修 士 論 文

ネットワーク実験環境の保存と復元に関する研究

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

野中 雄太

2008 年 3 月

修 士 論 文

ネットワーク実験環境の保存と復元に関する研究

指導教官　篠田陽一 教授

審査委員主査　篠田陽一 教授
審査委員　丹康雄 教授
審査委員　敷田幹文 准教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

610069 野中 雄太

提出年月: 2008 年 2 月

概 要

本論文では、ネットワーク実験設備上に構築された実験環境の保存と復元のシステムの提案・開発と、提案システムと既存の実験支援システムを組み合わせた技術開発を支援する手法の提案を行う。本論文での実験環境とは、技術検証のために実機を用いて構築されたネットワーク実験環境のことを言う。

実験環境の保存・復元の機能は、ネットワーク技術開発のサイクルや開発途中に行う実験の特徴からその重要さが見出せる。まず、実験は技術の不具合を発見し、その不具合を解消するために行われる。不具合を発見するためには、運用予定の環境に近い実験環境で実験を行うことが好ましい。この実験環境の構築には計算機へのソフトウェアの導入やネットワークの設定等を行う必要がある。実験環境の構築のためのコストは運用予定の環境が大きいほど大きくなる。そして、構築した実験環境で実験を行うが、実験は一度で終わるわけではなく、不具合を修正して何度も繰り返されることになる。修正後の再実験には、前回の実験と同様の実験環境が必要となる。また、技術開発では、開発された技術を実用化した後に不具合が見つかる場合や、機能の追加を行う場合があり、そのような場合も変更された技術の検証のため、以前の開発で用いた実験環境を必要とする。しかし、前回の実験から期間が開いた場合や、実験設備を複数人で共用している場合には、実験時の実験環境が保存されない場合がある。実験環境が破壊された場合、以前と同様の実験環境を再度構築して実験を行うことになるが、手動での実験環境の構築には様々な作業が必要であり、これを再度行うことは効率的ではない。また、構築した実験環境を変更せずに保存するのは、コストの面から現実的ではない。

そこで、構築した実験環境を保存・復元する仕組みが必要とされる。この仕組みを実現することで、実験のたびに実験者が手動で実験環境の構築を行う必要がなくなり、実験者の作業を削減することができ、技術開発を円滑に行うことができる。

現在までに大規模なネットワーク実験を効率化するための様々な実験支援システムが提案されている。実験支援システムは様々な機器で構成された実験設備と実験設備での実験を支援するための実験支援ソフトウェアで構成されている。実験支援ソフトウェアは、実験者の作成した実験記述を利用して実験の実行を自動化する機能を主としており、実験環境の構築も可能である。しかし、実験支援ソフトウェアの機能を超える処理を行う場合、実験者が手動で作業を行う必要がある。また、実験支援ソフトウェアでは、手動で構築した実験環境は保存できない。提案システムでは、手動で構築した実験環境も実験支援ソフトウェアで構築した実験環境も保存できる。そのため、提案システムの実現は実験設備上での実験に対してもコストの削減につながる。また、実験支援ソフトウェアによる実験実行のために必要となる実験記述の作成には習熟が必要であり、実験記述の作成は実験者の負担になる。提案システムでは保存した実験環境の状態から、実験支援ソフトウェアの実験記述を出力することで、実験全体の再現や自動化が行える。

また、技術検証に際しては類似した実験が行われることも多いため、ある実験で用いた実験環境を別の実験に用いることができれば、実験者が実験環境を構築する作業を削減できる。このとき、保存した実験環境を柔軟に復元することで、様々な実験に対応することができる。例えば、ある実験環境の規模を拡大できれば、小規模な実験から大規模な実験への対応も可能になる。さらに、複数の実験環境を組み合わせて復元することで、様々な実験環境を生成できる。

本論文では実験環境の保存・復元を実現を目的としたシステムの検討・設計・実装を行った。また提案システムと実験実行を主とした既存の実験支援システムを接続しネットワーク技術検証全体を容易にする手法を実現した。

検討の結果、本研究では実験環境の保存・復元に必要な特徴をネットワーク、ソフトウェア、実験の履歴と定めた。そして、保存した実験環境の特徴を全て復元する等価復元の機能と、実験の特徴からいくつかの特徴を復元する柔軟復元の機能を実現した。また、実験環境の規模の変更や、合成、再利用に必要な機能を実装した。そして、提案システムの評価を行い、実装したシステムによって実験環境の保存・復元が可能であること、他の実験支援ソフトウェアと連携して実験を効率化できることを確認した。

目 次

第1章　はじめに

1.1　背景

1.1.1　技術検証と実験環境

現在、インターネットは人々の生活に欠かすことのできない社会基盤として認識されている。そのため、動作検証が十分でない技術のインターネットへの導入は、インターネットに重大な影響を引き起こす可能性があるため避けるべきである。したがって、ネットワーク技術を実際に導入する前には検証を行う必要があり、その検証方法はこれまで様々なものが提案・実装されてきた。しかしながら、技術検証はインターネットに導入される実装を用いて行う必要があるため、最終的に実機またはそれに準じた仕組みを用いて行う必要がある。本論文では実機とはインターネットを構成している機器のこととする。この検証に用いる環境は、インターネットなどのその技術が運用を予定している環境に近いことが求められる。そのため、大規模な環境での運用を想定した技術の検証を行うためには、多数の機器を用いた大規模な環境を必要とする。

一方で、大規模な環境での実験には困難な点が多くある。まず、大規模な環境を手動で構築するのには人的・時間的コストが大きく、人的ミスも避けることができない。さらに、検証とは技術の不具合を発見・修正することが目的であるため、一度の実験で終わるものではなく、技術の不具合の発見後は修正した技術を再び同じ環境に導入して何度も実験を行うことになる。しかし、検証が長期間に及ぶときなど、実験に用いる機器を独占することが困難になる場合がある。このように機器を使用できなくなったとき、構築した環境は壊されることになり、検証を再開するときにはもう一度環境を構築しなければならない。以上のような問題は、検証に用いる環境が大きいほど顕著になる。

本論文で、「検証」とは技術の実装が設計通りに動作するかどうかを確認することを示し、「実験」とは検証を行うための作業の集合を言う。また、技術検証のために実機を用いて構築された環境を、「実験環境」と呼ぶ。

1.1.2　実験支援システム

ネットワーク技術の検証を支援するため様々な実験設備が設置されており、実験設備上で動作し、実験を効率化するための実験支援ソフトウェアも開発されている。実験支援ソフトウェアは、実験者が実験環境の構成や実験の手順を記した実験記述を利用し実験を自

動的に行うことができるものや、独自のインタフェースを用いて実験を容易に行えるようにするものがある。本論文では、実験設備と実験支援ソフトウェアを組み合わせたものを実験支援システムと呼ぶ。

実験支援ソフトウェアを利用することには多くの利点がある。まず、実験の自動化ができるため、検証全体にかかるコストを削減できる。さらに、実験記述を再利用することで同様の実験を再度行うことが容易になる。また、実験記述の一部を変更することにより、類似の実験を容易に実行できる。

ただし、多くの実験支援ソフトウェアは検証の全ての手順を網羅するようには設計されていない。詳細は後述するが、例として、手動で構築した実験環境の保存・復元を実現していないことや、実験を自動化するためには実験支援ソフトウェア固有の実験記述を作成しなければならないことがあげられる。

1.2 目的

ネットワーク技術の開発のための検証には、実機を用いた実験環境で実験を行うのが好ましい。しかし、実験のための準備である実験環境の構築を手動で行うのは手間がかかり、ネットワーク技術の検証では、実験の繰り返しや中断・再開によって同じ実験環境を何度も必要とすることが多い。また、既存の実験支援システムでは、特定の書式の実験記述に基づいて実験環境の構築を行うことはできるが、手動で構築した実験環境の保存や、実験環境からの実験記述の作成は考慮していない。そこで、本研究では、実験設備上に構築された実験環境の実験者の意志による保存・復元を実現することで実験環境の構築を支援し、ネットワーク技術の検証・実験を容易にすることを目的とする。

また、実験環境を保存・復元する仕組みを発展させることにより、技術検証をより効率よく行えるシステムの開発を目指す。

第2章　実験環境の保存と復元

2.1　検証と実験

2.1.1　検証と実験の手順と現状

宮地らはネットワーク技術の開発について、図 2.1 のように論理的検証と実験環境、大規模な実験環境を用いた検証を組み合わせた手順の提案を行った [1]。宮地らの提案では、論理的検証によりアルゴリズムの検証後、小規模な実験環境を構築し実験を行い、大規模な大規模実験環境での検証を行うとしている。ここで言う小規模な実験環境とは、実験者が個人で所有できるような数台から数十台の計算機で構成された環境であり、大規模な実験環境とは数十台以上の計算機で構成され、かつ実験支援ソフトウェアが導入されている環境のことである。検証に大規模な実験環境を必要とする技術では、上記のような検証手順が必要とされる。

図 2.1 では、検証の各フェーズにおいて問題が発生した場合に、同じフェーズでの検証を繰り返している。これは、不具合が起こった実装や設定を修正して再実験を行うことを示す。

また、宮地らは実験設備での実験の実行手順の提案も行った [2]。それによると、実験設備に導入されている支援ソフトウェアを用いない場合、実験の実行手順は以下のようになる。本論文において、シナリオの実行とは、構築された実験環境の上で評価のための手順でソフトウェアを動作させることを指す。

1. 実験とトポロジやシナリオの検討
2. 必要な実ノードなどの資源の用意
3. 各ノードの接続
4. ノードへのソフトウェアの導入
5. ネットワーク機器の設定
6. 構築した環境でのシナリオの実行
7. ログの解析

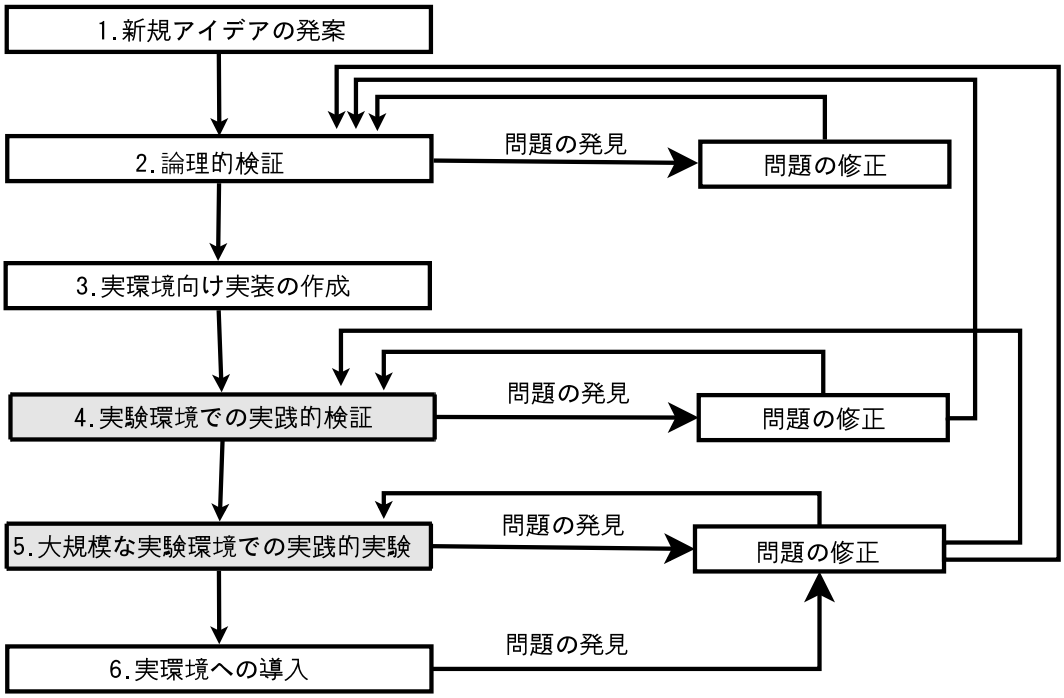


図 2.1: ネットワーク技術開発の手順 (文献 [1] より)

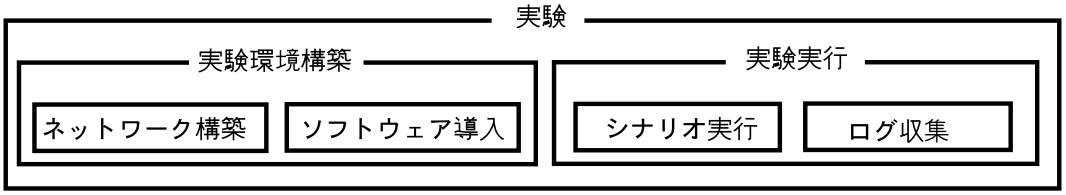


図 2.2: ネットワーク技術実験手順の分類

本論文では実験の実行手順のうち、2、3、4 を実験環境構築、5、6 を実験実行と呼ぶ (図 2.2)。実験の実行手順から、実験環境構築として様々な作業が必要になることがわかる。作業は検証に必要になる実験環境が大きいほど増大する。

以上のように、様々な検証を行い、目的とした環境に技術を導入する。しかし、技術開発はそれで終わりではなく、運用後に発生した不具合の修正や、機能の追加のために再度設計、実装、検証、運用のサイクルを繰り返すことになる。このことから、一般的技術開発は図 2.3 のように、技術の仕様を設計し、それの実装・評価を行い、実際に運用するという手順を繰り返すことになる。

このような技術開発の手順と実験の実行手順から、技術開発においては同様の検証が繰り返されること、規模の変更を行い類似した実験を行うことが多いという特徴を見出せる。

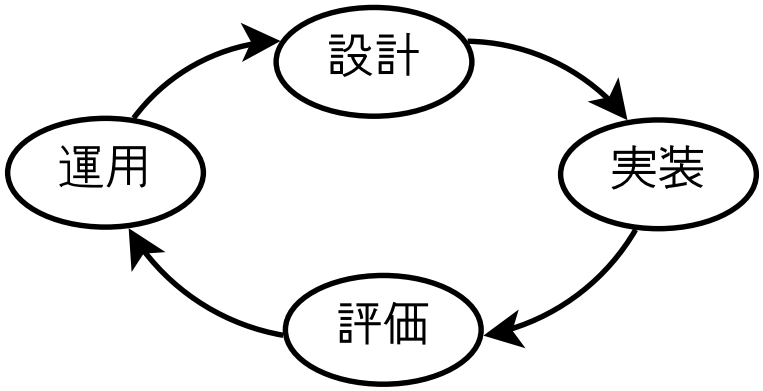


図 2.3: ネットワーク技術開発のサイクル

2.1.2 実験の特徴

技術開発手順と実験実行の手順から、実機を用いた実験の持つ特徴について述べる。

まず、前述の通り、技術開発の手順と実験の実行手順から、技術開発においては同様の検証が繰り返されることがあげられる。このとき、ある実験に用いた実験環境をそのまま用いれば実験環境構築の手間を省くことができる。しかし、前回の実験から時間が経る場合や、複数人で機器を共用している場合には、別の実験のための実験環境が構築されていることが考えられる。こういった場合には、再び実験環境を構築する必要がある。

