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

2003－ＱＡＩ－8　(1)
2003／7／7

Uniform Rendezvous Pointer: オーバーレイネットワーク間相互参照のための識別子空間

土井　裕介†　　石田　剛朗††　　斉藤　賢爾††　　門林　雄基†††

† 株式会社東芝 研究開発センター　〒212–8582 神奈川県川崎市幸区小向東芝町 1
†† 慶應義塾大学 政策・メディア研究科　〒252–8520 神奈川県藤沢市遠藤 5322
††† 奈良先端科学技術大学院大学 情報科学研究科　〒630–0101 奈良県生駒市高山町 8916–5
E-mail: †ydoi@isl.rdc.toshiba.co.jp, ††{ishi,ks91}@sfc.wide.ad.jp,
†††youki-k@is.aist-nara.ac.jp

あらまし　インターネット資源の再抽象化レイヤであるオーバーレイネットワーク統合への第一歩として、異なるオーバーレイ間で相互参照を行うための識別子空間、URP(Uniform Rendezvous Pointer) を提案する。まず、URP の必要性を議論した上で、求められる機能について考察する。具体的に、ネットワーク毎に異なるポリシーや識別子の検証可能性などについて評価し、あるべき URP の機能を提案する。

キーワード　オーバーレイネットワーク 異なるシステムの統合運用 ピアツーピア

Uniform Rendezvous Pointer: Identifier Space for Overlay Network Interreference

Yusuke DOI†, Takaaki ISHIDA††, Kenji SAITO††, and Youki KADOBAYASHI†††

† R&D Center, TOSHIBA Corp.　Komukai–Toshiba–Cho 1, Saiwai–Ku, Kawasaki, Kanagawa, 212–8582, JAPAN
†† Graduate School of Media and Governance　Endo 5322, Fujisawa, Kanagawa, 252–8520, JAPAN
††† Graduate School of Information Science　Takayama–Cho 8916–5, Ikoma, Nara, 630–0101, JAPAN
E-mail: †ydoi@isl.rdc.toshiba.co.jp, ††{ishi,ks91}@sfc.wide.ad.jp, †††youki-k@is.aist-nara.ac.jp

Abstract　As the first step towards application overlay network integration, we propose URP(Uniform Rendezvous Pointer). URP is the identifier space to point resources among different overlay networks. We discuss why we need URP and required functionalities for it. Especially, we evaluate some topics like policies, ability of verification, and so on. And finally we propose how URP should be.

Key words　Overlay network, Heterogeneous system integration, Peer-to-Peer

1. 背　　景

インターネットの普及とともに、その利用法の幅は広がりつづけている。特に、IP アドレスで表現できない資源を識別・表現するために、オーバーレイネットワーク (以下オーバーレイ) 等さまざま方法が考案されている。一方で、それらの協調動作のための枠組がほとんど存在しないか、存在したとしても限定的な解である、という問題がある。

本稿では、インターネットを用いて「統合分散環境」を実現するために、これらオーバーレイ同士を相互に接続する必要性について論じる。

1.1 「統合分散環境」に向けて

統合分散環境を、以下の 2 つの条件が成立する環境と定義する。

（1） **分散**: 全世界を継ぎ目無く統合するネットワークが存在し、その上に無数の資源が存在する。

（2） **統合**: 存在する無数の資源が、人間の目的達成を補助するために有機的に統合され、機能を提供する。

条件 1 は、IPv6 やネットワークアプライアンス等の普及により達成されると考えられる。一方で、条件 2 は、ネットワークを構成する資源の数や種類が多くなるにつれて困難になる。

条件 2 を達成するために、識別子による資源の表現能力に注目する。現在の Internet で多く用いられている識別子は、IP アドレス+ポート番号、そして URL である。IP アドレスはネットワークインターフェイスの識別子であり、これにポート番号を付ける事によってそのインターフェイスを持つ計算機上に存在するプロセスを指定する。同様に、URL は特定のプロセスが管理するファイルの識別子である。

これらの識別子は、「計算機上のプロセス」という粒度に縛られている。あらゆる資源を表現するために、その資源を管理するプロセスを指定する必要がある。一方、「実世界の人間」や「地理的な位置」等、特定のプロセスに強く関連づけられない種類の資源が次々とネットワークに登場しつつある。

