

通信制御プロトコル自律適合実現に向けた形式仕様記述言語の適用

柿島 純[†] 川上 博[†] 滝田 亘[†]

[†]NTT ドコモ ネットワーク研究所

(株)NTT ドコモ 〒239-8536 神奈川県横須賀市光の丘 3-5

E-mail: [†]kakishimaj@nttdocomo.co.jp

あらまし 現在、共通な IP ネットワーク上に多種多様な通信制御プロトコルが混在している。しかも、これらのプロトコルはネットワーク上で独立に稼働している。しかし、今後これらの独立なプロトコル同士の相互接続が頻繁に起こることが考えられる。さらに、ユーザ定義のプロトコルとも接続を行う必要がでてくるであろう。そこで、プロトコル同士の共有をどの範囲まで行えば通信が成立するかを見極めることを目的に、SIP(Session Initiation Protocol)を例に形式仕様記述言語を利用したアプローチの提案を行う。

Application of formal description techniques for auto adjustment of communication control protocols

Jun KAKISHIMA[†] Hiroshi Kawakami[†] Wataru Takita[†]

[†]Network Laboratories, NTT DoCoMo, Inc.

3-5 Hikarinooka, Yokosuka, Kanagawa, 239-8536 Japan

[†]E-mail kakishimaj@nttdocomo.co.jp

Abstract Heterogeneous NW control protocols are accommodated in the common IP BB. These protocols are operated independently. We need to interconnect these protocols frequently, and will interconnect user definition protocols. As an example of SIP (Session Initiation Protocol), we propose the approach for the scope specific of shared contents which can communicate as to each protocol by using formal description techniques.

1. はじめに

現在、共通な IP ネットワーク上に多種多様なサービスが提供されている。しかも、それらは ISP、事業者等によって独立に提供されている。しかし、今後互いに連携をする機会が多くなることが予想される。そこで、それらの相互接続を提供するサービスが想定できる。例えば、各事業者が独立にインスタントメッセージを提供している場合、キャリアのインフラ設備を利用して、それらの相互接続をサポートするというサービス提供が考えられる。それらを収容する際、標準化されていないユーザ定義の通信プ

ロトコルを収容する可能性もでてくる。このような環境下で、それらを連携する際どこまで共有する情報を持たせるべきかという観点から、プロトコル同士の自律的な適合を可能にする技術の創出を図る。本稿では、2章で研究の目的とアプローチを示す。さらに、呼制御プロトコルである SIP(Session Initiation Protocol)を例にとり、アプローチにそって検証を行う。3章で SIP 概要を説明する。4章で SIP 付加サービスを実装した端末同士を接続するための前提条件と課題を特定する。5章で、解決手段として形式仕様記述言語を用いる。6章で考察

及び今後の課題、7章でまとめを示す。

2. 本研究の目的とアプローチ

現在、IP 通信の普及と共に多様な通信制御プロトコルが IP ネットワーク上に混在している。さらに、プライベートな IP ネットワークが構築されている今、我々としてこのようなネットワークや、そこで使用されている通信制御プロトコルを収容する機会が増えてくるのが、想定される。その際、収容する各プロトコルに対してコンバータを作成する方法もある。しかし、アプリケーション技術で自律的にプロトコル間の接続調整を行える方法があれば、コンバータを逐次構築する必要もなく、さらに新しいプロトコルが出現する際も自律的に接続を行える可能性が出てくる。そこで、本研究の目的を以下のように設定する。

目的：多種多様なプロトコルに対して、アプリケーション層により自律的に適合できる技術を創出する

上記目的に対して、以下の研究アプローチを行う。

手順 1：既存の通信制御プロトコルに対して、どの範囲まで互いに情報を共有すれば、接続が可能になるか前提条件を設定し特定する

手順 2：上記前提条件を設定することにより発生する課題を特定する

手順 3：その課題に対して、自律的に適合できる技術を検討する

本稿では、SIP を例にアプローチ手順にそって検討を行う。図 1 にコンセプト概要を示す。

3. SIP 概要

SIP とは、インターネットに代表される IP ネットワーク上での、音声や映像等のストリーム型の通信をセッションと位置付け、その設定や解放、ならびに各種制御機能を提供するプロトコルである。本稿では、RFC 準拠の SIP[1]を対象とする。RFC 準拠によると、SIP はリクエスト/レスポンス方のインタラクティブ通信で、メッセージフォーマットは、Request-Line/Message Headers/Message Body から構成されている(図 2 参照)。Request-Line には、Method 名 (INVITE 等) を挿入する。Message Headers[1]

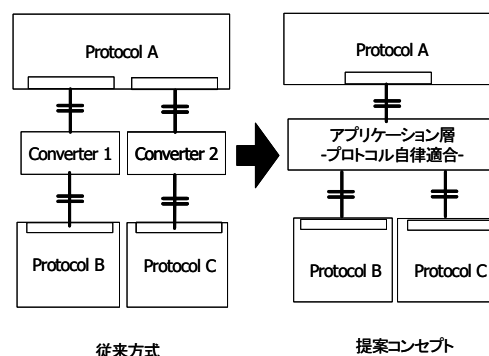


図 1 コンセプト概要

は図 3 に示すパラメータを記述する。例えば、Accept ヘッダ中に 2XX を記入すると、受信した Method が対応していることを表している。Message Body には付加情報を記述し、SDP(Session Description Protocol) [2]を使用することが多い。