また、特に大規模な実験を行う場合には、小規模な実験を行い実験対象の動作を確認した後に大規模な実験を行うことが多く、この場合小規模実験時の機器と同様な設定の機器を増加させることで大規模な実験を行うこととなることが多い。さらに、ある種のネットワーク技術の検証に用いる典型的な実験環境も存在する。このように、保存した実験環境と完全に同一ではないが、類似した実験も多く行われる。

実験環境を構成する機器を独占することができれば、実験環境を維持することで実験環境構築の作業を省けるが、大規模な実験環境を特定の実験のために維持し続けることは効率が悪い。そのため、実験環境を構成する機器は多人数で共用されることが多い。

次に、実験には一般的に使われている機器、ソフトウェアが多く使われる。例えばアプリケーションの検証をする場合には OS が必要になるなど、検証の対象となる新技術や機器以外にも様々な要素が必要となる。そのため、各実験で利用されるソフトウェアはある程度類型化できる。

これらから以下の特徴が見出せる。

1. 同一、または類似した実験を繰り返し行う
2. 実験環境を構成する機器は多人数で共用することが多い
3. 同一設定の機器を多数用いることが多い

4. 各実験で利用されるソフトウェアは共通部分がある

これらの特徴を考慮して、実験環境の保存・復元に求められる機能と、その機能を用いた実験の支援について検討する。

2.1.3 実験支援システムによる実験支援

技術開発の手順における小規模実験環境には実験支援ソフトウェアが存在しないことが多く、小規模実験が実験環境と技術の動作確認を兼ねることが多いため、小規模実験環境での実験は手動で行う場合が多い。これに対して大規模実験環境での実験は実験支援ソフトウェアを用いて自動化する必要がある。このように手動での実験と、実験支援ソフトウェアの実験記述を両方行うことは実験者には負担となる。

また、実験支援システムを用いた実験も完全に自動化できるわけではない。実験支援ソフトウェアで指定できる実験環境構築の粒度には限界があることや、実験記述の作成には習熟が必要なことから、大規模実験環境の構築もある程度まで手動で行い、実験環境構築の一部と実験実行のみを実験支援ソフトウェアで自動化することがある。しかし、既存の実験支援ソフトウェアには、手動で構築した実験環境を保存する機能は存在しない。実験は修正を行いながら繰り返されるため、このように実験を完全に自動化できていない場合には、実験環境が破壊された場合に何度も同じ作業を繰り返さなければならなくなる。

以上のように、既存の実験支援システムは一度の実験を支援する機能を持ってはいるが、技術検証のサイクル全体を支援するには至っていないのが現状である。

2.2 求められる機能

前述した実験の特徴から、実験環境の保存・復元に関して以下の機能が必要である。

1. 実験環境の保存・復元
   (a) 全ての属性の保存・復元
   (b) 限定した属性の復元
      i. 実験環境の等価な復元
      ii. 実験環境の柔軟な復元 (代替資源の利用、実験環境の部分的な復元、実験環境の自動選択)
2. 技術検証全体の支援 (実験環境の拡大、実験環境の合成、他の実験支援システムとの連携、実験履歴の管理)

1 はこれまで議論してきた実験が繰り返されること、既存の実験支援ソフトウェアでは実現されていないことから必要とされる。この機能によって、同じ実験環境での実験が繰

り返されると予測される時点の実験環境を保存しておけば、再び実験を行う場合に実験環境を構築するコストを削減できる。また、多人数で実験設備を利用する場合には、中断と再開という観点からも必要である。本論文では、実験環境の保存・復元を以下の 2 つの機能に分割して考える。

(a) は実験環境を構成する全ての要素を保存し、それを少しの変更もなく復元することである。しかし、多数の機器で構成された実験環境を考えた場合、機器同士を接続しているケーブルを流れる信号を保存することは困難である。また、機器の外部の条件、例えば実験設備の気温や、機器の温度による性能の変化を保存・復元することも困難である。このように、全ての属性を保存・復元することは理想的であるが、極めて困難である。

次に、(b) の保存・復元が考えられる。前述の通り、実験環境の全ての要素の保存・復元は困難であるが、実験者が必要とする特性を復元することは可能である。i の実験環境の等価な復元とは、実験者が必要とする実験環境の特性を最大限変更せずに復元することである。これは実験環境の保存・復元を考えたときにまず実現されるべき機能である。実験者の必要とする実験環境の特性については後述する。

ii は多人数での実験設備を共用する場合、実験設備を構成する機器が故障した場合を考慮したときに必要となる機能である。多人数での共用を前提とした実験設備上で実験環境の保存・復元を行う場合、保存時の実験環境で使用していた機器等の共用の資源が、復元時には他の実験者に使用されている場合が考えられる。このような場合、保存時とは異なった資源を用いて保存時と同様の実験環境を復元することができれば、実験者が実験環境を復元できる可能性が高まり、実験設備を有効に利用することができる。さらに、多数の実機を用いた実験設備では機器が故障する頻度も高くなるが、保存時の機器と異なった機器で復元できれば、故障時にも対応できる。

2 はある実験の実験環境を保存し同じ実験を再度行う、という一つの実験を支援するだけでなく、ある実験環境を別の実験への利用や、実験の全てのフェーズを支援するための機能である。

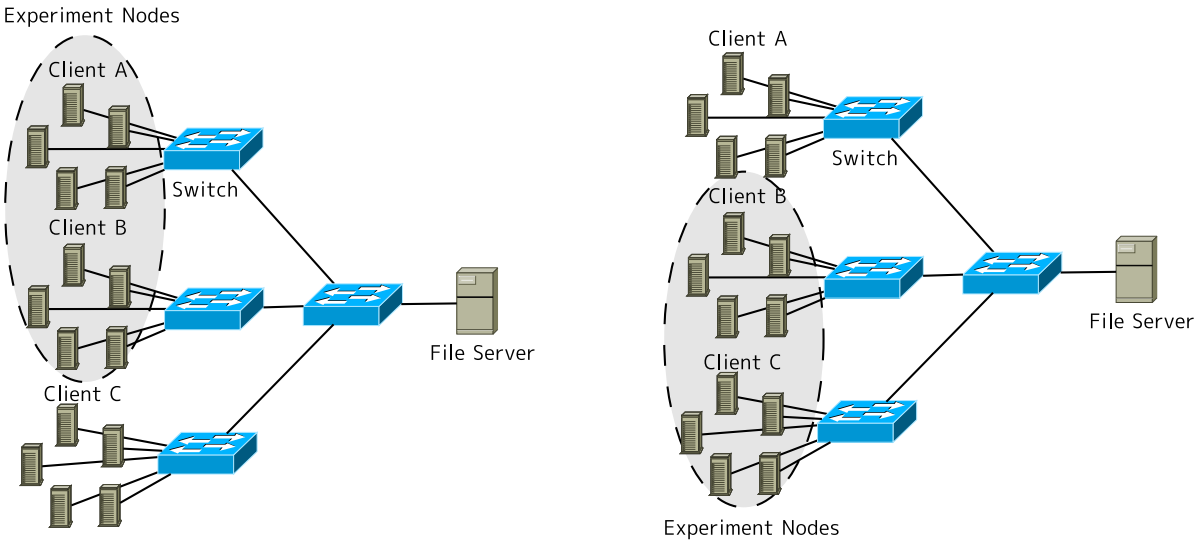
以降、実験環境の等価な復元、柔軟な復元と、技術検証全体の支援の機能について議論する。

2.2.1 実験環境の等価な復元

本稿では実験環境の等価な復元に必要な実験環境を構成する要素と、実験者が保存・復元を必要とする特性について議論する。

まず、本論文で対象とする実験設備は、実機によって構成されていることを前提としている。また、実機には様々なソフトウェアが導入されている。そして、実機同士が物理的・論理的に接続されることでネットワークが構築されている。以降、実験環境を構成する要素は、ハードウェア、ソフトウェア、ネットワークの 3 つに分類する。

ハードウェアには計算機、スイッチ・ルータ等のネットワーク機器など様々な種類があり、その仕様も様々である。そのため、構築した実験環境と完全に等価な実験環境を復元



(a) 保存時の実験環境　　(b) 復元時の実験環境

図 2.4: 柔軟な復元の例

するためには、保存時に使用していた機器と同じ仕様の機器を使用する必要がある。

次にソフトウェアであるが、これには機器に導入されているソフトウェアの種類やバージョン、ファイル構成などの静的な要素と、プロセスの状態や CPU・メモリの使用状況などの動的な要素がある。ソフトウェアを保存時と等価に復元するためには、これらを復元する必要がある。

最後にネットワークであるが、これには機器同士の物理的な接続と、機器のネットワークソフトウェアの設定によって構築される論理的な接続がある。そして、各機器のソフトウェアによって送出されるネットワーク上のトラフィックも、ネットワークの要素として考えられる。

以上のように、実験環境の等価な復元のためには様々な要素を考慮し、それらを保存・復元しなければならない。

2.2.2　実験環境の柔軟な復元

実験環境の等価な復元に加えて、実験設備の有効利用や、実験時間の短縮を考慮した場合に実験環境の柔軟な復元が求められる。柔軟な復元とは、保存時とは違う要素を用いて実験者の望む実験環境の特性を復元することとする。これには、代替の資源を用いた復元や、実験環境の部分的な復元がある。

図 2.4 は、代替機器群を用いて実験環境を復元した例である。この例では保存時の環境

の (図 2.4(a)) のクライアント A 群が利用不可能な場合に図 2.4(b) のようにクライアント C 群を代替として復元している。代替の資源としては、計算機の他にネットワーク機器や VLAN、IP アドレスといった論理的なものがあげられる。

また、保存した実験環境を部分的に復元できれば、保存した実験環境の他の実験への再利用が容易になる。例えば、実験環境のネットワークトポロジとソフトウェア構成を保存したとして、復元時にはネットワークトポロジだけを復元することが考えられ、この場合、ネットワークトポロジとソフトウェア構成の両方を復元する場合よりも短時間で復元可能になると予想される。

2.2.3 技術検証全体の支援

実験環境の等価な復元と柔軟な復元によって、実験の中断と再開や、実験の繰り返しに対応することは可能となる。これらに加えて、構築した小規模な実験環境を大規模に拡大する機能や、他の実験支援システムと連携する機能、別の実験に用いた実験環境を他の実験に利用する機能があれば実験に必要なコストをさらに削減することができる。

まず、構築した小規模な実験環境を大規模に拡大する機能について説明する。技術検証では、小規模な実験環境を構築し実験を行ってから大規模な実験を行うという特徴があった。このとき、小規模な実験で保存した実験環境を多数の機器での実験環境に拡大することができれば、小規模な実験から大規模な実験へ移行する場合の作業を削減することができる。

次に、他の実験支援システムとの連携について述べる。既存の実験支援システムの詳細については後述するが、実機を対象とした実験支援システムは手動で構築した実験環境の保存・復元を実現してはいない。本論文の対象は、実機で構成された実験環境の保存と復元であり、どのような手法で構築された実験環境であるかに依存しない保存・復元システムを提案・実装する。実験環境の構築の手法に依存しないということは、言い換えれば、手動で構築したものでも既存の実験支援システムで構築したものであっても保存・復元できる、ということになる。さらに、提案システムを用いて保存した実験環境から実験支援ソフトウェアの実験記述を生成することで実験の自動化も可能である。

既存の実験支援システムには実機で構成された実験設備を利用するものと、実機の構成に依存せずにソフトウェア単独で動作するものがある。ただし、既存の多くのソフトウェアも実機上で動作していることに変わりはないため、支援ソフトウェアを利用した実験環境も、実験環境全体を保存・復元することができる。

次に、実験環境の再利用や合成である。実験を行うたびに実験環境を保存していくと、実験環境が集積されることになる。過去に検証した技術と類似した技術を検証する場合には、実験に用いる実験環境も類似したものになる。そのため、これらの過去の実験をまとめ、目的の実験に合った実験環境を選択することができれば、実験環境を手動で構築する必要がなくなる。また、過去の実験環境のうちの単一の実験環境で目的の実験環境を構築できない場合に、保存された実験環境を複数利用することで構築できる場合もある。ま

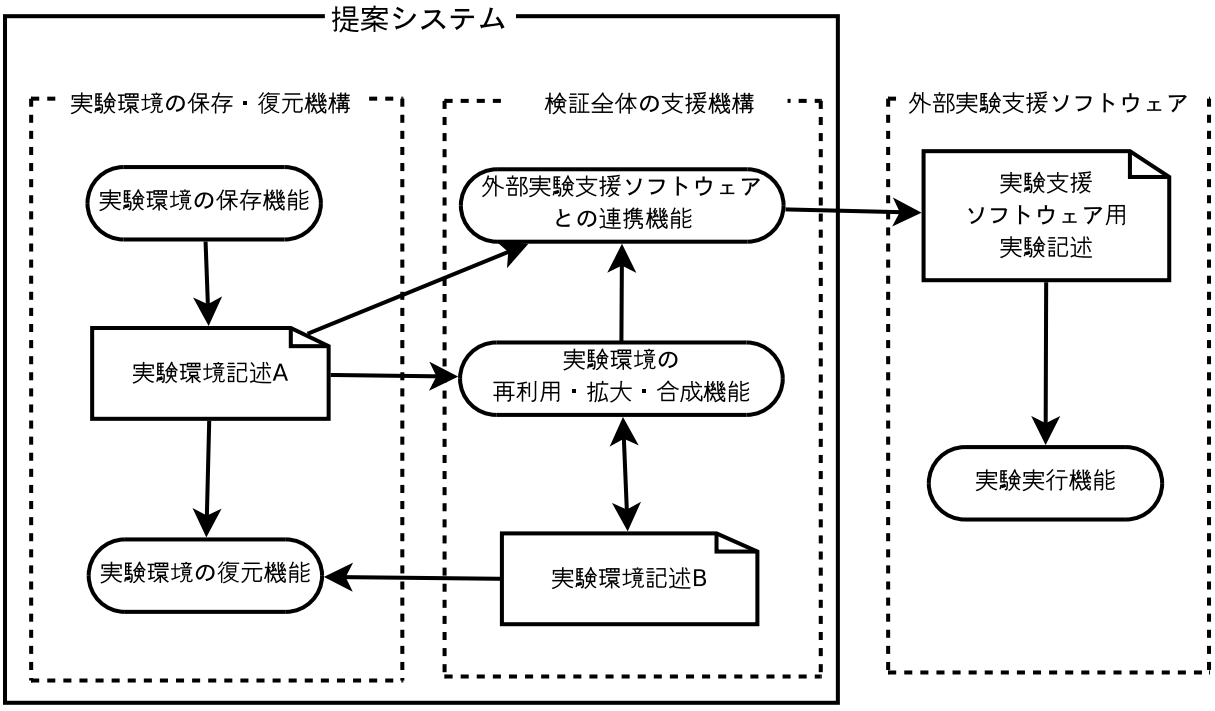


図 2.5: 提案システムの全体像

た、保存した実験環境を拡大・縮小して復元することができれば、規模を変更した実験も容易となる。

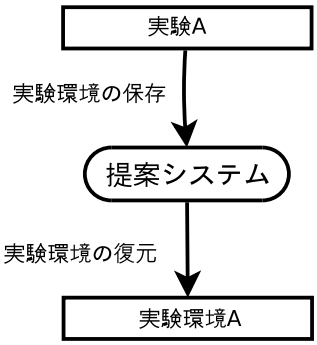
そして、実験環境を復元した場合に、保存時にどのような実験が行われたかを知ることができれば、実験の再開が容易になる。これは、実験環境の保存から復元までの時間が開いている場合や、大規模な実験環境を保存・復元した場合に、実験環境の構成の把握を助けることができるからである。

