

ホームネットワークのトポロジ検出と障害検知を実現する HTIP の実装とその活用

岡田崇^{†1} Sioutis Marios^{†1} 牧野義樹^{†1} 丹康雄^{†1}

概要：近年、ホームネットワークが複雑化しクラウドと連携した多様なサービスが提供されつつある。サービスの更なる展開にはネットワークの安定的動作が不可欠であるが、ホームネットワークに障害が発生した際の要因の発見はその複雑さゆえに特定することが困難である。本稿では、ホームネットワーク接続構成特定プロトコル HTIP のオープンソースソフトウェア 2 件の実装とそれらの特徴について述べる。またオープンソースソフトウェアを他社の実装と検証する相互接続試験環境について紹介し、長期間の運用から得た知見と課題について論ずる。

キーワード：ホームネットワーク、トポロジ検出、障害検知

An implementation and practical use of HTIP which implement a topology and fault detection of a home-network

TAKASHI OKADA^{†1} SIOUTIS MARIOS^{†1}
YOSHIKI MAKINO^{†1} YASUO TAN^{†1}

1. はじめに

近年、住宅には情報家電が普及し PC、タブレット端末、スマートフォンのみならず多種多様なデバイスがネットワークに接続し、各住宅に様々なホームネットワーク (以下、HN) が構成されている。また HN とそのサービスの発展により、住宅内のデバイスやセンサの情報に基づく快適なサービスが提供される基盤が整いつつある [2][3]。このような HN において、ユーザーがサービスを円滑に利用できるためには、デバイスやネットワークの障害検知と復旧の仕組みが必要である。本稿では、HN においてデバイスの情報を参照し、かつネットワークトポロジの検出を可能にするホームネットワークトポロジ特定プロトコル HTIP のオープンソース実装を 2 例紹介する。また本実装を他社の実装と共に長期間相互接続運用している実験住宅の実験環境と、運用中に発見した課題と知見について議論する。

2. ホームネットワークにおける障害

住宅の家電にネットワーク機能が付与された情報家電・デバイス・センサが広く普及しつつある。同時に HN は多種多様なデバイスとネットワークの複合体となり、その複雑性は増すばかりである。住宅には Ethernet、Wi-Fi、Bluetooth、PLC、ZigBee といったネットワークに AV 機器向けの DLNA、白物家電向けの ECHONET、遠隔制御向けの OSGi といったプロトコルやミドルウェアが混在している。このような多様な HN では、様々な新しいサービスが

提供されつつある。

HN サービスは管理プラットフォーム (PF) で管理されるアプリケーションとしてインターネットやクラウドで実行するケースも増えてきている。TTC TR-1046[3]では、HN サービスプラットフォームのアーキテクチャを規定している。

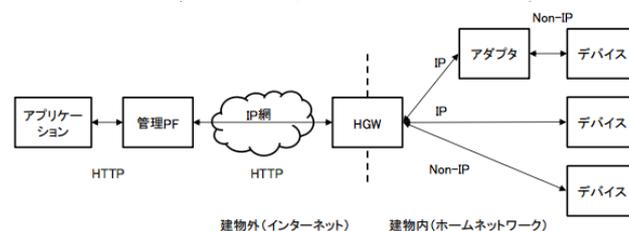


図 1 サービスプラットフォームのアーキテクチャ

HN のデバイスは、IP または非 IP のネットワークからホームゲートウェイ (以下、HGW) に接続し、インターネット・管理 PF・アプリケーションを利用する。管理 PF は、HN サービスとして実行されるアプリケーションを管理し、HN のデバイスの情報を蓄積しそれらのデータアクセスの API 機能を提供する。HN のデータは管理 PF に集約され、または管理 PF 間で連携され、アプリケーション開発者は HN の複雑性を隠蔽し開発を行うことができる。また HN サービス利用者は、管理 PF から多様なアプリケーションを利用できる。HN サービスがこのようなアーキテクチャへ変化するにつれて、HN の障害への対応も大きな課題とな

