社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－ＤＳＭ－41（10）
2006／5／12

Webサービスによる CUG 管理モデルと管理インタフェース

白石 展久， 塩尻 浩久

日本電気株式会社 知的資産 R&D ユニット ソリューション開発研究本部 ユビキタス基盤開発本部

〒108-8557 東京都港区芝浦 2-11-5

E-mail: {n-shiraishi@bq, h-shiojiri@ce}.jp.nec.com

あらまし 我々は、ユビキタス環境において、トランスポート層での閉域性を確保し、かつロケーションに依存しない透過的なリモートアクセス環境を提供する、オンデマンド仮想ネットワーク CUG システムの研究開発を行っている。本 CUG システムは、仮想ネットワークを動的に制御することが可能なため、適切な CUG 管理モデルと CUG 管理インタフェースを定義することにより、NMS のみならず、業務アプリケーション等からのトランスポート層レベルの通信制御を行うことが可能となる。本稿では、我々が開発している CUG システムを管理する、CUG、Member、Node 等を管理対象として抽象モデル化した CUG 管理モデル、およびその Web サービスによる管理インタフェースの設計について述べる。

キーワード 仮想ネットワーク，CUG，管理モデル，管理インタフェース，Web サービス

CUG Management Model and Management Interface Using Web Service

Nobuhisa SHIRAISHI, Hirohisa SHIOJIRI

Ubiquitous Platform Development Division, Solution Development Laboratories,

Intellectual Asset R&D Unit, NEC Corporation

11-5, Shibaura 2, Minato-Ku, Tokyo 108-8557, JAPAN

E-mail: {n-shiraishi@bq, h-shiojiri@ce}.jp.nec.com

Abstract The On-demand Virtual Network CUG System we developed provides transport-layer level security and location-independent transparent remote access for ubiquitous environment. By designing and providing the CUG Management Model and the CUG Management Interface of the Network CUG System, not only NMS but also other systems such as enterprise business systems can control the network flow in transport-layer level by the faculty of the Network CUG System. In this paper, we propose the CUG Management Model in which CUG, Member and Node are defined as the managed object, and the CUG Management Interface based on Web Service with those user can operate the Network CUG System.

Keyword Virtual Network , CUG , Management Model , Management Interface , Web Service

1. はじめに

1.1. オンデマンド仮想ネットワーク CUG システム

我々が開発しているオンデマンド仮想ネットワーク CUG システム(以下 CUG システム)は、既存の物理ネットワークの上に、オンデマンドでトランスポート層レベルの CUG である仮想ネットワークを構築し、この仮想ネットワーク上で、全てのアプリケーションの通信を行うシステムである。本 CUG システムのコンセプトを図 1 に示す。

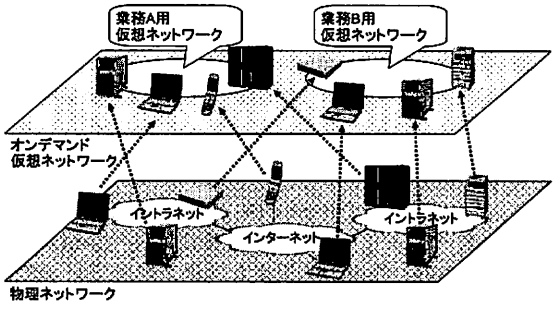

図 1: CUG システムのコンセプト

1.2. 本 CUG システムの特長

本 CUG システムには、以下の特長がある。

(1) ロケーションフリーな透過リモートアクセス環境

全ての通信を仮想ネットワーク上で行うため、実際のネットワーク環境に依存しない、ロケーションフリーな透過リモートアクセス環境を提供する。

(2) 仮想ネットワークに接続する全ての機器の認証と管理の実現

端末の接続対象となるネットワークが、システム上に仮想的に構築されたネットワークであるので、当該ネットワークに接続する全ての端末に認証を課すことが可能であり、当該ネットワークに接続されている全ての端末を把握・管理することが可能である。

(3) 短いライフタイムの仮想ネットワークの提供

接続対象となる仮想ネットワークを、必要なときにオンデマンドで動的に生成し、不要になった時に動的に削除することができるため、非常に短いライフタイムの閉域ネットワークのニーズに対応することができる。また、既存の VPN システムに比べ、より利用者からの要求に対してリアルタイムなアクセス制御を行うことも可能である。

