研究計画書
Auto IDシステムにおけるIDマッピング機構に関する研究

慶應義塾大学環境情報学部

自署：_____

学籍番号 70058118

平成 15 年 9 月 29 日

概要

RFID の普及により個体認識を行う技術が展開しつつある。既存の個体認識技術は、SCM や入室管理など、それぞれが単一の目的で利用されている。製品の製造段階で EPC と呼ばれる RFID タグを貼付し、製造から流通、小売に至る様々な状況で、汎用的な管理を目指す Auto-ID Center のような研究も行われている。
Auto-ID Center の展開する機構は、インターネット上に構築される。商品に関する様々な情報は、PML で記述されるため、読み取った RFID に対応する情報が格納される PML Server を発見する機構が必要となる。
Auto-ID Center では、この ID から IP アドレスへの変換機構を ONS として提案している。
提案されている ONS では、ID を基に生成した DNS クエリを DNS サーバ群に発行することにより IP アドレスを得る。この機構では、様々な製品に EPC をつけ、莫大な数の製品が流通する状況では、規模性が実現されない恐れがある。
本研究では、DNS を利用せずに ONS と同等の機能を持つ機構を提案する。DNS の代わりに分散ハッシュテーブルを用いることで、中央のインデックスにクエリが集中することを避け、規模性を実現する。
本研究により、あらゆる製品に ID を割り振り、その ID および付帯する情報を利用する様々なサービスを自由に構築することが可能となる。

1 背景

近年の RFID(Radio Frequency IDentification) の普及により個体認識を行う技術が展開しつつある。既存の個体認識技術は、SCM(Supply Chain Management) や入室管理などに利用されている。このように場面ごとに独立管理されている既存の技術を統合し、EPC(Electronic Product Code)[1] を製品につけることで一元的な管理を目指す MIT の Auto ID Center[2] のような研究が注目されている。

Auto ID システムでは、SCM における一つ一つの製品にグローバルユニークな ID をつけて製品の情報を管理している。つまり、SCM に関わるすべての製品の数だけ ID が必要になる。ID そのものに製品の情報を載せることはできない。よって、ネットワーク上に Auto ID のシステムを展開しようとした場合、製品の情報はネットワーク上のどこかのサーバに保持される。そのため、製品に関する情報を保持したサーバと製品についた ID とを結びつけ、膨大な ID 空間を管理できる機構が必要となる。

2 研究目的

本研究では、世の中の製品一つ一つに RFID を用いて一意な ID が割り振られた環境を想定する。前提として Auto ID システムではタグそのものに情報を記憶する機能はない。そして現在の Auto ID システムでは、製品についた ID から、その製品の持つ位置情報、温度情報などをインターネット上で扱える機構の議論が行われている。本研究ではそのような環境において必要不可欠な ID と製品のメタ情報を保持したサーバとのマッピングを行うシステムの構築を行う。

3 現在提案されている Auto ID システムの概要

現在 Auto ID Center で提案されているシステム概要を図 1 に示す。Auto ID システムは製品の近傍に展開する EPC タグ、リーダ、Savant、インターネット上に展開する ONS、および PML Server からなる。ID は EPC タグに格納され、PML Server で管理される情報は Savant で利用

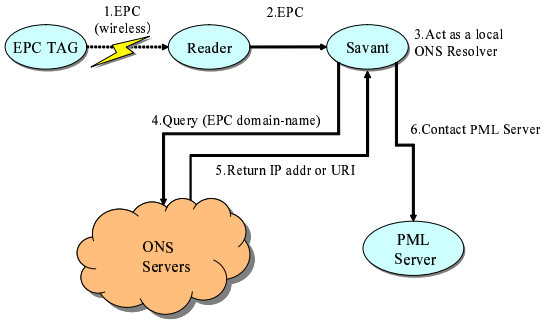

図 1: Auto ID におけるアーキテクチャ概要図

される。以下に図中の用語についての概要を示す。

- EPC および EPC リーダ
  バーコードのように製品の識別子として Auto ID Center で提案された次世代の製品コード体系。一意性が保障され製品の製造段階で貼り付けされる。また、ID は上書きさせない。具体的には"01.0000A89.00016F.000169DC0"のように 96bit の数字で表せる。EPC リーダはこの ID を無線で自動検出する役割を担う。
- Savant
  Savant[3] は EPC リーダが読み取った EPC のデータ取得、モニタリング、タスク管理をするソフトウェアである。ONS リゾルバとしての側面も併せ持ち、EPC から DNS クエリを生成し PML サーバのアドレスを取得する。
- PML(Physical Markup Language)
  PML[4] は製品の温度、位置などの属性情報を記述するための言語である。PML サーバは PML を用いて製品の属性情報を保持する。
- ONS(Object Name Service)
  ONS[5] は製品についた EPC とその製品自体の情報を保持した PML サーバの IP アドレスとのマッピングを行う。Savant で生成されたクエリを受信し、DNS を用いた処理を行い、目的の PML サーバの IP アドレスを取得する。
- Translation Format String
  ONS に対して PML サーバの IP アドレスを問い合わせるときに、DNS クエリ文字列を EPC から変換するためのビット列で 0 から 4 までの数字で構成される。