^{†1} 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology

る。TTC TR-1053[4]、TTC TR-1057[5]では、HN サービス向けプラットフォームの課題とその解決に求められるカスタマサポート機能として障害の要因・検出、情報の収集と通知について規定している。また TTC TR-1062[7]では、想定されるユースケースにおけるシーケンスを規定している。これらの技術レポートで規定された障害のユースケースに対し、障害を検出するプロトコルとして SNMP(Simple Network Management Protocol)がある。SNMP は、IP ネットワーク上のデバイスを管理するプロトコルであり、ネットワーク管理者がネットワークに接続するデバイスをモニタするための管理システムである。また ECHONET Lite は、エコネットコンソーシアムが策定した通信プロトコルであり、スマートハウス向けのデバイスが持つ機能をプロパティと呼ぶ属性やオペレーションとして定義している。ECHONET Lite のプロトコルは、このプロパティを参照しデバイスの状態をモニタできる。NETCONF は、ネットワーク機器の設定を RPC(Remote Procedure Call)を用いて行うプロトコルを提供する。

3. ホーム NW 接続構成特定プロトコル HTIP

3.1 HTIP とは

HN 接続構成特定プロトコル HTIP[8]は、HN において障害の切り分けを行うプロトコルである。ネットワークに接続したデバイスの機器情報を特定し、またネットワークポロジを特定することで、障害発生箇所切り分け、不具合対応が可能となることを期待できる。HTIP は、リンクレイヤブロードキャストドメインにおいて利用される。図に HTIP が利用される HN の構成を示す。

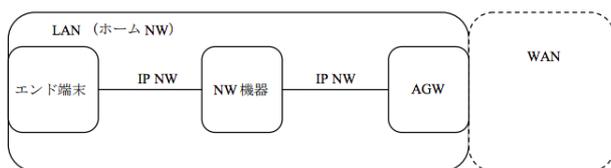


図 2 HTIP が利用される HN の構成

エンド端末は、IP アドレスを保持するデバイスである。NW 機器とは、ネットワークインタフェースを2つ以上持ち、MAC アドレステーブルを利用して、フレームやパケットをブリッジするデバイスである。Access Gateway (AGW) は、HN と WAN のインタフェース機能を持ち、エンド端末と NW 機器の両方の機能を持つ。

HTIP の機能を有するエンド端末(HTIP-エンド端末)は、以降に示す L3Agent または L2Agent が搭載されている。また HTIP 機能を有する NW 機器(HTIP-NW 機器)は、L2Agent が搭載されている。

HTIP を利用しない場合に HN から取得できる情報を図

3 に示す。AGW に対しいくつかのデバイスが存在していることだけが分かる。これに対し HTIP を利用し機器情報、接続構成情報を取得すると図 4 のような情報を把握することができる。このように HTIP を用いることで HN の機器の情報と接続構成情報から障害発生箇所の切り分けを行い、円滑に障害検知が実現できることが期待されている。

