Vol.2009-IOT-6 No.2
2009/6/27

Open Autonomic Networking(OAN) SDK を用いたネットワーク管理ツールへの応用

新 善文† 飯島智之† 木谷誠† 木村浩康† 黒崎芳行†

ネットワーク運用の自動化とネットワーク管理を IT システムと親和性の高いものにするために, NETCONF をベースに OAN SDK を開発した. これは Java の API を持ち，既存のソフトウェア開発ツールで扱い易いものである．これを利用することにより，ネットワーク管理ツールが容易に開発できることを示した．

Network Management Tools and Applications by Open Autonomic Networking(OAN) SDK

YOSHIFUMI ATARASHI† TOMOYUKI IIJIMA†
MAKOTO KITANI† HIROYASU KIMURA†
YOSHIYUKI KUROSAKI†

We developed Open Autonomic Networking Software Development Kit (OAN SDK) based on NETCONF. It is for network management systems increase affinity of IT systems. And we made some network management tools using OAN SDK. So it is very useful to make network management tools for its Java API.

1. はじめに

インターネットの利用は電子メール，ファイル転送だけでなく，WWW, P2P などいろいろなアプリケーションとともに発展を遂げ，ショッピングや行政サービスなどにも使われるようになってきた．今日ではインターネットはインフラとしての重要さが増すとともにその規模の拡大と安定性，安全性が重要となっている．それに伴い IP ネットワークやシステムの運用管理も複雑になってきている[1]．システム管理者は増加するネットワーク機器やサーバにそれぞれ複雑な設定をすることになるため，その負担は指数関数的に大きくなっている[2]．それにも関わらず，これまでネットワーク運用は革新的な技術の導入はなく，管理者の経験に基づいた旧態依然の手法が用いられていた[3]．IT システムは常に変化し続けているため，その変更や開発にあわせて，本来ネットワークも柔軟で敏速に変更可能とすべきである．

そこで我々は IT システムの変更や開発にあわせて日々更新が可能なネットワークを実現するためにネットワークの「運用の自動化」と「IT システムとの高い親和性」に取り組んだ. 運用管理の自動化を行うための仕組みとして, IETF (Internet Engineering Task Force)で仕様策定がすすんでいる netconf[4]を用いて API を定義するだけでなく，プログラム開発を促進するために SDK(Software Development Kit)を開発した．またこれを用いて運用管理ツールを作成し，SDK 応用の可能性を示した．

2. 従来技術

これまでネットワーク機器は，主にコマンドベースの CLI で制御していた．CLI はネットワーク管理者が利用することを前提として開発されている．CLI の仕様は機器や機種，提供ベンダに依存するが，操作する対象が人間と言う柔軟で知的な存在であるため，機器や機種，提供ベンダ毎に異なる仕様を吸収して運用することが可能であった．

しかし「運用の自動化」の実現を目標としてこれをコンピュータに代行させようとすると，複雑な処理を盛り込む必要がある．例えば CLI は人間が使うことを前提にしていたため戻り値の概念がなく，応答メッセージを解析するコードを用意しなければならない．また応答メッセージのフォーマットやメッセージの出現順序なども機器ごとに異なるため，対応する解析プログラムを書くにも，多大な労力を必要とする．

SNMP（Simple Network Management Protocol）は，ネットワーク管理のためのプロトコルであり，それをベースとして様々な試みがなされた[3]．しかしながら，基本設計が古く，そもそもオブジェクト指向でなく変数指向であるため，ネットワーク装置に設定を施す場合 MIB（Management Information Base）ツリーの変数を 1 つずつ設定し

† アラクサラネットワークス(株)
Alaxala Networks Corp.

ていかなければならなかった. 更に複数のマネージャが同時に MIB へ変更を加えようとした場合，衝突を回避する仕組みがなかった[5]．このような仕様のため SNMP は設定に不向きで，主に情報収集のためにしか利用されて来なかった[1][6]．その証拠にインターネットが日々進化しているにも関わらず，IETF では SNMP をさらに強化しようという SNMPv4 へ向けた活動はされておらず，「運用の自動化」を実現するための新たな運用方式が模索されている[7].

