GINE の GUI 機能の拡張

2005MT092 太田翔　　2006MI038 今川敬太
指導教員　後藤 邦夫

1　はじめに

近年 web サービスやインターネットを利用したアプリケーションが増加している．これらのサービスやアプリケーションは，実装する前にネットワーク障害等を想定して性能評価をする必要がある．性能評価には，さまざまなネットワークエミュレータが用いられる．

ネットワークエミュレータはホスト内に仮想ネットワークを構築し，実際にパケットを送受信することで，実ネットワークにより近い環境を模倣することができる．南山大学の後藤邦夫教授が考案した Goto's IP Network Emulator(以下，GINE[4]) もネットワークエミュレータの一つである．

しかしネットワークエミュレータ操作を全て手動で行うことは，非効率的である．上記の問題を解決するため，2010 年南山大学大学院浅野修士論文 [1] によって，GUI 操作による エミュレーションモデルの構築，保存・読出機能などの管理機能が GINE に追加された．

しかし GUI 操作において，構築した仮想ネットワーク構成を部分的に削除する機能は，GINE に存在しない．また現状の GUI 機能では構築するネットワークの規模が，ウィンドウの大きさで制限されしまうため，大規模な仮想ネッワークを構築することができない．

本研究では，インターネットのような大規模ネットワークを構築することを想定し，GINE の操作性向上を目的とし，GINE の GUI 機能を拡張する．

2　GINE の概要

本節では，GINE の概要について説明する．GINE を使用することで，多数のルータやリンクから構築された様々な大規模ネットワークをエミュレートすることが可能である．各リンクに遅延やパケットロスなどの通信障害を設定することが可能で，バンド幅も自由に設定することができる．加えて IPv6 にも対応している．上記の機能により，ネットワーク障害など，現実に近い形で実験ができる．

また, GINE は C++ クラスライブラリ (GNU commn C++)[2] を用いて記述されたプログラムであり，機能の追加が容易にできることも特徴である．実際に GINE には研究生により，幾度も新機能が追加されている．最近では，2010 年の浅野修士論文による，GNU common C++ にある Persistence を用いての保存/読出機能の実装及び，GUI 機能の簡易実装，2010 年星野&石野卒業論文 [3] による動的経路制御機能の追加等がある．

3　GINE の GUI 機能の概要

本節では，2010 年浅野修士論文で実装された GINE の GUI 機能について説明する．この GUI 機能は C++ 言語とアプリケーションフレームワークである Qt を用いてプログラミングされている．実装された GUI 機能は次の 4 点である．

- GINE クラスで生成するオブジェクトをボタン (ラベル) として配置する機能.
- オブジェクトボタンを描画フィールド（パレット）にドラッグ&ドロップで配置する機能.
- 各オブジェクトボタンをダブルクリックすると別ウィンドウが開き，詳細な情報を設定する機能.
- スタートボタンをクリックすると，スレッドがスタートして通信を開始する機能.
- ポーズボタンやストップボタンを配置し，時間経過に沿ったシナリオを実現する機能.

上記の GUI 機能の操作手順を次の図 1 に示す.

図 1　GUI 機能の概要

1. タイマを作成する.
2. GINE オブジェクトを生成するボタンリストを選択する.
3. ボタンをドラッグ&ドロップする.
4. ドロップしたボタンをダブルクリックする.
5. 設定した値の保存・キャンセルをする.

上記の GUI 機能を操作することで，プログラミングに詳しくないユーザでも簡単に仮想ネットワークを構築することが可能になった.

4　既存システムの改善案

本節では，本研究で提案する GINE の GUI 機能の改善案について説明する．改善案は次の 2 つである.

- 構築された仮想ネットワークを部分的に削除する

機能の追加.

- 拡張パレット機能の追加.

次節で詳しい概要を述べる.

4.1 仮想ネットワークを部分的に削除する機能の追加

2010 年浅野修士論文により, GINE に Network Name Space (以下 NETNS) の端末管理機能，オブジェクトの保存・読出機能が追加され，GUI 機能が簡易的に実装された．しかし，GUI 上で構築された仮想ネットワークを部分的に削除する機能は存在しない．よって，オブジェクト生成や入力操作等の誤りがあると，仮想ネットワーク構築を最初からやり直さなければならない．例えば次の図 2 では，2 つの NETNS と 1 つの Veth で構成されるネットワークを構築しているが，余分な Veth(Veth3) を配置してしまい，パレット内に使用しない Veth が存在している．