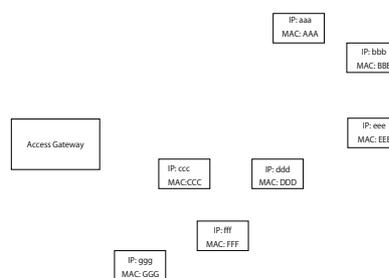


図 3 HTIP を利用しない場合の HN の情報

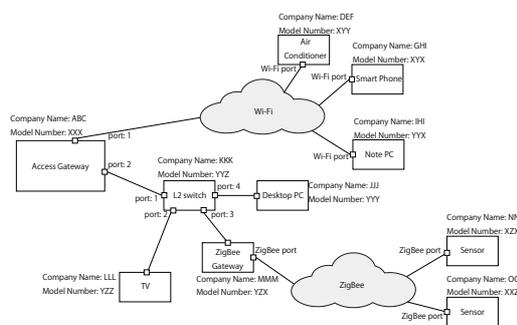


図 4 HTIP を利用した場合の HN の情報

3.2 L3Agent

L3Agent は、UPnP controlled device の機能を利用して機器情報を Manager へ通知する。機器情報とは区分、メーカーコード、機種名、型番、チャネル使用状態情報、電波強度情報、通信エラー率情報などで構成される。区分は TTC JJ-300.01 端末区分情報リスト[9]において、大区分として AV 系、PC 系、通信系、くらし系、娯楽系、プロトコル系、その他に分類され、例えば AV 系には更に 11 の小区分に分類され「テレビ」「セットトップボックス」「プロジェクタ」などに区分されている。以下に L3Agent の Device Description Document の例を示す。

```
<friendlyName>short user-friendly title</friendlyName>
<manufacturer>manufacturer name</manufacturer>
<manufacturerURL>URL to manufacturer site</manufacturerURL>
<modelDescription>long user-friendly title</modelDescription>
<modelName>(c)機種名</modelName>
<modelName>(d)型番</modelName>
<modelURL>URL to model site</modelURL>
<serialNumber>manufacturer's serial number</serialNumber>
```

図 5 機器情報の Device Description Document の例

3.3 L2Agent

エンド端末の L2Agent は、LLDP を利用して機器情報を Manager へ通知する。機器情報通知は、機器情報に ID を割り当て、TLV 形式で格納する。図に機器情報 ID を利用した機器情報 TLV のフレームを示す。

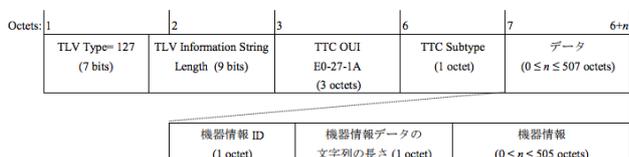


図 6 機器情報 ID を利用した機器情報 TLV

NW 機器の L2Agent は、LLDP を利用して機器情報と接続構成情報を Manager へ送信する。接続構成情報は、MAC アドレステーブル情報であり、ポートごとの MAC アドレスの値が格納される。

3.4 Manager

Manager は、HTIP-エンド端末、HTIP-NW 機器からの情報を取得し、HN の接続構成を特定する。Manager について、TTCJJ-300.00 の仕様では詳細を定めておらず実装依存としている。

Manager は、HN の HTIP-エンド端末、HTIP-NW 機器からの情報を蓄積し、これらの情報を基に接続構成を特定するが、美原らの手法[11]を用いることで HN 特有のネットワークトポロジを推定することができる。

4. HTIP の実装

HN のデバイスの機器情報及び接続構成を特定する HTIP のオープンソースソフトウェアを実装し、GitHub において公開した。本節では各オープンソースソフトウェアの実装とその普及活動について述べる。

4.1 HTIP のオープンソースソフトウェア

ホームネットワークトポロジ特定プロトコル HTIP のオープンソースソフトウェアを 2 件実装し、GitHub において公開した。これらのオープンソースソフトウェアは、HTIP

の仕様に準拠したソフトウェアであるが、HTIP の仕様は実装においていくつかの機能が実装依存としている。2 件のオープンソースソフトウェアはそれぞれ用途別に実装されている。以降詳細を述べる。

4.2 lwhtip

lwhtip[12]は、ユーザーが HTIP の機能を汎用 PC やマイコンボードで簡易に利用できることを目的に開発された。lwhtip は、汎用 PC やマイコンボード上の Linux オペレーティングシステムで実行されることを想定しており、C 言語で実装されている。C コンパイラ・autotools をインストールした最新の Linux オペレーティングシステム上で configure スクリプトと make コマンドによりコンパイル可能である。またライブラリで利用する関数群のドキュメント生成をサポートしている。ライセンスは MIT ライセンスとし、大学や企業が自由に利用、改変後再配布が可能である。

