

ネットワーク実験支援システムに必要な機能と設定記述

宮地利幸^{†‡§}

知念賢一^{†‡§}

篠田陽一^{§†‡}

北陸先端科学技術大学院大学

[†] 情報科学研究科 / [§] 情報科学センター / [‡] インターネット研究センター

[§] 情報通信研究機構 北陸リサーチセンター

Functions and Descriptions for Network Experiment Support System

Toshiyuki Miyachi^{†‡§}

Ken-ichi Chinen^{†‡§}

Yoichi Shinoda^{§†‡}

[†]School of Information Science / [§]Center for Information Science

[‡]Internet Research Center, Japan Advanced Institute of Science and Technology

[§]Hokuriku Research Center

National Institute of Information and Communications Technology

1 はじめに

新たなネットワーク技術のアイデアを検証する際や、その実装を実環境に導入する以前、また、既存の技術の挙動の観測のために、様々な手段を用いてネットワーク実験が行われている。

我々は、多数の実ノードを利用した実験を行うための施設である StarBED[1] での実験支援システム SpringOS を開発している。この経験をもとに、実験環境および支援システムに必要とされる機能について議論している。ここで言う機能は主に、実験ノードおよびトポロジの設定、シナリオ実行のためのものである。

実験の自動的な遂行のためには、実験前に実験トポロジおよび実験手順を記述する必要がある。利用者は、実験手順を記述することにより、実験環境および実験支援システムの機能を利用する。これらの手順は、実験環境が要求する文法にしたがって記述される。しかし、利用者が、実験のためだけに新たな言語を習得するのは効率的ではない。一般的に普及しているプログラミング言語と同一の文法で実験の設定記述が可能なら、多くの利用者は新たな言語の習得なしに実験を行える。しかし、既存の言語の文法では、実験設定言語とは性質が異なるため、そのまま利用するのは困難である。そこで実験設定に適した言語を新たに提案する。

そこで、ネットワークの実験遂行に最低限必要な機能および、その機能を利用するための言語を定義する。一般的なプログラミング言語の文法にしたがって記述された設定記述を、我々の定義した言語に変換し、実験遂行効率の向上をはかる。単純な機能の集合として言語を構成することで、様々な文法からの変換に対応できる。複

雑な機能は単純な記述の組合わせで実現する。

本論文では、これまでの SpringOS 開発の経験をもとに、次世代の支援システムのための基本となる機能と、記述言語について議論する。ただし、ここでは、必要な機能について述べるにとどめ、その実現方法については言及しない。SpringOS でのこれら機能の詳細や実装については、[2][3][4][5] にまとまっている。

2 実験支援システムの役割概要

本章では、実ノードを用いて実験を行う際の実験手順について考察し、それらを自動的に行う実験支援システムに必要な機能について議論する。

我々は、これまで実ノードを用いた実験環境である StarBED 上での実験支援システム SpringOS を開発してきた。ここでは、実ノードを用いて実験を行う際の一般的な手順を以下に示す。

1. 実験トポロジやシナリオの検討 実験の目的を考慮し、適切なネットワークトポロジ、各ノードの OS・ファームウェアや必要なアプリケーション、実行されるべき実験イベント発生手順(シナリオ)を検討する。
2. 必要なノードの用意 1の結果に基づいて必要なノードを用意する。
3. ノード接続 用意したノードを1の結果にしたがったトポロジに接続する。
4. ノードへのソフトウェアの導入 ノードへ必要な OS・ファームやアプリケーションを導入し、必要な設定をする。また、各ノードの IP アドレスや経路などを設定する。
5. シナリオの実行 構築された環境上で実験シナリオを遂行し、実験データを収集する。

6. ログの解析 得られた実験データを解析する。

ネットワーク実験支援システムで、これらの作業すべてを自動的に行えば、利用者は実験の詳細を決定し、それを前もって記述しておくだけで実験が自動的に行われる。これらの手順を自動的に行うために必要であると考えられる機能を示す。

ノード設定とシナリオの事前指定 実験トポロジを自動的に構築し、任意のシナリオで実験を進行させるためには、前もって支援システムに利用者の意図するトポロジおよびシナリオを入力する必要がある。このような要求を満たすための仕組みが必要となる。

ノードへの OS/ファームウェア導入 ノードへ利用者が指定した OS やファームウェアおよびアプリケーションを導入するための仕組みが必要である。

実ノードの設定 IP アドレスや経路情報など、実ノードを実験用のノードとして動作させるための設定を行う仕組みが必要である。

実験手順の遂行 指定された順序/タイミングでコマンドを実行する必要がある。これにより実験が遂行される。タイミング調整の一つとして複数のノードの同期機構は必須である。

