記述名処理用分散名前管理サーバの ＯＤＰの考え方に基づく設計

古宇田　フミ子
東京大学 工学部 電子情報工学科

属性等から構成される名前である記述名管理の全体図の設計を行なうことを目的とする。方法として、 ODP(開放型分散処理)の標準化で用いられているviewpoint languageの考え方を利用して、幾つかの側面から、記述名管理の構成を試みた。 ODPの記述は、抽象的で、実装を目指していないことから、モデル化された結果は、抽象的ではあるが、必要なインタフェースや操作、オブジェクト間の関係等の管理の概略が分かるものとなった。

1. はじめに

利用者の立場を考慮した新しい名前管理法を作ることを目的とする。分散環境において、利用者が能動的に計算機資源を利用することができるように資源の持つ意味や機能の記述により資源の識別が可能となる方式を開発し、簡単なインタフェースの形で利用者に提供可能とする記述名管理の全体像を考察する。

利用者からの問い合わせ記述は、利用者の立場を考慮して、不完全な聞き方であっても、対処可能な柔軟な方式が重要である。また、この返り値としては、単に識別子だけではなく、環境に応じた識別子や、必要に応じて、識別子の列を返せるようなものにする。

この問題を明確にするために、記述名名前管理システムの体系を、見る立場を幾つか変えて、それぞれの観点から、分けて捉える。捉え方の指針として、 ISOとCCITTとで行なっているＯＤＰ(Open Distributed Processing)の標準化[Na93][RMODP1]の手法を利用する。ＯＤＰでは、複雑な構造を視点毎に分けて、簡単化できる（と言われている）Viewpoint language(視点言語)が提案されている。これを利用して、記述名管理の全体図の設計を試みる。この設計により、記述名管理で重要な点は何か、ＯＤＰでは、どのような考え方が重要か、ＯＤＰの考え方で設計するとどのような構成になるか、等を思考実験を行なうことで、調べる。

各視点言語間の関係(consistency)はどのようなものかも調べたい。

A conceptual design of a descriptive name server based on the ODP viewpoint languages

Fumiko Kouda
The faculty of engineering, University of Tokyo

1.1 背景

ＯＤＰの標準化の手法では、複雑な分散処理システムを視点毎に分けて、簡単化して設計をし易くする方針を採っている。このように視点を目的、用途別に固定して、システム全体を眺めると、構成要素が明確になると考えられる。

一方、ＯＤＰ参照モデル[N7988][N8125]では実装を目指していないので、記述が抽象的で、見えにくい点がある。このような抽象的な記述から本当に設計ができるかどうかも試みたい。

利用者側からは、対象の識別子が分からないが、それを利用したいという問題がある。識別子を知らないと資源が利用できない不便な状況を解決したい。特に問題となるのは、分散環境で、他計算機の状況が変化した時に、変化情報を簡単な方法で、能動的に知り、使用できるようにすることである。この場合、別の方法が必要になる。対象が持つ別の側面、その中でも対象の持つ属性を述べた名前を利用して識別子を問い合わせることができるようになれば、利用者の知っている情報を用いることができるので、便利である。

このような観点から、[IS7498]で述べられている「属性を説明したものの名前」である記述名の定義に従って、記述名管理を考察する。

名前管理で基本要素項目は、register bind resolve である。これらが、記述名の場合、どのような構成になるかが問題となる。利用者にはどのような名前形式を見せるか、又、処理系で用いる名前はどうなるか、処理系ではどのように分散を考慮し、名前のバインドや名前解決を図るかを明らかにする必要がある。

本文では特に、利用者の記述名による問い合わせの処理法を中心に考え、bindとregisterの処理については、扱わない。

2. 予備知識 -- ODP 参照モデル part2 part3

2.1 ODPの記述的モデル(文法解説：Part2)の構成 [N7988]

記述的モデルでは、existence と activity に関する基本となるモデルの枠組み、アーキテクチャの概念、ＯＤＰにおけるコンフォーマンスからなる。これらの概念の中で、本文に関係する概念を以下に挙げる。