実行形式はエンド端末、NW 機器上で実行する L2Agent をサポートしている。

L2Agent (エンド端末)

lwhtip は L2Agent の機能を実装している。実行時に登録した機器情報を定期的に LLDP フレームとしてブロードキャストする。現在の実装では実行するデバイスのネットワークインタフェースの可能な全てから機器情報を送信する。

L2Agent (NW 機器)

ユーザーが簡易に HTIP の機能を利用できるため、lwhtip は汎用 PC やマイコンボードを L2 スイッチとして動作させることを想定している。複数の PC 本 L2Agent (NW 機器) は Linux の bridge 機能を有効にし、複数のネットワークインタフェースをブリッジする。例えば 2 つのネットワークインタフェース en0、en1 をブリッジインタフェース br0 でブリッジする場合以下のコマンドで実現できる。

```
# brctl addbr br0
# brctl addif br0 en0
# brctl addif br0 en1
```

L2Agent (NW 機器) は、ブリッジインタフェースの MAC アドレステーブル情報を参照し定期的にその情報を送信する。

lwhtip で構成するネットワーク

lwhtip の L2Agent、L2Agent (NW 機器) を接続することで簡易なネットワークトポロジを構成することができる。

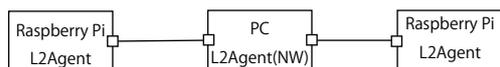


図 7 lwhtip で構成する簡易なネットワーク

図では 2 台の Raspberrry Pi 上で L2Agent を実行し PC に接続し、PC 上で L2Agent (NW 機器) を実行している。PC 上で tcpdump などのパケットキャプチャツールを利用してその挙動を確認することが可能である。

4.3 node_htip

node_htip[13]は、ユーザーが HTIP の情報を収集しそのネットワークトポロジを確認する目的で開発された。node_htip は、node.js で実行する JavaScript による HTIP 実装である。node.js 上で動作するため、汎用 PC 上での利用を想定しており node.js が動作するあらゆるオペレーティングシステム上で動作可能である。インストールは、GitHub からレポジトリをクローンした後 npm により簡易にインストール可能である。ライセンスは MIT ライセンスとし、大学や企業が自由に利用、改変後再配布が可能である。

node_htip は HTIP の L2 フレーム受信機能を備えた Manager の実行形式をサポートしている。

Manager

HTIP の仕様では、Manager の明確な機能は定義していないが、node_htip は以下の機能を有している。

- a)L2Agent から送信された機器情報の受信
- b)L3Agent から送信された機器情報の受信
- c)L2Agent (NW 機器) から送信された接続構成情報の受信
- d)L2Agent・L3Agent からの機器情報及び L2Agent (NW 機器) からの接続構成情報の保持と提供
- e)L2Agent・L3Agent からの機器情報及び L2Agent (NW 機器) からの接続構成情報の可視化

node_htip は e)の機能をデフォルトの実行形式としている。Manager は web サーバーを起動し、アクセス時に L2Agent、L2Agent から収集した情報を基に推定したネットワークトポロジを表示する。またネットワークトポロジ上のデバイスの機器情報を表示する。