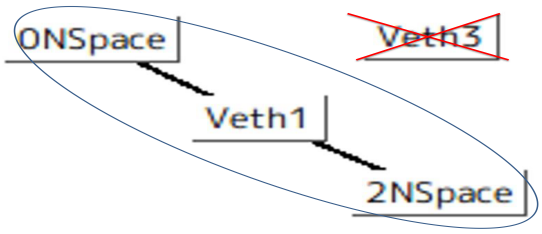

図 2　仮想ネットワーク構成中に生じた誤り例

図 2 の様に，些細な誤りを修正することができず，仮想ネットワーク構築を最初からやり直すことは，ユーザにとって不便である．そこで構築された仮想ネットワークを部分的に削除する機能を追加する．

4.2 拡張パレット機能の追加

現状の GUI 機能では，パレットは最初に開かれる 1 つしか存在しない．ラベルをドラッグ&ドロップしてパレットに仮想ネットワークを描いていくという仕様上，パレットがラベルで埋め尽くされてしまうと，それ以上仮想ネットワークを描くことができない．次の図 3 は実際にパレットがラベルで埋め尽くされた状態である．

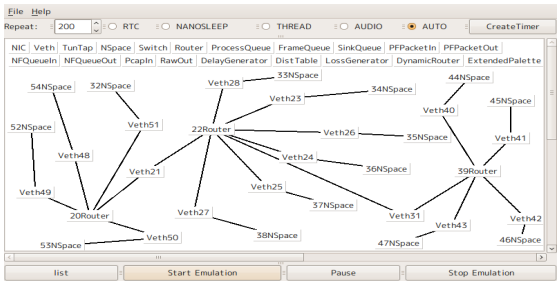

図 3　多数のラベルによりパレットが埋め尽くされた例

図 3 の様に，1 つのパレットだけでは大規模な仮想ネットワークを構築することができない．仮に構築出来たとしても，ユーザにとって仮想ネットワーク構成を把握することが困難になる．そこで拡張パレット機能を追加することで，複数のパレットによる大規模な仮想ネットワーク構築を可能にする．

5 既存システム改善の実現

本節では，既存システム改善の実現について述べる．

5.1 削除機能の実現

生成された GINE オブジェクトの情報は静的な map コンテナに保存される．GINE の map コンテナは ”ラベル名”, ” オブジェクト本体 ” の 2 つの要素から構成されている，仮想ネットワークの各オブジェクト情報を削除するため，GINE クラスに *deleteInstanceOf* 関数，GUI クラスに *remove* 関数を追加した．*deleteInstanceOf* 関数は，生成された GINE オブジェクトの情報を map コンテナから削除する役割を担う．*remove* 関数は，生成された GUI オブジェクトを削除する役割を担う．*deleteInstanceOf* 関数と *remove* 関数によって，仮想ネットワーク構成を部分的に削除する操作の流れを次の図 4 に示す．

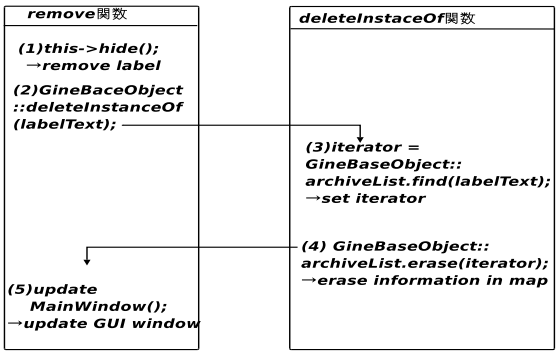

図 4　削除機能の操作の流れ

1. 削除ボタンが押されると，GUI オブジェクト (label) を消す.
2. GineBaceObject クラスの *deleteInstanceOf* 関数を呼び出す.
3. 削除する情報探し，イテレータに代入する.
4. map コンテナ内の GINE オブジェクトの情報を削除する.
5. 削除されたオブジェクトの情報を元に，GUI ウィンドウを更新する.

上記の 2 つの関数を用いることで，GUI オブジェクトと map コンテナ内の GINE オブジェクトの情報を削除することができる.

次に上記の関数を用いて構築した仮想ネットワークを部分的に削除してみる．構築した仮想ネットワーク構成を，次の図 5 に示す.