2.3 本研究の全体像と貢献

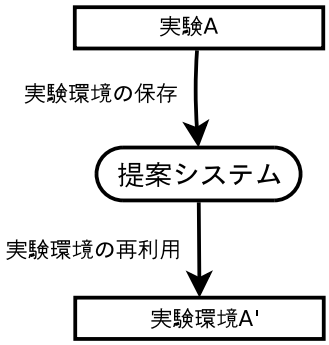
図 2.5 は提案システムの機能と全体像を示したものである。図 2.5 に記された機構のうち、実験環境の保存・復元機構と検証全体の支援機構は本論文で設計・実装し、外部実験支援ソフトウェアは既存のものを用いる。実験環境の保存・復元機構では、実験環境を実験環境記述として保存し、それを利用して実験環境の復元を行う。

検証全体の支援機構では、ある実験環境記述から外部実験支援ソフトウェアの実験記述の作成や、他の実験環境実験記述の作成を行い、検証全体の支援を行う。本論文では実験実行の自動化については対象としないが、外部の実験支援システムへのインタフェースを実現することで、他の実験支援システムとのシームレスな連携を実現する。

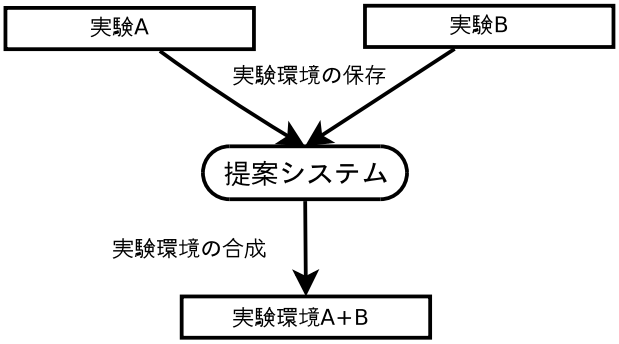
以上の機能により、実験環境の保存・復元をはじめとし、小規模実験から大規模実験へ



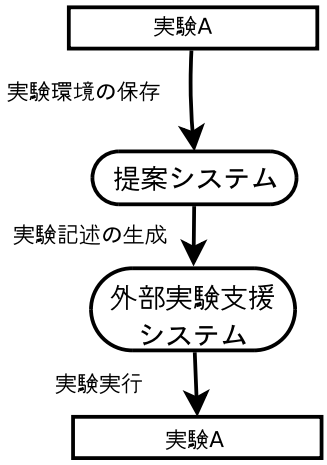
(a) 実験環境の保存・復元



(b) 実験環境の再利用



(c) 実験環境の合成



(d) 外部ソフトウェアとの連携

図 2.6: 提案システムを用いた検証

の移行や、保存した実験環境の類似の実験への転用、保存した複数の実験環境を組み合わせた実験環境を生成して実験を行うことも可能になる。また、復元した実験環境を用いた実験実行の自動化も実現できる (図 2.6)。これらの実現により、ネットワーク技術開発をトータルに支援するシステムを実現する。

図 2.7 は実験支援システム及び提案システムの位置づけを示したものである。横軸は実験の手順の支援に関して、縦軸は実験の繰り返しや再利用等の技術開発全体に関しての位置を表す。既存の実験支援システムは実験実行の支援を主とし、技術開発を通しての実験環境の変化に対応する機能は実現されていない。提案システムでは、実験環境の保存と復元により実験環境構築を支援し、かつ様々な実験環境で何度も行われる実験のサイクル全体を管理する。

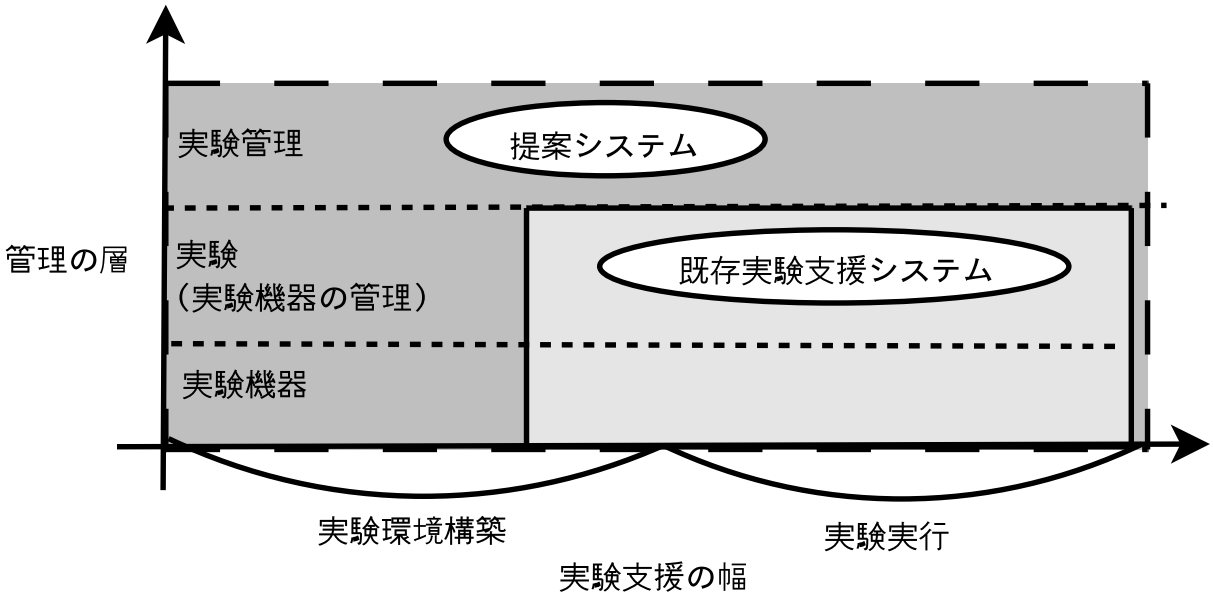


図 2.7: 提案システムの位置づけ

提案システムによって、ネットワーク技術検証のための実践的検証の手順は以下のようになる。

1. 小規模実験
   - ソフトウェアの導入
   - ネットワーク機器の設定
   - シナリオの実行
   - ログの解析
2. 実験環境の保存・復元
3. 大規模実験
   - シナリオの実行 (手動または自動)
   - ログの解析

本論文で提案するシステムには、実験者に以下の利点がある。

1. 実験設備の稼働率の上昇
2. 実験の実行時間の削減

3. 実験者の拘束時間、実験工数の削減

1は、保存時に利用していた機器以外の代替の機器を用いて保存時と等価な実験環境の復元を可能にすることによって実現する。実験環境を構成する要素の復元の取捨選択を可能にし、実験者の必要とする要素の復元に必要十分な機器を選択して実験環境を復元することで、実験設備内の機器を有効利用し、実験設備の稼働率を上昇させる。

2は、これまで手動、あるいは手動で実験環境を構築して動作を確認した後で実験支援システムの実験記述を作成して行っていた実験の準備段階を、構築の手法に依存せずに実験環境の保存・復元することで実現する。また、過去の実験に用いた実験環境をアーカイブしておき、その中から対象の実験に対し適当な実験環境を選択・復元することで実現する。

3は、提案システムによって実験手順を削減することによって実現する。実験手順の削減は、実験者の行う作業を減少させ、同時に拘束時間を削減することになる。

第3章　関連技術

ネットワーク技術の検証を行うために今日までに様々な手法や実験設備が提案・構築されてきた。その中で、ソフトウェアシミュレータにはNS-2が、VMware[4]を用いた仮想マシンを利用するものにVM Nebula[5]があり、多数の実機を用いるものにEmulab[6]、StarBED[7, 8]、GARIT[9]がある。

提案システムでは、これらの技術と連携することで技術開発全体を支援することを目指す。

NS-2

NS-2は実験に使用するネットワークトポロジや実験の手順を記述することによって自動的に実験を行うことのできるソフトウェアシミュレータである。これには最低1台の計算機があれば実験を行え、実験環境の構築も設定を記述することで自動的に行えるという利点がある。しかし、実機上で動く実装を利用することができず、実際に運用予定の実装に対する試験を行うことができない。そのため、NS-2を用いた検証だけでは検証の現実性を確保できない。

VMware

VMwareはWindows、Linux上でx86/x64アーキテクチャをエミュレートした仮想マシンを動作させることのできるソフトウェアである。これを用いることで少数の計算機で多数の計算機を必要とする検証を行うことが可能である。さらに、VMwareには多数の仮想マシンを一元的に管理する機能があり、仮想マシンを保存・復元する機能も持つ。しかし、1台の計算機に導入できる仮想マシンの数には限界があるため、ある程度以上大規模な検証は複数の計算機を利用することになる。そのとき、計算機同士を接続するネットワーク機器の設定を保存する機能はVMwareにはない。

VM Nebula

VM Nebulaはネットワークセキュリティに関する実験を行うことを目的とし、VMwareを利用して作成した仮想マシンを用いて構築した実験環境上で検証を行う実験環境であ

る。VM Nebula では、VMware の持つ仮想マシンを保存する機能を利用し、ワームやウイルスによって変更されたソフトウェアやネットワーク機器の設定を、保存した任意の時点の環境に戻すことができる。VM Nebula は特定の目的の実験への対応を目的としているため、本論文で対象とした汎用的な実験環境とはスコープが異なっている。また、複数の実験者で共用することは考慮されていない。現在は、VM Nebula を拡張し、多人数での利用等を考慮した VM Nebula2[10] の開発が進められている。

Emulab

Emulab は NS-2 と実機を利用した実験設備である。Emulab では実験者がソフトウェアを用いてネットワークの物理的・論理的トポロジを変更することが可能である。Emulab では実験者の要請を受けた実験を、実験機器が空き次第行う。このとき、割り当てられた利用時間を過ぎた実験環境を保存し、次の利用時に実験環境の設定を復元する。ただし、Emulab は独自のインタフェースで実験環境の構築と実行を行うため、Emulab を用いて構築した実験環境のみを対象としている。それに対して、本論文での提案した機構では、構築の手法に関係なく実験環境を保存・復元可能である。

Frisbee[11] は Emulab で使われているディスクイメージの保存機構であり、取得したディスクイメージをマルチキャストを用いて複数の計算機に配布する機能を有している。

GARIT/AnyBed

GARIT は奈良先端科学技術大学院大学を中心に開発されている、仕様が均一でない多数の機器で構成された実験設備である。AnyBed は GARIT のような様々な機器のある実験設備での実験を支援するためのソフトウェアである。AnyBed では実験トポロジの構築を行うことができるが、実験実行の自動化は対象とされていない。

StarBED,SpringOS

StarBED は一ヶ所に集められた計算機群とスイッチで構成された実験設備であり、実験者は VLAN を利用してネットワークトポロジを構築して実験環境を構築する。StarBED には実験支援ソフトウェア群として SpringOS が用意されている。これを利用することで実験者が実験に必要なネットワークトポロジや実験手順を記述することで自動的に実験を行える。ただし、現状では手動で構築した実験環境を保存・復元する機能はない。StarBED の特徴は、ネットワークスイッチと計算機のみで構成された実験設備であることと、その実験支援ソフトウェアがオープンなことである。そのため、同様の設備を個人や組織で構築し、利用することが可能である。本論文の提案する機構もそのような環境を保存・復元することを目的としている。そのため、提案システムと SpringOS を組み合わせることによって、実験の自動遂行から実験環境の保存・復元までの機能を備えた実験設備を構築で

きると考える。

NSS[12] は SpringOS の一部であり、実験設備上のソフトウェア配布を支援するために開発されたソフトウェアである。NSS では計算機のソフトウェア構成をディスクイメージとして取得し、それの多数の計算機に配布することができる。ただし、NSS はネットワークの保存・復元は対象としておらず、多数の計算機で構築された実験環境全体の保存と復元を行うことはできない。

第4章　設計

4.1　保存・復元対象の要素と必要な機能

2.2.1 節で、実験環境の要素をハードウェア、ソフトウェア、ネットワークに分類した。今後、ハードウェアの構成 (仕様) とハードウェア同士の物理的な接続を「物理トポロジ」、物理トポロジ上の機器のソフトウェア設定による機器同士の論理的な接続を「論理トポロジ」と呼ぶ。

まず、ハードウェアとして、計算機、スイッチ・ルータといったネットワーク機器、それらを物理的に接続する各種ケーブルが存在する。これらのハードウェア同士が物理的に接続されることによって、実験設備のネットワークの物理トポロジが構成されている。次に、計算機やネットワーク機器の設定によって構成される論理トポロジがある。これには、スイッチの VLAN によるトポロジ等がある。さらに、各計算機のソフトウェアがあり、この要素には計算機上のソフトウェアの構成やプロセスの状態等がある。

これら 3 つを保存・復元することで、実験環境全体の保存・復元が可能となる。また、論理トポロジとソフトウェアを保存・復元する仕組みを分割して実装することによって、ネットワークトポロジだけの復元といった部分的な実験環境の復元が可能になると考えられる。

また、前述した実験環境構築の結果を保存・復元するだけでなく、実験の履歴の管理も重要である。実験が進むにつれてネットワーク構築やソフトウェア導入等の実験者の作業は増加し、実験環境の初期状態からの変化も大きくなる。実験が中断・再開されるとき、中断した時点の状況が不明であると、どこまで実験を進めたのかがわからなくなってしまう場合がある。そのため、実験を保存した際、どのような手順で実験を進めていたか、どの段階で中断したのかを明確に知ることが必要となる。実験の履歴を管理することは、実験環境の再利用や他の実験支援システムとの連携にも有用である。

4.2　実験環境の保存・復元手法

4.2.1　ネットワークの保存・復元手法の検討

本節では物理トポロジ、論理トポロジで構成されるネットワークの特徴について議論する。

まず、実験環境構築するために物理トポロジを構築する必要があるが、これは手動で配

線作業を行う必要がある。この作業についてはソフトウェアで自動化することは困難である。

その一方、論理トポロジについてはソフトウェアで変更を行え、多くの共通のプロトコルで運用されているという特徴がある。つまり、ネットワーク機器のベンダや計算機のOSにより設定のためのコマンドや設定ファイルの書式が異なったとしても、それらの設定はTCP/IPをはじめとした共通のプロトコル群に準拠しているものも多く、それらに準拠した設定には互換性がある。