<X>-templateは、<X> の共通性質を述べる。instantiateで、実在物となる。xの候補としては、object, action, interfaceがある。

predicate としてのtype、これに対応する setとしてのclass を一般的に規定している。これと各種の<X>-templateとを組み合わせた、<X>-template type と <X>-template class を定義している。これらの間には、（集合の）包含関係が定義されている(subclass, superclass, subtype, supertype)。

instantiationは、<X>-templateと他の必要な情報を用いて、結果として<X>が存在するようになる過程であり、refinementは specificationの度合いを変える過程を表す。

contractは、template, QoS, 有効期間、無効にする条件からなる。

behaviour(振る舞い)は、起こり得る条件下でのaction(動作)の集合。一つのオブジェクトが他のオブジェクトと behavioural compatibilityであるとは、最初

のオブジェクトを、環境の違いを気づかせないように二番目と取り替えることができることを言う。

2.2 Part2における名前付けの考え方

名前管理の基本要素は、現在の所、 name space, naming context, naming graph, name resolution, naming domain, name authority object であると規定されている。名前解決は「初期の名前と名前コンテクストから、その名前で表されるある名と実体を見いだす過程」と定義される。名前グラフの節点は名前コンテクストを表し、枝は関係を表す。

2.3 ODPの規定的モデル(機能解説 : Part3) [N8125]

ODPで重要な概念であるViewpoint(視点)について、各視点毎に concept(概念)とそのrule(規則)が、説明され、各種のfunction(機能)説明がある。

現在規定されているViewpointの定義は、以下のようになっている。

Framework concepts

1. Enterprise viewpoint: a viewpoint on an ODP system that focuses on the purpose, scope and policies for the system

2. Information viewpoint: a viewpoint on an ODP system that focuses on the semantics of information and information processing activities in the system

3. Computational viewpoint: a viewpoint on an ODP system that focuses on the functional decomposition of the system into objectswhich are candidate for distribution

4. Engineering viewpoint: a viewpoint on an ODP system that focuses on the infrastructure required to support distribution.

5. Technology viewpoint: a viewpoint on an ODP system that focuses on the choice of technology to support the system.

2.4 各視点に特有な概念と構成規則 [N8125]

2.4.1 Enterprise Viewpoint

この視点では、役割、活動、義務、方針に基づいて、ＯＤＰシステムの目的を規定する。用いられる概念には、part2で規定されている概念と、performative action, agent, artefact, policy maker, administrator, arbitrator, resource, resource manager, community, voluntary community, federationがある。

ＯＤＰシステムと環境は、community （＝目的を達成するためのオブジェクトの集まり）として表される。ＯＤＰの目的は、contractにより表される。オブジェクトは役割を持ち、これにより、 agent か, artefactとなる。performative action はオブジェクトの状態（義務を負う、義務を遂行する、義務を放棄する、行為をする許可を得る、禁止する）を変化させる。これを起動できるオブジェクトが agentで、できないものがartefactである。community への出入り規則は、contractにより決められる。resourceは、消耗してもしなくてもよい。

2.4.2 Information Viewpoint

情報の視点では、ＯＤＰシステムの情報の意味 (semantics)と、ＯＤＰシステム内の情報処理の要求事項を定義する。

この視点の概念は、Part2で用いられる用語と、 schema, static schema, invariant schema, dynamic schema, integrity rule, cardinality constraint relation, relational image, partition が使われる。

情報は、スキーマの集合から構成される。スキーマは、情報オブジェクトと integrity rule を記述する。情報オブジェクトは位置独立である。 integrity ruleは、情報の状態と構造を述べ、cardinality constraintも含む。

不変スキーマ(invariant schema)は、情報オブジェクトの中で、時間や振る舞い(behaviour) に依らず同じ構造を表すスキーマである。静的スキーマ(static schema)は、ある時点での情報オブジェクトの状態と構造を定義するスキーマである。動的スキーマ(dynamic schema)は、情報オブジェクトの不変スキーマと関係する振る舞いを定義した動作(action)テンプレートである。