リソース管理機構 実験に必要なリソースを管理する。リソースには、実験に利用できるノードやネットワークを構築する際に利用する可能性がある VLAN 番号などが挙げられる。この機構は利用者から要求を満たすノードを割り当てる。また、複数の利用者により環境が共有される場合は、他の利用者と同じリソースを割り当てないような調停機構が必要である。

ノードの起動・停止機構 ノードの起動や停止、再起動などの処理を自動的に行う機構。これにより実験開始時にノードを起動したり、終了時にノードを停止する。またノードがハングアップしてしまった場合に対応できるよう、電源レベルでの制御ができることが望ましい。

これらの手順を手により行う場合と支援システムにより自動化された場合の比較を図 1 に示す。支援システムが存在しない場合、利用者は、各ノードの様々な制御は、コマンドの実行などにより手動で行うか、遠隔からスクリプトを実行することになる。すなわち利用者は実験実施中、常に作業を行わなければならない。また、実験の規模が大きくなった場合には、多数のノードを制御しなくてはならないため、実験の遂行は困難であり、さらに、人手による誤操作の問題は深刻である。一方、実験支援システムを利用すれば、実験の設定を事前に記述するのみで、実験は設定にしたがって忠実に行われ、新

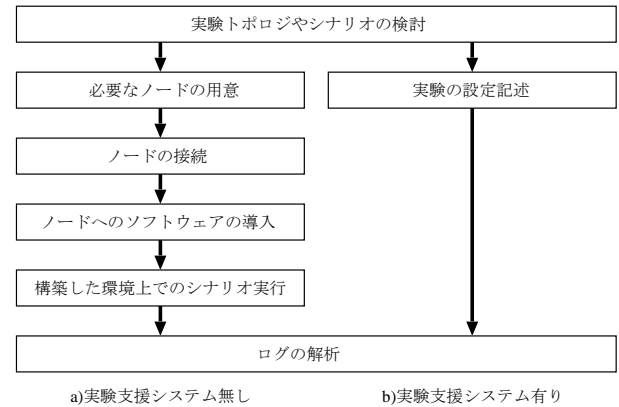


図 1: 実験手順の一例

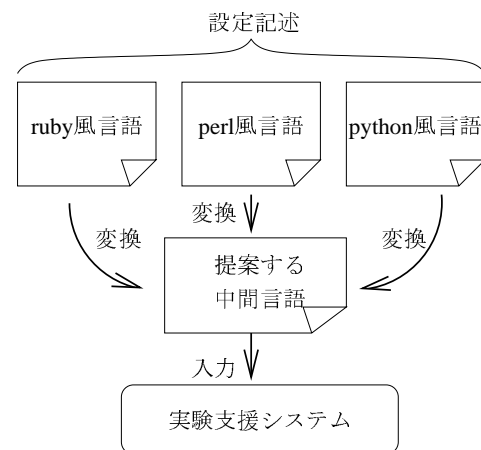


図 2: 設定記述の差異の吸収

たな技術の開発および実験結果の検討という、本来の目的に注力できる。

3 実験の要素機能と設定記述

2 章で実験に必要な機能の概要について述べた。本章では、利用者が実験環境または実験支援システムに要求する最低限必要な機能とその記述方法について議論する。複雑な機能は単純な機能を組み合わせることで実現する。多くの利用者にとって、perl や ruby、python などといった一般的な言語を利用できた方が実験を容易に行える。したがって、我々は一般的な言語で記述された設定を、図 2 のように、我々が定義した言語に変換する。これにより、利用者はより親しんだ言語で設定を記述でき、実験遂行の効率が向上する。

perl や ruby のような一般的な言語では、多くの場合、複数の要素の同期をとるための機能が用意されていない。しかし、ネットワーク実験ではノードの同期は必須であるため、新たな言語を定義する。