また、必要な資源を効率的に発見するためには、資源の特徴的性質を記述できる識別子空間が必要となる。その空間は一意に定義できるものではなく、発見の目的に応じて異なる空間が設計される。実際、単純な名前、資源の性質やサービスインターフェイス、セキュリティポリシーや匿名性等、目的に応じて異なるシステムが作られている。そして、システムの数だけ識別子空間は存在し、互換性は無い。

1.2 オーバーレイ

オーバーレイは、多様な識別子空間を実現するための一つの手段である。IP を基礎にしたインターネット上に、アプリケーションが必要とする識別子空間を実現し、その上で資源への到達性を実現する。

各種インスタントメッセージングシステム (Jabber [3] や ICQ [4] 等) や IRC [5] 等は、人やプレゼンスといった資源の表現のために発達している。一方、Tenbin [8] 等は、コンテンツという資源を扱う。Chord [7] 等の分散ハッシュテーブル (DHT) は、ハッシュ値という抽象的な資源の表現を扱う。他、Freenet [1] 等、IP ノードに対する匿名性の適用を主目的としたものなどもある。なお、本稿ではオーバーレイ上のノードと区別するため、インターネット層におけるノードを IP ノードと表記する。

目的の数だけ資源の抽象化の手法は異なり、その結果多数のシステムが生まれている。本稿では、これらのシステム全てを「オーバーレイ」という枠組で捉える。オーバーレイは、IP アドレスで表現できない資源を抽象化して取り扱い、資源への通信機能を提供する。

1.3 オーバーレイ相互接続の要求

オーバーレイ上に抽象化された資源は、ネットワークを経由して利用可能である。これを抽象ノードと呼ぶ。しかし、抽象ノードとして利用できるだけでは統合分散環境には不十分であり、これら資源を統合する (節 1.1 条件 2) 必要がある。ネットワークの価値を高めるには、目的達成のために最適な資源を自由かつ創造的に選び取り、要求を満たす資源の組み合わせを選択する必要がある。この資源選択の自由を提供する機能を「自由で創造的なランデブー」と呼ぶ。

現状では、異なるオーバーレイ上の資源を組み合わせて活用する事には困難がある。一方で、「自由で創造的なランデブー」はこの困難を乗り越えた先にある。また、資源の選択がオーバーレイの内部に閉じていると、「全ての要求を満たすオーバーレイ」を作らなければ目的は満たされない。複雑な目的の達成に向けて、単一の目的で作られたオー

バーレイをモジュール的に組み合わせる事で、自由かつ創造的な資源の組み合わせを実現でき、かつ価値創造がオーバーレイの内部に閉じないので、自発的に多数の応用が生まれるものと考えている(3.5)。

オーバーレイをまたがる抽象ノードの統合的利用に向けた最初のステップとして、様々なオーバーレイ上の抽象ノードを指し示すための識別子空間、Uniform Rendezvous Pointer(URP) を定義する。

2. 複数オーバーレイの統合に向けて

本節では、まず複数オーバーレイを統合する事により得られる具体的なシナリオについて紹介する。その上で、複数オーバーレイ利用のモデルを定義し、モデル中に存在する資源を一意に識別するために必要な機能を考える。

URP に対する要求は、複数のオーバーレイにまたがる文脈において、それぞれのオーバーレイにある資源を区別し、指定できる事とする。

2.1 複数オーバーレイのモデル

複数の異なるオーバーレイを利用するモデルを考える。

オーバーレイには、様々な手法で抽象化された資源が独立・並列に存在する。

資源は一意に指定できない場合がある。例えば Freenet では、検索を開始する場所によって得られる内容が異なる。この場合、資源を示す代わりに資源の探し方を指定する他ない。

オーバーレイは、IP ノード同士が結合して構成する。この構成方法にはいくつかのパターンが認められる。

（1） 既定の初期 IP ノード (well-known bootstrap nodes) のみで情報共有し、他の IP ノードはクライアントとして構成

（2） 既定の初期 IP ノードを中心に自動構成

（3） 近隣の IP ノードとの自動構成

（4） システム外の経路 (Out of Band) によって隣接 IP ノードを手動構成

（5） DNS 等のディレクトリサービスへの依存により構成

手法 1 および 2 は、既定の初期 IP ノードをネットワーク構成の足掛かりとする。参加する全ての IP ノードは既定の初期 IP ノードを知っており、それら IP ノードの利用が前提となっている。