静的情報は behaviourを持たず、静的スキーマで表され、動的情報は不変スキーマと動的スキーマの組み合わせで表される。

behaviour はstate の変化として表される。state はクラスの要素としてモデル化される。

2.4.3 Computational Viewpoint

計算的視点では、オブジェクト、その活動、それらの間の相互作用を分散透明性を持つ用語で定義する。これらは、interaction、 activity、 portability、failure の各規則からなる。この視点で用いられる概念は part2で定義された用語と、transaction, transactional properties, ACID properties, operation signature, environment constraint, operational interface template, operation execution, operation, transactional operation, operational interface, transactional interface, stream interface, stream signature, stream interface template, computational interface template, computational object template, binding object が用いられる。 ここで、transactional properties には visibility, consistency, recoverability, permanence, dependency が含まれる。図１に、computational object template の構造を示す。

2.4.4 Engineering Viewpoint

この視点では、 ODPシステムの機能の実行を可能とする、抽象的基幹構造を記述し、物理的分散とローカルシステム資源の管理に必要な抽象化を同定し、 ODP機能を支える異なるオブジェクトの役割を定め、オブジェクト間の参照点を定めることを行う。

この視点で用いられる概念は part2で定義された用語と、 nucleus object, capsule, cluster, active cluster, deactivated cluster, checkpoint, deactivation, reactivation, basic engineering object, stub object, binder object, interceptor, protocol object, channel, channel template, supporting object, communication interface, engineering interface

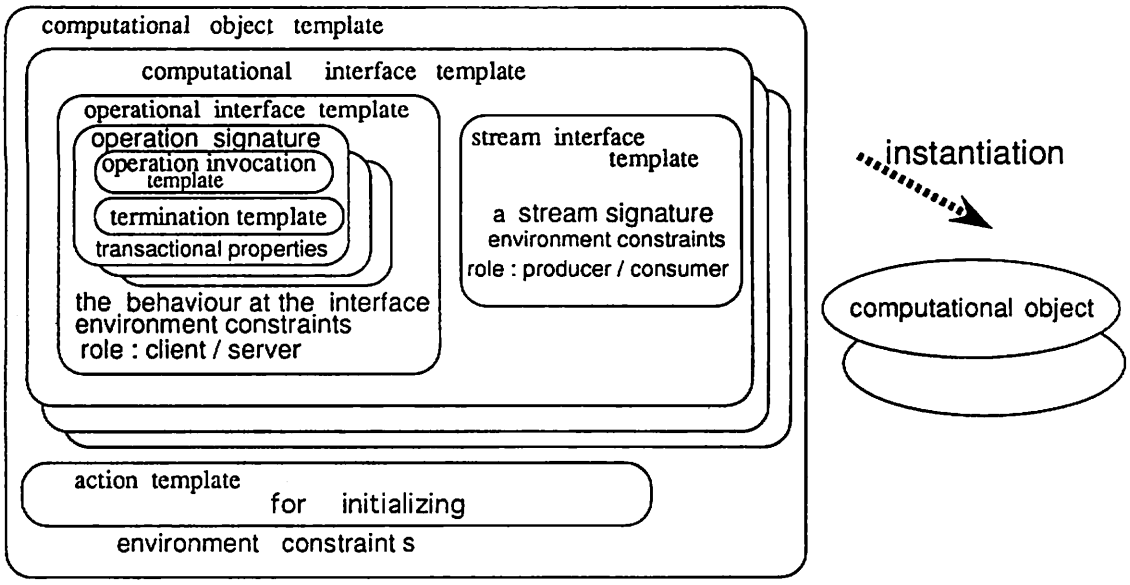

図 1　　computational object templateの構造

identifier, engineering location, interface reference, administrative domain, communication domain, node, migration, interface group がある。

basic engineering objectは、環境の制約を持つcomputational object の表現と定義されている。clusterは、basic engineering objectの集合で、capsuleは資源の割当とカプセル化の基本単位である。nucleus objectは、処理、貯蔵、通信機能の調整をするオブジェクトである。 stub object, binder object, protocol object, interceptor, channel, supporting object によってクライアント、サーバ間のチャネルの構造が規定される。特に、stub objectやbinder objectは、分散透明性の実現に寄与する。