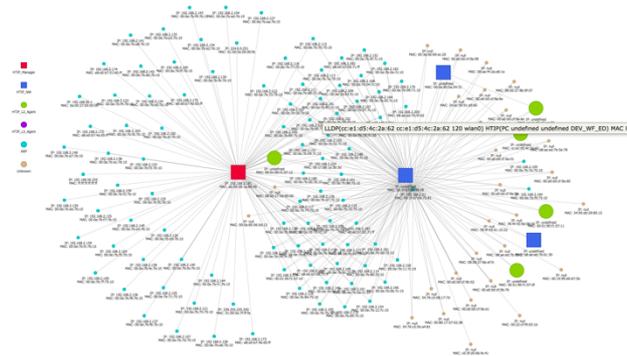


図 8 node_htip による接続構成情報の可視化

node_htip はライブラリにより構成されているため a)~d)の機能のみを利用することが可能である。つまり機器情報・接続構成情報を保持し提供する機能のみを実行し、異なるデバイスを可視化 web サーバーとすることも可能である。機器情報・接続構成情報の提供は REST 形式のアクセスに対し JSON フォーマットで提供される。

```
[{"mac": [{"mac": "08:00:27:00:00:00", "chassis_id": "08:00:27:00:00:00", "port_id": "08:00:27:00:00:00", "port_desc": "eth0", "device_mac": "08:00:27:00:00:00", "model_number": "08:00:27:00:00:00", "fdm": "08:00:27:00:00:00", "expire": "2017-07-26T02:33:26.479Z", "htip_type": "Unknown"}, {"mac": "08:00:27:00:00:00", "chassis_id": "08:00:27:00:00:00", "port_id": "08:00:27:00:00:00", "port_desc": "eth0", "device_mac": "08:00:27:00:00:00", "model_number": "08:00:27:00:00:00", "fdm": "08:00:27:00:00:00", "expire": "2017-07-26T02:33:26.479Z", "htip_type": "Unknown"}]
```

図 9 node_htip の接続構成情報(JSON 形式)の抜粋

4.4 オープンソースソフトウェアの適用範囲

それぞれ異なる目的のオープンソースソフトウェアを公開してきたが、いくつかの実装について利用の容易さ・開発の容易さ・実行負荷・メンテナンスの観点から検討する。

C 言語は多くのシステムのベースとなる言語であり、マイコンボードとの親和性も高い。特定のマイコンボード上ではそのままでは利用できないが、各ボード向け実装のリファレンスコードとしての役割を期待している。また汎用 PC 上で複数の仮想機械を実行し、仮想ネットワークトポロジを構築するような試用例に対し、軽量であり利用し易い。

JavaScript はユーザーの多いプログラミング言語であり、多くのライブラリが活発に開発・公開されているため、利用可能なユーザーも多い言語である。オープンソースソフトウェアとしてもまたオペレーティングシステムを問わない点も利点である。開発が活発に行われる反面 node.js、JavaScript、ブラウザの変化に伴い頻回なメンテナンスが必要である。

その他に、Java や Python・Ruby といったスクリプティング言語での実装が考えられる。Java はユーザー数も多くオペレーティングシステムを選ばないが、VM 上で実行する必要がある。スクリプティング言語は簡易に実装が可能であるが JavaScript、Java と共にマイコンボードへの移植は困難と言える。

実装	利用の 容易さ	開発の 容易さ	実行負荷	メンテ ナンス
lwhtip	×	×	○	○
node_htip	○	○	×	×
Java	○	×	×	○
Python,Ruby	○	○	×	×

4.5 オープンソースソフトウェアの普及活動

オープンソースソフトウェアは、2017年9月より GitHub で公開されると共に、一般社団法人 情報通信技術委員会（以下、TTC）の IoT ワーキングパーティ「IoT エリアネットワーク管理技術」において、その開発の方向性や実用化について議論を行ってきた。本ワーキングパーティは、大学や企業がデバイスの運用管理を実現するため、課題を明らかにし、相互に解決して HTIP を中心としたエコシステムの普及を図ることを目的とし、活動している。オープンソースソフトウェアを公開し、HTIP の運用管理・開発ガイドラインの作成・プラグフェストの開催など今後の HTIP 普及に向けた活動を行なっている。

5. HTIP 実験環境とその運用

本節では、オープンソースソフトウェアを含む複数の HTIP 実装の相互接続性を検証する HTIP 相互接続試験環境とその運用について、また運用により明らかになった課題について述べる。

5.1 HTIP 相互接続試験環境

石川県能美市の実験住宅内に富士通株式会社、SMK 株式会社、北陸先端学技術大学院大学の 3 社の HTIP 実装を持ち寄り、相互に接続性を試験する環境を構築した。