手法 3 は、リンク層ブロードキャスト等を利用した自動構成手法である。Apple Rendezvous や Sun Jini ではこの手法を利用している。

手法 4 は、一部の Gnutella 由来のシステムで利用される手法である。IRC や HTTP によって初期 IP ノードを別途取得してネットワークを構成する。

構成方法によって、全世界で共有されるネットワークが一つできる (手法 1、2、5) のか、それとも特定の 初期 IP ノードや物理リンクに関連づけられた独立したオーバーレイが多数発生する (手法 2、3) のか、という違いがある。本稿では、前者をユニバーサルネットワーク、後者をコミュニティネットワークと呼ぶ。また、コミュニティネットワークにおけるそれぞれの断絶したオーバーレイをコミュニティと呼ぶ。

コミュニティやオーバーレイは接続者にそれぞれ異なる要求を行う可能性がある。具体的には、ただ乗り (free riding) を防ぐために何らかの帯域制御を行う、サービスに対価を要求する、認証を要求する、あるいは匿名性を保証するなどといったポリシーである。同一のオーバーレイに複数のポリシーが混在する事も有り得るし、ポリシー毎に異なるコミュニティを形成する場合もある。

当然、ネットワークサービスである以上、共通のプロトコルを利用する事は大前提となる。

以上、本稿では、オーバーレイを以下の 4 要素に整理したモデルを考察の基礎とする。

（1） 抽象ノードの抽象

（2） ネットワーク構成/コミュニティ

（3） ポリシー

（4） プロトコル

2.2 資源指定方法の考察

本節では、前節で述べたモデルにおいて資源を指定するための方法について、どのような場面でどのような指定方法必要になるか考察する。

2.2.1 アクセス手段の指定

URP は、多様なオーバーレイ上に抽象化された抽象ノードを指し示す。抽象ノード利用には指し示すだけでは不十分であり、抽象ノードに対してどのプロトコルを利用するか指定できる必要がある。

従来の URI 式の数文字のプロトコル識別子 (`chord`、`freenet` 等) による指定方式の他に、より精密にプロトコルを示す指摘方式が必要な場合がある。例えば、Jini の様に、抽象ノードにアクセ

スするためのスタブクラスの定義により資源を抽象化する場合等がある。

2.2.2 ポリシー情報の提供

ネットワークへの参加の際、アイデンティティを示す必要や、資源の提供を求められる場合等が考えられる。

このような場合、IP ノードはポリシーに基づいて振舞うべきなのか知る事により、事前に資源を用意できる。また、ポリシーに従えないのであれば、無駄なアクセスのための時間・帯域の消費がなくなる。

2.2.3 コミュニティの指定

例えば、オンラインゲームのためのオーバーレイは、規模性やプレイヤーの快適性等を考えて、複数の「ワールド (=コミュニティ)」を持つ場合がある。

このような状況下で抽象ノードを識別するには、抽象ノードが属するコミュニティを指定する必要がある。コミュニティを示すには、初期 IP ノードの IP アドレスを示す等の方法がある。

2.2.4 依存/外部参照の指定

オーバーレイは他の情報に密に結合している場合がある。例えば、特定の地域に存在する抽象ノード群を抽象化したオーバーレイを考えた場合、地域を指し示す記述から正しいコミュニティを選択する事になる。このような場合、初期 IP ノードを外部のディレクトリサービス、例えば DNS や DHT などを利用し発見する。そのための手掛りを URP に含められると良い。

2.2.5 資源成立条件の限定

あるオーバーレイにおいて、時間経過した後もその内部の資源が有効である保証は無いし、オーバーレイのどの IP ノードから資源探索しても同じ資源に到達する保証も無い。

資源の有効性や同一性を確認するためには、資源が意味を持つための条件を示す方法がある。識別子に有効期限を与え、その期間のみ有効性を保証する方法や、MD5 チェックサムによって内容の同一性を確認する方法などが考えられる。