3. OAN (Open Autonomic Networking) SDK アプローチ

3.1 システム開発効率向上と IT システムとの親和性

IT システム開発において最もコストがかかるのがプログラム開発である．プログラムの生産性を向上させ，開発期間を短縮するため COBOL，PL/I，C，Java などのプログラミング言語が登場した．さらに現在のコンピュータシステムやストレージ管理には XML インタフェースが採用され, 相互接続もかなりできるようになってきている．その一方でネットワーク機器に対する制御は CLI の域を出ていない．アプリケーションからネットワーク機器の機能やネットワーク構成を制御するために，プログラム言語からの制御用インタフェースが求められている．

そのような制御用インタフェースがあれば，IT システムとも親和性の高いネットワーク管理アプリケーションが開発可能となる．例えば「会議室予約システムで会議室が予約されると，それと合わせて会議室に設置されたネットワーク機器のポートを開ける」などの業務と連動してネットワークを制御するネットワーク管理アプリケーションが開発可能となる．

これまでに述べた CLI や SNMP のネットワーク運用方式としての課題とその対策，およびその対策技術を表 1 にまとめた．

従来のネットワーク運用方式の課題を解決し「運用の自動化」と「IT システムとの高い親和性」を実現する技術として，「NETCONF」,「データモデル」,「Open Autonomic Networking SDK(OAN SDK)」をあげた．ネットワーク機能をオブジェクト化して制御可能なプログラム言語としては OAN SDK が採用した Java の他にも C++や Ruby などもあるが，開発者の人口，技術成熟度そしてそれに伴う NETCONF 実装のしやすさの観点から Java を選択した．

それぞれの対策技術の関係を図 1 に示す．それらの技術詳細とそれがどのように課題を解決するかを以下に述べる．

3.2 NETCONF

NETCONF は IETF にて標準化が進められているネットワーク機器制御のためのプロトコルである[4][8]. 図 1 に示すように NETCONF は設定項目の操作手順を標準化しており，ネットワーク機器とネットワーク管理ソフトウェアの両方が NETCONF を実装することにより利用できるようになる．設定対象に関しては今後議論がなされる見込みである[9].

SOAP の利用はネットワーク経由で複数のアプリケーションを連携可能にする Web サービスのメッセージ仕様である[10]．これにより NETCONF を用いたネットワーク管理ソフトウェアを他の IT システムと連携させることが容易となる．

表 1「運用の自動化」「IT システムとの高い親和性」の実現に必要な対策

課題	対策	対策技術
結果判定の枠組みに標準的な仕様がなく，機器や機種，提供ベンダ毎に異なる	標準プロトコルを策定する	NETCONF
MIB ツリー化した設定項目は，コンピュータから扱いにくい	設定項目を論理的に構造化する	データモデル
アプリケーションからネットワーク機器やネットワーク構成を制御する開発技術がない	ネットワークの機能をオブジェクト化した制御用インタフェースを提供する	OAN SDK (Java API)

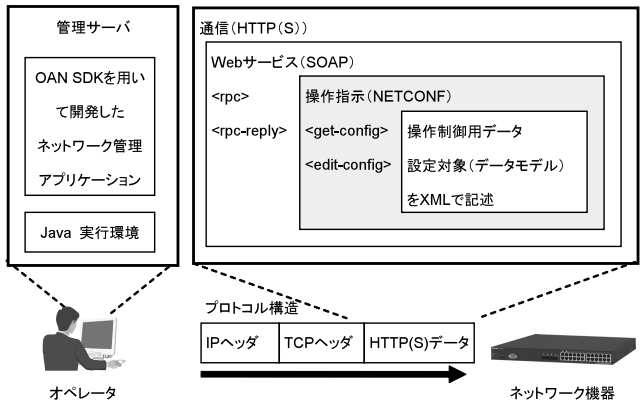

図 1　NETCONF，データモデル，OAN SDK の関係

3.3 データモデル

NETCONF はネットワーク機器とネットワーク管理ソフトウェア間の通信については規定しているが，どの対象をどう管理し制御するかは規定していない．例えば，LAN スイッチが備える VLAN 機能やフィルタ機能の設定をどのように変更するのかは決めていない．これらの機能の設定手順や方法を機器に依存しない形で共通化する必要があり，それが「モデル化」である．モデル化された設定項目が「データモデル」である．[11]