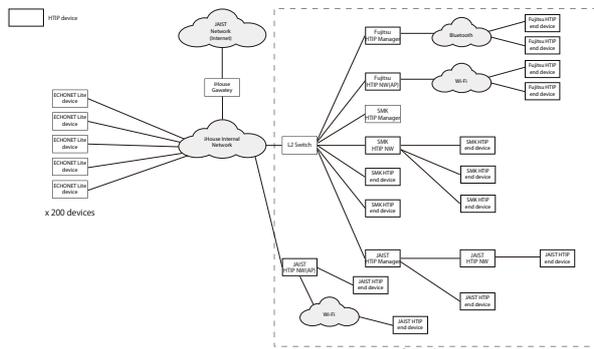


図 10 HTIP 相互接続試験環境概要

実験住宅内には 200 以上の情報家電、デバイスやセンサが設置されており、ECHO NET Lite を用いてその状態の参照及びデバイスの制御を行なっている。図の破線で囲まれた箇所が HTIP 相互接続試験環境であり 3 社の HTIP エ

ンド端末、NW 機器が接続されネットワークが構成されている。本試験環境は 2017 年 5 月より運用し、長期的に実行を続け障害の観測を行っている。

5.2 今後の課題

HTIP 相互接続試験

複数の HTIP 実装のデバイスが同一のブロードキャストドメインネットワークに存在する時、相互に接続可能か試験する必要がある。HTIP の仕様及び技術レポートでは未だこの試験手順については規定していない。本接続試験環境では、3 社の HTIP 実装の相互接続性の確認を行ってきたが、その運用から相互接続試験実施項目の策定とソフトウェアが必要であることが分かった。

現在の具体的な相互接続性の試験手順は、まず各社の HTIP Manager から機器情報、接続構成情報を取得し、これらを比較する。これらに相違があった場合 tcpdump などのパケットキャプチャツールを用いて相違の原因を確認する。しかし、この試験手順では実装が HTIP 仕様を網羅的に満たしているかの検証が行われておらず、また手動での試験となっている。今後 HATS[1]の試験実施要領で規定される試験及びコンFORMANCEステストツールなどが必要である。

トポロジの発見

L2 のネットワークトポロジ検出アルゴリズムが存在するが、HN はこれらのアルゴリズムが前提とする完全な MAC アドレステーブルの情報が収集されないケースがほとんどである。HTIP 相互接続試験環境においても破線で囲まれた HTIP L2Agent、L2Agent (NW 機器) で構成されたネットワークトポロジはその推定が可能であるが、破線外のネットワークは HTIP に対応しておらず、その推定ができていない。今後 HTIP のみではなく複数のプロトコル情報を基に完全では無くベストエフォート型のネットワークトポロジ検出アルゴリズムを検討することで、より住宅内の障害解決に役立つと考えられる。

ループの問題

上記のネットワークトポロジ検出における典型的な問題の一つは、マルチホーミングするデバイスによるループ問題である。

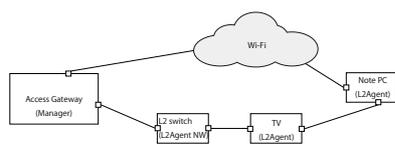


図 11 マルチホーミングデバイスによるループの問題

図では全てのデバイスに Manager, L2Agent, L2Agent (NW 機器) のいずれかの機能が備わっているが TV と Note PC が複数のネットワークインタフェースがあるため、Manager からは TV と Note PC の接続性を確認できない。また TV が HDMI Ethernet によりブリッジングを行う時 Manager へは 2 つのポートから Note PC の HTIP フレームを受信することになる。これらの問題はルールを規定することにより解決できる可能性があるがデバイス、ネットワーク、プロトコルの仕様ごとに検討をする必要がある。