実験トポロジは基本的にノードとリンクにより成り立つため、ノードとリンクの設定ができなければならない。

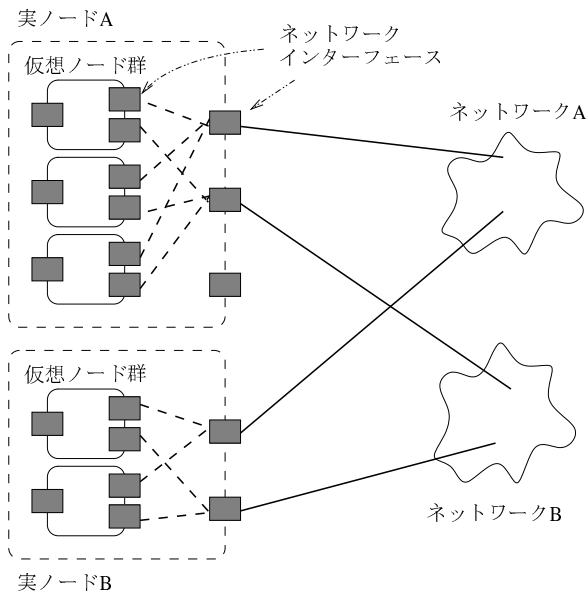


図 3: 仮想ノードおよび仮想インターフェースを利用した実験トポロジ例

実験シナリオを遂行するためには、プロセスの実行・終了などのアクションを実行できる必要がある。また、前述の通り、アクションを起こすタイミングを、他ノードとの同期により決定するための機能も必要である。

ノード数が足りない場合などに、VMware[6] や Xen[7] のような仮想ノード実現技術を利用して、ノードの多重化を行い、一台の実ノードを複数台に見せる場合がある。この場合は、実ノードと仮想ノードの両方を制御する必要がある。また、実験ノード上で VLAN の設定を行う際には物理インターフェースの上に論理インターフェースが設定されるため、同様に物理インターフェースと論理インターフェースの設定が必要となる。仮想ノードと仮想インターフェースを利用した場合の実験トポロジ例を、図 3 に示す。この要求を満たすため、階層的に実験要素を定義するような機能が必要である。

3.1 ノード

実験の主体となるのはノードであるため、ノードに関する多くの記述が必要である。また、ノードの記述は、ノード選定の条件と、実験を行うためにノードに導入されるべき設定がある。

ノード選定の条件となりうる事項は以下の 2 点である。

ネットワークインターフェースに関する設定

これは必要なネットワークインターフェースの数だけ行われる。設定された数によりノードが選択される。また、ネットワークインターフェースの識別子、種類および IP アドレスなど設定項目も含まれる。

ノードの性能に関する設定 実験対象の技術や実装により、一定値以上の性能をノードに要求する場合が多

い。このような場合に、ノードの性能や種類を定義することが必要となる。具体的には、CPU の種類、CPU のクロック数、メモリ容量、HDD 容量などがある。

これらの条件に基づいて、リソース管理機構により実験に利用できるノードが選定され、利用者に割り当てられる。実験によっては、リソース管理機構を介さず、意図的にノードを割り当てる場合があるため、それを満たす機能も必要である。

以下は、ノードを実験用の要素として動作させるために必要な設定事項である。

ノード名 ノードの識別子。実験中はここで指定した識別子で各ノードを判別する。

ノード種別 PC またはルータ、スイッチなど、ノードの種別を記述する。

ノード操作方法 ノードにより操作方法が異なる。特にスイッチなどはベンダーにより操作方法は様々である。どのような手順で設定および操作を行うかを指定する必要がある。

起動方法 ノードの起動方法には、PXE[8] や TFTP[9] を用いたディスクレスでの起動や、すでにノードの HDD にインストールされている OS を用いた起動などが考えられる。ここで記述された方法にしたがいノードが起動される。

OS の種類 指定した OS と実際に起動してきた OS が一致するかを確認することで、正しい OS が起動しているかどうかを確認できる。

ディスクイメージ/起動パーティション ID ノードにソフトウェアを導入する方法の一つとして、前もって用意しておいたディスクのイメージを書き込むことにより、ディスクの内容を複製する方法がある。このとき、書き込むべきディスクイメージの URI が指定される必要がある。また、ディスクレス環境で利用する場合も、カーネルや、メモリイメージの URI が必要である。一方、ノードにすでに導入されている OS から起動する場合は、パーティション・フラッシュメモリなどの起動元の指定が必要である。

割り当てる実ノード 実験用の実ノードを管理する機構が動作している場合は、必要な機能を持つノードを割り当てることもできるが、場合によっては意図したノードを割り当てたい場合もある。このような場合に、明示的にノードを指定できる必要がある。

実行するシナリオ このノードで実行されるシナリオを記述する。記述方法に関してはアクションの節で述べる。

ノードの階層化を実現するために、設定記述中に仮想ノードがどのノード上で動作するかを記述する。これに

より、理論的には無限にノードの階層化を行うことができる。また、仮想ノードを起動するための各種設定が必要であるが、仮想ノード実現技術によりその設定は異なるため、ここでは言及をさける。