2. CUG 管理モデルと CUG 管理インタフェースの必要性

本 CUG システムは、トランスポート層レベルの CUG である仮想ネットワークを構築し、管理する機能を持つ。そのため、本 CUG システムを外部から操作/管理する API およびその管理モデルを定義し、提供することによって、NMS からの CUG の操作や管理はもちろん、業務アプリケーション等の他のアプリケーションからも、CUG の操作を通じて、トランスポート層レベルの通信制御を行うことが可能となり、その業務アプリケーションのネットワーク制御プラグイン的な利用も可能となる。

本 CUG システムが提供する CUG は、論理的な仮想ネットワークであり、実ネットワークに比べ、低いオーバヘッドでネットワークの操作や制御を実現することができる。したがって、例えばあるジョブが生成される度に、そのジョブに対して新しい CUG による仮想ネットワークを生成して割り当て、ジョブ終了時には、その CUG を削除することにより、ジョブ毎に仮想ネットワークを動的に割り当てるなどの、非常にライフタイムのスケーラビリティが高いネットワークサービスを提供することができる。

ネットワーク管理モデルやプロトコルとしては、TMN[1][2]や SNMP[3]、MSDMS:WUWS[4] など、既に様々な仕様が存在する。しかしながら、仮想ネットワークを CUG として、より単純かつ直感的に、不足なく表現した管理モデルは存在しない。

そこで当グループでは、CUG による仮想ネットワークを抽象モデル化し、CUG を構成する ComPlace、Member、Node 等の構成要素を管理対象オブジェクトとして定義すると共に、その管理インタフェースを Web サービスにとして定義し、更にその上位 API として、より直感的に CUG を管理/操作できる Java クラス API を提供し、他の業務アプリケーションからの CUG 操作を可能とすることにより、本 CUG システムを、他の業務アプリケーションにおけるトランスポート層レベルでのネットワーク通信制御プラグインとして実現した。

3. CUG 管理モデル

3.1. CUG 管理モデル

本 CUG システムを管理/操作するための CUG 管理モデルを、図 2 に示す。

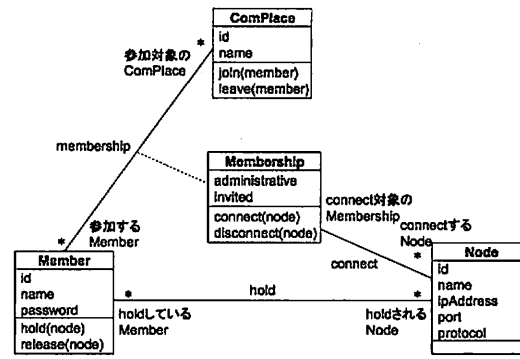

図 2: CUG 管理モデル

3.2. CUG 管理モデルを構成するクラス

本 CUG 管理モデルは、以下のクラスで構成される。

3.2.1. ComPlace クラス

本 CUG システムが提供する CUG を表現するクラス。参加している Member との間に、Membership クラスで表現される membership 関連を持つ。属性として、id と name を持ち、メソッドとして、CUG へ Member を参加させる join メソッド、CUG から Member を退出させる leave メソッドを持つ。

3.2.2. Member クラス

ComPlace へ参加し、本 CUG システムが提供する CUG を利用する、利用者を表現するクラス。具体的には CUG を利用する仮想的な利用者や、CUG の機能を利用して実現されるプリントサービスやストレージサービスなどのサービスなどを表す。参加対象の ComPlace との間に Membership クラスで表現される membership 関連を持ち、保持している Node との間に hold 関連を持つ。属性として、id と name と password を持ち、メソッドとして、Node を保持する hold メソッド、Node を解放する release メソッドを持つ。

3.2.3. Node クラス

Member が ComPlace へ接続するためのネットワーク上の接続点を表すクラス。この Node を保持している

Member との間に hold 関連を持ち、この Node を用いて接続している Membership との間に connect 関連を持つ。属性として、id と name 以外に、ipAddress、port、protocol 等の、ネットワーク接続点としての属性を持つ。本クラスを操作するメソッドは無い。

3.2.4. Membership クラス

「Member が ComPlace に参加している」という、Member と ComPlace との membership 関連を表現するクラス。Membership は、この Membership を通じて ComPlace に接続する Node との間に、connect 関連を持つ。属性として、その ComPlace への他の Member を参加、退出させることができる管理者権限があるかどうかを表す administrative 属性、本クラスで示される Member が本クラスで示される ComPlace への接続を促されているかどうかを表す invited 属性を持つ。メソッドとして、この Membership で Node を接続する connect メソッド、この Membership から Node を切断する disconnect メソッドを持つ。