2.4.5 Technology language

この視点では、ODPシステムの実装方法を述べている。この視点で用いられる概念は part2で定義された用語と、implementable standard (= template for a technology object), implementation (= instantiation の過程), IXIT (= implementation extra information for testing)である。

3. 目的とする記述名管理の概略、全体図（図２）

利用者の必要性は、対象の識別子が分からない場合、代替の方法として、対象の属性等からなる記述名を用いて、問い合わせを行い、対応する識別子の組を返り値として得ることである。

記述名管理システムでは、利用者の問い合わせに応じて、記述名の名前解決を行ない、この結果を利用者に返す。但し、ここで用いている名前解決の意味は、Ahmad[Ah87]が述べているように「locationを決定すること」や、Jacqmot[Ja90]らの定義にある「名前を物理アドレスに変換すること」ではなく、Part2 の定義に沿って、対応する識別子やそのグループが見つかることとする。但し、識別子から実体を求める処理は行わない。名前解決では、記述名のデータベースである

情報ベース(IB)と、利用者から提示された記述名の要素を記述名解決アルゴリズムを用いて比較する。IBに関係する要素が無い場合は、他ホストの記述名解決に問い合わせる。

対象が新たに生成された場合、利用者からの問い合わせに対処できるように、記述名向きの新規登録、変化した場合の更新処理を（異種）分散環境で行なう。

利用者とシステムの間でやり取りする情報は、記述名を用いた問い合わせ、識別子の（集合）からなる返り値、の二種類ある。これらを Univers[Bo90]で使われている naming program という呼び名に似せて、それぞれ、 Descriptive naming program(DNP), result naming program(RNP) と呼ぶ。

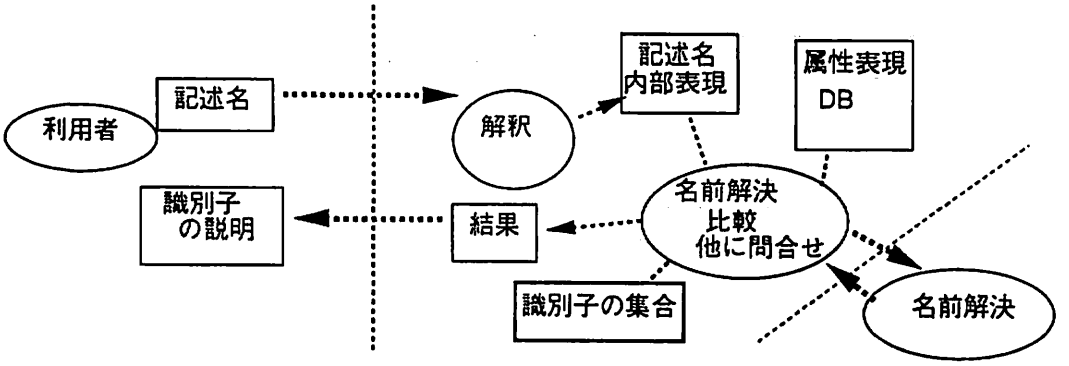

図２　記述名管理の概略、全体図

4. Viewpoint言語に基づく記述名名前管理サーバの設計

前章のモデルを、各視点での関係概念に適応することを試みる。

4.1 Enterprise Viewpoint における記述名管理

この視点では、community は、記述名管理と利用者グループが考えられる。それぞれの目的は、前者は、利用者からの記述名を名前解決し、対応する識別子を返す処理を行うことであり(contractのtemplateにはこの処理動作の記述がある)、後者は、識別子の分からない対象を属性等の記述名で捉え、これを提示し、結果を受け取る、こととなる。記述名管理のcommunity は、名前解決処理オブジェクト同士のcommunity、記述名登録処理オブジェクト同士のcommunity、記述名更新管理オブジェクト同士のcommunityに分けられる(図３)。

agent は、利用者、名前解決処理オブジェクト(ＲＶＯ)、記述名登録処理オブジェクト（ＲＧＯ）、記述名更新管理オブジェクト(ＵＰＯ)であり、artefactはDNP、RNP、IB、記述名解決アルゴリズム(RVAL)と見ることができる。これらのオブジェクトの役割(role)は、それぞれ、agent では、記述名を提示し結果の識別名（のグループ）を受け取る、記述名の名前解決を行なう、記述名を登録する、対象の変化に応じて記述名の管理表を更新する、artefactでは、記述名の記述、識別名の記述、記述名要素の情報管理表、名前解決処理となる。