3.2 ネットワーク

我々の提案する方式では、ネットワークをノードと同様に一つの要素として扱うこととする。各ノードのネットワークインターフェースを、ここで指定されたネットワークに接続することにより、ネットワークを形成する。

識別子 ネットワークの識別子。実験中はこの識別子で各ネットワークを判別する。

利用するアドレスレンジ アドレスレンジを設定し、接続したノードのネットワークインターフェースのIPアドレスの自動設定を行う。ノードのネットワークインターフェースにIPアドレスが静的に指定されていた場合は、ここで指定されたレンジ内のアドレスかどうかを判別することにより、設定ミスを検出するためにも利用できる。

種類 接続したノードのネットワークインターフェースの種類を確認するために利用する。

3.3 アクション

実験トポロジ構築後の実験シナリオ実行では、プロセスの実行を中心とした処理が必要となる。ここでは、プロセスの実行などをアクションと呼ぶこととする。実験の遂行にはアクションの制御が必須となる。各ノードの設定にアクションが必要である場合も多いため、実験トポロジの構築にもこれらの記述は利用される。

我々は、メッセージ交換によりノードの同期を実現しているため、メッセージ交換のためのアクションや、最低限の制御構文も必要である。

プロセスの実行 各ノードでのプロセスの実行
プロセスの終了 指定されたプロセスを終了させる
メッセージの送受信 これによりノードの同期や、ステータスの報告を行う

条件分岐 アクションの実行を制御するために、条件分岐が必要である。

ジャンプ 設定記述中の指定した位置に強制的に処理を移す必要がある。

数値演算 条件評価のために数値計算ができる必要がある。
文字列演算 我々の手法ではメッセージ交換によりノードの同期を行うため、文字列の比較などの処理が必要になる。

4 まとめ

実験を容易に行うためには、実験支援システムが必要である。我々は、これまでの SpringOS の開発経験をもと

に、実験を行うための実験環境および支援システムに必要とされる機能について議論を進めている。

実験を行うためには、ノード、スイッチそしてアクションの指定が必要になる。また実験遂行に関しては複数のノードの同期が必要となる。

本論文では、実験環境および支援システムに必要な最小限の機能および、設定記述について述べた。perl や ruby、python などの既存の言語は、広く利用されているため、その文法を習得している利用者は多いと考えられる。このような文法を用いて、実験の設定記述を行えば実験実行は容易になる。しかし、このような言語では、ノード同期の記述などが困難であるため、新たに実験遂行のための最低限の機能を持つ言語を定義する。利用者は既存の言語に近い文法で設定記述を行い、これを、我々が定義した言語に変換し、実験システムの入力とする。これにより、実験利用者が実験専用の文法を習得する必要はなくなる。

今後は、本論文で議論した機能の、具体的な実現方法について議論を進め、次世代 SpringOS へ導入する。

参考文献

- [1] The StarBED Project. [Online]. Available: <http://www.starbed.org/>
- [2] 宮地 利幸, 知念 賢一, 篠田 陽一, “StarBED における自動実験駆動機構,” in 第 6 回インターネットテクノロジーワークショップ (WIT2004) 論文集日本ソフトウェア科学会 研究会資料シリーズ No.36 ISSN1341-870X, 日本ソフトウェア科学会 インターネットテクノロジー研究会, Ishikawa, Japan, Dec. 2004, pp. 81–88.
- [3] Toshiyuki Miyachi, Ken-ichi Chinen and Yoichi Shinoda, “Automatic Configuration and Execution of Internet Experiments On An Actual Node-based Testbed,” in *1st International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities(Tridentcom)*, Trento, Italy, Feb. 2005, pp. 274–282.
- [4] 宮地 利幸, 知念 賢一, 篠田 陽一, “Springos/vm: 大規模ネットワークテストベッドにおける仮想機械運用技術,” in 情報処理学会研究報告書 2005-OS-99 ISSN 0919-6072, May 2005, pp. 105–112.
- [5] Ken-ichi Chinen, Toshiyuki Miyachi and Yoichi Shinoda, “A Rendezvous in Network Experiment — Case Study of Kuroyuri,” in *International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities(Tridentcom)*, Barcelona, Spain, Mar. 2006.
- [6] VMware. VMware. [Online]. Available: <http://www.vmware.com/>
- [7] University of Cambridge Computer Laboratory. Xen virtual machine monitor. [Online]. Available: <http://www.cl.cam.ac.uk/Research/SRG/netos/xen/>
- [8] Intel Corporation, *Preboot Execution Environment (PXE) Specification Version 2.1*, Sep 1990.
- [9] K. Sollins, “The TFTP Protocol (Revision 2), RFC1350,” July 1992.