次に動作手順を述べる。まず最初に製品に付けられた EPC タグからタグのリーダーを通して EPC を検知し、その ID 情報を Savant に渡す。次に Savant は Translation Format String を用いて EPC を DNS クエリに変換する。Savant は ONS に対して生成した DNS クエリを ONS に送る。ONS はその EPC に関する PML を保持した PML サーバの IP アドレスを Savant に送る。IP アドレスを受け取った Savant は PML サーバから EPC に関する PML を取得する。

4 Auto ID 導入による影響

次に Auto ID システムを物流の世界に当てはめ、具体的な場面を想定して考えた後、問題点を整理する。

4.1 各種業態による ID 使用の規模概算

我々に身近なところの例として飲料缶について考える。近年のペットボトルの台頭により日本国内の飲料缶の生産量はやや頭打ち状態にあるものの、2000 年の統計では 580 億本の飲料缶が生産されている。これを単純計算すると一日あたり約 1 億 6000 万本の飲料缶が生産されていることになる。飲料缶だけを考えても一ヶ月強で現在の IPv4 アドレス空間を上回る程の ID を必要としている。しかし、これだけの数にしても SCM の世界から見ればごく一部分に過ぎず、製品に ID をつけて管理する時に、いかに膨大な ID 空間を管理する必要があるかが容易に想像できる。

4.2 DNS を利用した ONS の問題点

本節では DNS を用いて ONS を動かした場合に考えられることについて検討する。

DNS の名前空間はツリー構造を用いて階層的に管理されている。この構造は Domain ツリーを頂上の Root Domain から下に辿っていくことで名前解決を実現している。しかし、このツリー構造は最初の始点が一つであるため、ここが機能しなくなるとシステム全体が機能しなくなる問題を抱える。事実、2002 年 10 月 21 日、世界の 13 個の Root Name Server に一斉に DoS 攻撃が仕掛けたれたことは記憶に新しい。[6]。このときは 7 台の Root Name Server に大きな影響が出たという事例がある。

5　問題解決へのアプローチ

上で述べたように物流をはじめとした実空間のオブジェクト一つ一つに ID を割り振るには膨大な ID 空間が必要となる。ONS にはその膨大な数の識別子を処理できる規模性が必要となる。つまり、規模性を持った ONS の構築は Auto ID システムにとって必要不可欠である。

5.1　機能要求

ここでは ONS 実現のために満たすべき要件を述べる。

- 互換性
  ONS 以外の機構の枠組みを大きく改変することなくシステムを構築する必要がある。
- Single Point of Failure の回避
  前章で述べたように DNS のは、Root Name Server が機能しなくなると、システム全体が機能しなくなる。
- スケーラビリティ
  前節で具体的な数字を示したように Auto ID システムでは膨大な量のオブジェクトを判別するための ID 空間を必要とする。96bit の ID 空間が果たして十分かという問題に関してはさらに議論を重ねる必要があるが、少なくともこの膨大な ID 空間でも運用管理ができることは必要不可欠である。具体的に記すと、EPC と PML とのマッピングに関して、PML サーバの検索、およびアドレスの登録を行えるシステムの構築は、Auto ID システムにとって必要不可欠である。

5.2　Distributed Hash Table

Distributed Hash Table(分散ハッシュテーブル、DHT) とは DHT に参加するノードそれぞれが共通のアルゴリズムを持ってハッシュ空間の分割管理を行い、データは同様のアルゴリズムによってハッシュ空間の一点に関連付けられるというものである。これによりどのデータがどのノードに存在するのかということが、中央のインデックスサーバを利用せずに分散して特定可能になり、かつ規模拡張性の非常に高い分散検索が実現可能となる。現在 Peer to Peer の分野における Chord[7] や Freenet[8] といったプロジェクトでも規模性を実現する手法としても利用、研究されている。

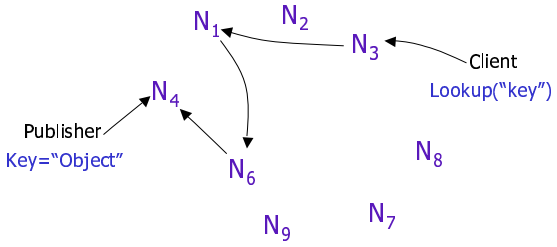

図 2: 分散ハッシュを使った検索モデル

図 2 に、その簡単な動作を示す。図中の Publisher は目的のデータを保持している。データの名前をハッシュ計算し、あるノードに関連付ける。クライアントは目的のデータ名のキーとしてハッシュ計算し、各ホストが持つルーティング情報を検索して目的のホストに到達する。