resourceは識別子の集合と考えられる。

4.2 Information Viewpoint における記述名管理

この視点は、スキーマで構成される。スキーマはtemplateであると定義されて

いる。このことから、スキーマは、個別の対象を表すのではなく、ある条件を満たす集合の共通性質を述べるものと見なければならない。

情報オブジェクトはスキーマと関係が深い。情報オブジェクトには、IB、DNP、RNP、と利用者オブジェクト、RVO、RGO、UPO 等があると見ることができる。

不変スキーマには、DNP やRNP の書き方の仕様形式を表したものや名前解決アルゴリズムの構造が対応する。静的スキーマには、情報ベース(IB)の構造が当てはまる。利用者オブジェクトの動的スキーマは、 1) create a DNP, 2) issue a DNP to RVO, 3) wait for the RNP, 4) get the RNP, 5) consume the RNP となる動作列が記される。また、ＲＶＯの動的スキーマは、1) wait for a DNP, 2) get a DNP, 3) analyse(parse) the DNP, 4) consult RVAL, 5) inquire other RVO to resolve, 6) get the result from the other RVO, 7) get the result of DNP from RVAL, 8) determine the RNP と、表される。RVALの動的スキーマは、1) receive elements of DNP, 2) get IB, 3) compare elements of DNP with IB, 4) produce a set of identifiers となる。

4.3 Computational Viewpoint

この視点もtemplateで定義されているので、computational object template をinstantiate して、computational objectを作ることから始める。このテンプレートはインタフェースを重視して書かれている(図１)。利用者オブジェクトはcomputational interface templateとして、operational interface templateを持ち、インタフェースでの振舞は、 DNPを発行し、RNP を受け取ることであり、operation signature(操作標識)は、DNP を作成し、結果を待つことであり、役割はクライアントとなる。

名前解決オブジェクトの場合は、computational interface templateを複数持つ。すべて、operational interface templateからなる。役割がサーバとなる場合は、二種類ある。1)一つは、インタフェースでの振舞は、利用者オブジェクトの発行したDNP を受け取ることや、他のRVO から、そこでは不明であることが明確化されたDNP の要素についての問い合わせを受け取ること、記述名要素をRVALに発行することであり、操作標識は、 DNPの要素を解析し、記述名要素に分解すること、である。2) 他の一つは、インタフェースでの振舞は、RVALからの結果(識別子のグループ)を受け取ること、利用者オブジェクトに結果としてのRNP を返すこと、結果としての識別子のグループ等を問い合わせ元のRVO に戻すことや、問い合わせ用に他のRVO にここのRVALでは解決できない記述名要素を発行することであり、操作標識は、RVALから戻った結果と、先に解決されている部分を合わせて、識別子のグループをRNP として作成すること、不十分ならば、不明な記述名要素を明確化し、問い合わせに回すこと、である。役割がクライアントになる場合は、3)インタフェースでの振舞は、DNP 要素のうち不明な要素を2)の処理から受け取り、これについて、他のRVO に(選んで)問い合わせ、結果としての識別子のグループ等を問い合わせたRVO から受け取ることであり、操作標識は、RVO 間でループが起こらないように、他のRVO を選び、問い合わせ形式を作る、この結果を2)の処理に回す処理を行うこととなる。

RVALでは、サーバの役割を持ち、インタフェースでの振舞は、IBを受け取ること、RVO から記述名要素を受け取ること、結果としての識別子のグループ等を問い合わせ元のRVO の2)の処理に戻すことであり、操作標識は、IBと記述名要素とを比較しながら名前解決を行ない、識別子のグループが導き出されれば成功とし、また、これ以上名前解決が不可能になった場合は処理続行不能として、処理を終了することとなる（図４）。