以上の特徴を考慮して、ネットワークの保存・復元手法を検討する。物理トポロジに関しては、Emulabのようにソフトウェアでパッチパネルを操作して構築することもできるが、ソフトウェアのみで変更することが困難であるため、保存時には保存時の物理トポロジの構成を取得・記録し、復元時には保存時の物理トポロジと復元時の物理トポロジを比較して利用する機器を選択することになる。そのための仕組みとして、実験設備上の物理トポロジの構成を手動で登録する仕組みや、自動で物理トポロジを把握する仕組みが必要となる。

論理トポロジを構成するための設定は、前述した通り機器のベンダや計算機のOSに関わらず同一の意味を持っているものが多い。各ベンダ等によって独自のプロトコルを用いる場合もあるが、一般的には共通のプロトコルで論理トポロジの構築は可能である。そのため、共通のプロトコルに準拠した論理トポロジに関する情報を機器のベンダやOSに依存しない情報に抽象化して保存し、復元時に論理トポロジの情報を利用する機器に合わせて設定することで、保存時と異なった機器を用いて、保存時と等価な実験環境を復元できる。また、論理トポロジは物理トポロジ上に構成されるため、論理トポロジを保存・復元するためには物理トポロジを把握しなければならない。そのための方法として、前述した物理トポロジを把握する仕組みを利用して取得することが考えられる。

ネットワーク機器の設定に関しては、telnetを用いて設定する方法や、各ベンダによって異なるネットワーク機器のインタフェースを統一化するNETCONF[13]を用いる方法がある。

4.2.2 ソフトウェアの保存・復元手法

ソフトウェアに関する要素は、導入されているOSやアプリケーションの構成などの静的な要素と、プロセスの動作状況などの動的な要素に分けられる。

まず、ソフトウェアの静的な要素の保存と復元の方法として以下が挙げられる。

1. 固定した構成のOSとアプリケーションを用いる方法
2. 計算機のハードディスクをディスクイメージとして保存・復元する方法
3. ファイルシステムとしてバックアップする方法
4. ファイル単位でバックアップする方法

実験の特徴として、OS 等の様々な実験で共通に使われるソフトウェアが存在することを述べた。固定した構成のソフトウェアとして、実験に多く使われる OS とソフトウェアをあらかじめハードディスクの特定のパーティションに用意しておけばソフトウェアの導入のコストを減少させることができるため、実験環境の構築時間も削減できる。

ディスクイメージとしてソフトウェア構成を保存する方法は、保存対象の OS やファイルシステムに依存せずに実験者が必要とするソフトウェア構成を再現できる。そのため、未知の OS やファイルシステムにも対応することが可能である。しかし、計算機によっては作成したディスクイメージを導入しても動作しない場合が考えられる。また、常にハードディスク上の保存対象のパーティションのデータ全てを書き出すため、保存・復元にかかる時間も大きくなる。

ファイルシステムとしてバックアップする方法は、差分バックアップや増分バックアップ等の手法を用いることで、ディスクイメージとして扱う方法に比べて高速にソフトウェア構成を保存・復元することが可能である。さらに、stackable filesystem と呼ばれるファイルシステムを用いることで、元のファイルシステムに変更を加えることなくファイルを追加したように扱うこともできる。stackable filesystem には unionfs[14] や aufs[15] がある。これにより、固定した構成のソフトウェアの変更点を効率的にバックアップすることもできる。しかし、OS によって利用できるファイルシステムが異なるため、実験に使われる各 OS のファイルシステムについてそれぞれバックアップ手法を実装しなければならない。

ファイル単位でバックアップする方法は、OS に依存せずに汎用的に用いることができる。また、ファイル単位でのバックアップを行う場合にも固定した構成のソフトウェアとの差分・増分をバックアップすることで、ディスクイメージを用いた場合よりも短時間での保存・復元が可能になると考えられる。

また、導入されているソフトウェアによって必要とする機器の仕様が異なるため、多人数での利用を考慮する場合には、復元時に用いる計算機上で保存時のソフトウェアが動作するかどうかを判定する必要がある。その方法として、物理トポロジを把握する際に計算機の仕様も同時に把握し、それを利用して対象ソフトウェアが動作するかどうかを判定する仕組みを利用することが考えられる。

ソフトウェアの動的な要素の保存・復元手法としては、数多く提案されているプロセスマイグレーションの方法を利用することが考えられる。ただし、それらには特殊な OS を必要とするなどの制約があるため、様々な OS に対応することが困難と考えられる。また、特殊な OS の導入は実験者の求める環境を構築する妨げになる場合もある。

4.2.3 実験履歴の管理手法

実験の再開を容易にするために、実験履歴を管理するための手法としては以下が挙げられる。

1. オペレーション履歴の保存

2. プロセス履歴の保存

3. 実験状況の概要を記述

1は、実験者が計算機に対して行ったオペレーションの履歴を保存する方法である。これによって、実験者が実行した実験の手順を保存することができる。ただし、計算機上では実験者が手動で実行したソフトウェア以外も動作している。この方法ではそれらの履歴は保存されない。

2は、実験者が実行したかどうかではなく、計算機で実行されたプロセスの履歴を全て保存する方法である。これによって、実験者が意識していない実験の履歴も保存できる。ただし、あまり重要ではない情報が保存される可能性があり、保存される情報が多大になることが考えられる。

3は保存時の概要を、文章等で保存する方法である。CVS 等ではプロジェクトを更新する際にその概要を記述する。実験の保存・復元に際しても、保存時の概要をメモしておけば復元時のリファレンスとして利用できる。ただし、実験に関しては各計算機に対する作業が多く、各計算機全てに対する作業の概要を記述するのはコストが高い。

4.2.4 検証全体の支援手法

まず、他の実験支援システムとの連携機能について述べる。実機を用いた実験設備を用いて実験を行う場合、何らかの方法で機器を設定する必要がある。そのため、既存の実験支援システムにおいても、実験設備内の物理トポロジや、機器の仕様などの情報を把握しており、それらの情報を利用して機器に指示を与え実験を行っている。

提案システムでも、保存・復元のために必要となる物理トポロジ等の情報を持つことになる。そのため、それらの情報を他の実験支援ソフトウェアの実験記述に出力することで実験支援システムとの連携を実現できる。また、プロセスの管理を行うことによって、プロセスの履歴から実験記述を生成することも可能になる。

次に、実験の再利用や複数の実験環境の合成であるが、これは保存した実験環境の中から適切なものを選択することと、それらを組み合わせて復元することで可能である。適切な実験環境を選択するために、実験環境がどのような実験に用いられたかなどの情報を管理する仕組みがあれば効率よく保存された実験環境を利用できる。

4.3 システム構成

4.3.1 実験設備の要件

本論文で対象とする実験設備は、図 4.1 のように、実験用ネットワークと管理用ネットワークが分かれているものとする。実験用ネットワークは実験者が検証に用いるためのネットワークであり、管理用ネットワークは保存・復元のために必要な通信を行うための

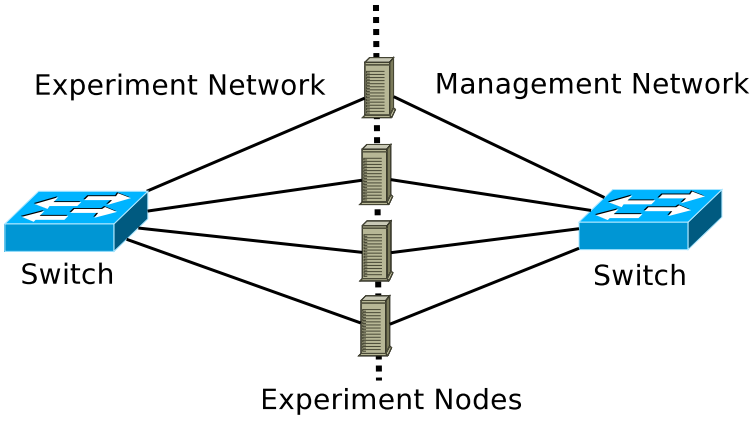


図 4.1: 実験用と管理用ネットワーク

ネットワークである。2 つのネットワークを分離したのは、検証に用いているネットワークに保存・復元のためのトラフィックが混ざることを防ぎ、検証の状況により到達性がなくなった計算機に対しても、保存・復元のための通信を行う必要があるという理由からである。

また、本論文で保存・復元の対象とする実験設備の機器は計算機と VLAN を扱えるネットワークスイッチのみとし、論理トポロジは VLAN を用いて仮想的に構築することとする。これは、検証に使われる実験環境の機器の多くは計算機であるため、これらの機器を用いた実験環境を保存・復元することで多くの技術の検証に対応できると考えたためである。例えば、計算機をルータとして動作させることによって、ルーティングを考慮した複雑な環境での検証にも対応できる。

設計した実験環境の保存・復元機構の概念図を図 4.2 に示す。まず、実験設備の物理トポロジや機器の仕様は資源管理機構 (Resource Manager) が把握する。また、資源管理機構では実験設備の使用状況も把握する。ネットワークの保存・復元機構 (Network Manager) は、実験者の指定した実験環境の物理トポロジを資源管理機構から取得し、物理トポロジにしたがって必要な機器の論理トポロジを保存・復元する。ソフトウェア保存・復元機構 (Software Manager) は実験者が指定した計算機のソフトウェア構成をハードディスクのパーティションのディスクイメージやファイル単位のバックアップとして保存・復元する。そして、各計算機にプロセス履歴を保存する仕組みを組み込んでおく。これにより、実験の履歴を保存する。

以下に、各機構の詳細について述べる。

4.3.2 資源管理機構

資源管理機構は実験設備で共用されている資源を把握し、ネットワーク保存・復元機構やソフトウェア保存・復元機構が実験環境を保存・復元する際に必要となる情報を与える

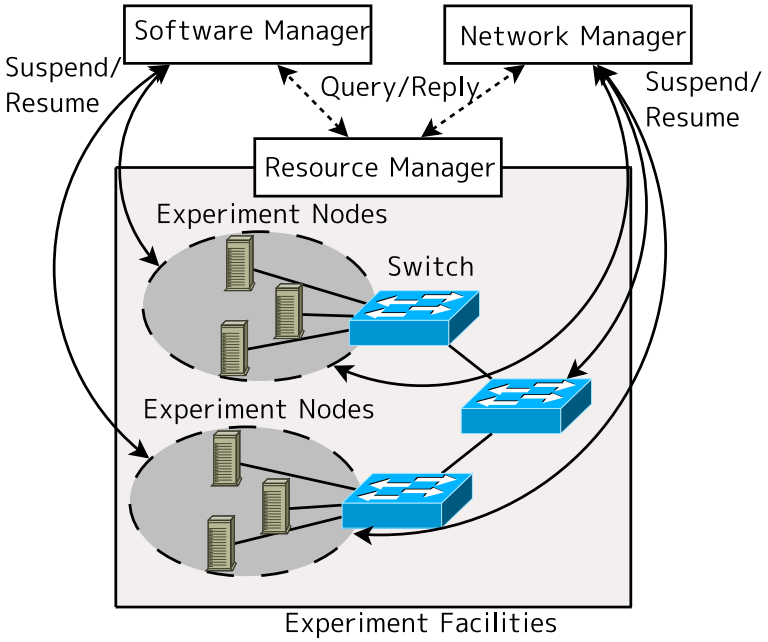


図 4.2: 提案手法概念図

機能を持つ。資源管理機構が把握する情報は以下の通りである。

- 物理トポロジ
- 機器仕様
- 機器識別子 (ホスト名)
- 資源利用状況

実験設備の物理トポロジを構成する要素は、計算機のネットワークインタフェース (以降 IF) とスイッチの IF との接続状態、スイッチの IF とスイッチの IF の接続状態、などの機器同士の接続状態である。資源管理機構は、各機器の各 IF がどの機器のどの IF に接続されているかを把握することで、実験設備全体の物理トポロジを把握する。また、保存時と復元時の実験環境のネットワークの同一性を保つためには、各機器間の帯域等のリンク特性も復元可能である必要がある。そのため、資源管理機構は各機器の IF の規格についても把握する。また、実験設備内の各機器を識別するためのホスト名も資源管理機構は保持する。

実験設備内の機器の仕様は、ソフトウェアを保存・復元する際に重要になる。ソフトウェアの動作に関連するものとして、CPU の種類・処理能力、メモリの種類・容量、チップセットの種類、ハードディスクの種類やディスク容量が挙げられる。CPU、メモリ、チップセットは、保存したソフトウェアが動作するかどうかと、動作速度に関連する。ハード

ディスク容量は保存したソフトウェアを導入できるかどうかに関連する。資源管理機構はこれらの保存・復元に必要な情報の把握も行う。

実験設備内の共用資源の利用状況は、多人数で実験設備を利用するために把握しておく。把握する情報としては、実験設備内の計算機の利用状況、実験設備内で利用されている VLAN 番号が挙げられる。

4.3.3 ネットワーク保存・復元機構

ネットワーク保存・復元機構は、実験環境のネットワークトポロジの保存・復元を行う。この機構は、実験環境の保存・復元時に資源管理機構と通信を行い、物理トポロジ等の情報を取得する。そして、保存・復元対象機器の設定情報の取得、対象機器の設定を行い、実験環境の論理トポロジを保存・復元する。論理トポロジを保存・復元するために必要な情報は以下の通りである。この情報と、資源管理機構が持つ情報を組み合わせることで物理トポロジと論理トポロジからなるネットワークトポロジを保存・復元できる。

- 計算機の IF の IP アドレス、経路情報
- 計算機の IF が接続されたスイッチの IF の VLAN 番号
- 計算機の IF の仕様

図 4.3 はネットワーク保存・復元機構の動作を示したものである。図 4.3(a) は保存時の、図 4.3(b) は復元時の動作を示す。

保存時、実験者はネットワークを保存したい実験環境上の計算機群をネットワーク保存・復元機構に指定する。計算機群を指定されたネットワーク保存・復元機構は、資源管理機構と通信を行い、計算機群とスイッチの名前とそのスイッチのポートとの対応に関する情報などの、物理トポロジの情報、計算機群の仕様の情報を取得する。その後、ネットワーク保存・復元機構は指定された計算機群と接続されているスイッチの IF の VLAN 番号を取得する。また、指定されたホスト名の計算機群の持つ経路情報と実験用 IF に設定された IP アドレスを計算機群から取得する。そして、計算機群のホスト名、実験用 IF の用途 (管理用ネットワークに接続された IF か、実験用ネットワークに接続された IF か)、各インタフェースの規格、各 IF に接続されたスイッチのホスト名、VLAN 番号、機器の仕様を記した実験環境記述 (Environment Description) を出力する。このとき、実験環境記述は、ネットワーク機器のベンダや OS に依存しない書式で出力される。