4. 前提条件と課題

接続対象とする端末には、基本サービスである音声は Method/Message Headers/Message Body に関してすべて統一されているものとする。付加サービスに関して、まずどちらかの端末に実装され、片方には実装されていないケースが想定される。この場合は、付加サービスを削除し、基本サービスのみを接続する手段が考えられる。そのため、削除することによる基本サービスの影響があるか否かを検討する必要がある。まず Method に関して考えてみる。Method に関して、INVITE、ACK、BYE、OPTIONS、CANCEL、REGISTER、INFO、SUBSCRIBE、PUBLISH、NOTIFY、UPDATE、MESSAGE、REFER が標準化されている。付加サービスに関しては、主に SUBSCRIBE、PUBLISH、NOTIFY、UPDATE、MESSAGE が使用されるケースが多いが、受信側にこれらを受信できない場合は、非対応を表示する Header Field 中の accept 中に、非 2XX を返信することで送信側は送信した Method を終了し、基本サービスの Method に移行する方法がある。従って、基本サービスに影響はないと言える。次に、両方の端末に対して共通な付加サービスを実装していることを考える。Method/Message Headers/Message Body がすべて一致していれば、

Request-Line
Message Headers
空白行
Message Body

図 2 : SIP メッセージのフォーマット

問題がない。次に、Method が共用されていない場合は前述したように、Header Field 中の非 2xx メソッドを使用することで、付加サービスは処理されないため、Method の一致は必須である。次に、Headers に関して一致されていない場合を考えてみる。Header Field 中の Allow 部分に対して、それぞれの端末が対応しているパラメータを記述し、Info Method を使用して Headers を送信することで、それぞれのパラメータの一致か不一致が確認できる。不一致の場合は、Update Method を使用して変更が可能になるため、不一致があるパラメータを一致させることは可能である。しかし、ここでパラメータの定義をどこまで共有させるかに課題がある。RFC では、IANA[3] (The Internet Assigned Number Authority) で、Header Field の共有化を図るよう勧告されているが、強制ではないため独自定義を行うことも想定される。例えば、プレゼンスサービス[4]における状態定義は、用途によって定義が異なるため、すべてにおいて共通化を図ることは困難である。そこで、以下の課題がでてくる。

課題: すべての定義において共有化が図れない Head Field 中のパラメータに対して、どこまで共通化を図れば通信が成立するのか

次に法人同士で Method のやり取りを行う際に、Security の問題が発生する。例えば、Dos アタックのような攻撃を他の事業者より受けた際

Header Field	where	proxy	MESSAGE
Accept		R	-
Accept		2xx	-
Accept		415	m
Accept-Encoding		R	-
Accept-Encoding		2xx	-
Accept-Encoding		415	m*
Accept-Language		R	-
Accept-Language		2xx	-
Accept-Language		415	m*
Alert-Info		R	-
Alert-Info		180	-
Allow		R	o
Allow		2xx	o
Allow		r	o
Allow		405	m

図 3 Message Header の抜粋

の対処法も必要になるであろう。以上の課題を認識しつつ、本稿では対象外とする。

5. 形式仕様記述言語

SIP においては、Head Field のパラメータのメタ定義をどのように構築するかという課題に置き換えられる。プレゼンスのメタ定義の従来技術として XML (Extensible Markup Language) を使用して、木構造を構築することでメタ情報の属性化を図る手段がある。RFC3265[5]では図 4 のように規定を行っている。しかし、木構造を構築する際も定義の共有を図らなければならない。そこで、すべての定義を共有しなくても、「ある定義された共有情報を使って、未共有の定義を扱うことはできないか」という観点の元、これを満たす手段について検討した。その結果、LOTOS 等の形式仕様記述言語[6]を候補とした。形式仕様記述言語とは、ソフトウェアの仕様レベルでさまざまな解析や検証を、機械的に行う必要があるために、その仕様を一定の形式的な規則に従って、厳密に記述するための形式記述技法である。従来は、仕様を合成した際の競合を特定するなど[7][8]、仕様の矛盾点を特定するために使用されていた。本稿では前述した課題に対して、定義同士の命題を記述するために形式仕様記述言語を使用する。命題は論理代数式で表すこととする。ここで、SIP のプレゼンスを例に考えてみる。営業などでプレゼンスを使用する場合、外出中という状態が必要になる。外出中の目的として、取引先訪問

がある。取引先まで、電車又は車で行く方法がある。プレゼンスとして<外出>、<電車内>、<車内>、<訪問中>を定義する必要がある。形式仕様記述言語を使って、以下のような命題を作成する。

外出1 ∧ 電車内 → 訪問中1 (1)