3.3. CUG 管理モデルにおけるクラス間の関連

本 CUG 管理モデルのクラス間には、以下の関連がある。

3.3.1. membership 関連

Member が ComPlace に参加していることを表現する関連。この関連は、Membership クラスによって表現される。membership 関連は、ComPlace クラスの join メソッドと leave メソッドによって変更される。Member が ComPlace の join メソッドによって ComPlace に参加すると、対応する Membership が生成される。Member が ComPlace の leave メソッドによって ComPlace から退出すると、対応する Membership が削除される。

3.3.2. hold 関連

Member が Node を保持していることを表現する関連。Member は使用する Node を hold することにより、その Node を使用して Membership を通じて ComPlace に接続することができる。1 つの Node を、複数の Member が hold することが可能である。hold 関連は、Member クラスの hold メソッドと release メソッドによって変更される。

3.3.3. connect 関連

Member がその Node を使って Membership で関連付けられている ComPlace に接続中であることを表現する関連。ある Node が複数の Member による ComPlace への参加に使用されることは出来ない。しかしながらある Node が、単一の Member によって、複数の ComPlace に参加に使用されることは可能である。connect 関連は、Membership クラスの connect メソッドと disconnect メソッドによって変更される。

3.4. CUG 管理モデルにおける管理操作

本 CUG 管理モデルにおける管理操作として、Node を使用して Member を ComPlace で通信可能な状態にする手順と、Node を ComPlace から切断し、member を ComPlace から退出させる手順を、例として以下に示す。

3.4.1. Node を使用して Member を ComPlace で通信可能な状態にする

本 CUG 管理モデルにおける管理操作として、ComPlace を作成し Node を使用して Member が ComPlace で通信可能な状態にするまでの CUG 管理モデルへの操作を図 3 に示す。なお、Member の ComPlace への参加操作と、Member の Node の保持操作の順序は任意であり、どちらの操作を先に行ってもよい。

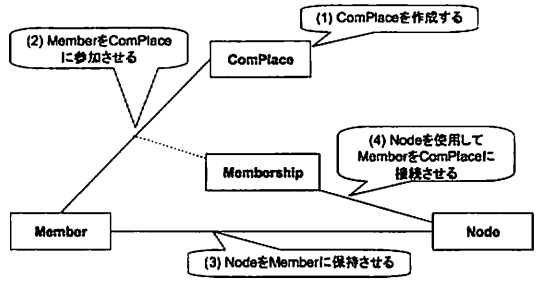

図 3: Node を使用して Member を ComPlace で通信可能な状態にする手順

(1) ComPlace を作成する

ComPlace を作成することにより、CUG を作成する。この際、CUG を作成した Member に関しては、administrative 属性が有効な、対応する Membership が生成される。

(2) Member を ComPlace に参加させる

ComPlace の join 操作により、通信させようとする Member を ComPlace に参加させる。この際、対応する Membership が作成される。なお、この join 操作は、対象となる ComPlace に対して管理権限を持っている Member(対象となる ComPlace との間に administrative 属性が有効な Membership を持つ Member)によってしか行うことができない。

(3) Node を Member に保持させる

Member に対する hold 操作により、接続に使用する Node と Member との間に hold 関連を持たせる。使用する Node が存在しない場合には、Node を新規に作成する必要がある。

(4) Node を使用して Member を ComPlace に接続させる

接続する Member と接続対象の ComPlace との関連を保持する Membership に対して、connect 操作を行い、接続に使用する Node と Membership との間に connect 関連を持たせる。

3.4.2. Node を ComPlace から切断し、Member を ComPlace から退出させる

次に Node を ComPlace から切断し、Member を ComPlace から退出させるまでの CUG 管理モデルへの

操作を、図 4 に示す。なお、以下に示す全ての操作(Node の ComPlace からの切断、Member の Node の解放、Member の ComPlace からの退出)は、いずれも任意の順番で行うことができる。

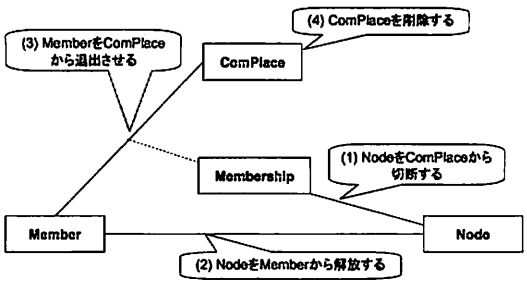

図 4: Node を ComPlace から切断し、Member を ComPlace から退出させる手順

(1) Node を ComPlace から切断する

Node を、ComPlace から切断するには、その Node と connect 関連を持つ、該当する ComPlace との Membership に対して、disconnect 操作を行う。