UML（Unified Modeling Language）を用いてモデル化するとそれぞれの設定項目の関係が視覚的にも明確となり，管理アプリケーションの開発が容易になる．モデル化されたフォーマットに則ってネットワーク機器とネットワーク管理ソフトウェアが連携することにより，ネットワーク機器のベンダや機種や依存することなく，ネットワークを制御が可能となる．モデル化されたフォーマットはコンピュータからの可読性が高い XML で記述する．XML で記述することによってデータが階層化され，コンピュータで扱いやすい設定項目構造となる．

3.4 Open Autonomic Networking Software Development Kit (OAN SDK)

NETCONF の下位トランスポートプロトコルに SOAP を利用すると，図 2 に示すようにネットワーク制御用インタフェースは WSDL（Web Service Description Language）で提供することになる．WSDL は Web サービスに対するインタフェースを記述するための標準仕様で， XML で記述されているためコンピュータに直接取り込んでネットワーク機器にアクセスさせることができる．

しかし，現在アプリケーション開発者にとって WSDL を直接編集してネットワーク機器を制御するのは容易なことではない．なぜなら WSDL を扱える開発者の数がまだ少なく，WSDL によるアプリケーション開発の土壌が整備されていないからである．そこで WSDL から Java API を自動生成するツールを利用し，それにより生成された Java API を使うことによって，ネットワーク機器の制御ができるようになる．Java API により，ネットワーク機能がオブジェクト化され，ネットワークの機能や複雑さが隠蔽される． この API をさらに使いやすくするために SDK を作成し，OAN SDK と名付けた．これを利用することにより開発ツール/統合開発環境の NetBeans[12]や Eclipse[13]などが利用でき，ネットワーク技術者以外のプログラマがネットワーク管理ソフトウェアやツールを開発しやすくなる．

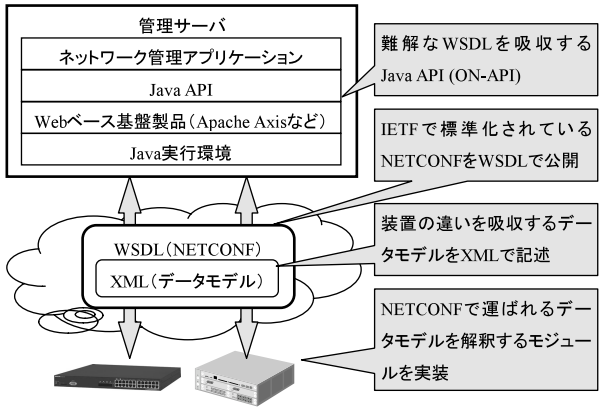

図 2 Java API(OAN SDK)と WSDL の関係

4. OAN SDK の開発

4.1 実装方法の検討

OAN SDK を開発するにあたり，ネットワーク機器への機能追加をどの程度おこなうのか，どこをインタフェースとして規定するかあるいはプロトコルとして実現するかといった違いにより，従来の CLI, SNMP の他に 3 つの方法を検討した．この比較検討を表 2 に示す．

実装方法の検討で比較したのは以下の 4 つである．

1. CLI, SNMP ：従来の管理手法
2. 機器内高機能化とプロトコルによる実装 ：プロトコルの定義，開発が必要
3. 機器内高機能化と API で実装 ：機能を機器内に実装したものを API でコール
4. シンプルな API による実装:この API を介してサーバと連携が前提となる実装

また評価基準は以下の 5 つとした．

a. プログラムからの制御
b. プログラム作成の容易さ
c. 運用のしやすさ
d. 開発環境の充実度
e. 製品品質と信頼性の確保

表 2 実装のための比較検討

	評価項目	1	2	3	4
		CLI , SMTP	機器内高機能化と protocol	機器内高機能化と API	Simple API
a	プログラムからの制御	×(△)	-	○	○
b	機能追加の容易さ	×	×	×	○
c	運用のしやすさ	△	○	○	○
d	開発環境の充実度	×(△)	×	○	○
e	製品品質と信頼性の確保	△	×	×	○

ここで表 2 の内容を詳しく説明する. 1a, 2c という表記は実装方法の番号 1-4 と評価項目の記号 a-e を組み合わせて，表内の位置を示している．

CLI, SNMP はプログラムから扱う場合はプログラムとしてのインタフェースでなく，スクリプトして実装する必要があり，機器の状態が正確に把握できないため 1a,は×(△)とした．1b の機能追加はベンダ側での開発が必要なので×．1c の運用については現状どおりなので△, 1d の開発環境はスクリプト処理のライブラリがあるので×(△), 1e の信頼性の確保は現状どおり△とした．