6　設計

上で述べた機能要求をもとに，分散ハッシュテーブルを用いる ID マッピング機構について述べる。

6.1　アーキテクチャ概要

本研究では分散ハッシュを階層化させることで、ID 空間を可変長に分割できる。それによりデータベースの検索効率を上げる。図 3 に本研究で提案するシステムのアーキテクチャ概要図を示す。

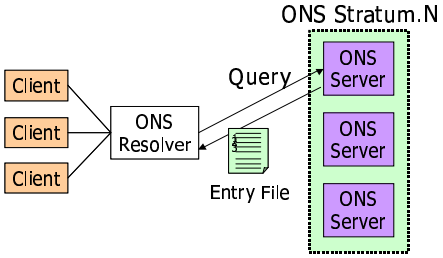

図 3: 提案するアーキテクチャ概要図

本システムアーキテクチャにはクライアント、リゾルバ Server、ある階層上 (図 3 では第 N 階層) にある ONS Stratum.N からなる。

- クライアント
  クライアントは読み取った EPC をそのままの状態でリゾルバに送る。

- ONS リゾルバ
  ONS リゾルバはクライアントから EPC を受け取る。受け取った EPC のビット列に対して先頭から数ビットずつを区切って、その区切りごとにハッシュ関数にかける。求められたハッシュ値とエントリファイル (後述) が一致するエントリを見つけ、一つ下の階層のどのサーバに問い合わせるかを知る。
- ONS Stratum
  本システムでは、ハッシュリストにより階層ごとに分けられた ONS サーバ群を一つのまとまりとして”ONS Stratum”と呼ぶことにする。その後に 1 回目の ONS リゾルバの問い合わせから n 回目に検索対象とされたといたサーバ群であることを表す N を付けている。
- ONS Server
  一つ下の階層にある問い合わせ先のリストと何ビットをハッシュにかけるのかというビット長情報をリゾルバに送る。リゾルバにエントリファイルとハッシュ計算に用いるビット長を提供する。エントリファイルは、ハッシュ値を対応するノードの IP アドレスのリストからなる。

6.2 動作概要

本研究では、ONS の課題を考慮して分散型のシステムである DHT を利用する。図 4 に処理の流れを示す。

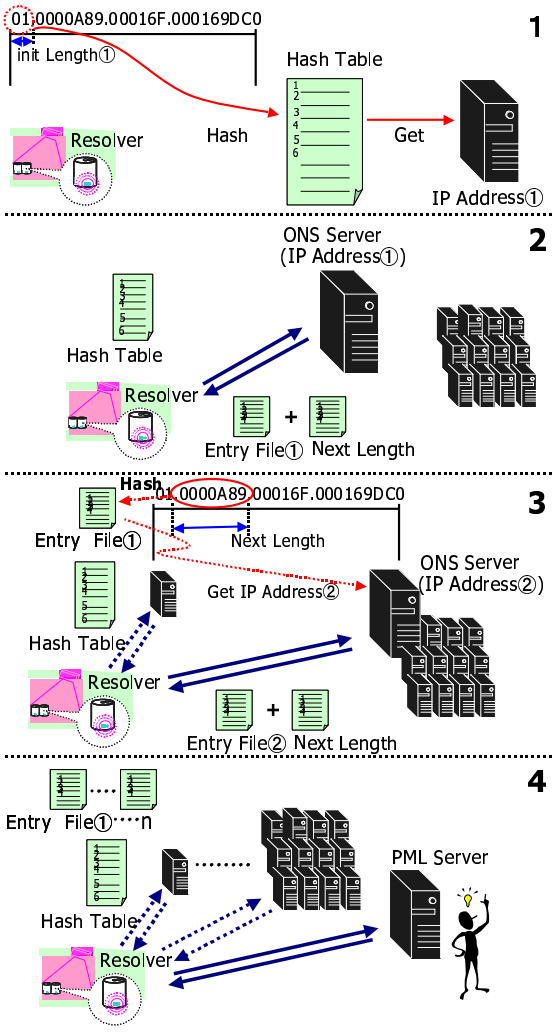

図 4: 提案するモデルでの処理の流れ

1. ONS リゾルバはあらかじめ予約されたビット長 (図 5 での init Length) 分の EPC ビット列をハッシュ計算し、ハッシュ値を求める。保持しているエントリファイルを参照し、次に問い合わせすべき ONS サーバの IP アドレスを取得する。