(2) Node を Member から解放する

Member に対する release 操作により、接続に使用した Node と、Member との間の hold 関連を削除する。

(3) Member を ComPlace から退出させる

ComPlace の leave 操作により、参加していた Member を ComPlace から退出させる。この際、対応する Membership が削除される。削除される Membership が、Node と connect 関連を持っている場合は、その connect 関連は強制的に削除される。

(4) ComPlace を削除する

ComPlace を削除し、CUG を削除する。削除する ComPlace に参加中の Member が存在する場合には、それらの Member は強制的に退出させられる。すなわち、削除する ComPlace に関連する Membership が存在する場合には、それらの Membership は強制的に削除される。

3.5. 本 CUG 管理モデルの特長

本 CUG 管理モデルでは、仮想ネットワークを表現する ComPlace と、ネットワーク機器(接続端点)を表現する Node とを直接関連付けるのではなく、その仮想ネットワークを利用する利用者を表す Member クラスを導入している。ユビキタスな環境においては、1 人の利用者が複数の接続端点を使用して、ネットワークに接続することが考えられ、そのような環境では、その複数の接続端点を束ね、それを利用する Member というクラスの概念を導入することにより、より適切な管理モデルを構成することができる。

また、1 つの接続端点が、複数の利用者によって所有されており、状況によって、異なる利用者の権限によってネットワークに接続される場合がある。このような状況を表現するため、「ネットワークへの接続権限」を表現する Membership というクラスの概念を導入した。

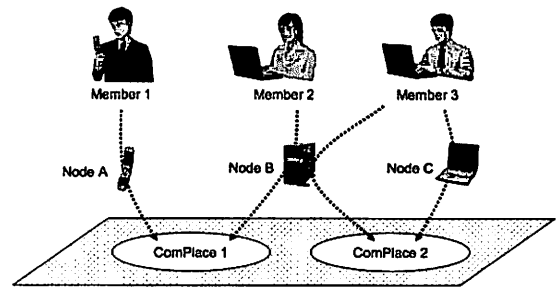

図 5: 本 CUG 管理モデルで管理できるネットワーク

4. CUG 管理インタフェース

4.1. CUG 管理インタフェースの概要

本 CUG システムの CUG 管理インタフェースの概要を図 6 に示す。CUG を管理操作するための通信部分は、一般利用者用 Web サービスと、システム管理者用 Web サービスとで構成され、そのインタフェース仕様は、各々独立した Web サービスインタフェース定義(WSDL)として定義されている。利用者は、任意の Web サービス実装ライブラリを用いて、この WSDL に従った管理アプリケーションを実装することができる。また、より直感的かつ簡便に CUG の管理操作を行うための Java クラス API が、この Web サービスインタフェースの上位の API として提供されている。

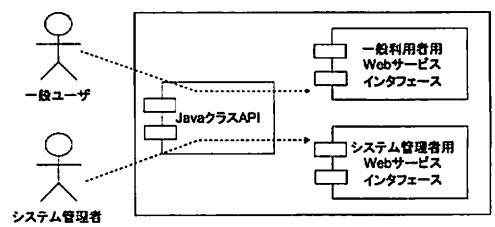

図 6: CUG 管理インタフェースの概要

4.2. Web サービスによる管理インタフェース定義

本 CUG システムの Web サービスによる管理インタフェースは、一般利用者用 Web サービスインタフェースとシステム管理者用 Web サービスインタフェースの、2 つの独立したインタフェースとして提供される。また、Web サービスによる管理操作の際には、Web サービスの利用者は、管理対象である Member に一意に対応付けられ、本 Web サービスインタフェースを通じて操作を行う利用者に対応する Member の ID が、Web サービスによる操作の際、その SOAP ヘッダに格納される。

4.2.1. 一般利用者用 Web サービス

一般利用者用 Web サービスインタフェースは、CUG システムの一般的な利用者が、各 CUG に関連する操作を行うためのインタフェースである。WSDL では、各操作機能は、オブジェクト指向的な、各管理対象オブジェクトに対する操作ではなく、あらゆる管理対象オブジェクトに対する操作が、単一の port に対する operation として実装されている。本インタフェースでは、以下の機能が提供される。