実験環境の復元時には実験環境記述を読み込み、それに基づいてスイッチ及び計算機群を設定し、ネットワークトポロジを復元する。実験設備のどの機器を用いてネットワークトポロジを復元するかは、保存時の機器での復元、実験者の指定した機器による復元、資源管理機構が割り当てた任意の機器による復元が考えられる。それらのネットワーク保存・復元機構はどの場合でも資源管理機構に使用権限を確認する。これは、他の実験者が使用中の機器の使用を避けるためである。また、保存時のネットワークトポロジをどこま

で厳密に復元するかは実験者が指定する。例えば、リンク特性が違ってもネットワークの形さえ同じであれば復元を許す、等の指定が考えられる。

この機構により、以下のようなパターンで実験環境のネットワークトポロジの保存と復元が可能となる。

1. 同構成での復元

   保存時と同じ計算機、IP アドレス、VLAN 番号を用いて復元する

2. 代替計算機を用いた復元

   実験環境を構成する計算機を指定して復元する

3. 代替 VLAN を用いた復元

   保存時に利用していた VLAN 番号が使用中である場合、別の VLAN 番号を用いて復元する

4. 代替リンク特性を用いた復元

   保存時とはリンク特性が異なるネットワークトポロジでの復元

4.3.4 ソフトウェア保存・復元機構

前述の議論よりソフトウェア構成を保存・復元する機構を設計した。なお、提案システムでのソフトウェアの保存・復元においては、プロセス状態の保存・復元は扱わない。これは、特殊な OS 等が必要となるため、実験環境の多様性を失わせる可能性があるためである。

まず、計算機のハードディスクを複数のパーティションに分割し、各パーティションを固定ソフトウェア用と実験者が任意に OS を導入できるパーティションに用いることにした。また、固定ソフトウェア用のパーティションには実験に必要とされるソフトウェアをあらかじめ導入しておく。あらかじめ導入しておくソフトウェア群を標準構成と呼び、必要なソフトウェアについては後述する。

これにより、標準構成のソフトウェアを利用する実験者にも、独自のソフトウェアを用いた実験を行いたい実験者にも対応できる。保存・復元方法としては、標準構成ソフトウェアはファイル単位で変更点をバックアップし、任意の OS 用のパーティションはディスクイメージとして保存・復元する。

図 4.4 に、ソフトウェア保存・復元機構 (Software Manager) の構成と動作を図示した。図 4.4(a) と図 4.4(b) はそれぞれ、ソフトウェア構成の保存・復元にディスクイメージとファイルを用いる場合の動作となる。

どちらの場合も、ネットワークを利用して計算機の起動パーティションを指定する。この機能を持つのが、Network Boot System である。これには Preboot eXecution Environment(PXE)[16] 等を利用する。またディスクイメージとしてソフトウェア構成を保存・復元する機構が図

中の Partition Agent であり、標準構成ソフトウェアのファイルシステムのバックアップとレストアを行うのが図中の Backup Agent である。

保存時、計算機のソフトウェア構成をファイル単位でバックアップする場合、Software Manager は Backup Agent を用いてファイルシステムの変更点をバックアップする。Backup Agent はあらかじめ標準構成ソフトウェア群に組み込んでおく。計算機のハードディスクのディスクイメージを保存する場合、Software Manager は Partition Agent を用い計算機のハードディスクの対象パーティションのバイナリデータを取得し、ファイルサーバに転送する。

復元時、保存した計算機のディスクイメージを復元する場合、Partition Agent はディスクイメージを保存してあるファイルサーバからディスクイメージを取得し、復元対象の計算機のハードディスクに書き込む。そして、ネットワークを利用して起動パーティションを指定して起動させる。その後、ファイル単位で保存・復元する場合には Backup Manager による復元が行われる。

図 4.5 は類似した実験を複数回行った場合の実験回数と構築時間についての一般的な検証を、構築を手動で行った場合、ソフトウェアをディスクイメージで復元した場合とファイル単位で復元した場合での予測を比較したものである。

手動で実験環境を構築する場合には、実験回数に対して構築時間はほぼ一定であると考えられる。ディスクイメージを利用する方法では、1 回目の構築は手動で行う必要があるため手動とほぼ同じ時間がかかるが、2 回目以降は提案システムを用いて復元することができるため手動構築時よりも短い時間で構築できる。また、ディスクイメージはほぼサイズが一定になるので、何度構築を行っても 2 回目にかかる時間とほぼ変化はないと考えられる。ファイル単位で実験環境を復元し、実験環境を構築する場合には、1 回目の構築から標準構成のソフトウェアが利用できるので他の方法と比べて一回目の構築時間は短い。2 回目の以降の構築では、差分ファイルの量によって構築時間が異なる。差分ファイルが極端に多い場合は、ディスクイメージを用いた構築よりも時間がかかることも考えられる。

ソフトウェアの標準構成の検討

標準構成のソフトウェア群は、実験を行うために必須であるものと、実験を容易に行うために導入が推奨されるものに分けられる。以下にそれらを示す。

- 必須
    1. OS
    2. Backup Manager
- 推奨
    1. ソフトウェア導入用ソフトウェア

2. 実験支援ソフトウェア
3. 実験に使用されることの多いソフトウェア

OS は計算機を利用するために必須である。ただし、OS の種類に関しては検討を要する。これは実験に用いる可能性が高いものを導入するべきである。Backup Manager は実験者が導入したソフトウェアの保存・復元するために必須となる。

また、実験に使用されることの多いソフトウェアを導入しておくことで、実験環境の構築に要する時間を短縮できる。ただし、実験によって使用されるソフトウェア全てをあらかじめ準備しておくことは困難であるので、ソフトウェア導入用のソフトウェアがあることが好ましい。これには、各 OS で用いられているパッケージ管理用のソフトウェアや、ネットワークを通してソフトウェアを得るための FTP クライアントなどが考えられる。そして、実験支援ソフトウェアを導入しておくことで、実験環境構築後の作業をより効率的に行えるようになる。

4.3.5 実験履歴管理機構

実験履歴を管理するための手法として、いくつかの手法を述べたが、本実装では各計算機のプロセスの履歴を保存する手法を利用することにする。これは、実験者によるオペレーションの履歴を保存する手法よりも情報量が多いため、必要な情報を保持できる可能性が高いことと、手動で実験の概要を記述するよりも実験者にかかる負担が小さいためである。

プロセス履歴を保存する機構は、ファイル単位でのソフトウェア構成の保存・復元対象となる標準ソフトウェア構成用のパーティションに導入しておく。そして、ソフトウェアの構成の保存時にプロセス履歴を保存する。

4.3.6 検証全体の支援

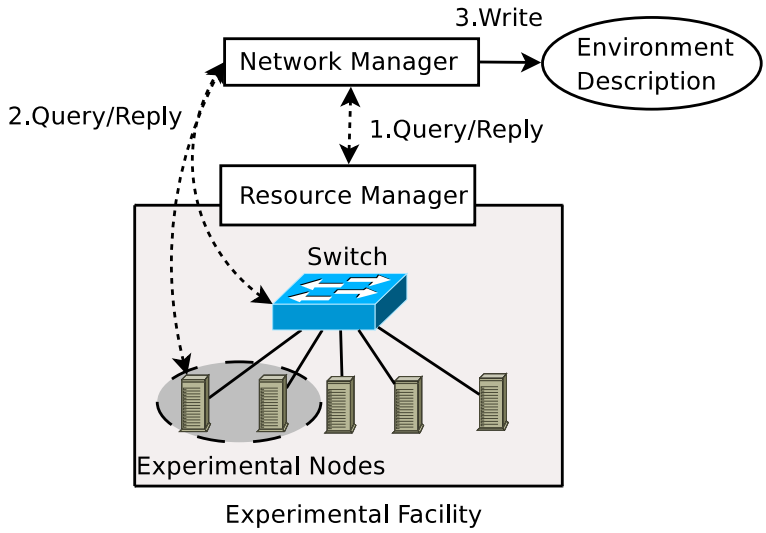
以上に設計した各機構が保有する情報を実験環境記述として保存することで実験環境の保存・復元を行う。この実験環境記述には、実験環境の保存・復元に必要な情報が含まれることになる。

そこで、資源管理機構の持つ物理トポロジと、ネットワーク、ソフトウェアの保存・復元機構の取得する論理トポロジ・ソフトウェア構成の情報の中から、実験実行用の支援システムの実験環境構築に必要な情報を選択し、対象の実験記述を生成することで実験支援システムとの連携が可能である。さらに、提案システムでは実験時のプロセス履歴も保存する。この情報を利用することで、実験記述の実験実行部の生成も可能であり、実験支援システムとの実験の工程においての連携が可能となる。

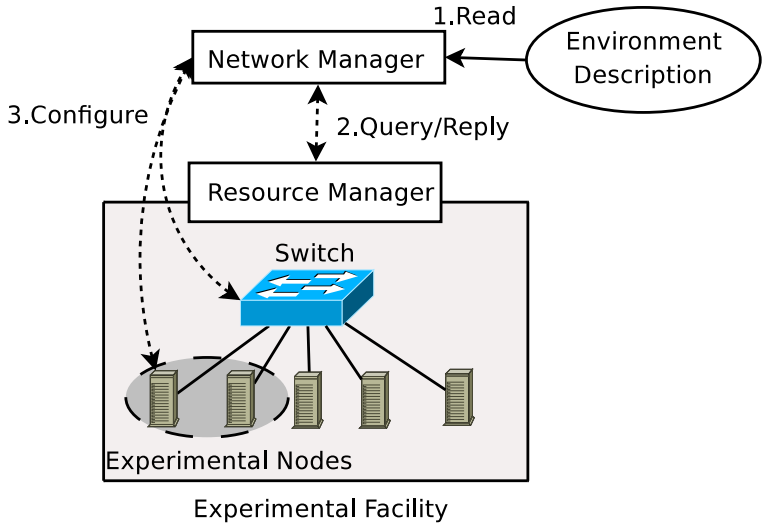
図 4.6 は手動で構築した実験環境と実験環境上での実験を、他の実験支援システムとの連携させるイメージを示している。手動 (あるいは実験支援ソフトウェア) で構築した実

験環境を、提案システムで保存し、保存時に生成される実験環境記述を元に実験支援ソフトウェアの実験記述を出力する。この出力を実験記述テンプレートと呼び、実験者はそれを利用して次の実験の実験記述を作成する。そして、作成した実験記述を用いて実験を実行することになる。加えて、実験の終了後にさらに実験環境を保存することによって、その実験を次の実験のための実験記述テンプレートとして出力することができる。このようにすることで、繰り返される実験を統合的に支援することが可能である。

また、複数の実験環境記述を利用して復元を行うことで実験環境の合成も可能である。

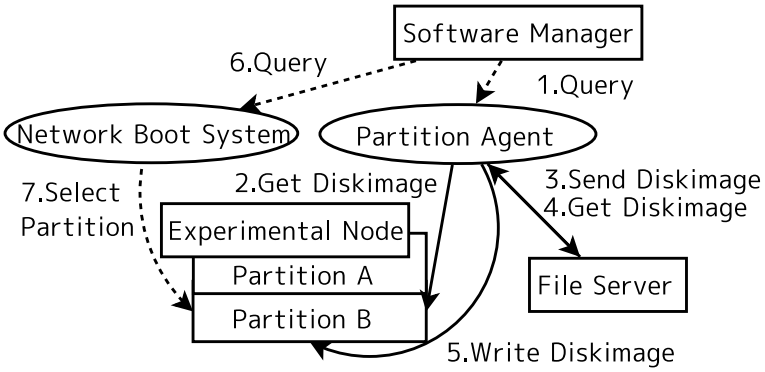


(a) 保存時の動作

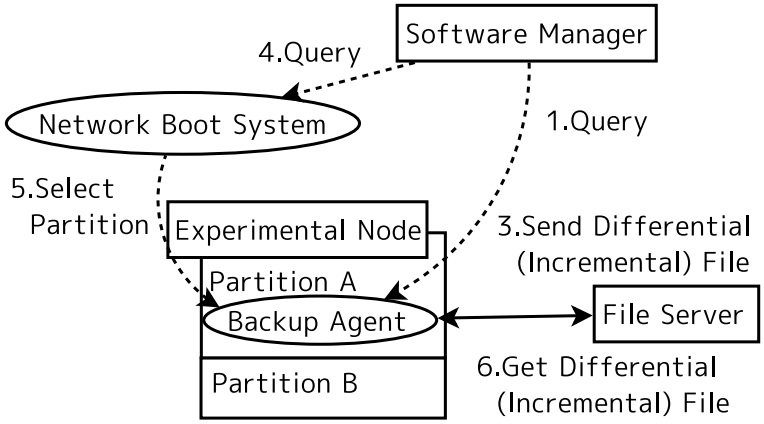


(b) 復元時の動作

図 4.3: ネットワーク保存・復元機構の動作



(a) ディスクイメージ



(b) ファイル単位

図 4.4: ソフトウェア保存・復元機構の動作

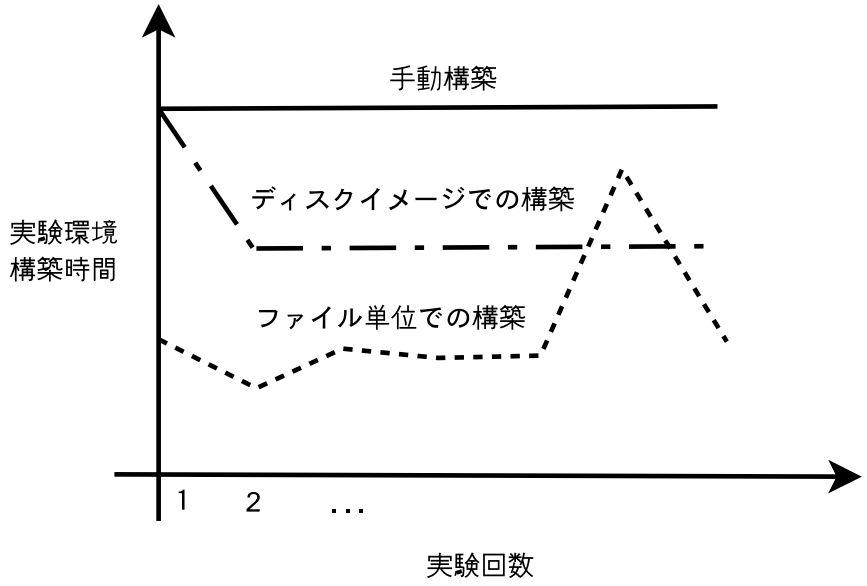


図 4.5: 実験回数と構築時間

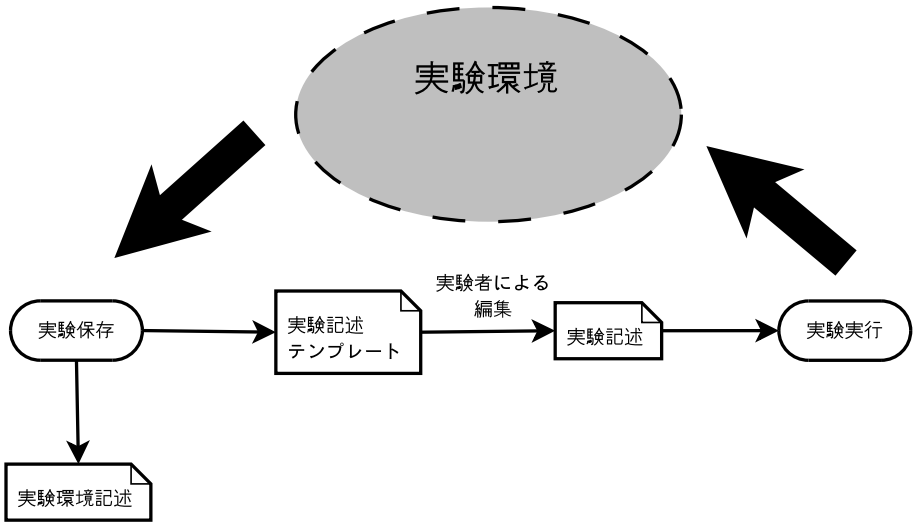


図 4.6: 実験支援システムとの連携による技術検証

第5章　実装

提案システムを情報通信研究機構 北陸リサーチセンターの実験施設である StarBED 上 [7, 8] に実装した。StarBED に導入したのは、StarBED が本論文で対象としている管理用ネットワークと実験用ネットワークが存在している実験設備であり、さらに多数の機器を用いた大規模な実験環境を構築できるためである。