2. 取得した IP アドレスの ONS サーバにリゾルバは問い合わせを行い、ONS サーバが保持している次に問い合わせるべきサーバのアドレス群を記したエントリファイルと次に何ビット分ハッシュ計算するかを示すビット長 (図 5 での Next Length) を取得する。

3. リゾルバは取得したビット長をハッシュにかけ、求まったハッシュ値とエントリファイルとを比較し、次に問い合わせをすべき ONS サーバの IP アドレスを得る。このとき、同時にエントリファイルをローカルにキャッシュして検索効率を高める。

4. リゾルバは上記 3 の作業を繰り返し PML サーバのアドレスを解決する。

6.3 考慮すべき課題

上記のモデルを実現するにあたり、以下のような問題を検討する必要がある。

- 動的な情報更新に対する解決手法
  リゾルバはエントリファイルをキャッシュする。そして、そのキャッシュの有効期限を変えることで、ONS の動的な更新と階層構造の変化に対応できるのかもしれない。

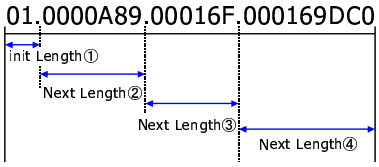

図 5: ハッシュ関数におけるビット長イメージ

- どれくらいのアクセスに耐えられるのか
- リゾルバが ONS サーバではなく PML サーバであると知る手がかりとして、ONS サーバから戻ってきた情報が IP アドレスであるか、または Next Length が 0 より小さいとき、などが考えられる
- DHT を階層化させることによるメリット
  本研究で提案するモデルでは、DHT を階層化することによってアドレス空間の大きさを部分ごとに変えることを可能としている。これにより ID 空間を必要な大きさごとに分け、規模性を確保する。

7 関連研究

- Chord
  Chord Project は P2P システム上で、拡張性に優れたシステムの構築を目的として研究を行っている [9]。Chord では、各 Peer ごとに DHT を持たせ、N 個のノードがあるシステムにおいて検索時間を log(N) のオーダーにまで抑えることを提案している。

8 本研究による貢献

実空間内のありとあらゆる製品に一意に識別子を割り振り、製品がもつ情報をいろいろな形で管理しようという試みが社会的に広く認識されるようになった。

MIT の Auto ID Center が展開する Auto ID System もその一つで、実空間内の製品の情報をインターネット上で管理する機構には ID と製品の情報を結びつけるためのシステム構築は必要不可欠である。

本研究により、一意に製品に ID を割り振り、その ID およびその ID のついた製品に付帯する情報を利用する様々なサービスを自由に構築することが可能となる。

9 これまでの研究活動

学部 3 年時より Auto ID に関する輪講や WIDE Project[10] の Spears Working Group に参加し RFID を使った実空間ネットワークにおけるアプリケーション実験に参加するなどの研究活動を行ってきた。今年の春は RFID によって人の位置を検知し、自動でチャットスペースや仮想共有スペースなどに招待をかけるという、ネットワークを利用して、実空間でのランデヴーを提供するアプリケーションの開発に携わった。

10 政策・メディア研究科に進学を志望する理由

今後 IPv6 や RFID の普及が進み、我々の日常生活のありとあらゆるものが ID を持ち、ネットワーク上で管理されていくことが予想される。そのような世界における新しいシステムのひとつとして着目されている Auto ID の技術、システムにはまだ様々な問題が残されている。
Auto ID の技術について研究を進めていこうと考えている私にとって、Auto ID Center アジア初の拠点が設置された SFC の環境は必要不可欠である。
以上の理由から私は政策・メディア研究科への進学を強く希望する。

参考文献

[1] David L.Brock,
”The Electronic Product Code(EPC) A Naming Scheme For Physical Objects”, 1/1/2001

[2] MIT Auto ID Center, 1999,
http://www.autoidcenter.org

[3] Oat Systems, MIT AutoID Center, ”The Savant - Version 0.1 (Alpha)” Oat Systems & MIT Auto-ID Center, 2/1/2002

[4] David L. Brock, ”The Physical Markup Language - A Universal Language for Physical Objects”, 2/1/2001

[5] Oat Systems, MIT AutoID Center, ”The Object Name Service - Version 0.5 (Beta) Oat Systems” & MIT Auto-ID Center, 2/1/2002

[6] ルートサーバへの一斉 DoS 攻撃,
http://www.zdnet.co.jp/news/0210/23/njbt_01.html

[7] The Chord Project, *http://www.pdos.lcs.mit.edu/chord/*

[8] The Freenet Project, *http://freenetproject.org/*

[9] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan, Chord, " A Scalable Peer-to-peer Lookup Protocol for Internet Applications ", IEEE/Acm Transactions on Networking

[10] WIDE Project, *http://www.wide.ad.jp*