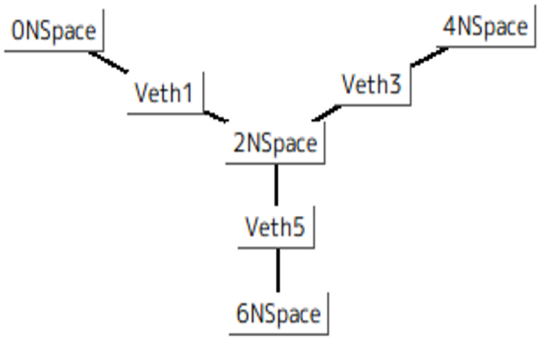

図 5　仮想ネットワークを構築する

図 5 の仮想ネットワークは，4 つの NSpace と 3 つの Veth で構成されている．構築された仮想ネットワークの GINE オブジェクトの情報を次に示す．

図 5 の GINE オブジェクト情報

```
======List GINE Objects(Begin)====
size : 7
objname(in list) : ClassID :
objname(from obj) : xPos : yPos
0NSpace : NSpace : 0NSpace : 140 : 91
2NSpace : NSpace : 2NSpace : 328 : 177
4NSpace : NSpace : 4NSpace : 600 : 137
6NSpace : NSpace : 6NSpace : 240 : 306
Veth1 : Veth : Veth1 : 221 : 132
Veth3 : Veth : Veth3 : 454 : 156
Veth5 : Veth : Veth5 : 277 : 242
=======List GINE Objects(End)=======
```

次に Veth3，Veth5 を削除をする．部分的に削除した仮想ネットワーク構成を，次の図 6 に示す．

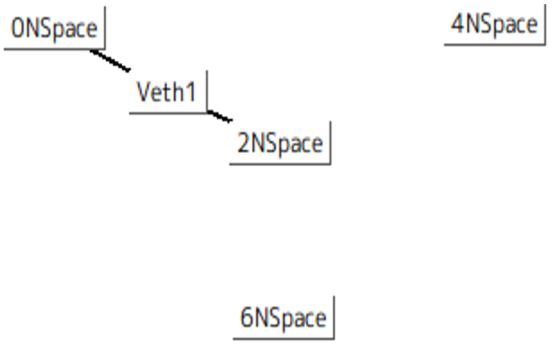

図 6　仮想ネットワークを部分的に削除する

図 6 では Veth3，Veth5 の GUI オブジェクトが削除されている．削除された仮想ネットワークの GINE オブジェクト情報を次に示す．

図 6 の GINE オブジェクト情報

```
======List GINE Objects(Begin)====
size : 5
objname(in list) : ClassID :
objname(from obj) : xPos : yPos
0NSpace : NSpace : 0NSpace : 140 : 91
2NSpace : NSpace : 2NSpace : 328 : 177
4NSpace : NSpace : 4NSpace : 600 : 137
6NSpace : NSpace : 6NSpace : 240 : 306
Veth1 : Veth : Veth1 : 221 : 132
=======List GINE Objects(End)=======
```

上記の結果から，GUI オブジェクトと GINE オブジェクト情報が適切に削除されていることが分かる．

5.2　拡張パレット機能の実現

拡張パレット機能を追加するため， ExtendedPalette クラスと拡張パレット用のレイアウトを作成し，保存する GINE オブジェクト情報を変更した．

ExtendedPalette クラスの追加

GineBaseObject に ExtendedPalette クラスを追加した．ExtendedPalette クラスでは操作は行われない．GUI 上の ExtendedPalette ラベルの作成時に生成され，ラベルの位置情報等を保存する為に用いられる．

拡張パレット用のレイアウトを作成

拡張パレットには保存や読出し等のボタンは必要ない．ラベルをドラッグ&ドロップする機能のみを持たせたパレットを作成した．

保存する GINE オブジェクト情報の変更

現状の GUI では，構築された仮想ネットワークの情報を静的に作成された map コンテナに書き込んでいる．map コンテナ内のオブジェクト本体には，ラベルの位置情報等が含まれており，読み込んだ際に対応するラベルをパレットの適切な位置に配置できるようになっている．しかし拡張パレット機能を追加することでパレットが 1 つではなくなり，位置情報だけではどこのパレットに配置するかが分らなくなる問題が生じる．対策として，map コンテナを各パレット毎に用意し対応するコンテナに情報を保存する案と，オブジェクト本体に位置情報だけでなく配置するパレットの情報も含ませる案を提案した．前者の案は作成する拡張パレットの数だけ map コンテナを作成しなければならず処理が複雑になる為，後者の案を採用した．各オブジェクトに std::string 型の変数を作成し，対応するパレットの名前を代入することで，読出し時に適切なパレットに配置をすることが可能になった．

拡張パレット機能を GUI へ組み込む

実際に拡張パレットを開くには，ExtendedPalette ラベルの右クリックメニューから ”open extended palette ” を選択する．開かれた拡張パレットは対応する ExtendedPalette ラベルの名前をタイトルに持つ．開かれる際に map コンテナを読み込み，そのパレットと対応するラベルを拡張パレットに描くことで，一度パレットを閉