5.1　資源管理機構

資源管理機構として、SpringOS の一部である experiment resource manager(ERM) を用いた。ERM は、StarBED 上の機器について、以下のような情報を保持する。

- 各機器のホスト名
- 計算機の IF の情報

  管理用 IF か実験用 IF か、規格、接続スイッチホスト名、接続スイッチ IF
- スイッチ間の接続状態
- スイッチのコマンド体系

5.2　ネットワーク保存・復元機構

Network Manager は実験者に指定された計算機の接続されているスイッチの設定を、ベンダに依存しない書式で実験環境記述として出力し、それを用いて復元を行う。なお、Network Manager は本研究で実装した。Network Manager は実験者に保存対象の計算機のホスト名を指定されると、ERM と協調して以下の動作を行う。

1. ERM と通信を行い、計算機の IF の情報、スイッチのコマンド体系を取得する
2. 計算機の IF の情報から、実験用 IF が接続されているスイッチにログインする
3. 計算機の IF が接続されているスイッチ IF の VLAN 番号を取得する
4. ERM から得た情報と、スイッチから得た VLAN 情報を基に、実験環境記述を出力する

保存時に、Network Manager はスイッチにログインして設定を取得する。スイッチの各ベンダのコマンド体系に合わせた手順でコマンドを入力し、取得した設定をベンダに依存しない書式に変更する。現在対応しているベンダとコマンド体系は Foundry Networks[17] の IronWare[18] と Cisco Systems[19] の Cisco IOS[20] である。

復元時の Network Manager の動作は以下の通りである。

1. ERM と通信を行い、計算機の IF の情報、スイッチのコマンド体系を取得する
2. 計算機の IF の情報から、実験用 IF が接続されているスイッチにログインする
3. 計算機の IF が接続されているスイッチ IF の VLAN 番号を設定する

Network Manager は、復元時に、実験者が復元対象の計算機を指定しなければ保存時の機器で復元を行い、復元対象の計算機が指定された場合はその計算機で復元する。また、保存時の VLAN 番号が他の実験者に使用されていた場合には、実験者からの指示により別の VLAN 番号を利用しての復元も可能である。このとき、利用する VLAN 番号は使用されていない VLAN 番号から自動的に決定される。

5.3 ソフトウェア保存・復元機構

Software Manager は実験者に指定された計算機のソフトウェア構成を保存・復元する。ソフトウェア構成の保存と復元はオプションとして指定でき、保存しないことも可能である。実験者はディスクイメージとして保存するか、ファイル単位で保存するかを指定する。

ディスクイメージとしてソフトウェアを保存・復元する Partition Agent には、SpringOS の一部である NI を利用した。ファイル単位で保存・復元する Backup Manager には、rsync アルゴリズム [21] を利用したファイルバックアップ用ソフトウェアである rdiff-backup[22] を用いた。また、計算機の起動パーティションを指定する Network Boot System には PXE を利用した。

ディスクイメージとして保存する場合の Software Manager の動作は以下の通りである。

1. Network Boot System を使用し、保存対象の計算機を Partition Agent が導入されたディスクイメージから起動
2. Partition Agent が対象パーティションのディスクイメージをファイルサーバに送信
3. 実験環境記述にディスクイメージのファイル名を記述
4. Network Boot System を使用し、保存対象の計算機の起動方法を最初の状態に戻す

ディスクイメージとして復元する場合の Software Manager の動作は以下の通りである。

1. Network Boot System を使用し、保存対象の計算機を Partition Agent が導入されたディスクイメージから起動
2. Partition Agent が実験環境記述で指定されたディスクイメージをファイルサーバから取得し、対象計算機に導入
3. Network Boot System を使用し、保存対象の計算機の起動方法を最初の状態に戻す

ファイル単位で保存する場合の Software Manager の動作は以下の通りである。なお、あらかじめ標準構成のソフトウェアを導入しておく必要がある。

1. 指定された計算機の Backup Agent が標準構成のファイル群との差分をファイルサーバに保存
2. 実験環境記述に差分ファイルの情報を記す

ファイル単位で復元する場合の Software Manager の動作は以下の通りである。

1. 指定された計算機の Backup Agent が現在のファイル群と保存されたファイル群との差分をファイルサーバから取得

また、一つの計算機のソフトウェア構成を複数の計算機に導入することも可能である。

5.3.1 ソフトウェア標準構成

ソフトウェアの標準構成として、以下を選択し導入した。

- OS:Linux(Fedora Core 5)
- Backup Manager:rdiff-backup とその他実装したスクリプト
- ファイル導入用ソフトウェア:yum
- 実験支援ソフトウェア:kuroyuri slave

OS は、ソフトウェア開発に適している Linux を選択した。Fedora は Linux ディストリビューションの一つである。Fedora を選択した理由は、ソフトウェア開発に必要なソフトウェアが多く導入されており、実験での使用頻度の高いソフトウェアがあらかじめ導入されていると考えられるためである。yum は Fedora が採用しているパッケージ管理ソフトウェアである。

また、各計算機のソフトウェアをファイル単位で保存・復元するために、rdiff-backup とそれを管理用の計算機から一括で管理するためのスクリプトを導入している。そして、実験実行を支援するための実験支援ソフトウェアとして、SpringOS の一部である kuroyuri slave を導入した。kuroyuri slave は、実験を管理する機能を持った SpringOS の kuroyuri master の指示を受けて各計算機での実験を自動実行する機能を持つ。

5.4 実験履歴保存機構

実験の履歴を管理するために、実験中のプロセスの履歴を保存する機構の実装について述べる。

プロセス履歴の取得の方法は OS によって異なる。本実装では標準構成のソフトウェア群の OS として Linux を採用したので、Linux のプロセスを取得するソフトウェアである psacct を用いた。

実験開始から実験終了までの実験履歴保存機構の維持機構の動作は以下の通りである。

1. 実験開始時に psacct を起動し、プロセス履歴の取得を開始
2. Software Manager での保存時に、プロセス履歴を記録したファイルをファイルサーバに送信
3. Software Manager での復元時には、プロセス履歴の取得を再開

5.5 実験環境記述の書式と検証全体の支援

実験環境記述と、それを他の実験支援システムの実験支援ソフトウェアの実験記述の形式に変換する機能の実装について述べる。

実験環境保存時に生成される実験環境記述の書式は以下の通りである。

計算機　[ディスクイメージファイル名]　管理用 IF 識別子　実験用 IF　接続スイッチ名　接続スイッチポート　VLAN 番号　(対象ファイル群識別子)

[] がついているフィールドはディスクイメージで、() がついているフィールドはファイル単位でソフトウェアを保存する場合のオプションである。

本論文で実装対象としているのは StarBED であり、StarBED には SpringOS という実験支援ソフトウェアがある。そのため、実験環境記述を SpringOS の実験記述に変換し、実験記述テンプレートを出力する機能を実装した。また、複数の実験環境記述を利用して復元を行うことで、実験環境の合成も可能となっている。実験環境の合成は、実験環境の復元を複数回行うか、実験環境記述を手動で合成することで実現した。

第6章　評価

本章では提案するシステムを評価する。評価は、提案システムが設計通りに動作するかどうかを確認するための各機構単体の試験と、提案システムを用いた実験が可能かどうかと、提案システムが実験者にかかるコストを削減できるかどうかを確認するための複合試験で行った。

6.1　単体試験

提案システムでは以下の機能を実現した。

- ネットワークの保存・復元
    1. 保存時の計算機・VLAN を利用した復元
    2. 代替計算機での復元
    3. 代替 VLAN を利用した復元
    4. 他ベンダのスイッチでの復元
- ソフトウェア構成の管理
    1. ディスクイメージとしての保存・復元
    2. ファイル単位での保存・復元
- 実験履歴の管理
    1. プロセスの履歴の保存
- 他の実験支援システムとの連携
    1. 実験環境記述の SpringOS 用の実験記述への変換

以降、これらの機能について評価を行った結果を示す。

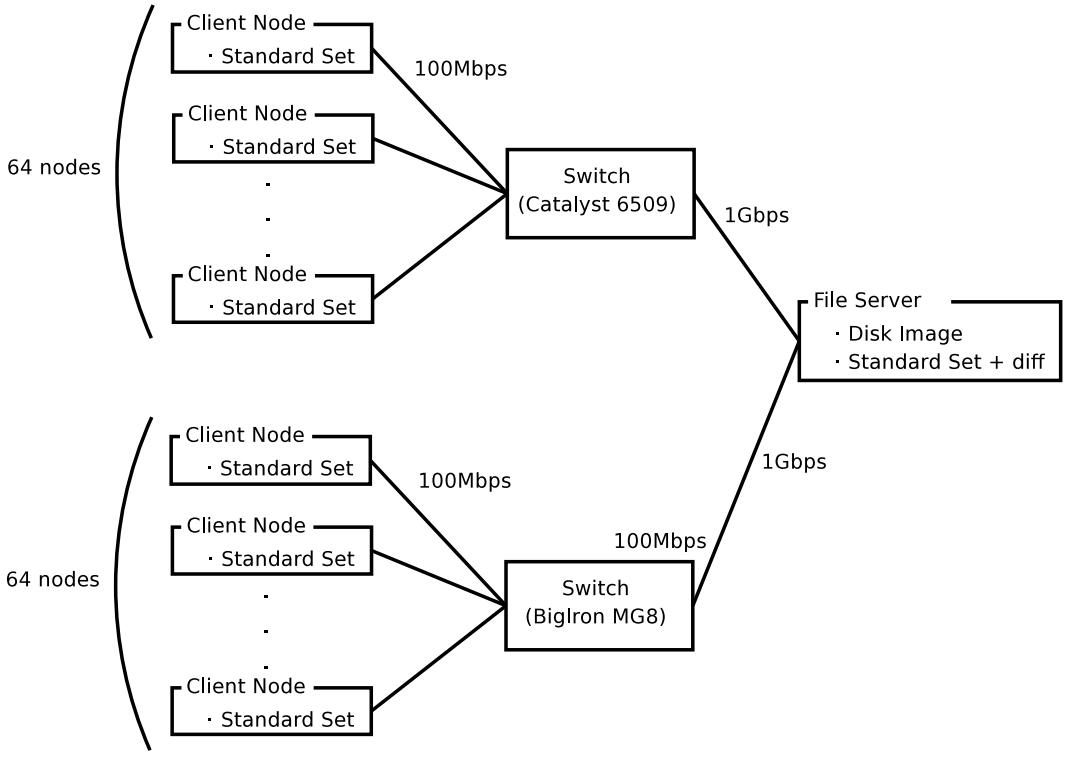


図 6.1: 単体試験のための実験環境

6.1.1 単体試験の実験環境

単体試験に用いた実験環境は図 6.1 の通りである。単体試験に用いた計算機 (Client Node) は 128 台で、64 台ずつスイッチ BigIron MG8、Catalyst 6509 に接続されている。計算機からスイッチまでの帯域は 100Mbps であり、スイッチとディスクイメージやファイルの差分を保存するためのファイルサーバまでの帯域は 1Gbps である。

6.1.2 ネットワーク保存・復元

ネットワーク保存・復元機能の動作の確認手順は以下の通りである。この手順での動作確認をそれぞれのスイッチに接続されている計算機群に対して行った。

1. 実験環境のネットワークを保存
2. 保存時の計算機・VLAN を用いた復元 (等価な復元)
3. 柔軟な復元

(a) 代替計算機を用いた復元

(b) 代替 VLAN を用いた復元

(c) 他ベンダのスイッチに接続された計算機を用いた復元

それぞれの結果について、以下に述べる。計算機の等価な復元では、保存時の計算機と同じ計算機の実験用 IF が、保存時の実験用 IF に設定されていた VLAN に所属していることを確認した。

柔軟な復元では、まず保存した実験環境の計算機とは異なる代替の計算機を指定しての復元の結果、指定した計算機の実験用 IF が保存時の計算機の実験用 IF に設定されていた VLAN に所属していることを確認した。次に、保存時の VLAN 番号が使用中だった場合に代替 VLAN 番号の利用を許す指定を行って復元を行った結果、使用中の VLAN 番号を自動的に避けて状態を復元する機能が動作していることを確認した。

そして、他ベンダのスイッチに接続されている代替の計算機を指定しての復元を行った結果、保存時と同じスイッチを用いて復元を行った場合と同じ結果が得られた。これより、コマンド体系に依存せず実験環境の VLAN を復元できることが確認できた。

6.1.3 ソフトウェア保存・復元

ソフトウェア保存・復元機能の動作確認の手順は以下の通りである。

1. ディスクイメージでの保存・復元

   (a) 指定パーティションのディスクイメージとしての保存

   (b) ディスクイメージを用いての復元

2. ファイル単位での保存・復元

   (a) 保存時と標準ソフトウェア群との差分を保存

   (b) 保存時のソフトウェア構成の復元

どちらの方法に関しても、保存したソフトウェア構成の復元できたことを確認した。

6.1.4 実験履歴管理

実験履歴の管理機能として導入した、プロセス履歴保存機能の動作確認を行い、実験開始から終了までのプロセスの履歴が保存できたことを確認した。ただし、実験者が実行したもの以外の実行履歴も含まれているため、実験履歴を参照する際には注意が必要となる。