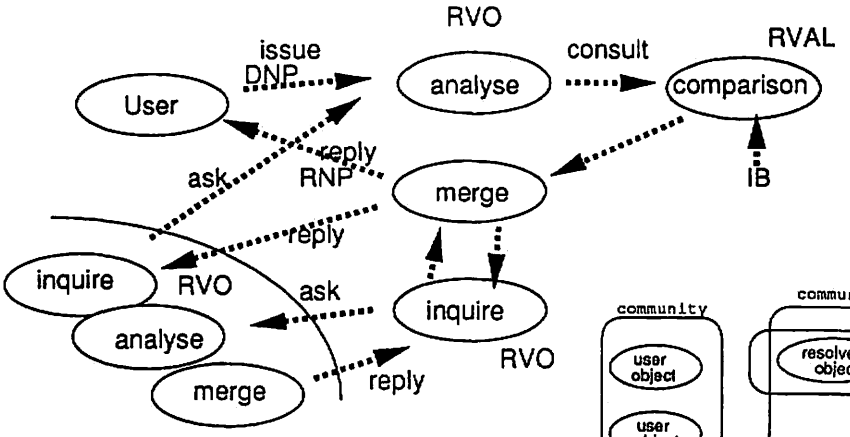

図 4　　computational object

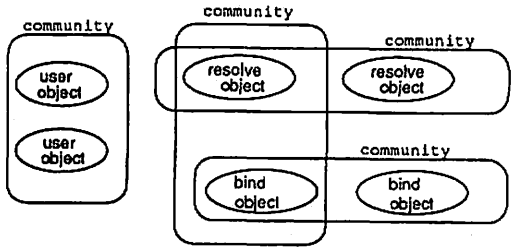

図 3　community　(Enterprise Viewpoint)

4.4 Engineering Viewpoint

この視点では、分散環境での構成法が重要になる。basic engineering object は、定義より計算的オブジェクトの表現であるから、利用者オブジェクト、RVO、RGO、UPO 等を対応させたものとなる。RVO は、工学視点では、分析、決定、問い合わせ、の３つの工学オブジェクトに分けられる。問い合わせオブジェクトは、stub object につながる。stub object と binder objectによりDNP やRNP の要素の異機種に対する変換等の管理を行う。クラスタとしては、RVO に対応する３つのオブジェクト、IB、記述名解決アルゴリズム(RVAL)のグループ、DNP、RNP、のグループ、記述名登録処理オブジェクト(RGO) と、記述名更新管理オブジェクト(UPO) 、IB、のグループがある。これらのクラスタと、stub object 等チャネルを構成するオブジェクトを併せて、一つのカプセルとなる。

5. 考察

5.1　記述名管理とＯＤＰ手法に基づく設計

ODP の手法に沿った記述名管理の構成法は、処理に各種の面があることを気づかせる。情報視点は、動的スキーマにより、行うべき処理が示され、計算的視点では、これらをインタフェースと、操作という面から捉えている。このような捉え方により、処理系の全体図は見易くなったと思われる。この構成法を有用にするためには、幾つかの残された問題点がある。例えば、具体化するための精密化

が必要な点がある。即ち、

利用者に柔軟な問い合わせを可能にするためには、DNPとRNPの構造の精密化、即ち、不変スキーマ、DNP RNP の書き方の仕様形式を厳密に定め、ここに柔軟な表現を可能にできるようにする必要がある。更に、RVALでの名前解決のスキーマの具体化、IBの構造の明確化等が必要となる。

名前管理で重要な要素のうち、registerとbindに関係する項目は本文で扱えなかった。この点も残された課題である。

工学視点では、ODP モデルで抽象的な視点ではあるが、モデルの枠組みが明確にされている。分散透明性の機能についても述べられている。本文ではこれに沿って、考察したが、充分とは言えない。精密化が必要になる。

記述名管理での名前解決と通常の識別名の名前解決の間に差異が見られる。通常の場合は、naming graphにより、階層化された名前が次々と解決されるが、記述名の場合は、記述名を要素分解し、RVALとIBとで対処する形を取っている。解決の流れが異なることが分かる。