機器内高機能をプロトコルで実装するものは，2a はプログラムからの制御でなく，自律運用をねらうものなので-とした．2b はプロトコルの開発に非常に多くの工数がかかるので×とする. 2c の運用についてはコマンド入力でプロトコルを enable することによりその機能を扱えるはずなので○. 2d の開発環境は機器ベンダでないとできないので×．2e の信頼性の確保は新たなプロトコルは相互接続性の検証や追加された機能の検証が雪だるま式に増えるので×とした．

機器内高機能化を API で実装するものはプログラムから扱う API を定義するので，3a は○．3b の機能追加はベンダ側で機能追加の開発を行う必要があるため×．3c の運用については追加された機能は API で扱うことができるので○. 3d の開発環境はサポートする言語やライブラリによるが統合開発環境が使えるだろうということで○とした．3e の信頼性の確保は機器内に機能追加されたことにより，検査項目が増えるため×とした．

シンプルな API をネットワーク機器側に実装し，運用管理サーバと連動させる 4a は API により，プログラムから制御できるので○．4b の機能追加は運用管理サーバ側で実現でき，ベンダにたよらず可能となるので○．4c の運用は運用管理サーバから制御でき，運用工数削減効果が見込めるので○．4d の開発環境は統合開発環境，ツールを利用できるので○．4e の製品品質と信頼性の確保については機器内に機能の追加をおこなわないので検査工数を増やすことがなく，機器はベンダで保証された状態で運用できるので○とした．

以上の比較検討により，我々はシンプルな API をネットワーク機器側に実装し，運用管理サーバと連動させることが最善の実装方法だと判断した．したがって，この方針で OAN SDK を開発した．

4.2 ON-API の実装

Java API である Open Networking API(以後，ON-API と略す)を実装したネットワーク管理ソフトウェアとネットワーク機器の構成を図 3 に示す．ネットワーク機器は HTTP メッセージを受信するために HTTP デーモンを稼動させておく．また，HTTP メッセージ内に納められた NETCONF メッセージを HTTP デーモンから受信するため NETCONF デーモンを稼動させておく. NETCONF デーモン内には XML パーサを用意し, NETCONF メッセージの解析を行い管理アプリケーションからの指示を認識する．

ON-API を用いて開発されたネットワーク管理ソフトウェアによるネットワーク機器設定シーケンスを図 3 により解説する．

(1) 管理ソフトウェアがネットワーク機器の設定要求を発行する．
(2) 設定要求は NETCONF メッセージのフォーマットに整形され，SOAP エンベロープ内に納められ，サーバからネットワーク機器へ送られる．
(3) SOAP エンジン部にて受信メッセージから SOAP エンベロープを外し，NETCONF メッセージを取り出す. NETCONF メッセージ は NETCONF デーモンへ送られ，XML パーサにて解析される．解析された結果に基づきネットワーク機器の設定を反映する．
(4) 設定結果が NETCONF デーモンへ返され，NETCONF デーモンは戻り値を持った NETCONF メッセージを生成する．
(5) SOAP エンジン部は NETCONF メッセージを SOAP エンベロープに納めサーバへ返信する．
(6) アプリケーションは戻り値に応じて予め設定しておいた処理を実行する．

4.3 ON-API の機能

現在，ON-API がサポートする機能と各機能にて設定可能なインスタンス変数例を表 3 に示す．サポートする機能は，ネットワーク管理アプリケーションよって自動化できると望ましい機能から実装している．

したがって，エンタープライズネットワーク内で頻繁に変更がなされる VLAN 機能や容易に多数の設定をする傾向にある Filter 機能などを優先的にサポートした．

設定項目については，管理ソフトウェアで主に利用されると考える項目を検討し，各機能の動作に必要となる基本的な項目からサポートしている．

アプリケーション開発のためには表 3 に示した機能毎の設定項目をモデル化（データモデルの規定）する必要がある．

4.4 OAN SDK の利用方法

OAN SDK は ON-API を扱いやすいようにするためにパッケージ化したものである．OAN SDK によりサーバ上で開発したネットワーク管理ソフトウェアは ON-API にてネットワーク機器を制御する．