2.2.6 ユーザ識別子の記述

特定のオーバーレイに対して、「何者として」アクセスするのかを指示する識別子は、アクセス制御を簡略化できる。

2.2.7 Uniform, not Universal

URP は、全世界 (Universal) の資源を統一的に表現するものではない。ある特定の環境においてのみ意味を持つ識別子、文脈に応じて示す資源が変わる識別子が存在する事を許す。

一方で、URP はどのような環境においても画一的 (Uniform) に生成・解釈できる必要がある。

3. 提案する URP

ここまでの議論を受けて、複数のオーバーレイの中で抽象ノードを識別し、かつオーバーレイの多様さを柔軟に吸収できるポインタを提案する。

3.1 URP により示されるもの

節 2.2.5 での議論にあるように、オーバーレイ上の抽象ノードは一意に特定できる場合もあれば、そうでない場合もある。従って、URP は抽象ノードの位置を直接示す事を前提にできない。この制約下で抽象ノードを示すために、抽象ノードの探索方法を示すという方法が利用できるべきである。

従って URP は、「探索方法」を示す事により抽象ノードを指示する識別子である、と言える。

3.2 要素の検討

ここまでの議論に登場した、オーバーレイの要素を整理すると以下のようになる。

「何を利用して探索するか」:

- `accessor`: プロトコル識別子

「どこを探索するか」:

- `community`: 初期 IP ノード/ 外部参照

「何を探索するのか」:

- `resource_id`: 実際の資源を識別する記述

「どのように探索するか」:

- `identity`: アクセス者のアイデンティティ
- `condition`: 資源探索成立の条件
- `preference`: ポリシー要求等

資源を同定するため、最低限、`accessor`、`community`、`resource_id` が必要である。「どのように探索するか」は必要に応じて利用すれば良い。

3.3 表記案

URLの表記方法に則り、基本形は以下のように提案する。

```
{accessor}://{community}/{resource_id}
```

また、全ての要素を記述するために、以下の表記を提案する。

```
{accessor}://{identity}@{community}
?{condition}!{preference}/{resource_id}
```

(改行は含まない)

URPの各項目について以下に述べる。

「何を利用して探索するか」: accessorは、資源へのアクセス方法を規定する。アクセス方法の規定には (1) 明文化されたプロトコルによるアクセス (2) 抽象クラス定義によるアクセス の2通りが考えられる。

明文化されたプロトコルによるアクセスへのaccessorは、従来のURLにおけるhttpあるいはftp等のschemeと同様、短い識別子による記述を行う (例: chord can iwat 等)。

一方、「抽象クラス定義によるアクセス」は、未知ないしアクセス方法が一定しない資源へのaccessorを提供する。URPによって表現される資源の利用法はある程度抽象化できる可能性がある。例えば、getやput等の基本的なデータ操作がある。このような抽象化されたデータ操作を、抽象クラスとして共有した上で、資源へのaccessorとしての具象クラスをネットワーク経由で供給できる。抽象クラスの定義については節4.6で議論する。

「どこを探索するか」: communityは、オーバーレイに接続するために必要な情報である。内容はオーバーレイの構成方法 (2.1参照) によって異なる。既定の初期IPノードを持つ場合は空欄で良いし、DNS等の外部参照を利用する場合はその参照に必要な式ないし名前を記述する。

「何を探索するか」: 各オーバーレイ内部で、資源を示す識別子である。オーバーレイに対応して、抽象ノードの識別子、ファイル名、問い合わせ文字列等が書かれる。

「どのように探索するか」: identityは、{userid}:{credential}、あるいは{userid}のみになる。useridは各オーバーレイ固有のユーザ識別子である。credentialは、識別子に対応づけられた秘密である。

例えば、ファイルのMD5チェックサムをconditionとしてURPに含める場合は、md5sum=16136a7....のように記述する。

また、URPによる資源探索プロセスに対価 (地域通貨・資源等) が要求される場合がある。それら資源をどこまで提供するかここに記述できる。例えば、利用する通貨単位がmh(Man Hour)だった場合、conditionにcost<0.4mhとして記述する事で、URPによる資源探索は0.4mh以下で終了しなければならない (さもなければ中断)、という記述が可能になる。

preferenceには、資源を探索する際、資源利用者側に選択肢がある事柄について記述する。例えば匿名性が実現可能なネットワークでは、利用者が匿名であるか無いかを選択できる。探索方法を指定するURPにおいてこういった選択肢を記述する事は、簡易な資源利用を促すことができる。