5.2 ＯＤＰ流の考え方の設計は？

ISO 7層モデルでは、データは下層に行くほど、ヘッダが加わって、長くなる、という特徴があった。ODP では、各視点間ではそのようなことはなく、それぞれが独立して、「均等の詳しさ」を持ち、それぞれの見方に基づいて構成される。

Part2[N7988]の基本モデルの枠組みでは、以下の点が特徴的であると思われる。1) observable or not、即ち、動作(action)に関しては、外から見える振る舞い(behaviour)か、見えない内部動作か、を区別し、2) template, instantiation, specification, refinement等のオブジェクトに基づく捉え方をしている。また、3) predicate or set 、内包と外延の二通りの表記がある(例、type, class)。4) verb or noun 、即ち、動作としてか、状態かの区別をしている(例、bind)。part3、規定的モデルでは、 templateを基本にして設計し、instantiateにより実在物を実現しようとしている。各種の template の内容をきっちりと設計できることが分散設計の鍵になると思われる。

Computational language の概念の記述は、Pascalのプログラムのような構成を採っている。この理由は、後方参照を避けたため、と思われる。

視点毎の構成法は、分散に関わる部分と、関係ない部分を切り離して、構成することができる。

Engineering Viewpointで、初めて分散処理に関係する項目を陽に考慮しなければならなくなる。分散に関係した項目は比較的、機械的に作ることができる。一方、他の視点の考慮事項は、多様性があり、自由度が大きいので、純粋に分散処理に関わる項目に比して占める割合が大きいように思う。

5.3 Viewpoint間のconsistency

ODP part3 では、consistency constraints を以下のように定義している：L_1とL_2を視点言語とし、S_1をL_1のspecification、S_1'とS_2をL_2のspecificationとした場合、S_1の変換(transformation)がS_1'となった時、S_1'とS_2とがbehaviourally compatible になれば、S_1とS_2は consistent である。

このことを、情報視点と計算的視点でのRVALについて調べる。情報視点では、RVALは、一つの不変スキーマである。計算的視点に変換されると、サーバとして名前解決を行なう。ここで、RVALとは異なる名前解決アルゴリズムRNを用いて、記述名要素を入力として、対応する識別子のグループがRVALの場合と同じように導き出せれば、情報視点のRVALとRNは、consistency constraints を満たす、と言うように理解できる。

6. おわりに

利用者の立場を考慮した記述名名前管理法を、ODP の視点言語の考え方に沿って設計することを試みた。本文では、名前管理処理で重要な要素のうち、名前解決の部分について、モデル化を行なった。現在、ODP は、CD版であることや、抽象的に記述されているので、分かりにくい点があり、モデルも抽象的な結果になってしまった。しかし、モデル化の結果、処理に必要な情報の種類、インタフェースでやり取りする情報や処理、オブジェクト間の関係等の概略は掴むことができた。今後は、この観点を基にして、モデルを精密化することが課題である。また、名前管理処理の他の重要な要素の設計も残された課題である。

参考文献

[Ah87] M. Ahamad An Efficient Algorithm for Name Resolution in Computer Networks, Computer Networks and ISDN Systems 13 (1987) 301 - 311

[Bo90] Mic BOWMAN, Larry L. PETERSON and Andrey YEATTS Univers: An Attribute-based Name Server, SOFTWARE-Practice and Experience, Vol.20, No.4, 403-424(April 1990)

[Ja90] C. Jacqmot and E. Milgram Naming and network transparent process migration in loosely coupled distributed systems, 67 - 78, IFIP, 1990

[Na93] 中川路、田中、浅野 開放型分散処理の標準化の概要、電子情報通信学会誌（近日掲載）

[N8125] Draft Recommendation X.903: Basic Reference Model of Open Distributed Processing - Part3 : Prescriptive Model, ISO/IEC JTC SC21 N8125 June 1993

[N7988] Recommendation X.902: Basic Reference Model of Open Distributed Processing Part2: Descriptive Model, ISO/IEC JTC1 SC21 June 1993

[RMODP1] Draft Recommendation X.901: Basic Reference Model of Open Distributed Processing - Part1: Overview and guide to use ISO/IEC JTC1/SC21/WG7 N807 June 1993