NetBeans や Eclipse といった開発ツールを使うことによってデータモデルをベースにした ON-API を利用できる．このメソッドを利用することによって例えば VLAN の各インスタンス変数を設定，参照することができるようになる．インスタンス変数の値を設定する場合は set 系メソッドを，値を参照する場合は get 系メソッドを使用する．VLAN ID が 100 の VLAN を作成したい場合には setVlanid()メソッドに 100 を引数として設定し，setVlanid(100)とコールすれば良い．ネットワーク機器に既に設定されている VLAN の VLAN ID 値を参照したい場合，getVlanid()メソッドを利用すればよい．

これらのメソッドを利用することによって，従来よりも容易にネットワーク管理ツールが開発できるようになる．ネットワーク管理ツールの開発には OAN SDK をインストールした開発環境を用いればよい．

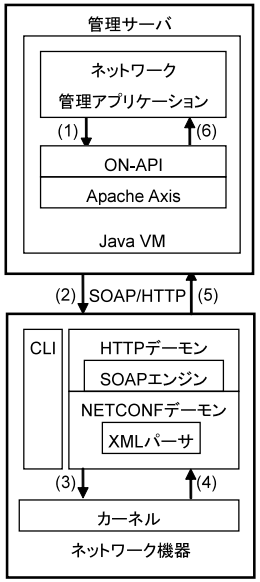

図 3　ON-API 実装サーバとネットワーク機器の構成

表 3　ON-API 機能例

機能	インスタンス変数
VLAN	VlanId, VlanName, etc.
Line	PortId, Speed, Name, etc.
Link Aggregation	LaId, LaType, etc.
Node	Name, Location, etc.
Filter	SourceIP, DestinationIP, etc.
Route	Destination, NexthopAddress, etc.

5. OAN SDK を用いたネットワーク管理ツール

OAN SDK を用いたネットワーク管理ツールとして MAC アドレス管理ツールを作成した．このツールは昨今のセキュリティ強化のためネットワーク接続された PC，サーバなどの MAC アドレスを管理し，不正アクセスを防ぐことができる．

また Config ファイル管理ツールも作成した．Config ファイル管理ツールは複数のネットワーク機器から Config ファイルを取得，保存でき，差分の比較などの機能を持つ．管理ツールのこれらの画面イメージを図 4, 5 に示す．

図 4　MAC アドレス管理ツール

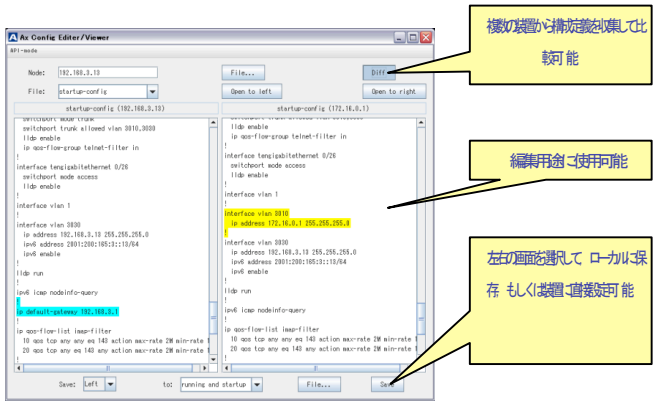

図 5　Config ファイル管理ツール

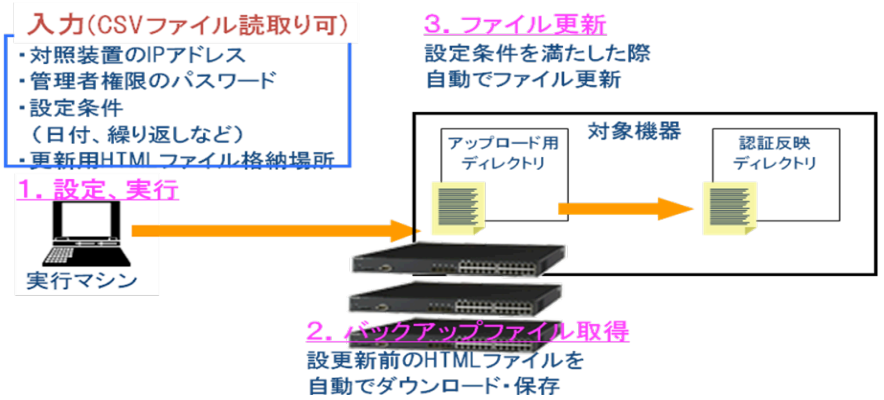

図 6 web 認証画面変更ツール

さらにネットワーク機器の持つ web 認証機能の画面設定を変更するツールを作成した．これは 1) 設定条件ごとに画面を自動更新，2) 複数装置に同時にファイル更新，3) HTML ファイルのバックアップ取得，をねらったものである．