また、自発的な資源の提供を意味する記述もpreferenceに適当だろう。

3.4 提案するURPによるモデルの充足

節2.1で述べた複数オーバーレイモデルに対して、提案するURPは必要な資源を指し示せるかを考察する。

まず、オーバーレイの構成手法 (モデル要素2) について考える。ユニバーサルネットワークではオーバーレイの構成のために必要な情報は既定のものとして与えられるため、URPの記法に関わらずネットワークを構成可能である。一方、初期IPノードが必要なコミュニティネットワークでは、communityより初期IPノードを伝達できる。

ネットワークのポリシーの記述 (モデル要素3) は、preferenceあるいはconditionによって行う。preferenceは、計算資源の提供や匿名性など、確定的な意思を記述し、conditionはファイルのチェックサムや支払える対価など、条件を記述する。

プロトコルの記述 (モデル要素4は、accessorによって行う。accessorは同時に、抽象ノードの抽象 (モデル要素1を定める。chordにとっては抽象ノードは単なるデータであるが、通貨取り引きのためのプロトコルを利用する場合は、通貨あるいは取り引きを行う者を抽象する。

しかし、accessorによって抽象ノードの抽象とプロトコルの両方を定めて良いかについては、議論の余地がある (節4.6)。

3.5 具体的応用例

ファイル共有を行うオーバーレイは多数存在する。それらには、ファイルを共有するためのコスト(HDD 領域・通信帯域・CPU 時間・電気料金等) や独自のコンテンツの提供に対するインセンティブが乏しいという問題がある。このようなオーバーレイを、ここではファイル共有オーバーレイと呼ぶ。

インセンティブの問題を解決するために設計された Mojonation では、内部にファイル交換と対価の支払いの仕組みの両方を持つオーバーレイを構築した。ファイル公開や交換のための資源の提供に対して対価の支払いを求める事によって、より質の高い資源の提供を促進するという狙いであった。しかし、この統合されたシステムは普及に困難があり、オーバーレイとしては成功しなかった。結局、対価のやりとりをあるオーバーレイの内部のみで行う事により価値創造をその中に限定してしまい、インセンティブとして成り立たなかったものと考えられる。

これに対して、対価のやりとりを行うオーバーレイをファイル交換オーバーレイと結合させて利用するというアプローチが考えられる。例えば、対価の一つとして「手形」のような地域通貨を扱うオーバーレイ [6] が存在する。このようなオーバーレイをここでは対価交換オーバーレイと呼ぶ。

この 2 つのオーバーレイを統合して利用できれば、ファイル交換と対価の支払いを対応づけを行う事は容易である。例えば、先行して普及すると考えられるファイル交換オーバーレイに対して、対価交換オーバーレイを扱えるプラグインソフトを導入する。対価交換オーバーレイにおける相手の信用に関する問い合わせや価格の擦り合わせ等の情報を、ファイル交換オーバーレイ上で URP により交換する。URP によってこのプラグインソフトが呼び出され対価交換が成立する、といったシナリオが考えられる。

図 1 は、このシナリオの模式図である。2 人の利用者が、ファイル交換オーバーレイと対価交換オーバーレイを同時に使用して、ファイルの交換と対価交換を URP によって対応づけている様子を示している。対価交換オーバーレイはファイル交換オーバーレイと独立に存在しているので、ここで交換された価値は URP によって広く再利用できる。すなわち、URP により価値創造の領域が広が

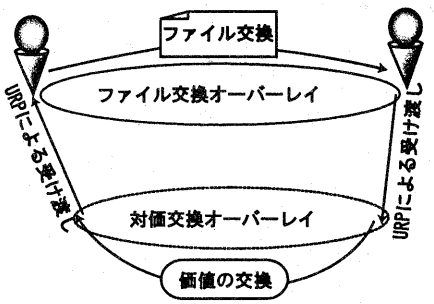

図 1 ファイル交換オーバーレイと対価交換オーバーレイの URP による統合 (模式図)

Fig. 1 Integration of file exchange overlay(upper) and value exchange overlay(lower). Independent value exchange overlay is used to extend ability of file exchange overlay as exchanged URP can bind them together.