外出2 ∧ 車内 → 訪問中2 (2)

上記(1)の命題は「外出中であつ電車内ならば、取引先へ訪問中である」という意味である。上記(2)の命題は「外出中であつ車内ならば、取引先へ訪問中である」という意味である。上記命題に対して帰納法を使うことで、<電車内>、<車内>の定義を事前に共有しなくても可能になることを示す。ここで、帰納法に関して簡単な例を使って説明する。

ケース: これらの豆はこの袋から出したものである

結果: これらの豆は白い。

ケース、結果より以下のルールが導き出せる。

ルール: この袋から出した豆はすべて白い

このように帰納法とは、ケースと結果より普遍的なルールを推論する方法のことである。

先ほどの命題(1)、(2)に関して外出、訪問中の定義は事前に共有しているとする。

外出1 = 外出2、訪問1 = 訪問2 (3)

ここで帰納法を適用すると(1)、(2)はケース、(3)は結果と見なせるので、電車内=車内という普遍的なルールが導かれる。電車内と車内は異なる状態ともみられるが、外出中、訪問中と組み合わせることで同じことを定義していると解釈ができる。このことから、電車内と車内は事前に定義を共有しなくてもいいことを表している。ここで、電車内と車内が同じ定義というルールから、それぞれの状態を表すことができないように思えるが、命題(1)、(2)がケースとして併用された場合に同じ定義という意味であり、各命題が独立に存在する場合は、定義が可能である。このように、形式仕様記述言語と帰納法を使うことで、記述内容の定義すべてを事前に共有しなくても済むことがわかる。

6. 考察と課題

「Head Field 中のパラメータに対して、どこまで共通化を図れば互いに通信が成立するの

```

Event = ( "Event"/"o" ) HCOLON event-type
      *( SEMI event-param )
event-type = event-package *( "." event-template )
event-package = token-nodot
event-template = token-nodot
token-nodot = 1*( alphanum / "-" / "!" / "%" / "*"
                / "_" / "+" / "/" / "" / "~" )
event-param = generic-param / ( "id" EQUAL token )

Allow-Events = ( "Allow-Events" / "u" ) HCOLON
                event-type *( COMMA event-type )
Subscription-State = "Subscription-State" HCOLON
                    substate-value *( SEMI subexp-params )
substate-value = "active" / "pending" / "terminated" /
                extension-substate
extension-substate = token
subexp-params = ("reason" EQUAL event-reason-value)
                / ("expires" EQUAL delta-seconds)
                / ("retry-after" EQUAL delta-seconds)
                / generic-param
event-reason-value = "deactivated"
                  / "probation"
                  / "rejected"
                  / "timeout"
                  / "giveup"
                  / "noresource"
                  / event-reason-extension
event-reason-extension = token

```

図4 プレゼンスのメタ定義 (RFC3265 準拠)

か」という課題に対して、プレゼンスを例にとり形式仕様記述言語と帰納法により、すべての定義を事前に共有しなくてもいいことを示した。本例の場合は命題「AならばB」という、AとBに共有する定義が共に含まれている場合に成り立つことがわかる。しかし、プレゼンス状態定義がすべて上記命題上で成り立つわけではない。従って、帰納法が適用できるよう形式仕様記述言語で命題を記述する必要がある。つまり、プレゼンスの状態種別によって、命題の記述方式が異なることになる。すべてのプレゼンス定義を共有する方法と比較して、運用上効率を図れるかは今後の検討課題である。又、本稿ではSIPのプレゼンスに対して、形式仕様記述言語と帰納法を適用したが、最終目的は未標準化の通信プロトコルを対象に、どの範囲まで共有すれば通信が可能になるかということである。従って、他の通信プロトコルを対象に形式仕様記述言語と帰納法を適用することで、ある程度の共有情報の傾向を特定することが必要になる。

7. まとめ

本稿では SIP のプレゼンスを例に取り、すべての定義を通信者同士で共有しなくても、通信が成り立つ手段として形式仕様記述言語と帰納法を使うことで実現することを示した。ただし、限定された範囲でのみ適用される方法なので、SIP 以外のプロトコルに関しても 2 章で示したアプローチにそって、適用範囲の特定を今後行っていく。

参考文献

- [1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [2] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002.
- [3] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", CP 26, RFC 2434, October 1998
- [4] Rosenberg, J., "A Presence Event Package for the Session Initiation Protocol (SIP)", RFC 3856, August 2004.
- [5] Roach, A. B., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.
- [6] Bolognei, T., Brinksma, Ed., Introduction to the ISO Specification Language LOTOS, in the formal Description Technique LOTOS, Elsevier Sci. Pub., pp.23-73, 1989
- [7] N. Shiratori, H. Kaminaga, K. Takahashi and S. Noguchi, A Verification Method for LOTOS Specifications and its Application, Protocol Specification, Testing and Verification IX, pp.59-70, North-Holl, 1990.
- [8] 臼井伸幸 高橋 薫 神長裕明、白鳥 則朗 : 形式仕様の開発における機能要求への反映、信学技報 IN-95-36(1995)