これらを CLI での操作不要で，複数装置に自動で更新を実行できるようにした．操作としては対象装置の IP アドレス，管理権限パスワード，アップロードする HTML 保存パスを登録するだけである．

6. 評価

ここでは Web 認証画面変更ツールの評価をおこなう．Web 認証画面更新ツールはネットワーク管理に詳しくない開発者 1 名で，約 2.6k step(開発期間 2 か月)であった．これは開発者のスキルを考慮しても効率よく開発できたことを示している．

またこのツールの管理作業の効率化を見るために装置 2 台の設定について従来方法とこのツールとの比較をおこなった．その結果，従来 CLI で 200 秒かかっていたものが，スクリプトを書いて 8.5 秒，このツールではさらに 2.5 秒に短縮できた．よって，CLI に比べると約 80 倍，スクリプトと比較しても 3.4 倍の効率化が確認できた．

7. おわりに

OAN SDK を利用することにより，ネットワーク管理ツールが容易に開発でき，しかもネットワーク機器は信頼性があり，保守サポートされているものを使用することができる．これはネットワークシステム運用全体での品質向上に寄与する．

本論文ではネットワーク管理ツールを開発し，OAN SDK の有効性を示したが，API の利用方法は他にもある．たとえば，ネットワーク以外の認証機能(IC カードなど)を使ってネットワークの VLAN を切り替えるといったセキュリティシステムが構築できる．他にもアプリケーションからネットワークが制御できないために，経験的な方法でのネットワークを使用していたものが，きちんと管理された状態でネットワークが本来持っている機能を使いきることができるようになる．

今回のツール開発の経験において，ツールの作成以前に何をしたいのか，その時に管理する情報は何かが問題となった．そのためにデータモデルを考えることに開発の多くの時間を費やした．これは本来の目的を明確にすることに工数をさいたことであり、これが思いがけずツールの完成度向上に効果があった。この経験より OAN SDK の利用がツールの開発効率向上だけでなく，ツールの完成度やユーザ満足度の向上にも効果的である．

謝辞

ネットワーク管理ツールの開発に多大な協力をいただいた（株）日立製作所の金居秀明氏に感謝の意を表する．

参考文献

1 Mi-Jung, C. et al.: XML-based configuration management for IP network devices, IEEE Communications Magazine, Vol.42, No.7, pp.84-91 (2004).
2 宮本貴朗，田村武志，鈴木亮司：大規模ネットワークにおける VLAN 管理システム，情報処理，Vol.41，No.12，pp.3234-3243(2000).
3 長田智和，谷口祐，玉城史朗：大規模分散ネットワーク運用管理システムの提案，情報処理学会研究報告，Vol.2000，No.113，pp.31-36(2000).
4 IETF, Network Configuration (Netconf), http://www.ietf.org/html.charters/netconf-charter.html.
5 Gerhard, M. et al: Using Netconf for Configuring Monitoring Probes, Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP, pp.1-4(2006).
6 George, P. et al.: On management technologies and the potential of web services, IEEE Communications Magazine, Vol.42, No.7, pp.58-66 (2004).
7 Aiko, P. et al: XML-Based Management of Networks and Services, IEEE Communications Magazine, Vol.42, No.7, pp.56-57 (2004).
8 Rob, E.: NETCONF Configuration Protocol, RFC4741 (2006).
9 Tomoyuki, I. et al.: VLAN data model for NETCONF, draft-iijima-ngo-vlandatamodel-01, work in progress (2007).
10 Ted, G.: Using NETCONF over the Simple Object Access Protocol (SOAP), RFC4743 (2006).
11 Tomoyuki Iijima, et al.: Development of NETCONF-Based Network Management Systems in Web Services Framework, IEICE Trans. Communication, Vol. E92-B, No.4, pp1104-1111(2009).
12 NetBeans, http://www.netbeans.org/.
13 Eclipse, http://www.eclipse.org/.