このような問題は複数のオープンソースソフトウェアの実装、HTIP 相互接続試験環境の構築と長期間の運用、ワーキングパーティの議論により発見し議論してきた問題である。今後 HTIP の高度化と普及のため、オープンソースソフトウェアの継続的メンテナンスと普及、相互接続試験実施項目、ネットワークポロジ検出アルゴリズムの検討が課題となる。

また HTIP は今後 IoT デバイスへの適用を検討しており、HN とは異なる特徴を踏まえた運用規定や技術レポートを議論するため、オープンソースソフトウェアや相互接続試験環境の継続的開発を行っていく。

6. おわりに

HN の現状と障害への対応の課題を明らかにし、これまで提案された技術レポートやプロトコルを整理した。HN 接続構成特定プロトコル HTIP の概要を示し、そのオープンソースソフトウェア 2 件の実装について述べた。オープンソースソフトウェアは、Linux 向けの C 言語による L2Agent 実装、node.js 上での JavaScript による Manager 実装を公開し、それぞれの特徴を整理した。またオープンソースソフトウェアと他社との実装を検証する相互接続試験環境について述べ、その運用により明らかになった課題と今後の対応について議論した。

謝辞 本研究は、「総務省研究開発課題 IoT 共通基盤技

術の確立・実証 II. 効率的かつ安定的な IoT デバイス接続・エリアネットワーク運用管理技術の確立」の助成を受け実施された。

参考文献

- [1] “HATS 推進会議”. <https://www.hats.gr.jp/>, (参照 2018-01-25).
- [2] 丹康雄. ホームネットワークの現状と標準化動向. 電子情報通信学会 通信ソサイエティマガジン. 6 巻(2012-2013)2 号.
- [3] “TR-1046 ホームネットワークサービスを実現するサービスプラットフォーム”.
http://www.ttc.or.jp/jp/document_list/pdf/j/TR/TR-1046v1.pdf (参照 2018-01-25).
- [4] “TR-1053 サービスプラットフォームにおけるカスタマサポート機能”. http://www.ttc.or.jp/jp/document_list/pdf/j/TR/TR-1053v1.1.pdf, (参照 2018-01-25).
- [5] “TR-1057 ホームネットワークにおけるカスタマサポート機能ガイドライン”.
http://www.ttc.or.jp/jp/document_list/pdf/j/TR/TR-1057v1.1.pdf, (参照 2018-01-25).
- [6] “TR-1061 JJ-300.00 機能実装ガイドライン～非イーサネットデータリンク層、複数 LLDPDU、障害切り分け情報対応～”.
http://www.ttc.or.jp/jp/document_list/pdf/j/TR/TR-1061v1.pdf, (参照 2018-01-25).
- [7] “TR-1062 ホームネットワークサービスにおけるカスタマサポートユースケース”.
http://www.ttc.or.jp/jp/document_list/pdf/j/TR/TR-1062v1.1.pdf, (参照 2018-01-25).
- [8] “JJ-300.00 ホーム NW 接続構成特定プロトコル”.
http://www.ttc.or.jp/jp/document_list/pdf/j/STD/JJ-300.00v3.pdf, (参照 2018-01-25).
- [9] “JJ-300.01 端末区分情報リスト”.
http://www.ttc.or.jp/jp/document_list/pdf/j/STD/JJ-300.01v2.pdf, (参照 2018-01-25).
- [10] “東芝 HEMS”. <http://feminity.toshiba.co.jp/feminity/>, (参照 2018-01-25).
- [11] Yoshiyuki Mihara, Shuichi Miyazaki, Yasuo Okabe, Tetsuya Yamaguchi and Manabu Okamoto: “Identifying Link Layer Home Network Topologies Using HTIP.” Proc. of IEEE Consumer Communications and Networking Conference, pp.892-899, 2017.
- [12] “lwhtip”, <https://github.com/Tan-Lab/lwhtip>, (参照 2018-01-25).
- [13] “node_htip”, https://github.com/Tan-Lab/node_htip, (参照 2018-01-25).