り、交換される対価をインセンティブとして機能させる事が容易になる。

このように、単独のオーバーレイで全ての仕組みを実現するのではなく、自由で開かれた、創造的なランデブーのための空間を用意し、単純なオーバーレイをモジュールとして連携させる事により、応用が自己創出的に発展していく事が期待できる。

4. 議論: オーバーレイ統合に向けて

URP により、オーバーレイ等の利用法、あるいはオーバーレイによる通信アーキテクチャにどのような影響があるか、以下に考察する。

4.1 利点: 記述の容易さ

URP により、オーバーレイへのアクセス方法の記述が容易になる。ユーザの目に見える範囲では、メールや口頭での通知等が容易に行える。実際のアクセスを行うプログラムがアクセス方法を判断すれば良いため、利用者に対する負荷が軽くなる。

4.2 URP の応用可能性

URP は複数のオーバーレイを連携せせる事を目的として作られた。複数のオーバーレイによる連携には、節 3.5 で述べたように、対価交換のような 2 者の直接即時的な連携が考えられる他、より大きな規模での連携を考える事もできる。

例えば、URP を収集する検索エンジンのクロウラー (収集プログラム) を考えてみよう。ある曖昧な検索式に応じて、具体的な「資源の探し方」のリストを提供する検索エンジンは、多様なオーバー

レイの中に資源が抽象化されて存在する世界の利便性を確実に向上させる。

また、URP により、異なるオーバーレイ中に表現された抽象ノードの羅列を表現できる。具体的には、インスタントメッセージングにおける友人リストのようなものに、AIM、MSN Messenger、Jabber 等の利用者を URP によって表現し、格納でき、また単一のクライアントプログラム (あるいは他のプログラムを駆動するメタ・プログラム) に対して URP を渡す事で、適切なプロトコルでメッセージをやりとりできる。

このように、URP によって、オーバーレイに存在する「囲い込み」を乗り越え、オーバーレイ選択に自由と柔軟性を導入できる。筆者らは、この自由と柔軟性は、ネットワーク環境を「統合分散環境」に進めるために必須の性質の一つであると考えている。

4.3 議論:「秘密の共有」の崩壊

匿名掲示板などを「コミュニケーション」の切口で分析すると、そこには「秘密の共有」と「自己顕示」という二つの側面が出てくる。特に秘密の共有は、自己と相手の心理的な相互信頼を確立するための手段となっている。コミュニケーションを目的とするオーバーレイでも同様である。つまり、URP が指し示すような資源を発見する事に価値を見いだし、発見した者を「仲間」として迎え入れるような人的コミュニティの形があるかもしれない。

こういった人的コミュニティにとって、URP のような簡易なアクセス手段の記述は、共有される秘密へのアクセスを容易にし、相対的にその秘密の価値を下げるのだろうか。

また、URP により「検索」がオーバーレイに対しても可能になった場合の影響についても考察が必要だろう。Google 等の検索エンジンによって、個人的なページやそのリンク集を経由した情報の探索 (俗にネットサーフと呼ばれていた方法) は減少した。検索エンジンは、URL によって無数の情報を発見し、検索者に提供できるようになった。その結果、WWW 上にひっそりと存在した人的コミュニティは一気に白日に晒され、その一部は公衆化する事を嫌い、あるいはそもそも非合法故に WWW の外 (ファイル共有ソフト等) に潜った。

URP によって、多数のオーバーレイが現在の WWW と同様にアクセス可能になった場合、こういった人的コミュニティはどこに行くのか、どういったインパクトを与えるのか等も考える必要があるだろう。

4.4 普及への障害: 複数方式への対応

URP によって簡易に資源を指定しできるようになっても、その資源が簡便に利用できるだけの機構が存在しなければ、URP によるメリットは乏しくなる。一方、将来いくら可能性があろうとも、現状でメリットが乏しいものを実装するインセンティブは低い。既存のフレームワークの URL 処理部への統合等、URP を利用する環境を発展的に成長させるための現実的な壁を壊していく手段を考える必要がある。

4.5 議論: URP 利用者の情報

`condition` や `preference` はあくまで利用者個々の事情であり、資源へのポインタの伝達手段である URP に含めるべきでは無いとする考え方もある。しかし、筆者は敢えて URP にこういった情報を入れるべきであると考える。なぜならばオーバーレイはあらゆる意味で不均一である可能性が高いからである。

