ietf.org/html/rfc8259
[RFC8288]
Web Linking. M. Nottingham. IETF. October 2017. Proposed Standard. URL: https://tools.ietf.org/html/rfc8288
[RFC8392]
CBOR Web Token (CWT). M. Jones; E. Wahlstroem; S. Erdtman; H. Tschofenig. IETF. May 2018. Proposed Standard. URL: https://tools.ietf.org/html/rfc8392
[SHACL]
Shapes Constraint Language (SHACL). Holger Knublauch; Dimitris Kontokostas. W3C. 20 July 2017. W3C Recommendation. URL: https://www.w3.org/TR/shacl/
[WOT-ARCHITECTURE]
Web of Things (WoT) Architecture. Matthias Kovatsch; Ryuichi Matsukura; Michael Lagally; Toru Kawaguchi; Kunihiko Toumura; Kazuo Kajimoto. W3C. May 2019. URL: https://w3c.github.io/wot-architecture/
[WoT-Binding-Templates]
Web of Things (WoT) Protocol Binding Templates. Michael Koster. W3C. 5 April 2018. W3C Note. URL: https://www.w3.org/TR/wot-binding-templates/
[X509V3]
ITU-T Recommendation X.509 version 3 (1997). "Information Technology - Open Systems Interconnection - The Directory Authentication Framework" ISO/IEC 9594-8:1997.. ITU.
[xmlschema11-2-20120405]
W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes. David Peterson; Sandy Gao; Ashok Malhotra; Michael Sperberg-McQueen; Henry Thompson; Paul V. Biron et al. W3C. 5 April 2012. W3C Recommendation. URL: https://www.w3.org/TR/2012/REC-xmlschema11-2-20120405/

G.2 Informative references

[draft-ietf-ace-oauth-authz]
Authentication and Authorization for Constrained Environments (ACE) using the OAuth 2.0 Framework (ACE-OAuth). L. Seitz; G. Selander; E. Wahlstroem; S. Erdtman; H. Tschofenig. IETF. 27 March 2019. Internet-Draft. URL: https://tools.ietf.org/html/draft-ietf-ace-oauth-authz-24
[iana-uri-schemes]
Uniform Resource Identifier (URI) Schemes. IANA. URL: https://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml
[json-ld11]
JSON-LD 1.1. Gregg Kellogg; Pierre-Antoine Champin. W3C. 10 May 2019. W3C Working Draft. URL: https://www.w3.org/TR/json-ld11/
[json-ld11-api]
JSON-LD 1.1 Processing Algorithms and API. Gregg Kellogg. W3C. 10 May 2019. W3C Working Draft. URL: https://www.w3.org/TR/json-ld11-api/
[JSON-SCHEMA-VALIDATION]
JSON Schema Validation: A Vocabulary for Structural Validation of JSON. Austin Wright; Henry Andrews; Geraint Luff. IETF. Internet-Draft. URL: https://tools.ietf.org/html/draft-handrews-json-schema-validation-00
[LDML]
Unicode Technical Standard #35: Unicode Locale Data Markup Language (LDML). Mark Davis; CLDR Contributors. URL: https://unicode.org/reports/tr35/
[LINKED-DATA]
Linked Data Design Issues. Tim Berners-Lee. W3C. 27 July 2006. W3C-Internal Document. URL: https://www.w3.org/DesignIssues/LinkedData.html
[OPENAPI]
OpenAPI Specification: Version 3.0.1. Darrel Miller; Jason Harmon; Jeremy Whitlock; Kris Hahn; Marsh Gardiner; Mike Ralphson; Rob Dolin; Ron Ratovsky; Tony Tam. OpenAPI Initiative, Linux Foundation. 7 December 2017. URL: https://swagger.io/specification/
[owl2-manchester-syntax]
OWL 2 Web Ontology Language Manchester Syntax (Second Edition). Matthew Horridge; Peter Patel-Schneider. W3C. 11 December 2012. W3C Note. URL: https://www.w3.org/TR/owl2-manchester-syntax/
[owl2-overview]
OWL 2 Web Ontology Language Document Overview (Second Edition). W3C OWL Working Group. W3C. 11 December 2012. W3C Recommendation. URL: https://www.w3.org/TR/owl2-overview/
[rdf11-concepts]
RDF 1.1 Concepts and Abstract Syntax. Richard Cyganiak; David Wood; Markus Lanthaler. W3C. 25 February 2014. W3C Recommendation. URL: https://www.w3.org/TR/rdf11-concepts/
[rdf11-mt]
RDF 1.1 Semantics. Patrick Hayes; Peter Patel-Schneider. W3C. 25 February 2014. W3C Recommendation. URL: https://www.w3.org/TR/rdf11-mt/
[RFC3987]

Internationalized Resource Identifiers (IRIs). M. Duerst; M. Suignard. IETF. January 2005. Proposed Standard. URL: https://tools.ietf.org/html/rfc3987
[Rijgersberg-et-al-2013]
Ontology of Units of Measure and Related Concepts. Hajo Rijgersberg; Mark van Assem; Jan Top. Semantic Web journal, IOS Press. 2013. URL: http://www.semantic-web-journal.net/content/ontology-units-measure-and-related-concepts
[SemVer]
Semantic Versioning 2.0.0. Tom Preston-Werner. 26 December 2017. URL: https://semver.org/
[smartM2M]
ETSI TS 103 264 V2.1.1 (2017-03): SmartM2M; Smart Appliances; Reference Ontology and oneM2M Mapping. ETSI. March 2017. Published. URL: http://www.etsi.org/deliver/etsi_ts/103200_103299/103264/02.01.01_60/ts_103264v020101p.pdf
[string-meta]
Strings on the Web: Language and Direction Metadata. Addison Phillips; Richard Ishida. W3C. 16 April 2019. W3C Working Draft. URL: https://www.w3.org/TR/string-meta/
[vocab-ssn]
Semantic Sensor Network Ontology. Armin Haller; Krzysztof Janowicz; Simon Cox; Danh Le Phuoc; Kerry Taylor; Maxime Lefrançois. W3C. 19 October 2017. W3C Recommendation. URL: https://www.w3.org/TR/vocab-ssn/
[WOT-SECURITY-CONSIDERATIONS]
Web of Things (WoT) Security and Privacy Considerations. ; Michael McCool; Elena Reshetova. W3C. March 2019. URL: https://w3c.github.io/wot-security/

↑