じても続きから描くことが出来るようになっている．次の図 7 で，右クリックメニューが表示されいることを示す．

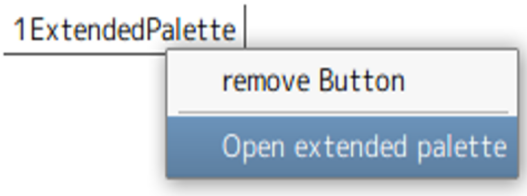

図 7　右クリックメニューの表示

6　システムの実験

本節では追加した GUI 機能の実験について説明する．

6.1　実験方法

本研究で追加した機能を用いて，エミュレーションできるかどうか実験する．まず拡張パレット機能を用いて仮想ネットワークを構築する．そして拡張パレット上で，構築した仮想ネットワークのエミュレーションが可能か確認する．次に，構築された仮想ネットワーク構成を部分に削除する．そして削除された仮想ネットワーク上で，エミュレーションが可能か確認する．確認するための操作は次の通りである．

- 構築した仮想ネットワーク上の Nspace から端末を開き，ifconfig コマンドを用いて，ネットワークインターフェースが生成されていることを確認する．
- ping コマンドを用いて，各 NSpace 間で通信できているかどうか確認する．

次の図 8 に，構築した仮想ネットワーク構成を示す．

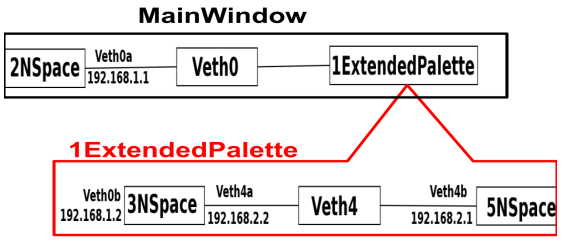

図 8　実験用仮想ネットワーク構成

6.2　実験結果

仮想ネットワーク構築後，各 NSpace の端末で ifconfig コマンドを用いて，インターフェース情報を確認した結果，設定した Veth のインターフェースが作られていた．次に 2NSpace の Veth0a に IP アドレス 192.168.1.1 を設定し，3NSpace の Veth0b に IP アドレス 192.168.1.2 を設定した．そして ping コマンドを用いて通信を行った結果，各 NSpace 間で通信ができていることが確認できた．次に Veth0 を remove ボタンで削除した．そして Veth0 と繋がっている，2NSpace と 3NSpace で ifconfig コマンドを用いて，インターフェース情報を確認した結果，設定した Veth インターフェースが削除されていた．次に 2NSpace で ping を用いて通信しようとした結果，connect: Network is unreachable という表示がでたことから，ping が送れないことを確認できた．上記の結果から，GINE に削除機能と拡張パレット機能を，適切に追加できてたことが分かった．

7　おわりに

本研究によりネットワークエミュレータ GINE に新機能である削除機能と拡張パレット機能が追加できた．削除機能を用いることで，GUI 上で構築した仮想ネットワークを部分的に削除することが可能になり，GINE の GUI 機能の操作性が向上した．また，拡張パレット機能を用いることで，一つのラベルで LAN などのネットワークを表すことができ，容易に大規模な仮想ネットワークを構築することができる．今後の課題を次に述べる．

- GUI にさらなる機能を追加し，様々な仮想ネットワークを構築できるようにする．次に追加する必要のある機能の一例を述べる．
  - GUI 上で星野&石野の卒業論文で提案された Quagga を操作する機能．
  - EGP 等を用いた実験をするための AS 機能．

本研究によって GINE の GUI の操作性が向上し，プログラムを書くことができないユーザにとっても，ネットワークを容易に学習することができるようになったことを期待する．そして今後，研究室の後輩らによってさらに GINE に新機能が搭載され，さらなる改良がなされることを期待する．

参考文献

[1] 浅野 洋介： GINE を用いた大規模ネットワークエミュレーションと管理機能の追加，修士論文，南山大学 大学院 数理情報研究科 数理情報専攻 (2010).

[2] Free Software Foundation： *Gnu common C++*, http://www.gnu.org/software/commoncpp/ (accessed Jan. 2010).

[3] 星野聡介，石野佑弥： 大規模ネットワークエミュレーションにおけるトポロジ構成と経路制御，卒業論文，南山大学 数理情報学部 情報通信学科 (2010).

[4] Y. Sugiyama and K. Goto : Design and implementation of a network emulator using virtual network stack, In *Proc. of the Seventh International Symposium on Operations Research and Its Applications (ISORA2008), Lecture Notes in Operations Research, Vol.8*, pp. pp.351–358, (Nov. 2008).