例えば、同一のオーバーレイのプロトコルにおいて、匿名が「望まれる」場合と、実名が「望まれる」場合がある。実際に匿名で参加するか実名で参加するかは利用者が決めて良い事である一方、URP を伝える側がオーバーレイ側の希望を伝える経路があっても良い筈だ。これは匿名性に限らず、資源提供の緩いポリシーの記述にも役立つ。

一方、利用者側のクライアントプログラムには、`preference` および `condition` を評価し、場合によっては URP による資源探索をブロックしたり、これらのフィールドをユーザの要求に合うように書き変えたりする機構が必要になるだろう。これはブラウザにおける P3P [2] のようなものになるかもしれない。

4.6 議論: 抽象ノードの抽象と操作

オーバーレイにおける抽象ノードの抽象 (クラス化) およびその抽象に対する操作 (メソッド) については、さらなる議論が必要である。

オーバーレイにおいては、様々な資源が様々な軸によって抽象化され、提供される。これら抽象化された抽象ノードに対する操作にはどのようなものがあるか、オーバーレイによって異なる可能性がある。

一方、共通の資源の抽象クラスを導入する事で、URP における資源の扱いを画一化できる可能性がある。この場合、accessor としてプラグイン (あるいはスタブクラス) を用意する事で、ありとあらゆる資源にアクセスする機構をネットワーク側から取得できる事になる。但し、この方法は資源のあり方に制約をかける事になる。

また、URP に操作を記述すべきかという議論がある。URP によって資源と探索者をランデブーさせるが、ランデブーした後の操作は記述できなくても良いのか。URL の場合は、GET 要求なのか POST 要求なのか、あるいは HEAD 要求なのかは文脈で決定できたが、その前提を URP に対して仮定して良いのか。例えば、accessor として chord.join://... 等と書ける事によるメリットがあるかもしれない。

4.7 今後の課題

効果の検証が第一の課題である。そのためには、実際に URP を応用するシステムを作る事が望ましい。具体的には、節 3.5 で述べた応用例のシステム試作を目指す。

また、並行して、URP の取り扱いに関する API 等、利用環境を定める作業が必要である。URL を扱うルーチンに取り入れるにはどうすべきか等も検討する予定である。

5. 結　　論

本稿では、目的別に生じたオーバーレイの自然な発展としての統合において、URP が必要となる理由について述べた。その上で、URP に必要かもしれないいくつかの機能を列挙し、求められる条件を満たすためにどの機能が必須であり、どの機能がオプショナルかを考察した。

当然、本稿で述べた URP は、現状存在するオーバーレイに対する限定された考察に基づく以上、将来発生する全てのネットワークをカバーしているという保証は無い。一方で、Universal でないが Uniform であるポインタおよびその記述方法がオーバーレイ統合に寄与する事は本稿で述べた。

本稿で述べたような、実際に取り扱う様々な性質の資源を様々なオーバーレイの形で抽象化する、というアーキテクチャは未だに十分成立しているとは言えない。今後の研究課題として、オーバーレイをより積極的に活用したアーキテクチャを実際に作成し、「統合分散環境」としての評価を行っていく必要がある。

文　　献

[1] I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong. Freenet: A distributed anonymous infor-mation storage and retrieval system. In *Pin Designing Privacy Enhancing Technologies*, 2001.

[2] Lorrie Cranor, Marc Langheinrich, Massimo Marchiori, Martin Presler-Marshall, and Joseph Reagle. The platform for privacy preferences 1.0 (p3p1.0) specification. W3C Recommendation, April 2002.

[3] Jabber Software Foundation. Jabber.org website. http://www.jabber.org/.

[4] ICQ Inc. Icq website. http://web.icq.com/.

[5] C. Kalt. *Internet Relay Chat: Architecture*, April 2000. RFC 2810.

[6] Kenji Saito. Peer-to-peer money: Free currency over the Internet. In *Proceedings of the Second International Human.Society@Internet Conference (HSI 2003), Lecture Notes in Computer Science*. Springer-Verlag, 2003. (to appear).

[7] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM*, August 2001.

[8] 下川俊彦, 吉田紀彦, 牛島和夫. ネームサーバを用いた柔軟な負荷分散. インターネットコンファレンス '99, pp. 107–116, December 1999.