(1) 各管理オブジェクトの参照／更新／削除
(2) Member オブジェクト以外の各管理オブジェクトの生成
(3) 管理オブジェクト間の関連の操作
- Member の ComPlace への参加/退出 (Membership の生成と削除)
- Member による Node の取得／解放
- Node の Membership への接続／切断
- Member の ComPlace への招待
- Member に対する ComPlace の管理権限の設定

ComPlace の変更と削除は ComPlace の管理権限を持つ Member しか行うことができない。また、Member に関する変更、Membership に関する変更は、操作を行っている利用者自身に対応する Member およびその Member に対応する Membership に対してしか、行うことが出来ない。

4.2.2. システム管理者用 Web サービス

システム管理者用 Web サービスインタフェースは、CUG システム自体の管理者のために提供される管理インタフェースである。WSDL では、各操作機能は、オブジェクト指向的な各管理対象オブジェクトに対する操作ではなく、あらゆる管理対象オブジェクトに対する操作が、一般ユーザ用 Web インタフェースとは別の、単一の port に対する operation として実装されている。本インタフェースでは、以下の機能が提供される。

(1) 各管理オブジェクトの強制削除
(2) Member オブジェクトの生成
(3) 各管理オブジェクト間の関連の強制削除
- Member の ComPlace からの強制退出 (Membership の強制削除)
- Member からの Node の強制解放
- Node の Membership からの強制切断
- Membership の強制変更(Member の ComPlace への招待状態の強制変更、Member の ComPlace の管理権限の強制剥奪)

4.3. CUG 管理用 Java クラス API

本 CUG 管理インタフェースでは、Web サービスによる管理インタフェースの上位 API として、より直感的かつ簡便に CUG の管理操作を行うため、Java クラスによる API が提供されている。本 CUG 管理インタフェースで提供する、CUG 管理用 Java クラス API を図 7 に示す。

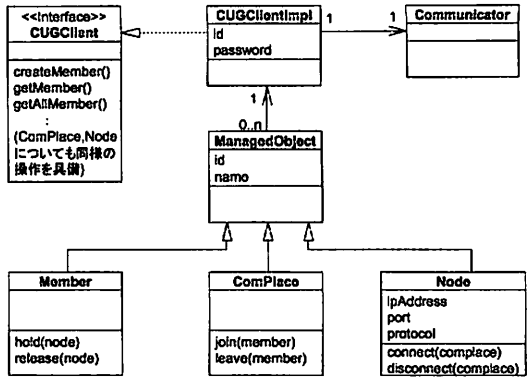

図 7: CUG 管理用 Java クラス API

本 Java クラス API では、一般利用者用 Web サービスインタフェースで提供される管理操作機能と、システム管理者用 Web サービスインタフェースで提供される管理操作機能の両方の機能を、同一の Java クラス API として提供する。また、本 Java クラス API では、Factory Method パターンを採用し、利用者は、管理対象である ManagedObject の派生クラスである Member、ComPlace、Node を直接作成せず、CUGClient インタフェースの操作である get や create 系の操作によって、対応する Java オブジェクトを取得する。本 Java クラス API を構成する各クラスを、以下に説明する。

4.3.1. ManagedObject クラス

管理対象を表現する各クラス(Member クラス、ComPlace クラス、Node クラス)のスーパークラス。各管理対象オブジェクトの共通の属性である id と name をメンバとして持つ。また、当該オブジェクトを参照するために使用した CUGClientImpl オブジェクトへの参照を持ち、当該オブジェクトに対する操作は、全てこの CUGClientImpl オブジェクトを通じて、その CUGClientImpl オブジェクト作成時に指定した Member の権限で行われる。

4.3.2. Member, ComPlace, Node クラス

実際の管理対象である Member、ComPlace、Node を表現するクラス。ManagedObject クラスの派生クラス。これらのクラスは、コンストラクタによって直接生成することはできず、CUGClientImpl クラスのインスタンスの get, create 系のメソッドによって、そのオブジェクトを取得する。

4.3.3. CUGClient インタフェース

CUG 管理機能のクライアントとしてのインタフェース定義。Member、ComPlace、Node 等の各管理オブジェクトに対する get や create 等の操作が定義されており、利用者は、これらの操作によって、管理対象オブジェクトを取得する。本インタフェースの実装クラスのオブジェクトを構築するためには、利用者に対応する Member の ID とパスワードによる認証を行う必要