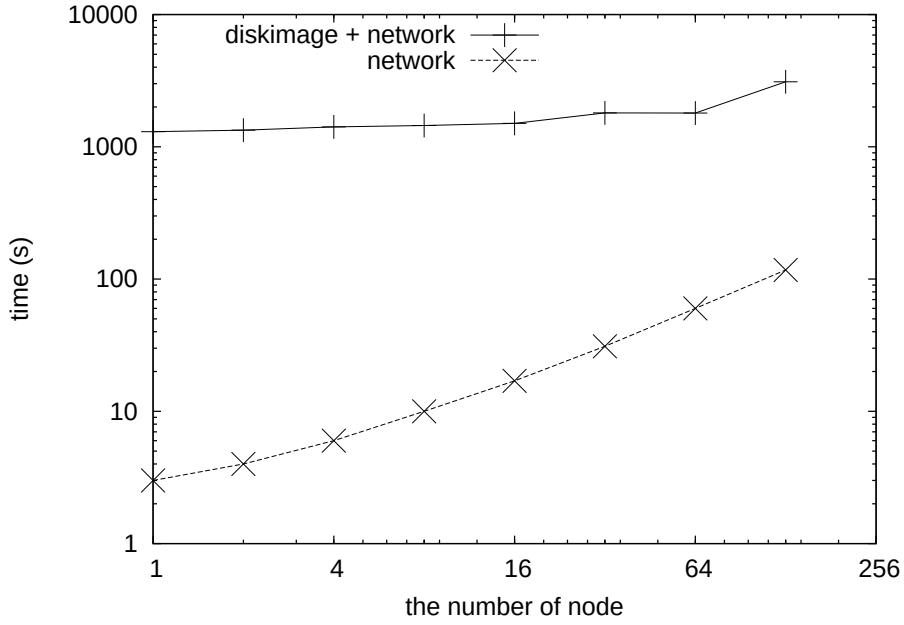


図 6.2: 実験環境保存時間

6.1.5 検証全体の支援

ネットワークとソフトウェアの保存・復元、実験履歴管理機能の動作が確認できたため、実験環境記述を用いた実験環境の保存と復元は実現できた。そのため、その実験記述を別の実験に用いることで実験環境の再利用は可能である。

また、生成される実験記述の一部を改変することで、実験環境の拡大と合成も実現できることを確認した。実験環境の拡大の場合は、ある実験環境を保存した実験環境記述に、増設する計算機の設定を書き加えることで可能となる。そして、実験環境の拡大は、いくつかの実験環境記述を合成すること可能である。これらについても、実際に動作確認を行い、実現できたことを確認した。

6.1.6 保存・復元時間の比較

図 6.1 の実験環境の保存・復元対象の計算機の数と、保存と復元にかかった時間について、ネットワークのみの保存・復元を行った場合、ネットワークの保存・復元に加えてディスクイメージとしてソフトウェアの保存・復元も行った場合の比較を図 6.2、図 6.3 に示す。保存・復元時間の計測は、利用する計算機の数を 1 台から 128 台まで増加させながら行った。

この結果、実験環境の部分的な復元であるネットワークのみの保存・復元は短時間で行えることが確認できる。ただし、ソフトウェアの保存・復元では、サーバのチューニン

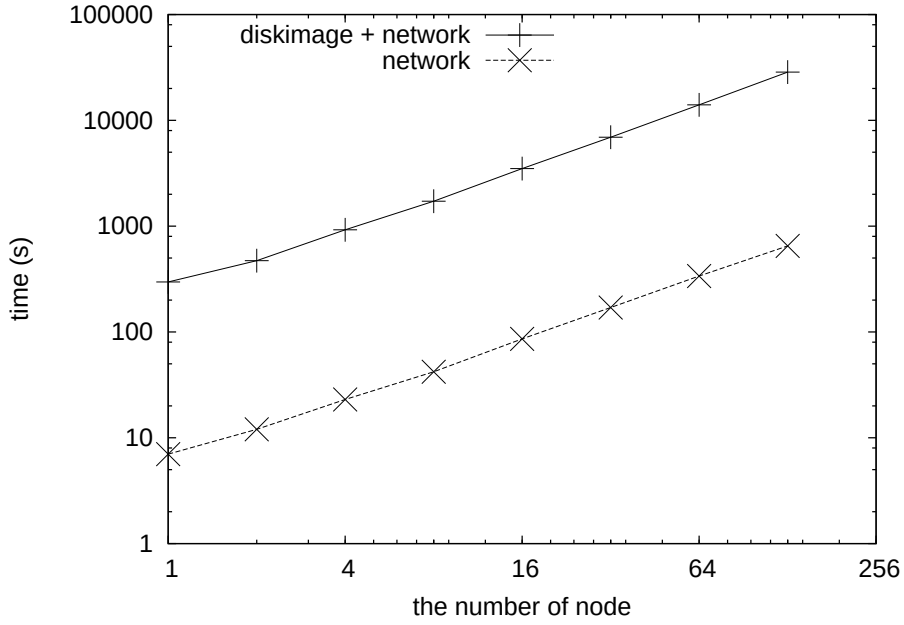


図 6.3: 実験環境復元時間

グやネットワークトポロジによって所要時間が変化するため、所要時間の削減は可能である。

6.2 複合試験

複合試験は、提案システムを用いて実験を行えるかどうかと、実験のコストを削減できるかどうかを評価するために行った。以下にその詳細を示す。

6.2.1 複合試験の実験環境

提案システムを用いて実験を行えるかどうかと、実験のコストを削減できるかどうかを評価するための実験として、文献 [23] で紹介した、StarBED を用いたサーバ負荷試験を模倣したものを行った。評価に用いた実験環境を図 6.4 に示す。

ここでは、各クライアントノードの、標準構成のソフトウェア用のパーティションに標準構成のソフトウェアに加えてファイル取得のために実装したソフトウェアである CCFTP を導入する。また、サーバノードには FTP 及び HTTP サービスを起動させる。そして、サーバノードに対し、各クライアントノードからファイルの取得を行い、ファイル取得の成功率や転送時間を測定する。

各計算機の仕様は表 6.1、表 6.2 以下の通りである。

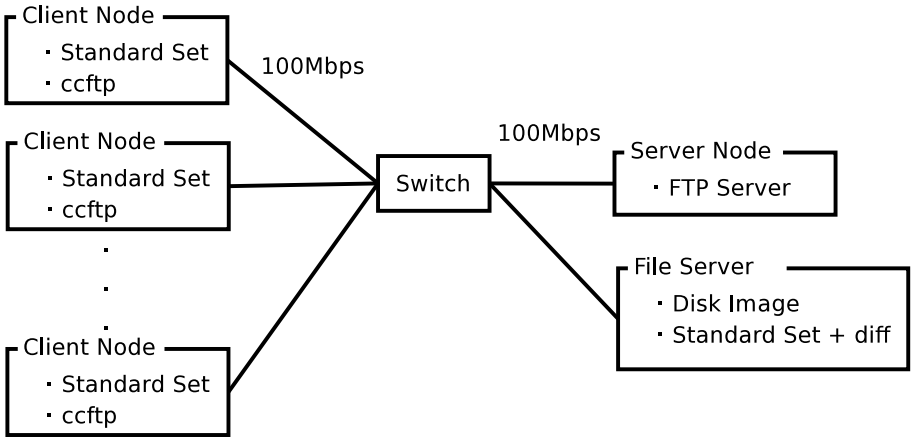


図 6.4: 提案システム評価のための実験環境

6.2.2 評価手順

以下が提案システム評価のための実験の手順となる。

1. 標準構成のソフトウェアを用意
2. 標準構成のソフトウェアに CCFTP を導入
3. クライアントノードのネットワークを保存し、ソフトウェアをディスクイメージ、ファイル単位で保存
4. 1 台で実験実行し、実験環境の動作確認
5. 初期の実験環境に復元
6. クライアントノードを 8、16、32、64 台と増やす

表 6.1: ファイルサーバの仕様

Chipset	Intel E7230
CPU	Pentium4 3.2GHz HT acceptable
Memory	4.0GB
OS	Red Hat Enterprise Linux ES release 3

7. 実験実行

8. 5 に戻る

復元するのはクライアントノードである。クライアントの数は、手動で構築した 1 台から初めて、それを 8、16、32、64 と増加させた。

6.2.3 評価項目

評価実験に対する評価項目は以下の通りである。

- 提案した実験の実行の手順が実現できるか
- 復元時に、ネットワーク、ソフトウェア、プロセス履歴が保存できているか
- 保存・復元時間を測定

6.2.4 結果

保存・復元の確認

実験環境の復元を行ったそれぞれの段階において、ネットワーク、ソフトウェア構成が正しく復元できているかどうかを確認した。また、プロセスの履歴については、各計算機で実行したプロセス (例えば実験用ソフトウェアの取得) が記録されているかどうかを確認した。確認は、復元時のクライアントノード、クライアントノードが接続されているスイッチにログインするなどして行った。

その結果、各段階においてネットワーク、ソフトウェア構成が復元されたこと、プロセス履歴が記録されていることが確認できた。このことから、実装したシステムでの実験環境の保存・復元が可能だといえる。

実験環境記述と実験記述の生成

検証実験を保存した際の実験環境記述を SpringOS の実験記述に変換した結果について述べる。検証実験の実験環境を保存したときの実験環境記述は以下のようになった。

表 6.2: クライアントの仕様

Chipset	ServerWorks LE
CPU	Pentium3 1GHz
Memory	500MB

```
sintcld001 image sintcld001_200801151534.gz
0 manage mgsw2 11/1 1 experiment silaswb001 7/15 850
sintcld002 …
…
```

この内容から、SpringOS の実験記述を生成すると、結果は以下のようになった。

```
1  nodeclass clclass1 {
2    method "HDD"
3    disktype "IDE"
4    partition 5
5    ostype "Linux"
6    diskimage "ftp://install:install@172.16.210.9/sintcld001_200801151534.gz"
7    netif media fastethernet
8    #実験実行手順を記述
9  }
10 nodeclass clclass2 {
11 …
12  }
13 …
```

1 行目はノードクラス clclass1 の定義である。

2 行目は、ハードディスクを利用することを示す。

3 行目はハードディスクのタイプである。ハードディスクのタイプは各計算機のホスト名から判別する。

4 行目はディスクイメージを導入するパーティションを示している。これはディスクイメージを取得するときに指定されたパーティションが記録される。

5 行目はパーティションに導入される OS のタイプである。これは StarBED の計算機のパーティションに関連付けられた名前であり、保存・復元対象のパーティションによって決定される。

6 行目はディスクイメージが保存されたファイルサーバとディスクイメージのファイル名、そしてファイルへのアクセス方法を示している。

7 行目はその計算機の実験用 IF の規格である。

そして、それ以降にはその計算機で動作される実験実行の手順を記述することになる。この部分は、プロセス履歴保存機構が保存したプロセス履歴を利用して実験者が記述する。

保存・復元時間の測定

提案システムの有用性を確認するため、実験環境の保存・復元に関して必要とされる時間を測定した。

図 6.5、6.6 は、ディスクイメージ、ファイル単位それぞれの場合のソフトウェア構成の保存・復元時の所要時間のグラフである。図 6.5 は各手法での保存時の台数と所要時間の関係を示している。図 6.6 は、各手法での復元時の台数と所要時間の関係を示している。

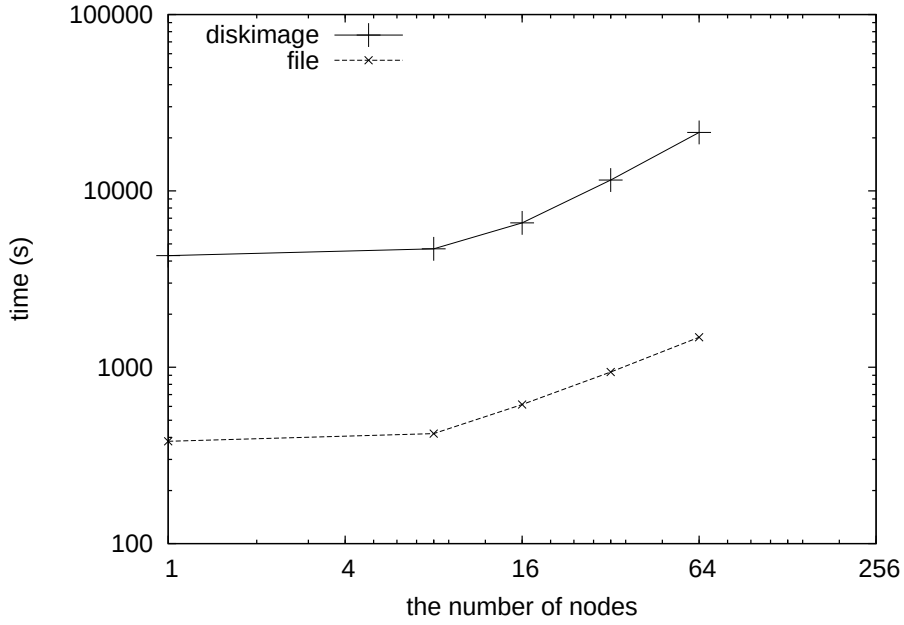


図 6.5: 実験環境の保存時間

また、各所要時間にはネットワークの保存・復元にかかる時間を含む。所要時間は、ディスクイメージでの保存・復元に比べ、ファイル単位で保存・復元を行った場合の方が少なかった。これは、評価実験で用いるソフトウェア構成と標準構成との違いが少ないことに起因していると考えられる。また、保存と復元にかかる時間はファイルサーバのスペックの向上や、ネットワークの帯域の増加によって削減できる。なお、保存したある計算機のソフトウェア構成を複数の計算機に配布するために必要な時間は、各機器の保存時のソフトウェア構成を復元するときの所要時間とほぼ同じだった。

提案実験手順の妥当性

手順に関しては、図 6.7 が今回の評価実験の実験環境の構築の部分の手順をすべて手動で行った場合と、既存の実験支援ソフトウェアのみを用いて行った場合、そして、提案システムを用いた場合の検証の工程を示したものである。

評価実験では、小規模な実験環境を構築して実験を行ってから大規模な実験環境での実験を行っている。手動での検証では、小規模・大規模な実験の両方で実験環境の構築をすることになる。

既存の実験支援ソフトウェアのみを用いた場合では、手動で構築した小規模実験で実験の動作を確認した上で、大規模な実験での環境構築のための実験記述を作成することになる。この実験記述を作成しての実験環境の構築は習熟が必要であることは既に述べた。また、実験環境を手動で構築し、実験を実験支援ソフトウェアで実現することはコストが