がある。本インタフェースの実装クラスによって取得された各管理オブジェクトに対する操作(join/leave, connect/disconnect, hold/release)は、全てこの CUGClient インタフェースの実装クラスである CUGClientImpl の生成時に指定された Member の権限で行われる。

4.3.4. CUCClientImpl クラス

CUGClient インタフェースの実装クラス。CUG 管理システムと実際に通信を行う Communicator クラスへの参照を持つ。

4.3.5. Communicator クラス

Web サービスによる CUG 管理通信を行うクラス。一般ユーザ用 Web サービスインタフェースおよびシステム管理者用 Web サービスインタフェースを使用して、CUG の管理操作の通信を行う。WSDL で定義された Web サービス操作と 1 対 1 に対応するメソッドを持ち、Web サービス実装ライブラリごとの差分は、本クラスで吸収される。また今後、Web サービス以外の通信インタフェースを定義した場合にも、その通信仕様を、本クラスに実装することによって、CUGClientImpl クラスに対しては、通信プロトコルに依存しない抽象的な通信機能を提供する。

4.4. 本 CUG 管理インタフェースの特長

本 CUG 管理インタフェースでは、各管理対象オブジェクトへの操作を、Web サービスにおいて単一の port に対するフラットな operation 群として定義することで、ライブラリ提供者にとって比較的実装しやすい、単純な WSDL によるインタフェース定義となっている。また、一般利用者用 Web サービスインタフェースと、システム管理者用 Web サービスインタフェースとを分離・独立して定義することにより、一般利用者に対して、公開する必要のない操作定義を隠蔽している。

また、これらの Web サービスインタフェースの上位 API として、共通の Java クラス API を提供することにより、より簡潔な API を実現している。本 Java クラス API によって、リスト 8 のような、Java による簡潔かつ直感的な CUG 操作のプログラミングが可能である。

```
CUGClient client = new CUGClientImpl("member0");

// ComPlace を作成する
ComPlace cp1 = client.createComPlace("ComPlace1");

// Member, Node オブジェクトの取得
Member   m1 = client.getMember("Member1");
Node     n1 = client.getNode("Node1");

cp1.join(m1); // Member を ComPlace に参加させる
m1.hold(n1); // Node を Member に保持させる
n1.connect(cp1); // Node を ComPlace に接続する
```

リスト 8: Java クラス API を使用した
CUG 操作の Java ソースコード例

5. 今後の課題

本稿で述べた CUG 管理 Web サービスインタフェースでは、管理対象オブジェクトに対する操作を、単一の port に対するフラットな operation 群として定義している。この仕様は、ライブラリ提供者にとっては、単純であるため実装しやすいというメリットを持つ反面、利用者にとっては、使いにくいというデメリットを持つ。本 CUG 管理インタフェースでは、このデメリットを補うべく、Java クラス API を提供しているが、今後は WSDM:MUWS 等を導入し、管理対象をそのままリソースとしてオブジェクトの形で見せるような Web サービスインタフェースも設計したい。しかしながら、WSDM:MUWS を導入した場合、そのインタフェース定義が複雑になり、ライブラリ提供者にとっては実装しにくくなるというデメリットをもたらす可能性もあるため、管理インタフェースへの WSDM:MUWS の導入にあたっては、これらのメリットとデメリットを慎重に吟味して、検討する必要がある。

6. まとめ

本稿では、当グループが開発している CUG システムの特長を示し、本 CUG システムが他の業務アプリケーションのネットワーク制御プラグインとして利用可能なことと、その際には CUG 管理モデルと CUG 管理インタフェースが重要であることを示した。次に、当グループで設計した CUG 管理モデルと、Web サービスによる CUG 管理インタフェース、およびその上位 API である CUG の管理と操作のための Java クラス API を示した。また、Web サービスによる CUG 管理インタフェースへの MSDM:MUWS の導入を、今後の課題として挙げた。

謝辞

本稿で述べた CUG 管理モデルおよび CUG 管理インタフェースの設計と実装にあたっては、有限会社グラッツの今井克則様と沼田暁子様に多大な御協力を頂いたことを、深く感謝申し上げます。

参考文献

[1] "Generic Network Information Model", ITU-T Recommendation M.3100

[2] "IP Network Management Information Agreement", TMF 611

[3] "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Framework", RFC 3411

[4] "Web Service Distributed Management: Management Using Web Service (MUWS 1.0)", OASIS Web Service Distributed Management (WSDM) TC