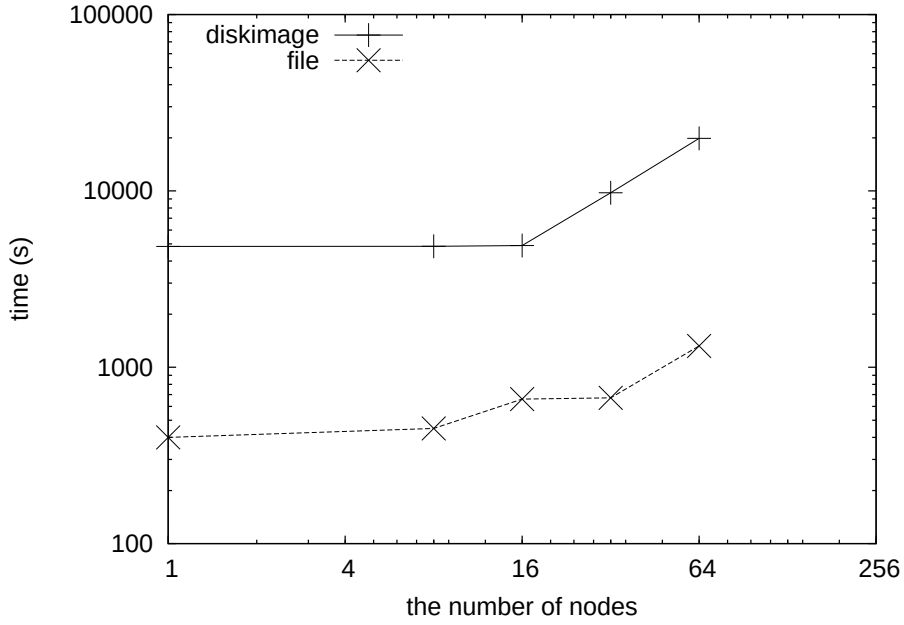


図 6.6: 実験環境復元時間

大きい。

提案システムを用いた場合、小規模実験で用いた実験環境から出力した実験環境記述を利用し、それを多少変更するだけで大規模に拡張することができるため、環境構築のための実験記述を必要としない。また、実験支援システムと組み合わせて、実験実行までの手順を自動化させる場合にも、対象実験支援システムの実験記述の書式で実験環境構築の部分を出力することができる。さらに、小規模実験時に保存されたプロセスの履歴を参照して、容易に実験記述の実験実行部を記述することが可能となる。

このように、実験のコストを削減することが可能である。

6.3 考察

単体試験と、前述した評価実験により、保存した実験環境の等価な復元、柔軟な復元の実現を確認した。また、外部実験支援ソフトウェアとの連携が可能であることを確認した。そして、単体試験の結果、実験環境の合成や再利用も可能になっていること、実験の履歴をプロセスの履歴として保存できていることを確認した。これにより、外部実験支援システムとの連携、ひいてはネットワーク技術の検証を統合的に支援できるシステムを構築できた。

また、ソフトウェアをディスクイメージで保存・復元した場合とファイル単位で保存・復元した場合の保存時間、復元時間を測定し、比較した。その結果、標準構成のソフトウェア群からの変化が少ない場合にはファイル単位での保存・復元が有効である。

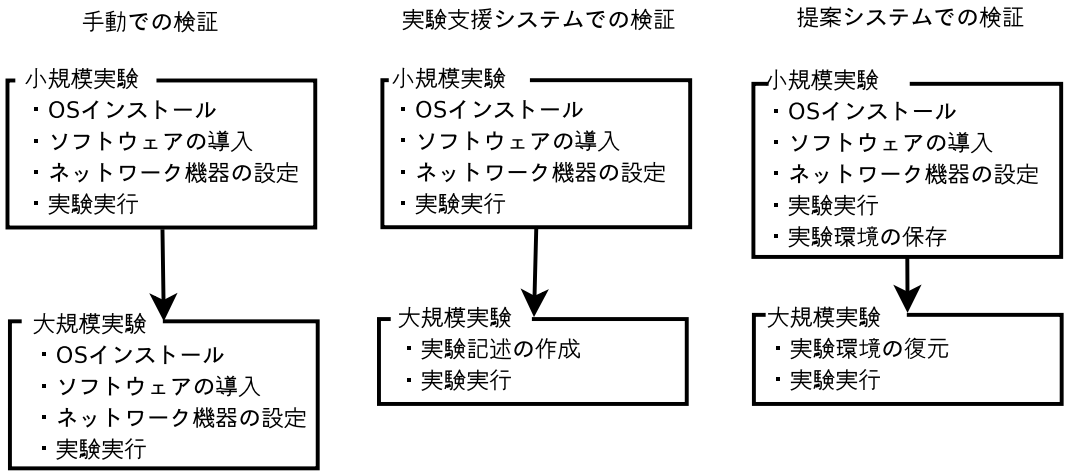


図 6.7: 各種検証手順

そして、提案した手順での実験が可能であることを確認した。各手法による実験環境の保存・復元時間の比較と、工数の比較から、提案システムは実験者の拘束時間、実験時間、実験工数の短縮が可能であり、さらに、あるネットワーク機器、計算機から保存したネットワーク、ソフトウェア構成を他の機器に導入できることから実験設備の稼働率も増加させる。

第7章　提案システムの方向性

7.1　実験環境のアーカイブと活用

提案システムにより、実験環境の保存と復元は可能となり、保存した実験環境の再利用と合成も実現できた。ただし、保存した実験環境の再利用を考えたとき、自分が行った実験であればどのような実験を行ったのかをある程度把握することは可能であるが、他の実験者の構築した実験環境の利用する場合、実験者が多くの実験環境を保存した場合等には、どの実験環境がどのような意味を持っているのかを把握すると効率が良い。

そのための機能として、提案システムで実装した実験履歴の管理に加え、同じ実験に用いられた実験環境の時系列での変化や、実験環境がどのような実験のために構築され、どんな結果を出したのかを俯瞰できる機能が必要となるだろう。

7.2　実験記述の自動生成

現在の実装では、実験環境のネットワーク設定と資源管理機構が把握している情報から実験記述テンプレートを生成し、それを調整し、実験実行の手順をすることで実験記述を作成する。また、提案システムではプロセス履歴を保存することで、実験の履歴を管理しようと試みた。このプロセス履歴から、実験実行の手順を抽出することができれば、実験記述そのものを作成することも可能になるだろう。

また、実験環境の構築から実験実行までの一連の流れを記した実験記述を生成することができれば、ファイルのバックアップを必要とせずに実験環境を構成する機器のソフトウェア構成を復元することが可能になるだろう。この手法ではソフトウェアの取得元から対象が失われていた場合等に対応することができなくなることが考えられるが、提案システムと組み合わせることでさらに効率の良い実験環境の保存・復元が可能になるだろう。例えば、パッケージ管理ソフトウェア等で取得できるファイルは保存せず、その他のファイルは保存する等、保存・復元に要する時間や必要なディスク容量が削減されることが考えられる。

7.3　その他の用途への転用

提案システムは実験、特に実験の準備の段階を効率的に行うことを目的としている。実験環境の保存・復元は、ネットワーク技術の検証だけでなく様々な用途で使うことができるだろう。

例えば、実機を使ったネットワーク技術の教育では、実験に利用できる機器や時間が限られている。実験者が構築した実験環境を保存・復元できることで、実験者にとっては実験の中断・再開が容易になるメリットがあり、機器の管理者にとっては各機器の設定を元に戻す作業を効率化できる。そして、教育者にとっては、保存された実験環境を容易に復元できることで、時間や機器の使用状況に左右されずに評価を行えることになるだろう。

さらに、提案システムは実験環境を保存するだけでなく、実際に運用されている環境を保存するためにも利用できると考えられる。実際に運用されている環境で提案システムを利用するためには、現在実験用ネットワークと管理用ネットワークの 2 つのネットワークを用意して確保しているネットワークの到達性を、何らかの方法で確保する必要がある。同時に、ネットワークの管理者等が資源管理機構に環境を把握させることで、環境の保存が可能になる。問題が発生した環境を保存し、それを実験支援システム等に導入して検証を行えば、運用されている環境に影響を与えずに不具合を修正することが可能となる。

第8章　おわりに

本研究では、実験環境の保存と復元のシステムを実現し、実験の繰り返しや中断・再開のたびに必要になっていた実験環境の構築を容易にすることと、本研究で実装した実験環境の保存・復元による実験環境の構築と実験実行を主とした既存の実験支援システムを接続し、ネットワーク技術検証全体を容易にすることを目的とし、システムの検討、設計、実装を行った。

そして、提案システムを用いたネットワーク実験の提案を行い、それを評価・考察した。その結果、提案システムにより実験環境の保存・復元が可能できたことを確認した。実験環境の保存と復元では、実験環境の保存した要素を全て復元する等価な復元、保存した要素の一部と代替の資源を用いて復元する柔軟な復元を実現できた。また、実験環境の拡大や、再利用、合成も容易に実現できることが確認できた。さらに、提案システムを用いた実験を実際に行った。これにより保存した実験環境からの外部の実験支援ソフトウェア用の実験記述の生成を実現した。加えて、提案システムを利用することで従来行われてきたネットワーク技術を対象とした検証手法よりも、実験時間、実験工数を削減できることを確認した。

謝辞

研究を行うにあたり、主指導教員である篠田陽一教授には多くの御助言や御指導をいただきました。深く感謝し、心よりお礼申し上げます。また、適切な御指摘、御指導を賜りました主テーマ審査委員である丹康雄教授、敷田幹文准教授、副テーマ指導教員である鳥澤健太郎准教授に深く感謝致します。

情報通信研究機構 三輪信介研究員、宮地利幸研究員、本学 知念賢一助教には、実験環境の保存・復元を実現のために必要な基盤を提供していただき、適切な御意見と多大な御協力をいただきました。心より感謝致します。

情報通信研究機構北陸リサーチセンター 佐野正行氏、竹中ゆかり氏にはStarBEDでの実験において多大な御助力をいただきました。心より感謝致します。

奈良先端科学技術大学院大学 砂原秀樹教授、倉敷芸術科学大学 小林和真教授、メディアエクスチェンジ株式会社 高田寛取締役には研究会等の活動において多大な叱咤激励をいただきました。心より感謝致します。

本学 宇多仁助教、中川晋一客員准教授には様々な場面で多くの御助言をいただきました。心より感謝致します。

本研究室 高野祐輝氏、LATT Khin Thida氏、安田真悟氏、井上朋也氏、SABER ZRELLI氏、Nguyen Lan Tien氏、芳炭 将氏、NGUYEN, Nam Hoai氏、平佐誠道氏、上野隆資氏、中井浩氏、三浦良介氏、松井 大輔、梅木 孝志氏、佐川 喜昭氏、千装俊幸氏、石渡優祐氏には様々な場面で御指導、御意見、御協力をいただきました。心より感謝致します。

最後に、研究や生活を支えてくれた家族に感謝致します。

参考文献

[1] 宮地利幸. 大規模実証環境の実現と実験支援によるネットワークサービスの検証技術. 北陸先端科学技術大学院大学博士論文. 2007 年 3 月.

[2] 宮地 利幸, 中田 潤也, 知念 賢一,Razvan Beuran, 三輪 信介, 岡田 崇, 三角 真, 宇多 仁, 芳炭将, 丹康 雄, 中川 晋一, 篠田 陽一. StarBED:大規模ネットワーク実証環境. 情報処理 Vol.49 No.1, 情報処理学会, pp.57-70. 2008 年 1 月.

[3] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[4] VMware. http://www.vmware.com/.

[5] 三輪 信介, 滝澤 修, 大野 浩之. 仮想 PC インターネットセキュリティ実験環境『VM Nebula』の設計と構築. 暗号と情報セキュリティシンポジウム (SCIS). 電子情報通信学会, 2003 年 2 月.

[6] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Bard, and Abhijeet Joglekar. An integrated experimental environment for distributed systems and networks. pp.255-270, Boston, MA, December 2002. USENIX-ASSOC.

[7] StarBED Project.http://www.starbed.org/.

[8] Toshiyuki Miyachi, Ken-ichi Chinen, and Yoichi Shinoda. StarBED and SpringOS: Large-scale General Purpose Network Testbed and Supporting Software. Valuetools 2006, Pisa, Italy, ISBN 1-59593-504-5, Oct, 2006.

[9] M. Suzuki, H. Hazeyama, Y. Kadobayashi. AnyBed: a testbed-independent topology configuration tool. In proceedings of Second Inter-national Conference on Testbeds and Research Infrastructures for the Development of Networks Communities (TridentCom2006), Mar. 2006.

[10] 三輪 信介, 大野 浩之. インターネットセキュリティ実験環境『VM Nebula2』の設計と構築. 暗号と情報セキュリティシンポジウム (SCIS2006). 電子情報通信学会, 2006 年 1 月.

[11] Mike Hibler, Leigh Stoller, Jay Lepreau, Robert Ricci, Chad Barb. Fast, Scalable Disk Imaging with Frisbee. In Proceeding of the 2003 USENIX Annual Technical Conference. June 2003.

[12] 三角真, 宮地利幸, 知念賢一, 篠田陽一, 実ノードを利用したネットワークシミュレーションにおけるノードへの OS の導入及びパラメータ設定機構の開発, 情報処理学会研究報告書 DPS-116, ISSN 0919-6072, pp. 95-100, 鳥取, 2004 年 1 月.

[13] RFC4741. NETCONF Configuration Protocol. December 2006.

[14] Charles P. Wright, Jay Dave, Puja Gupta, Harikesavan Krishnan, Erez Zadok, and Mohammad Nayyer Zubair. Versatility and Unix Semantics in a Fan-Out Unification File System. Stony Brook U. CS TechReport FSL-04-01b. Oct 2004.

[15] aufs. http://aufs.sourceforge.net/aufs.html.

[16] Intel and Systemsoft. Preboot Execution Environment (PXE) Specification Version 2.1. Sep 1999.

[17] Foundry Networks. http://www.foundrynetworks.co.jp/

[18] Foundry Networks. Multi-Service IronWare Operating System overview. 2006.

[19] Cisco Systems, Inc. http://www.cisco.com/jp/index.shtml

[20] Cisco Systems, White Paper: Cisco IOS Reference Guide. 2006.

[21] Andrew Tridgell. Efficient Algorithms for Sorting and Synchronization. PhD thesis at The Australian National University. February 1999.

[22] rdiff-backup. http://www.nongnu.org/rdiff-backup/index.html.

[23] 野中 雄太, 知念 賢一, 宇多 仁, 宮地 利幸, 篠田 陽一, StarBED を用いたサーバ負荷試験の実現, マルチメディア, 分散, 協調とモバイル シンポジウム (DICOMO)2007, 論文集, 情報処理学会, pp. 199-204, 鳥羽, 2007 年 7 月.

本研究に関する発表論文

- 野中 雄太, 知念 賢一, 宇多 仁, 宮地 利幸, 篠田 陽一, StarBED を用いたサーバ負荷試験の実現, マルチメディア，分散，協調とモバイル シンポジウム (DICOMO)2007, 論文集, 情報処理学会, pp. 199-204, 鳥羽, 2007 年 7 月 (ヤングリサーチャ賞受賞).
- 野中 雄太, 知念 賢一, 宮地 利幸, 三輪 信介, 篠田 陽一. 実機ベース汎用大規模実験環境の状態の保存・復元機構. インターネットコンファレンス 2007（IC2007）, pp.49-58, 福岡, 2007 年 10 月.